111年-案件彙報管理系統(XRC) 掃描報告

專案名稱	111年-案件彙報管理系統(XRC)
掃描開始	2022年11月30日 下午 05:41:39
預設集合	OWASP TOP 10 - 2021
掃描時間	00h:08m:24s
被掃描的程式行數	62671
被掃描的檔案數	243
報告建立時間	2022年11月30日 下午 05:52:10
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300
團隊	Users
Checkmarx版本	9.4.2
掃描類別	完整的
來源	LocalPath
漏洞密度	8/1000 (漏洞/LOC)
掃描注釋	
可見性	公開

過濾器設置

嚴重程度：

包含在內: 高風險, 中風險, 低風險, 資訊

排除在外: 無

結果狀態：

包含在內: 校驗, 不可利用, 確認, 緊急, 推薦不可用

排除在外: 無

被分配給

包含在內: 全部

類別

包含在內:

未分類	全部
Custom	全部
PCI DSS v3.2.1	全部
OWASP Top 10 2013	全部
FISMA 2014	全部
NIST SP 800-53	全部
OWASP Top 10 2017	全部
OWASP Mobile Top 10 2016	全部
OWASP Top 10 API	全部
ASD STIG 4.10	全部
OWASP Top 10 2010	全部
OWASP Top 10 2021	全部

排除在外:

未分類	無
Custom	無
PCI DSS v3.2.1	無
OWASP Top 10 2013	無
FISMA 2014	無
NIST SP 800-53	無
OWASP Top 10 2017	無
OWASP Mobile Top 10 2016	無
OWASP Top 10 API	無
ASD STIG 4.10	無
OWASP Top 10 2010	無
OWASP Top 10 2021	無

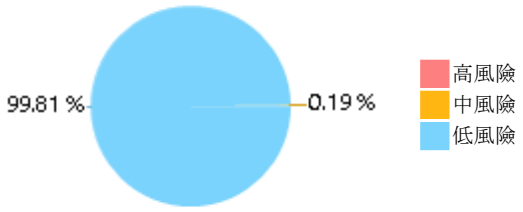
結果限制

未定義限值

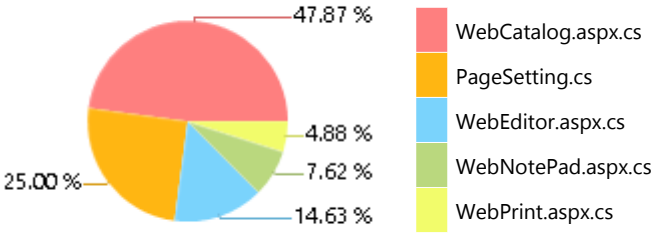
選中的問詢

選中的問詢列出在 掃描結果摘要

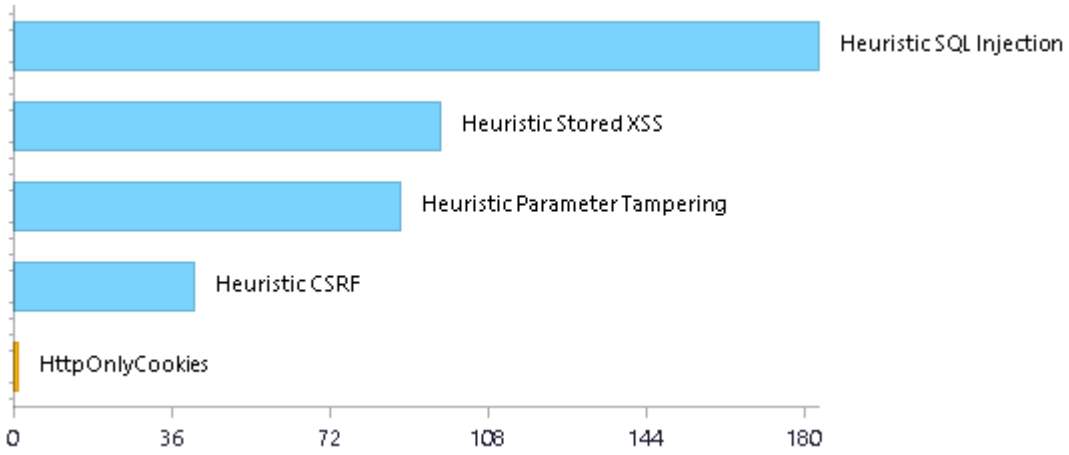
掃描結果摘要



最容易受攻擊的檔案



數量最多的前5類漏洞



掃描總結 - OWASP Top 10 2017

有關可見性和風險的詳細資訊及闡述參見： OWASP Top 10 2017

Category	Threat Agent	Exploitability	Weakness Prevalence	Weakness Detectability	Technical Impact	Business Impact	Issues Found	Best Fix Locations
A1-Injection	App. Specific	EASY	COMMON	EASY	SEVERE	App. Specific	184	25
A2-Broken Authentication*	App. Specific	EASY	COMMON	AVERAGE	SEVERE	App. Specific	0	0
A3-Sensitive Data Exposure	App. Specific	AVERAGE	WIDESPREAD	AVERAGE	SEVERE	App. Specific	43	39
A4-XML External Entities (XXE)	App. Specific	AVERAGE	COMMON	EASY	SEVERE	App. Specific	0	0
A5-Broken Access Control	App. Specific	AVERAGE	COMMON	AVERAGE	SEVERE	App. Specific	202	98
A6-Security Misconfiguration	App. Specific	EASY	WIDESPREAD	EASY	MODERATE	App. Specific	0	0
A7-Cross-Site Scripting (XSS)	App. Specific	EASY	WIDESPREAD	EASY	MODERATE	App. Specific	97	27
A8-Insecure Deserialization	App. Specific	DIFFICULT	COMMON	AVERAGE	SEVERE	App. Specific	0	0
A9-Using Components with Known Vulnerabilities*	App. Specific	AVERAGE	WIDESPREAD	AVERAGE	MODERATE	App. Specific	8	8
A10-Insufficient Logging & Monitoring	App. Specific	AVERAGE	WIDESPREAD	DIFFICULT	MODERATE	App. Specific	0	0

***** 專案掃描結果不包括所有相關的查詢。應該變更預設和/或篩選器以包括所有相關的標準查詢。

掃描總結 - OWASP Top 10 2021

Category	Issues Found	Best Fix Locations
A1-Broken Access Control	114	29
A2-Cryptographic Failures	0	0
A3-Injection	281	52
A4-Insecure Design	211	140
A5-Security Misconfiguration	8	8
A6-Vulnerable and Outdated Components	8	8
A7-Identification and Authentication Failures	1	1
A8-Software and Data Integrity Failures	6	6
A9-Security Logging and Monitoring Failures	49	49
A10-Server-Side Request Forgery	0	0

掃描總結 - OWASP Top 10 2013

有關可見性和風險的詳細資訊及闡述參見： OWASP Top 10 2013

Category	Threat Agent	Attack Vectors	Weakness Prevalence	Weakness Detectability	Technical Impact	Business Impact	Issues Found	Best Fix Locations
A1-Injection	EXTERNAL, INTERNAL, ADMIN USERS	EASY	COMMON	AVERAGE	SEVERE	ALL DATA	184	25
A2-Broken Authentication and Session Management	EXTERNAL, INTERNAL USERS	AVERAGE	WIDESPREAD	AVERAGE	SEVERE	AFFECTED DATA AND FUNCTIONS	0	0
A3-Cross-Site Scripting (XSS)	EXTERNAL, INTERNAL, ADMIN USERS	AVERAGE	VERY WIDESPREAD	EASY	MODERATE	AFFECTED DATA AND SYSTEM	97	27
A4-Insecure Direct Object References	SYSTEM USERS	EASY	COMMON	EASY	MODERATE	EXPOSED DATA	126	22
A5-Security Misconfiguration	EXTERNAL, INTERNAL, ADMIN USERS	EASY	COMMON	EASY	MODERATE	ALL DATA AND SYSTEM	0	0
A6-Sensitive Data Exposure	EXTERNAL, INTERNAL, ADMIN USERS, USERS BROWSERS	DIFFICULT	UNCOMMON	AVERAGE	SEVERE	EXPOSED DATA	41	37
A7-Missing Function Level Access Control	EXTERNAL, INTERNAL USERS	EASY	COMMON	AVERAGE	MODERATE	EXPOSED DATA AND FUNCTIONS	76	76
A8-Cross-Site Request Forgery (CSRF)*	USERS BROWSERS	AVERAGE	COMMON	EASY	MODERATE	AFFECTED DATA AND FUNCTIONS	41	7
A9-Using Components with Known Vulnerabilities*	EXTERNAL USERS, AUTOMATED TOOLS	AVERAGE	WIDESPREAD	DIFFICULT	MODERATE	AFFECTED DATA AND FUNCTIONS	8	8
A10-Unvalidated Redirects and Forwards	USERS BROWSERS	AVERAGE	WIDESPREAD	DIFFICULT	MODERATE	AFFECTED DATA AND FUNCTIONS	33	19

***** 專案掃描結果不包括所有相關的查詢。應該變更預設和/或篩選器以包括所有相關的標準查詢。

掃描總結 - PCI DSS v3.2.1

Category	Issues Found	Best Fix Locations
PCI DSS (3.2.1) - 6.5.1 - Injection flaws - particularly SQL injection	310	47
PCI DSS (3.2.1) - 6.5.2 - Buffer overflows*	0	0
PCI DSS (3.2.1) - 6.5.3 - Insecure cryptographic storage	0	0
PCI DSS (3.2.1) - 6.5.4 - Insecure communications*	0	0
PCI DSS (3.2.1) - 6.5.5 - Improper error handling*	49	45
PCI DSS (3.2.1) - 6.5.7 - Cross-site scripting (XSS)	97	27
PCI DSS (3.2.1) - 6.5.8 - Improper access control*	82	82
PCI DSS (3.2.1) - 6.5.9 - Cross-site request forgery	41	7
PCI DSS (3.2.1) - 6.5.10 - Broken authentication and session management*	0	0

***** 專案掃描結果不包括所有相關的查詢。應該變更預設和/或篩選器以包括所有相關的標準查詢。

掃描總結 - FISMA 2014

Category	Description	Issues Found	Best Fix Locations
Access Control*	Organizations must limit information system access to authorized users, processes acting on behalf of authorized users, or devices (including other information systems) and to the types of transactions and functions that authorized users are permitted to exercise.	38	1
Audit And Accountability*	Organizations must: (i) create, protect, and retain information system audit records to the extent needed to enable the monitoring, analysis, investigation, and reporting of unlawful, unauthorized, or inappropriate information system activity; and (ii) ensure that the actions of individual information system users can be uniquely traced to those users so they can be held accountable for their actions.	0	0
Configuration Management*	Organizations must: (i) establish and maintain baseline configurations and inventories of organizational information systems (including hardware, software, firmware, and documentation) throughout the respective system development life cycles; and (ii) establish and enforce security configuration settings for information technology products employed in organizational information systems.	14	10
Identification And Authentication*	Organizations must identify information system users, processes acting on behalf of users, or devices and authenticate (or verify) the identities of those users, processes, or devices, as a prerequisite to allowing access to organizational information systems.	6	6
Media Protection	Organizations must: (i) protect information system media, both paper and digital; (ii) limit access to information on information system media to authorized users; and (iii) sanitize or destroy information system media before disposal or release for reuse.	0	0
System And Communications Protection*	Organizations must: (i) monitor, control, and protect organizational communications (i.e., information transmitted or received by organizational information systems) at the external boundaries and key internal boundaries of the information systems; and (ii) employ architectural designs, software development techniques, and systems engineering principles that promote effective information security within organizational information systems.	0	0
System And Information Integrity*	Organizations must: (i) identify, report, and correct information and information system flaws in a timely manner; (ii) provide protection from malicious code at appropriate locations within organizational information systems; and (iii) monitor information system security alerts and advisories and take appropriate actions in response.	313	70

***** 專案掃描結果不包括所有相關的查詢。應該變更預設和/或篩選器以包括所有相關的標準查詢。

掃描總結 - NIST SP 800-53

Category	Issues Found	Best Fix Locations
AC-12 Session Termination (P2)	0	0
AC-3 Access Enforcement (P1)*	38	1
AC-4 Information Flow Enforcement (P1)	0	0
AC-6 Least Privilege (P1)	0	0
AU-9 Protection of Audit Information (P1)	0	0
CM-6 Configuration Settings (P2)	0	0
IA-5 Authenticator Management (P1)	0	0
IA-6 Authenticator Feedback (P2)	0	0
IA-8 Identification and Authentication (Non-Organizational Users) (P1)	0	0
SC-12 Cryptographic Key Establishment and Management (P1)	0	0
SC-13 Cryptographic Protection (P1)	0	0
SC-17 Public Key Infrastructure Certificates (P1)	0	0
SC-18 Mobile Code (P2)	0	0
SC-23 Session Authenticity (P1)*	41	7
SC-28 Protection of Information at Rest (P1)*	6	6
SC-4 Information in Shared Resources (P1)*	0	0
SC-5 Denial of Service Protection (P1)*	12	12
SC-8 Transmission Confidentiality and Integrity (P1)*	0	0
SI-10 Information Input Validation (P1)*	216	43
SI-11 Error Handling (P2)*	14	10
SI-15 Information Output Filtering (P0)	97	27
SI-16 Memory Protection (P1)*	0	0

***** 專案掃描結果不包括所有相關的查詢。應該變更預設和/或篩選器以包括所有相關的標準查詢。

掃描總結 - OWASP Mobile Top 10 2016

Category	Description	Issues Found	Best Fix Locations
M1-Improper Platform Usage	This category covers misuse of a platform feature or failure to use platform security controls. It might include Android intents, platform permissions, misuse of TouchID, the Keychain, or some other security control that is part of the mobile operating system. There are several ways that mobile apps can experience this risk.	0	0
M2-Insecure Data Storage*	This category covers insecure data storage and unintended data leakage.	0	0
M3-Insecure Communication*	This category covers poor handshaking, incorrect SSL versions, weak negotiation, cleartext communication of sensitive assets, etc.	0	0
M4-Insecure Authentication*	This category captures notions of authenticating the end user or bad session management. This can include: -Failing to identify the user at all when that should be required -Failure to maintain the user's identity when it is required -Weaknesses in session management	0	0
M5-Insufficient Cryptography	The code applies cryptography to a sensitive information asset. However, the cryptography is insufficient in some way. Note that anything and everything related to TLS or SSL goes in M3. Also, if the app fails to use cryptography at all when it should, that probably belongs in M2. This category is for issues where cryptography was attempted, but it wasnt done correctly.	0	0
M6-Insecure Authorization	This is a category to capture any failures in authorization (e.g., authorization decisions in the client side, forced browsing, etc.). It is distinct from authentication issues (e.g., device enrolment, user identification, etc.). If the app does not authenticate users at all in a situation where it should (e.g., granting anonymous access to some resource or service when authenticated and authorized access is required), then that is an authentication failure not an authorization failure.	0	0
M7-Client Code Quality*	This category is the catch-all for code-level implementation problems in the mobile client. That's distinct from server-side coding mistakes. This would capture things like buffer overflows, format string vulnerabilities, and various other code-level mistakes where the solution is to rewrite some code that's running on the mobile device.	0	0
M8-Code Tampering	This category covers binary patching, local resource modification, method hooking, method swizzling, and dynamic memory modification. Once the application is delivered to the mobile device, the code and data resources are resident there. An attacker can either directly modify the code, change the contents of memory dynamically, change or replace the system APIs that the application uses, or	0	0

	modify the application's data and resources. This can provide the attacker a direct method of subverting the intended use of the software for personal or monetary gain.		
M9-Reverse Engineering*	This category includes analysis of the final core binary to determine its source code, libraries, algorithms, and other assets. Software such as IDA Pro, Hopper, otool, and other binary inspection tools give the attacker insight into the inner workings of the application. This may be used to exploit other nascent vulnerabilities in the application, as well as revealing information about back end servers, cryptographic constants and ciphers, and intellectual property.	0	0
M10-Extraneous Functionality*	Often, developers include hidden backdoor functionality or other internal development security controls that are not intended to be released into a production environment. For example, a developer may accidentally include a password as a comment in a hybrid app. Another example includes disabling of 2-factor authentication during testing.	0	0

* 專案掃描結果不包括所有相關的查詢。應該變更預設和/或篩選器以包括所有相關的標準查詢。

掃描總結 - OWASP Top 10 API

Category	Issues Found	Best Fix Locations
API1-Broken Object Level Authorization	38	1
API2-Broken Authentication	0	0
API3-Excessive Data Exposure	12	8
API4-Lack of Resources and Rate Limiting	0	0
API5-Broken Function Level Authorization	0	0
API6-Mass Assignment	0	0
API7-Security Misconfiguration	16	16
API8-Injection	0	0
API9-Improper Assets Management*	0	0
API10-Insufficient Logging and Monitoring	49	49

***** 專案掃描結果不包括所有相關的查詢。應該變更預設和/或篩選器以包括所有相關的標準查詢。

掃描總結 - Custom

Category	Issues Found	Best Fix Locations
Must audit	0	0
Check	0	0
Optional	0	0

掃描總結 - ASD STIG 4.10

Category	Issues Found	Best Fix Locations
APSC-DV-000640 - CAT II The application must provide audit record generation capability for the renewal of session IDs.	0	0
APSC-DV-000650 - CAT II The application must not write sensitive data into the application logs.	0	0
APSC-DV-000660 - CAT II The application must provide audit record generation capability for session timeouts.	0	0
APSC-DV-000670 - CAT II The application must record a time stamp indicating when the event occurred.	0	0
APSC-DV-000680 - CAT II The application must provide audit record generation capability for HTTP headers including User-Agent, Referer, GET, and POST.	0	0
APSC-DV-000690 - CAT II The application must provide audit record generation capability for connecting system IP addresses.	0	0
APSC-DV-000700 - CAT II The application must record the username or user ID of the user associated with the event.	0	0
APSC-DV-000710 - CAT II The application must generate audit records when successful/unsuccessful attempts to grant privileges occur.	0	0
APSC-DV-000720 - CAT II The application must generate audit records when successful/unsuccessful attempts to access security objects occur.	0	0
APSC-DV-000730 - CAT II The application must generate audit records when successful/unsuccessful attempts to access security levels occur.	0	0
APSC-DV-000740 - CAT II The application must generate audit records when successful/unsuccessful attempts to access categories of information (e.g., classification levels) occur.	0	0
APSC-DV-000750 - CAT II The application must generate audit records when successful/unsuccessful attempts to modify privileges occur.	0	0
APSC-DV-000760 - CAT II The application must generate audit records when successful/unsuccessful attempts to modify security objects occur.	0	0
APSC-DV-000770 - CAT II The application must generate audit records when successful/unsuccessful attempts to modify security levels occur.	0	0
APSC-DV-000780 - CAT II The application must generate audit records when successful/unsuccessful attempts to modify categories of information (e.g., classification levels) occur.	0	0
APSC-DV-000790 - CAT II The application must generate audit records when successful/unsuccessful attempts to delete privileges occur.	0	0
APSC-DV-000800 - CAT II The application must generate audit records when successful/unsuccessful attempts to delete security levels occur.	0	0
APSC-DV-000810 - CAT II The application must generate audit records when successful/unsuccessful attempts to delete application database security objects occur.	0	0
APSC-DV-000820 - CAT II The application must generate audit records when successful/unsuccessful attempts to delete categories of information (e.g., classification levels) occur.	0	0
APSC-DV-000830 - CAT II The application must generate audit records when successful/unsuccessful logon attempts occur.	0	0
APSC-DV-000840 - CAT II The application must generate audit records for privileged activities or other system-level access.	0	0
APSC-DV-000850 - CAT II The application must generate audit records showing starting and ending time for user access to the system.	0	0
APSC-DV-000860 - CAT II The application must generate audit records when successful/unsuccessful accesses to objects occur.	0	0

APSC-DV-000870 - CAT II The application must generate audit records for all direct access to the information system.	0	0
APSC-DV-000880 - CAT II The application must generate audit records for all account creations, modifications, disabling, and termination events.	0	0
APSC-DV-000910 - CAT II The application must initiate session auditing upon startup.	0	0
APSC-DV-000940 - CAT II The application must log application shutdown events.	0	0
APSC-DV-000950 - CAT II The application must log destination IP addresses.	0	0
APSC-DV-000960 - CAT II The application must log user actions involving access to data.	0	0
APSC-DV-000970 - CAT II The application must log user actions involving changes to data.	0	0
APSC-DV-000980 - CAT II The application must produce audit records containing information to establish when (date and time) the events occurred.	0	0
APSC-DV-000990 - CAT II The application must produce audit records containing enough information to establish which component, feature or function of the application triggered the audit event.	0	0
APSC-DV-001000 - CAT II When using centralized logging; the application must include a unique identifier in order to distinguish itself from other application logs.	0	0
APSC-DV-001010 - CAT II The application must produce audit records that contain information to establish the outcome of the events.	0	0
APSC-DV-001020 - CAT II The application must generate audit records containing information that establishes the identity of any individual or process associated with the event.	0	0
APSC-DV-001030 - CAT II The application must generate audit records containing the full-text recording of privileged commands or the individual identities of group account users.	0	0
APSC-DV-001040 - CAT II The application must implement transaction recovery logs when transaction based.	0	0
APSC-DV-001050 - CAT II The application must provide centralized management and configuration of the content to be captured in audit records generated by all application components.	0	0
APSC-DV-001070 - CAT II The application must off-load audit records onto a different system or media than the system being audited.	0	0
APSC-DV-001080 - CAT II The application must be configured to write application logs to a centralized log repository.	0	0
APSC-DV-001090 - CAT II The application must provide an immediate warning to the SA and ISSO (at a minimum) when allocated audit record storage volume reaches 75% of repository maximum audit record storage capacity.	0	0
APSC-DV-001100 - CAT II Applications categorized as having a moderate or high impact must provide an immediate real-time alert to the SA and ISSO (at a minimum) for all audit failure events.	0	0
APSC-DV-001110 - CAT II The application must alert the ISSO and SA (at a minimum) in the event of an audit processing failure.	0	0
APSC-DV-001120 - CAT II The application must shut down by default upon audit failure (unless availability is an overriding concern).	0	0
APSC-DV-001130 - CAT II The application must provide the capability to centrally review and analyze audit records from multiple components within the system.	0	0
APSC-DV-001140 - CAT II The application must provide the capability to filter audit records for events of interest based upon organization-defined criteria.	0	0
APSC-DV-001150 - CAT II The application must provide an audit reduction capability that supports on-demand reporting requirements.	0	0
APSC-DV-001160 - CAT II The application must provide an audit reduction capability that supports on-demand audit review and analysis.	0	0
APSC-DV-001170 - CAT II The application must provide an audit reduction capability that supports after-the-fact investigations of security incidents.	0	0
APSC-DV-001180 - CAT II The application must provide a report generation capability that supports on-demand audit review and analysis.	0	0

APSC-DV-001190 - CAT II The application must provide a report generation capability that supports on-demand reporting requirements.	0	0
APSC-DV-001200 - CAT II The application must provide a report generation capability that supports after-the-fact investigations of security incidents.	0	0
APSC-DV-001210 - CAT II The application must provide an audit reduction capability that does not alter original content or time ordering of audit records.	0	0
APSC-DV-001220 - CAT II The application must provide a report generation capability that does not alter original content or time ordering of audit records.	0	0
APSC-DV-001250 - CAT II The applications must use internal system clocks to generate time stamps for audit records.	0	0
APSC-DV-001260 - CAT II The application must record time stamps for audit records that can be mapped to Coordinated Universal Time (UTC) or Greenwich Mean Time (GMT).	0	0
APSC-DV-001270 - CAT II The application must record time stamps for audit records that meet a granularity of one second for a minimum degree of precision.	0	0
APSC-DV-001280 - CAT II The application must protect audit information from any type of unauthorized read access.	0	0
APSC-DV-001290 - CAT II The application must protect audit information from unauthorized modification.	0	0
APSC-DV-001300 - CAT II The application must protect audit information from unauthorized deletion.	0	0
APSC-DV-001310 - CAT II The application must protect audit tools from unauthorized access.	0	0
APSC-DV-001320 - CAT II The application must protect audit tools from unauthorized modification.	0	0
APSC-DV-001330 - CAT II The application must protect audit tools from unauthorized deletion.	0	0
APSC-DV-001340 - CAT II The application must back up audit records at least every seven days onto a different system or system component than the system or component being audited.	0	0
APSC-DV-001570 - CAT II The application must electronically verify Personal Identity Verification (PIV) credentials.	0	0
APSC-DV-001350 - CAT II The application must use cryptographic mechanisms to protect the integrity of audit information.	0	0
APSC-DV-001360 - CAT II Application audit tools must be cryptographically hashed.	0	0
APSC-DV-001370 - CAT II The integrity of the audit tools must be validated by checking the files for changes in the cryptographic hash value.	0	0
APSC-DV-001390 - CAT II The application must prohibit user installation of software without explicit privileged status.	0	0
APSC-DV-001410 - CAT II The application must enforce access restrictions associated with changes to application configuration.	0	0
APSC-DV-001420 - CAT II The application must audit who makes configuration changes to the application.	0	0
APSC-DV-001430 - CAT II The application must have the capability to prevent the installation of patches, service packs, or application components without verification the software component has been digitally signed using a certificate that is recognized and approved by the orga	0	0
APSC-DV-001440 - CAT II The applications must limit privileges to change the software resident within software libraries.	0	0
APSC-DV-001460 - CAT II An application vulnerability assessment must be conducted.	0	0
APSC-DV-001480 - CAT II The application must prevent program execution in accordance with organization-defined policies regarding software program usage and restrictions, and/or rules authorizing the terms and conditions of software program usage.	0	0
APSC-DV-001490 - CAT II The application must employ a deny-all, permit-by-exception (whitelist) policy to allow the execution of authorized software programs.	0	0
APSC-DV-001500 - CAT II The application must be configured to disable non-essential capabilities.	0	0
APSC-DV-001510 - CAT II The application must be configured to use only functions, ports, and protocols permitted to it in the PPSM CAL.	0	0

APSC-DV-001520 - CAT II The application must require users to reauthenticate when organization-defined circumstances or situations require reauthentication.	0	0
APSC-DV-001530 - CAT II The application must require devices to reauthenticate when organization-defined circumstances or situations requiring reauthentication.	0	0
APSC-DV-001540 - CAT I The application must uniquely identify and authenticate organizational users (or processes acting on behalf of organizational users).	0	0
APSC-DV-001550 - CAT II The application must use multifactor (Alt. Token) authentication for network access to privileged accounts.	0	0
APSC-DV-001560 - CAT II The application must accept Personal Identity Verification (PIV) credentials.	0	0
APSC-DV-001580 - CAT II The application must use multifactor (e.g., CAC, Alt. Token) authentication for network access to non-privileged accounts.	0	0
APSC-DV-001590 - CAT II The application must use multifactor (Alt. Token) authentication for local access to privileged accounts.	0	0
APSC-DV-001600 - CAT II The application must use multifactor (e.g., CAC, Alt. Token) authentication for local access to non-privileged accounts.	0	0
APSC-DV-001610 - CAT II The application must ensure users are authenticated with an individual authenticator prior to using a group authenticator.	0	0
APSC-DV-001620 - CAT II The application must implement replay-resistant authentication mechanisms for network access to privileged accounts.	0	0
APSC-DV-001630 - CAT II The application must implement replay-resistant authentication mechanisms for network access to non-privileged accounts.	0	0
APSC-DV-001640 - CAT II The application must utilize mutual authentication when endpoint device non-repudiation protections are required by DoD policy or by the data owner.	0	0
APSC-DV-001650 - CAT II The application must authenticate all network connected endpoint devices before establishing any connection.	0	0
APSC-DV-001660 - CAT II Service-Oriented Applications handling non-releasable data must authenticate endpoint devices via mutual SSL/TLS.	0	0
APSC-DV-001670 - CAT II The application must disable device identifiers after 35 days of inactivity unless a cryptographic certificate is used for authentication.	0	0
APSC-DV-001680 - CAT I The application must enforce a minimum 15-character password length.*	0	0
APSC-DV-001690 - CAT II The application must enforce password complexity by requiring that at least one upper-case character be used.	0	0
APSC-DV-001700 - CAT II The application must enforce password complexity by requiring that at least one lower-case character be used.	0	0
APSC-DV-001710 - CAT II The application must enforce password complexity by requiring that at least one numeric character be used.	0	0
APSC-DV-001720 - CAT II The application must enforce password complexity by requiring that at least one special character be used.	0	0
APSC-DV-001730 - CAT II The application must require the change of at least 8 of the total number of characters when passwords are changed.	0	0
APSC-DV-001740 - CAT I The application must only store cryptographic representations of passwords.	0	0
APSC-DV-001850 - CAT I The application must not display passwords/PINs as clear text.	0	0
APSC-DV-001750 - CAT I The application must transmit only cryptographically-protected passwords.	0	0
APSC-DV-001760 - CAT II The application must enforce 24 hours/1 day as the minimum password lifetime.	0	0
APSC-DV-001770 - CAT II The application must enforce a 60-day maximum password lifetime restriction.	0	0
APSC-DV-001780 - CAT II The application must prohibit password reuse for a minimum of five generations.	0	0
APSC-DV-001790 - CAT II The application must allow the use of a temporary password for system logons with an immediate change to a permanent password.	0	0

APSC-DV-001795 - CAT II The application password must not be changeable by users other than the administrator or the user with which the password is associated.	0	0
APSC-DV-001800 - CAT II The application must terminate existing user sessions upon account deletion.	0	0
APSC-DV-001820 - CAT I The application, when using PKI-based authentication, must enforce authorized access to the corresponding private key.	0	0
APSC-DV-001830 - CAT II The application must map the authenticated identity to the individual user or group account for PKI-based authentication.	0	0
APSC-DV-001870 - CAT II The application must uniquely identify and authenticate non-organizational users (or processes acting on behalf of non-organizational users).	0	0
APSC-DV-001810 - CAT I The application, when utilizing PKI-based authentication, must validate certificates by constructing a certification path (which includes status information) to an accepted trust anchor.	0	0
APSC-DV-001840 - CAT II The application, for PKI-based authentication, must implement a local cache of revocation data to support path discovery and validation in case of the inability to access revocation information via the network.	0	0
APSC-DV-001860 - CAT II The application must use mechanisms meeting the requirements of applicable federal laws, Executive Orders, directives, policies, regulations, standards, and guidance for authentication to a cryptographic module.	0	0
APSC-DV-001880 - CAT II The application must accept Personal Identity Verification (PIV) credentials from other federal agencies.	0	0
APSC-DV-001890 - CAT II The application must electronically verify Personal Identity Verification (PIV) credentials from other federal agencies.	0	0
APSC-DV-002050 - CAT II Applications making SAML assertions must use FIPS-approved random numbers in the generation of SessionIndex in the SAML element AuthnStatement.	0	0
APSC-DV-001900 - CAT II The application must accept FICAM-approved third-party credentials.	0	0
APSC-DV-001910 - CAT II The application must conform to FICAM-issued profiles.	0	0
APSC-DV-001930 - CAT II Applications used for non-local maintenance sessions must audit non-local maintenance and diagnostic sessions for organization-defined auditable events.	0	0
APSC-DV-000310 - CAT III The application must have a process, feature or function that prevents removal or disabling of emergency accounts.	0	0
APSC-DV-001940 - CAT II Applications used for non-local maintenance sessions must implement cryptographic mechanisms to protect the integrity of non-local maintenance and diagnostic communications.	0	0
APSC-DV-001950 - CAT II Applications used for non-local maintenance sessions must implement cryptographic mechanisms to protect the confidentiality of non-local maintenance and diagnostic communications.	0	0
APSC-DV-001960 - CAT II Applications used for non-local maintenance sessions must verify remote disconnection at the termination of non-local maintenance and diagnostic sessions.	0	0
APSC-DV-001970 - CAT II The application must employ strong authenticators in the establishment of non-local maintenance and diagnostic sessions.	0	0
APSC-DV-001980 - CAT II The application must terminate all sessions and network connections when non-local maintenance is completed.	0	0
APSC-DV-001995 - CAT II The application must not be vulnerable to race conditions.	0	0
APSC-DV-002000 - CAT II The application must terminate all network connections associated with a communications session at the end of the session.	0	0
APSC-DV-002010 - CAT II The application must implement NSA-approved cryptography to protect classified information in accordance with applicable federal laws, Executive Orders, directives, policies, regulations, and standards.	0	0
APSC-DV-002020 - CAT II The application must utilize FIPS-validated cryptographic modules when signing application components.	0	0
APSC-DV-002030 - CAT II The application must utilize FIPS-validated cryptographic modules when generating cryptographic hashes.	0	0

APSC-DV-002040 - CAT II The application must utilize FIPS-validated cryptographic modules when protecting unclassified information that requires cryptographic protection.	0	0
APSC-DV-002150 - CAT II The application user interface must be either physically or logically separated from data storage and management interfaces.	0	0
APSC-DV-002210 - CAT II The application must set the HTTPOnly flag on session cookies.	0	0
APSC-DV-002220 - CAT II The application must set the secure flag on session cookies.	0	0
APSC-DV-002230 - CAT I The application must not expose session IDs.	0	0
APSC-DV-002240 - CAT I The application must destroy the session ID value and/or cookie on logoff or browser close.	0	0
APSC-DV-002250 - CAT II Applications must use system-generated session identifiers that protect against session fixation.	0	0
APSC-DV-002260 - CAT II Applications must validate session identifiers.*	0	0
APSC-DV-002270 - CAT II Applications must not use URL embedded session IDs.	0	0
APSC-DV-002280 - CAT II The application must not re-use or recycle session IDs.	0	0
APSC-DV-002290 - CAT II The application must use the Federal Information Processing Standard (FIPS) 140-2-validated cryptographic modules and random number generator if the application implements encryption, key exchange, digital signature, and hash functionality.	0	0
APSC-DV-002300 - CAT II The application must only allow the use of DoD-approved certificate authorities for verification of the establishment of protected sessions.	0	0
APSC-DV-002310 - CAT I The application must fail to a secure state if system initialization fails, shutdown fails, or aborts fail.	0	0
APSC-DV-002320 - CAT II In the event of a system failure, applications must preserve any information necessary to determine cause of failure and any information necessary to return to operations with least disruption to mission processes.	0	0
APSC-DV-002330 - CAT II The application must protect the confidentiality and integrity of stored information when required by DoD policy or the information owner.*	0	0
APSC-DV-002340 - CAT II The application must implement approved cryptographic mechanisms to prevent unauthorized modification of organization-defined information at rest on organization-defined information system components.	0	0
APSC-DV-002350 - CAT II The application must use appropriate cryptography in order to protect stored DoD information when required by the information owner or DoD policy.	0	0
APSC-DV-002360 - CAT II The application must isolate security functions from non-security functions.	0	0
APSC-DV-002370 - CAT II The application must maintain a separate execution domain for each executing process.	0	0
APSC-DV-002380 - CAT II Applications must prevent unauthorized and unintended information transfer via shared system resources.	0	0
APSC-DV-002390 - CAT II XML-based applications must mitigate DoS attacks by using XML filters, parser options, or gateways.	0	0
APSC-DV-002400 - CAT II The application must restrict the ability to launch Denial of Service (DoS) attacks against itself or other information systems.*	0	0
APSC-DV-002410 - CAT II The web service design must include redundancy mechanisms when used with high-availability systems.	0	0
APSC-DV-002420 - CAT II An XML firewall function must be deployed to protect web services when exposed to untrusted networks.	0	0
APSC-DV-002610 - CAT II The application must remove organization-defined software components after updated versions have been installed.	2	2
APSC-DV-002440 - CAT I The application must protect the confidentiality and integrity of transmitted information.	0	0
APSC-DV-002450 - CAT II The application must implement cryptographic mechanisms to prevent unauthorized disclosure of information and/or detect changes to information during transmission unless otherwise protected by alternative physical safeguards, such as, at a minimum, a Prot	0	0

APSC-DV-002460 - CAT II The application must maintain the confidentiality and integrity of information during preparation for transmission.	0	0
APSC-DV-002470 - CAT II The application must maintain the confidentiality and integrity of information during reception.	0	0
APSC-DV-002480 - CAT II The application must not disclose unnecessary information to users.	0	0
APSC-DV-002485 - CAT I The application must not store sensitive information in hidden fields.	0	0
APSC-DV-002490 - CAT I The application must protect from Cross-Site Scripting (XSS) vulnerabilities.	97	27
APSC-DV-002500 - CAT II The application must protect from Cross-Site Request Forgery (CSRF) vulnerabilities.*	41	7
APSC-DV-002510 - CAT I The application must protect from command injection.	0	0
APSC-DV-002520 - CAT II The application must protect from canonical representation vulnerabilities.	0	0
APSC-DV-002530 - CAT II The application must validate all input.	38	1
APSC-DV-002540 - CAT I The application must not be vulnerable to SQL Injection.	183	24
APSC-DV-002550 - CAT I The application must not be vulnerable to XML-oriented attacks.	0	0
APSC-DV-002560 - CAT I The application must not be subject to input handling vulnerabilities.*	18	10
APSC-DV-002570 - CAT II The application must generate error messages that provide information necessary for corrective actions without revealing information that could be exploited by adversaries.	24	20
APSC-DV-002580 - CAT II The application must reveal error messages only to the ISSO, ISSM, or SA.*	0	0
APSC-DV-002590 - CAT I The application must not be vulnerable to overflow attacks.*	0	0
APSC-DV-002630 - CAT II Security-relevant software updates and patches must be kept up to date.	0	0
APSC-DV-002760 - CAT II The application performing organization-defined security functions must verify correct operation of security functions.	0	0
APSC-DV-002900 - CAT II The ISSO must ensure application audit trails are retained for at least 1 year for applications without SAMI data, and 5 years for applications including SAMI data.	0	0
APSC-DV-002770 - CAT II The application must perform verification of the correct operation of security functions: upon system startup and/or restart; upon command by a user with privileged access; and/or every 30 days.	0	0
APSC-DV-002780 - CAT III The application must notify the ISSO and ISSM of failed security verification tests.	0	0
APSC-DV-002870 - CAT II Unsigned Category 1A mobile code must not be used in the application in accordance with DoD policy.	0	0
APSC-DV-002880 - CAT II The ISSO must ensure an account management process is implemented, verifying only authorized users can gain access to the application, and individual accounts designated as inactive, suspended, or terminated are promptly removed.	0	0
APSC-DV-002890 - CAT I Application web servers must be on a separate network segment from the application and database servers if it is a tiered application operating in the DoD DMZ.	0	0
APSC-DV-002910 - CAT II The ISSO must review audit trails periodically based on system documentation recommendations or immediately upon system security events.	0	0
APSC-DV-002920 - CAT II The ISSO must report all suspected violations of IA policies in accordance with DoD information system IA procedures.	0	0
APSC-DV-002930 - CAT II The ISSO must ensure active vulnerability testing is performed.	0	0
APSC-DV-002980 - CAT II New IP addresses, data services, and associated ports used by the application must be submitted to the appropriate approving authority for the organization, which in turn will be submitted through the DoD Ports, Protocols, and Services Management (DoD PPS	0	0
APSC-DV-002950 - CAT II Execution flow diagrams and design documents must be created to show how deadlock and recursion issues in web services are being mitigated.	0	0
APSC-DV-002960 - CAT II The designer must ensure the application does not store configuration and control files in the same directory as user data.	0	0
APSC-DV-002970 - CAT II The ISSO must ensure if a DoD STIG or NSA guide is not available, a third-party	0	0

product will be configured by following available guidance.		
APSC-DV-002990 - CAT II The application must be registered with the DoD Ports and Protocols Database.	0	0
APSC-DV-002990 - CAT II The application must be registered with the DoD Ports and Protocols Database.	0	0
APSC-DV-002995 - CAT II The Configuration Management (CM) repository must be properly patched and STIG compliant.	0	0
APSC-DV-003000 - CAT II Access privileges to the Configuration Management (CM) repository must be reviewed every three months.	0	0
APSC-DV-003010 - CAT II A Software Configuration Management (SCM) plan describing the configuration control and change management process of application objects developed by the organization and the roles and responsibilities of the organization must be created and maintained.	0	0
APSC-DV-003020 - CAT II A Configuration Control Board (CCB) that meets at least every release cycle, for managing the Configuration Management (CM) process must be established.	0	0
APSC-DV-003030 - CAT II The application services and interfaces must be compatible with and ready for IPv6 networks.	0	0
APSC-DV-003040 - CAT II The application must not be hosted on a general purpose machine if the application is designated as critical or high availability by the ISSO.	0	0
APSC-DV-003050 - CAT II A disaster recovery/continuity plan must exist in accordance with DoD policy based on the applications availability requirements.	0	0
APSC-DV-003060 - CAT II Recovery procedures and technical system features must exist so recovery is performed in a secure and verifiable manner. The ISSO will document circumstances inhibiting a trusted recovery.	0	0
APSC-DV-003070 - CAT II Data backup must be performed at required intervals in accordance with DoD policy.	0	0
APSC-DV-003080 - CAT II Back-up copies of the application software or source code must be stored in a fire-rated container or stored separately (offsite).	0	0
APSC-DV-003090 - CAT II Procedures must be in place to assure the appropriate physical and technical protection of the backup and restoration of the application.	0	0
APSC-DV-003100 - CAT II The application must use encryption to implement key exchange and authenticate endpoints prior to establishing a communication channel for key exchange.	0	0
APSC-DV-003110 - CAT I The application must not contain embedded authentication data.	6	6
APSC-DV-003120 - CAT I The application must have the capability to mark sensitive/classified output when required.	0	0
APSC-DV-003130 - CAT III Prior to each release of the application, updates to system, or applying patches; tests plans and procedures must be created and executed.	0	0
APSC-DV-003150 - CAT II At least one tester must be designated to test for security flaws in addition to functional testing.	0	0
APSC-DV-003140 - CAT II Application files must be cryptographically hashed prior to deploying to DoD operational networks.	0	0
APSC-DV-003160 - CAT III Test procedures must be created and at least annually executed to ensure system initialization, shutdown, and aborts are configured to verify the system remains in a secure state.	0	0
APSC-DV-003170 - CAT II An application code review must be performed on the application.	0	0
APSC-DV-003180 - CAT III Code coverage statistics must be maintained for each release of the application.	0	0
APSC-DV-003190 - CAT II Flaws found during a code review must be tracked in a defect tracking system.	0	0
APSC-DV-003200 - CAT II The changes to the application must be assessed for IA and accreditation impact prior to implementation.	0	0
APSC-DV-003210 - CAT II Security flaws must be fixed or addressed in the project plan.	0	0
APSC-DV-003215 - CAT III The application development team must follow a set of coding standards.	0	0
APSC-DV-003220 - CAT III The designer must create and update the Design Document for each release of the application.	0	0

APSC-DV-003230 - CAT II Threat models must be documented and reviewed for each application release and updated as required by design and functionality changes or when new threats are discovered.	0	0
APSC-DV-003235 - CAT II The application must not be subject to error handling vulnerabilities.*	0	0
APSC-DV-003250 - CAT I The application must be decommissioned when maintenance or support is no longer available.	0	0
APSC-DV-003236 - CAT II The application development team must provide an application incident response plan.	0	0
APSC-DV-003240 - CAT I All products must be supported by the vendor or the development team.	0	0
APSC-DV-003260 - CAT III Procedures must be in place to notify users when an application is decommissioned.	0	0
APSC-DV-003270 - CAT II Unnecessary built-in application accounts must be disabled.	0	0
APSC-DV-003280 - CAT I Default passwords must be changed.	0	0
APSC-DV-003330 - CAT II The system must alert an administrator when low resource conditions are encountered.	0	0
APSC-DV-003285 - CAT II An Application Configuration Guide must be created and included with the application.	0	0
APSC-DV-003290 - CAT II If the application contains classified data, a Security Classification Guide must exist containing data elements and their classification.	0	0
APSC-DV-003300 - CAT II The designer must ensure uncategorized or emerging mobile code is not used in applications.	0	0
APSC-DV-003310 - CAT II Production database exports must have database administration credentials and sensitive data removed before releasing the export.	0	0
APSC-DV-003320 - CAT II Protections against DoS attacks must be implemented.	0	0
APSC-DV-003340 - CAT III At least one application administrator must be registered to receive update notifications, or security alerts, when automated alerts are available.	0	0
APSC-DV-003360 - CAT III The application must generate audit records when concurrent logons from different workstations occur.	0	0
APSC-DV-003345 - CAT III The application must provide notifications or alerts when product update and security related patches are available.	0	0
APSC-DV-003350 - CAT II Connections between the DoD enclave and the Internet or other public or commercial wide area networks must require a DMZ.	0	0
APSC-DV-003400 - CAT II The Program Manager must verify all levels of program management, designers, developers, and testers receive annual security training pertaining to their job function.	0	0
APSC-DV-000010 - CAT II The application must provide a capability to limit the number of logon sessions per user.	0	0
APSC-DV-000060 - CAT II The application must clear temporary storage and cookies when the session is terminated.	0	0
APSC-DV-000070 - CAT II The application must automatically terminate the non-privileged user session and log off non-privileged users after a 15 minute idle time period has elapsed.	0	0
APSC-DV-000080 - CAT II The application must automatically terminate the admin user session and log off admin users after a 10 minute idle time period is exceeded.	0	0
APSC-DV-000090 - CAT II Applications requiring user access authentication must provide a logoff capability for user initiated communication session.	0	0
APSC-DV-000100 - CAT III The application must display an explicit logoff message to users indicating the reliable termination of authenticated communications sessions.	0	0
APSC-DV-000110 - CAT II The application must associate organization-defined types of security attributes having organization-defined security attribute values with information in storage.	0	0
APSC-DV-000120 - CAT II The application must associate organization-defined types of security attributes having organization-defined security attribute values with information in process.	0	0
APSC-DV-000130 - CAT II The application must associate organization-defined types of security attributes	0	0

having organization-defined security attribute values with information in transmission.		
APSC-DV-000160 - CAT II The application must implement DoD-approved encryption to protect the confidentiality of remote access sessions.	0	0
APSC-DV-000170 - CAT II The application must implement cryptographic mechanisms to protect the integrity of remote access sessions.	0	0
APSC-DV-000190 - CAT I Messages protected with WS_Security must use time stamps with creation and expiration times.	0	0
APSC-DV-000180 - CAT II Applications with SOAP messages requiring integrity must include the following message elements:-Message ID-Service Request-Timestamp-SAML Assertion (optionally included in messages) and all elements of the message must be digitally signed.	0	0
APSC-DV-000200 - CAT I Validity periods must be verified on all application messages using WS-Security or SAML assertions.	0	0
APSC-DV-000210 - CAT II The application must ensure each unique asserting party provides unique assertion ID references for each SAML assertion.	0	0
APSC-DV-000220 - CAT II The application must ensure encrypted assertions, or equivalent confidentiality protections are used when assertion data is passed through an intermediary, and confidentiality of the assertion data is required when passing through the intermediary.	0	0
APSC-DV-000230 - CAT I The application must use the NotOnOrAfter condition when using the SubjectConfirmation element in a SAML assertion.	0	0
APSC-DV-000240 - CAT I The application must use both the NotBefore and NotOnOrAfter elements or OneTimeUse element when using the Conditions element in a SAML assertion.	0	0
APSC-DV-000250 - CAT II The application must ensure if a OneTimeUse element is used in an assertion, there is only one of the same used in the Conditions element portion of an assertion.	0	0
APSC-DV-000260 - CAT II The application must ensure messages are encrypted when the SessionIndex is tied to privacy data.	0	0
APSC-DV-000290 - CAT II Shared/group account credentials must be terminated when members leave the group.	0	0
APSC-DV-000280 - CAT II The application must provide automated mechanisms for supporting account management functions.	0	0
APSC-DV-000300 - CAT II The application must automatically remove or disable temporary user accounts 72 hours after account creation.	0	0
APSC-DV-000320 - CAT III The application must automatically disable accounts after a 35 day period of account inactivity.	0	0
APSC-DV-000330 - CAT II Unnecessary application accounts must be disabled, or deleted.	0	0
APSC-DV-000420 - CAT II The application must automatically audit account enabling actions.	0	0
APSC-DV-000340 - CAT II The application must automatically audit account creation.	0	0
APSC-DV-000350 - CAT II The application must automatically audit account modification.	0	0
APSC-DV-000360 - CAT II The application must automatically audit account disabling actions.	0	0
APSC-DV-000370 - CAT II The application must automatically audit account removal actions.	0	0
APSC-DV-000380 - CAT III The application must notify System Administrators and Information System Security Officers when accounts are created.	0	0
APSC-DV-000390 - CAT III The application must notify System Administrators and Information System Security Officers when accounts are modified.	0	0
APSC-DV-000400 - CAT III The application must notify System Administrators and Information System Security Officers of account disabling actions.	0	0
APSC-DV-000410 - CAT III The application must notify System Administrators and Information System Security Officers of account removal actions.	0	0
APSC-DV-000430 - CAT III The application must notify System Administrators and Information System Security Officers of account enabling actions.	0	0
APSC-DV-000440 - CAT II Application data protection requirements must be identified and documented.	0	0

APSC-DV-000520 - CAT II The application must audit the execution of privileged functions.	0	0
APSC-DV-000450 - CAT II The application must utilize organization-defined data mining detection techniques for organization-defined data storage objects to adequately detect data mining attempts.	0	0
APSC-DV-000460 - CAT I The application must enforce approved authorizations for logical access to information and system resources in accordance with applicable access control policies.	0	0
APSC-DV-000470 - CAT II The application must enforce organization-defined discretionary access control policies over defined subjects and objects.	0	0
APSC-DV-000480 - CAT II The application must enforce approved authorizations for controlling the flow of information within the system based on organization-defined information flow control policies.*	0	0
APSC-DV-000490 - CAT II The application must enforce approved authorizations for controlling the flow of information between interconnected systems based on organization-defined information flow control policies.	0	0
APSC-DV-000500 - CAT II The application must prevent non-privileged users from executing privileged functions to include disabling, circumventing, or altering implemented security safeguards/countermeasures.	0	0
APSC-DV-000510 - CAT I The application must execute without excessive account permissions.	0	0
APSC-DV-000530 - CAT I The application must enforce the limit of three consecutive invalid logon attempts by a user during a 15 minute time period.	0	0
APSC-DV-000560 - CAT III The application must retain the Standard Mandatory DoD Notice and Consent Banner on the screen until users acknowledge the usage conditions and take explicit actions to log on for further access.	0	0
APSC-DV-000540 - CAT II The application administrator must follow an approved process to unlock locked user accounts.	0	0
APSC-DV-000550 - CAT III The application must display the Standard Mandatory DoD Notice and Consent Banner before granting access to the application.	0	0
APSC-DV-000570 - CAT III The publicly accessible application must display the Standard Mandatory DoD Notice and Consent Banner before granting access to the application.	0	0
APSC-DV-000580 - CAT III The application must display the time and date of the users last successful logon.	0	0
APSC-DV-000630 - CAT II The application must provide audit record generation capability for the destruction of session IDs.	0	0
APSC-DV-000590 - CAT II The application must protect against an individual (or process acting on behalf of an individual) falsely denying having performed organization-defined actions to be covered by non-repudiation.	0	0
APSC-DV-000600 - CAT II For applications providing audit record aggregation, the application must compile audit records from organization-defined information system components into a system-wide audit trail that is time-correlated with an organization-defined level of tolerance	0	0
APSC-DV-000610 - CAT II The application must provide the capability for organization-identified individuals or roles to change the auditing to be performed on all application components, based on all selectable event criteria within organization-defined time thresholds.	0	0
APSC-DV-000620 - CAT II The application must provide audit record generation capability for the creation of session IDs.	0	0

***** 專案掃描結果不包括所有相關的查詢。應該變更預設和/或篩選器以包括所有相關的標準查詢。

掃描總結 - OWASP Top 10 2010

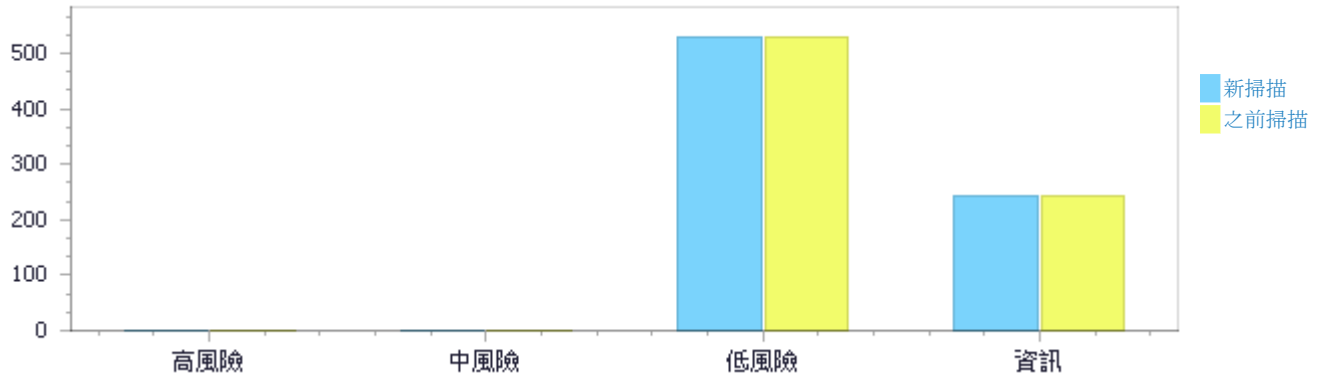
Category	Issues Found	Best Fix Locations
A1-Injection*	0	0
A2-Cross-Site Scripting (XSS)	0	0
A3-Broken Authentication and Session Management*	0	0
A4-Insecure Direct Object References	0	0
A5-Cross-Site Request Forgery (CSRF)	0	0
A6-Security Misconfiguration*	1	1
A7-Insecure Cryptographic Storage	0	0
A8-Failure to Restrict URL Access	0	0
A9-Insufficient Transport Layer Protection	0	0
A10-Unvalidated Redirects and Forwards	33	19

***** 專案掃描結果不包括所有相關的查詢。應該變更預設和/或篩選器以包括所有相關的標準查詢。

掃描結果分佈

與2022/11/30 下午 03:58的專案掃描比較

	高風險	中風險	低風險	資訊	總共
新問題	0	0	0	0	0
反覆出現的問題	0	1	530	241	772
總共	0	1	530	241	772
已修復的問題	0	0	0	0	0



掃描結果分佈

	高風險	中風險	低風險	資訊	總共
校驗	0	1	530	241	772
不可利用	0	0	0	0	0
確認	0	0	0	0	0
緊急	0	0	0	0	0
推薦不可用	0	0	0	0	0
總共	0	1	530	241	772

掃描結果摘要

漏洞類別	事件	嚴重程度：
HttpOnlyCookies	1	中風險
Heuristic SQL Injection	183	低風險
Heuristic Stored XSS	97	低風險
Heuristic Parameter Tampering	88	低風險
Heuristic CSRF	41	低風險

Heuristic DB Parameter Tampering	38	低風險
Client Potential DOM Open Redirect	18	低風險
Client DOM Open Redirect	15	低風險
Improper Exception Handling	12	低風險
Information Exposure Through an Error Message	12	低風險
Client JQuery Deprecated Symbols	8	低風險
Password in Configuration File	6	低風險
Client Side Only Validation	3	低風險
DebugEnabled	2	低風險
Information Exposure via Headers	2	低風險
Unencrypted Web Config File	2	低風險
Client Regex Injection	1	低風險
Missing Content Security Policy	1	低風險
Use Of Hardcoded Password	1	低風險
Insufficient Logging of Exceptions	90	資訊
Exposure of Resource to Wrong Sphere	73	資訊
Insufficient Logging of Sensitive Operations	49	資訊
Pages Without Global Error Handler	23	資訊
Hardcoded Absolute Path	6	資訊

10個最容易受攻擊的檔案

高級和中級漏洞

檔案名稱	找到的問題
WebBatchPrint.aspx.cs	1

已掃描的檔案

檔案名稱	檔案大小 KB	檢驗和
.ActiveScans/1132233	0	e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
.SourceControl/0000000243_000045571357_00-235867128	0	e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
App_Data/logs/archives/log.2022-03-25.txt	2	44c362fcc577a68798e47b995bb50b55f2f9f504074432ce08fc13206a07a798
App_Data/logs/log.current.txt	2	8dd99d9cc0d4715f04c888c0595e70c783a14a069aa71077778ef028b9f97fa6
Authority.ComDepRoleUser.aspx	2	78aba23aa5b99ebba78055749f85220701c517c9a5f0a4b7ded2902450d64c55
Authority.RoleComDepUser.aspx	3	406fcae3a78678dcf7b9aae4c30ebd4a5c07321f253059b844066ccc919ece18
Authority.RoleUserList.aspx	1	410286280522978aa86a646eeed6991764aba52c83db67e75cdd34304df0ce9b
bin/Cdsys.KM.Struct.dll	9	2a84f362819b25e1426d148c73e81366f78cc8725b441d646dbb0b88f8848ef7
bin/Cdsys.KM.TitleBannerC.dll	4	a003c1b592ef7fca0f8dc999091163f9b7967e503ca4e01ea3cc42e8a8de4859
bin/Cdsys.KM.Tree.dll	5	77d187f134dfa49a7d4ccaa4b918e1e395b4738c490a73d6fc8f74a53328a6ab
bin/Cdsys.km.UserIPEmail.dll	11	5152d1f83ffca3461a964713fdd1f01d6e06593915dc5571dc46c51ff26507a8
bin/Cdsys.KM.Utility.dll	203	cb1b084928d7173c15625b2774de459bbae1c02ab1b4923444f1302ed7f24a9c
bin/ChangeDocGroupName.dll	6	d88ea3bf5af0b7010d0dc4731fc70177006a9da735421f4f480e077941530f79
bin/ChangeDocName.dll	6	ed56835e6ec275d5f250f47676b139ca5ac9138d8750bcfa3911d694cdd7021b
bin/ChangeDocShowName.dll	7	29045fbeb29ab3049ba9370655f0fd62d0ae4f3d7a4d85124b2022187d860aa5
bin/ChangeMarquee.dll	5	ae912a61fdd4549bc35c28a5b95694c5154775ffe8e15510ae1de03ff6382823
bin/ChangeSystemName.dll	5	3f0062457870822838979230 4e7c67a557b5f02f7c740ccdb77166151fd828b2
bin/Commons.dll	9	c67075834afd9645de11e338f1e2ee462cb09adec273e809e7f06d95c137607a
bin/ICSharpCode.SharpZipLib.dll	132	a74ee6f9e54ae51073362003a62d5a70804b4dce13bbaac62b1773431a6334ca
bin/ICSharpCode.SharpZipLib.xml	564	ed45bcbc288668cf5c992a609e81b416832b5cdc972080137c6dd95e4ff368e0
bin/log4net.dll	109	a997fa053dae2e39badd8e43fb4a97f42ff638ec01785f086b2d26c14b02a2db
bin/Newtonsoft.Json.dll	490	882d1e04b099e1be116c6abdee0fc0bb326

		b1c715a7db6de0b14da083a404167
bin/Newtonsoft.Json.xml	694	2d46b37b086d6848af5f021d2d7a40581ce78aadd8ee39d309aee4771a0eeccf
bin/NLog.dll	661	5612ce9d1dc0fbdee4b532b9e2fb38678d04074f1ad153276e4038d8f1e82145
bin/NVelocity.dll	148	7f83dd1effdc061f34c497020ccda9a4868f83cb0db23681b7d9f986181e242a
bin/SecureCnnStr.dll	9	67b4c332ce637814bfc03ec1cb5383887cf72514f886de75872320783f765a04
bin/SSODLL.dll	7	c3ab81c7980541a14868f3e46de79d2c176ad79a585e92b318540d57f4fb4b43
bin/TCCKM.dll	348	13b44e990023a818d832654856f5ed33cc2f4b27aa8117d583a6cac8edf33c5a
bin/WebAutoClassMtn.dll	45	08bb7c03693de43e6629d50ecf620cbac084b3397dc5f2f3186c51791faca609
bin/WebCategoryReport.dll	16	2002e8c76dcb99d8228e21dbc46b6fd06ae57cef5c4816a42b1340e1f4da0191
bin/WebChangeDocName.dll	7	1fdc18b5976e2b025b55d80794b8df1d56d1b3d3f58f852acfb2c65cabf52959
bin/WebCmpLogin.dll	14	f9b5c102b1f7b9728a2c85e1ed2c664a4b3ac3992f92d2c4c26f29c05dbf6935
bin/WebCreateStdDoc.dll	10	a7c593a61bddb6074f7e22615852221edb32356cee5f017809b781e1e05b937c
bin/WebDeptQueryCount.dll	12	c3c1d9e448f96e58488afe5c4b913643d90db83c2396a92e43e990fe8419e5cd
bin/WebDocCount.dll	41	6a7d12c665f3f9652b5a6df942412a926f7c34f33c838974b6931cf59424c396
bin/WebDocCountCompare.dll	31	286e712c916aefdf09b2308fad616f84c26ba0f54553cdd6566a0fb906cc65d5
bin/WebDocQuery.dll	17	87e6d58e32487aeb2d2330afa3163d567208f1720559eb541f97fb854365535a
bin/WebDocRecoverMtn.dll	19	682252038dfe448b27b9d2f7df697ca04cb35f612e8ee06008bbc1a0de6ab96f
bin/WebDocReview.dll	14	a55362a9329d33fc2cf5b0f8a57d9a27edaf71c7a230733a7114f8722609a74f
bin/WebLogEvent.dll	20	5d39210cfcfdbb83b7797bb7bcd5c335d99d41889cd2b29221eb05c29f20cd81
bin/WebReport2019.dll	158	71f4ae90996b26342f68c00853bf6b53cabea8cda2199993bbecfb3ecde4965f
bin/WebReportMtn.dll	6	cae06733ff407789055e46c123ab90af312933b9eec571c6bd53ee9e4754c888
bin/WebSelect.dll	50	47d74234e4e6f7726184fdb8972d626e5593c9a2863c9ac1380a74fa36c6b1c3
bin/WebShowTool.dll	7	8fd2356b0e3b192858bd4a00bf2a8c104b054a097f51fea60fd5da0ca37d051c
bin/WebSite.dll	8	5e2ebcd54279747c2551eed061224a0c6967c0d66fb538abe1063ebb2f5cb936
bin/Websys.KM.Authority.Control.CBCCASE.dll	85	8b5933e5e68245674809f1b4b504f3df19e415e3a37a4883d918effa50a42413
bin/Websys.KM.Authority.Control.CBCCASE.dll.config	6	68b5258f6e313b1d5c62b7c6bd4f0c844bd

		7ef44541e70a90856b18fb7862d35
bin/Websys.KM.Authority.Core.CBCCASE.dll	37	747f2114344794c1b53334bac0bc7a13ffcad8ed6fca810338b4e72ba98e14be
bin/Websys.Utility.M.dll	19	67965b8a98b8992656fa2df98cc4e9b6d3d1a3594b0a522e2d1827575d9d8edf
bin/WebTreeCodeMtn.dll	28	efd67e810ef3f9b70f6aa49cc70be6b5fbc55dff3778c0d4e2435df2244172ab
bin/WebUserIPEmail.dll	14	b3613e5e82ced927d4c639d3f717a88d0913582ee4aaeb306495c993ba9f54cd
bin/WebUserLogin.dll	17	b20759329faf06f0b614828bd580793a40adf97bce4b18d234ebf15aa337fd1a
bin/XmlWebControl.dll	19	4f1e31cc7b7422d3c396dfdc05645d310b937bd250def352b0aa9aaa6c6c5d55
ChangeDocGroupName.aspx	6	861e85c7ac7a06c5c112e6e2a1f6554ac3fb38bba5ea975ba9ca1ef09cc3bc45
ChangeDocName.aspx	6	11459321c352e3bb47ac93075dc8a1375b472fd4d7b7067bd6ecd57c268167b4
ChangeDocShowName.aspx	6	6c2dd372df5d8a84e75b772b4aaabe3629c59fbaf4e592543055f0c8a7098b53
ChangeMarquee.aspx	6	007b465b36101110cf8238daddf2d9aabab5bf9411dce1e05fb2b4e8aa357d98
ChangeSystemName.aspx	6	bad3b39160cd897f806123a5673c3b2bce00f704e8eb4c382b473e09b91ad0fa
CompanyManagement.aspx	1	5a64dc34c697db15ea90d1b92b084fcbf2daf1db67577fe664fde13949a0dce8
Controls/Banner.ascx	2	074ebdf83b528966049ca798467b509d183ee549562a3edc0e0060014efa18d6
Controls/CdsysTitleBannerC.ascx	1	b77adcaf7ef6578d248af5370c973921bad7a515435460c7c57dae4705b814c9
Controls/CheckBoxControl.ascx	1	a3ee0ac372e6c8f739c7e20aefd7c2052e4210f7399b0f648d6646e291310158
Controls/CompanyList.ascx	11	08ffea0a294c410b7f6e7198499cec81208bbafd684fd016288c50f89bb88276
Controls/DepartmentList.ascx	7	a08168ca40da25da43e3d6d8807a8f4b7ea949db3eb7b58cbe36cfcba29f3bbb
Controls/Footer.ascx	1	50f74fe4cf923f99c2d18570eb8130eece0c4eceefae96c5531bf65ac3384007
Controls/MemberEdit.ascx	7	a2f5a509e601a1d6ea91411292f35285424f94c842048c71f65c49a52b4dceb0
Controls/MemberList.ascx	8	d965b8b1227da2d4398b3884e589f12bd3db58486abaabc297a91c9aa038df89
Controls/MemberMatrix.ascx	2	9972a9ecc1a517aaa69893d8484bc89b9e287ab44a002485f25eae3687f29825
Controls/MessageBox.ascx	1	d77c6d2aca9bff86fd87adfdf3a07159e6cba2768bcc8c20150bfb83f52ac99c
Controls/OUPickerButton.ascx	2	22cf4669ddc6aa63bfe6678b3789b23a2eae6388b09a66b9bc63e04b9355a97f
Controls/RoleAuthority.ascx	1	a1a3e8a6f6211cda5ee1d7e08abd76b2899bcc2404435e3cf0bccd3ef0a4d991
Controls/RoleEdit.ascx	6	391f2a6ce9c078b942931e6303af79e2a67a

		69359811591093b993cf24933a7e
Controls/RoleList.ascx	1	8bfef1024d67903e73ff0611f9d1f2e1caf7e20dcf268245d6b2abc39bba842c
Controls/UserAuthority.ascx	1	ff7ac8736f6dad013b57be9074ce006afbe67044b7a526f979baf3a3e668050a
CrossFrameScript.html	1	1009e7453b899b0a5fae0d398a073400cd59c3966500ff5ae007dd3e860b718b
DepartmentManagement.aspx	1	2f3a49afe41d1c9b05cf1ce637e4ac0ae589b03c95f54fcee1717b5f685028d8
jquery/js/esapi.js	1	3571eb18784ff7bd0600972163ddecdfeeb244e25c1608f6e9310de1ef2ab37c
jquery/js/jquery.contextMenu.js	80	ac541971c90bcd897c4299f8c002a92a231bbbf8e98dbbd3499edd697a295d2a
jquery/js/jquery-3.6.0.min.js	144	7ba35f08effa051a83bf35c443b57be603bd1b098a2d3e6680fc0e788f3d1627
jquery/js/yearpicker.js	14	bba84cb0526e1118693c60256d59d54ae612958e5b9a965b6c22b98e475f77ad
jquery-ui/1.13.2/jquery-ui.min.js	368	1fe9de1998df7c2810ed3423c6e4b87a7beabbd4a174b15b783009d4bdcc7a29
jquery-ui/js/clipboard.min.js	21	92a5b25b29fd7edcad5922b0a31e79dadadb69322fc2926cb8d7e650dd462a2e
jquery-ui/js/datepicker-zh-TW.js	2	0fa75d5422c174ab70b67d6465e0f043073a06f4847c35e7448095b0c826cf6e
jquery-ui/js/jquery.msgbox.min.js	10	f96d895b72936f1382ca48b1b717af18259db76d979a70dba813c6f2f4f07240
jquery-ui/js/jquery.redirect.js	7	6d69ae5c4892d35573385da52afebec92fb02feaf7670b0684c1b2aa6f2cfb98
jscolor/jscolor.js	53	f3ca21de7a02dcadadfb2e9221496913f859ce5bdb88dfeb31d4cb3a0c1e37b6
KM_js/AntiClickJack.js	1	20687a246aee6025de17a181dc324c117ccd28647c737a1a512618cbc1b2b700
KM_js/cdsJavaFunction.js	17	cd9362654e9f764dd86ab668c89e89224aa86af2eb287eb67eb15a1f60ac5534
KM_js/DateTime.js	14	f2ef5594ff18ca10ede080976f644621047c849dced185f513fbf6effe533e99
KM_js/disableRightClick.js	1	2c621174c35d530a4654b8f4528ba2a5c95b15d377e1f9f33e14375f0716b4cf
KM_js/Hashtable.js	4	9884d25c9e0f8456c377b961fe4f577299135ea9fa23618a6d8a3866729d7a0a
KM_js/logout.js	4	99b5d57ebac01b215b483adeda09d943df8743a24dcdccd852364847caa6cfff
KM_js/PopUpClose.js	7	469f781fd978997c49eeb22ea5c6439341939b0882cd338fd1b6d6cfb860a602
KM_js/TreeFunction.js	11	781de9f5a03012e43e0c0e32ef91f6811d22d75a41231aa8d19a7bb9ccc174a1
KM_js/WebAutoBatchUpd.js	18	f7630a6d1f3c9601394eacd6a04b03cd3edd0069e6d02875023cbdfdc7aafdad
KM_js/WebAutoClassMtn.js	20	cd00fe407b064a7674d644b658c2398b0c2f3b5bab39f39b482726459bc52714
KM_js/WebBatchPrint.js	2	206c9d4b641a931af8fbc4276db4ed4423d

		517bb5f16e387650a7c411c350d16
KM_js/WebCatalogG.js	54	3a1dfb9a6658c776ceda56a4983ed817582 27b587785c0626139d24ca2dc3ec0
KM_js/WebCompanySel.js	10	f5f6071b1e57c0f95aa8332d5bc2470365fde 6c21bdfbc055a91535a936ddf95
KM_js/WebFileUp.js	7	ae6139e1dd8ebc28827ad34f73c807bf47e9 3289bf1f550a50c69e8a8e85604e
KM_js/WebOptionSelect.js	13	485d6520e88ee6d7f2d7ffba9d0ba39244c0 de33693227d8d82903aefe18b5b2
KM_js/WebReport2019.js	5	11fd2b674738f3954f186febefeaaad141681 39bec59ad238b3d4c1361001715
KM_js/WebStdSelect.js	16	5f56c5301d39a427151fc2124b197fdc3356 6f7c94137903c18378f1b3c9f04e
KM_js/WebTreeCodeMtn.js	17	837871ae30c7d22b9ad98f5f7d621fe4e20b 5524fb0ef626ac5abe8e22f83b25
KmJavaScript.cs	32	2ee5d2d722c5f4c0685e213aee79a60fc46b e42ddf7a2eaf0209c641aaad19c5
KmXmlUI.cs	14	8c8ddb519e6d75d19817057c36a68395668 cd085c6ef18348a91144cdfa1e308
LogKeyError.html	1	f26f6fb871c0dbb98ad1ece8099abed45a76 fcfe9e87136b2e9770bf52386ccb
MemberInfo.aspx	2	7fd3e8ea0d23d4f42c5e42c086cd76e4ca80 997b1d825e6c62dc72f4eb3701e9
MemberManagement.aspx	1	2be519781632f4baa078de552b6c480fd49 0e3dbedba7545e60880ed7069c85b
MemberPermission.aspx	1	6a4fd3bd6ca388b3dcf26fed4b34dd2d871a 04942383132464fdc19b58452911
MenuControl_A.Html	6	6b4441fd0a6072b88b3cf5b4037c88343a8f 8c4dcca1f5fc0d7d96006d6ffe1d
NLog.config	2	01390d83641cad74f0237831ca912688316 ec7a76314c5cda63bc448c3a93ba3
obj/Debug/KM_CBC.csproj.FileListAbsolute.txt	9	299bb331f5a10e27578788de77cd60aab3e bc6a15487426fd4679e9a87c8efd0
obj/Debug/TCCKM.dll	348	114e7fda695e29300bea15ed7d637835948 fa89cf8cb4e642d88c04eed283a37
obj/Debug/TemporaryGeneratedFile_036C0B5B-1481-4323-8D20-8F5ADCB23D92.cs	1	e3b0c44298fc1c149afbf4c8996fb92427ae4 1e4649b934ca495991b7852b855
obj/Debug/TemporaryGeneratedFile_5937a670-0e60-4077-877b-f7221da3dda1.cs	1	e3b0c44298fc1c149afbf4c8996fb92427ae4 1e4649b934ca495991b7852b855
obj/Debug/TemporaryGeneratedFile_E7A71F73-0F8D-4B9B-B56E-8E70B10BC5D3.cs	1	e3b0c44298fc1c149afbf4c8996fb92427ae4 1e4649b934ca495991b7852b855
OUPickerButtonTest.aspx	2	bc3d78e029fec28441d452fd66e020ad6964 7d46d294885f53b9f9beb887a3f4
PageSetting.cs	138	4108703ebd5410bb6a13bd0302a09ca5792 4050296705fab5c64363bd702524e
Properties/AssemblyInfo.cs	2	b9ff67fc5f47562a4c1db4069e254c0c10ce2 1c5a2e426bd06d2e0f0c723f460
RoleManagement.aspx	1	4db1c1ae57373eefd0a40888367af84d91b3 5bb86378f2156c3a73b7a3010e33
Scripts/jquery-3.6.0.min.js	144	7ba35f08effa051a83bf35c443b57be603bd

		1b098a2d3e6680fc0e788f3d1627
ShowDocument.aspx	3	61f3e176ef044bddf60039e6442b5caf23d8d7563bea097b73e85125f33f9722
ShowDocument.aspx.cs	9	12cd76496039d14d038264d34e6167ec1fe7893edc30f701aca49f8e8bd19487
ShowDocument.aspx.designer.cs	6	8fcdc58b609c624276ad673af2a62674ae1fe11cc418d98b8adf280ed42143a6
ShowLogDocument.aspx	2	88a2d3265aa4a3b10750ed669c7f85ce428faf5132b29d3a35d0e9f723698fe9
ShowLogDocument.aspx.cs	3	9eb25616d156d166aa767a4392f7ff60a2feef6fbee919836de572ab8c324dfc
ShowLogDocument.aspx.designer.cs	2	a80306b2d76122acafb4eca0d23714d91419828a2c8cc825dac97d51bc728855
Web.config	9	7f19e0d58ae444acebcebc0c4381d29a4dd3d4f5988cd7f48c0ff59e458494e5
WebAttachLink.aspx	2	9cf1df7d7eb9e7ba7c5ada15b99e38457558e936c6a7a28c429d9da254e0992a
WebAttachLink.aspx.cs	7	2788f8354b0a7be603f1857a31449110104e00bd4c4a13b9d35921b938e17ca9
WebAttachLink.aspx.designer.cs	3	40d1c883cd4b751a0afba0c5b578ab954bbca1faf7231e9d9d62740d50fd990d
WebAutoBatchUpd.aspx	21	897ec6a6f09700ac928b245d238360e89e9a8043b75a169294bd8f563817f667
WebAutoCategoryBuild.aspx	1	f9b00598b3e00a4a4bba3074b43e3f7f681719f1960fe46ede71c59cef289796
WebAutoCategoryRun.aspx	1	6416dd4364b675cdee904fac78673e405387ff1eefcd20532b22386d42776915
WebAutoClassMtn.aspx	19	5d10e95df9f69de3cd5d17381f9aa4c578e9788e0537ae72ce2e113af4a1930d
WebBatchPrint.aspx	8	311faecb6825e1daa580e1c0947c5ef76119c5c1f2671b229b89be2cb0d5c5a7
WebBatchPrint.aspx.cs	50	60695ed700b2c391fa7bdac4fadd379862253d6053801327cc3e3a296ff9abd4
WebBatchPrint.aspx.designer.cs	9	2e183f7d2987a0cf0172ba4658a7109863c874535fd844133846b6f5e8ab4fea
WebBulletin.aspx	2	499078b1529356dd33cf884025ab13ccf476a15ab1815e02b9656986f1d0d433
WebBulletin.aspx.cs	4	b6789f2c758d7df7ca573590f4b6c40aeadfbb81f88b2ff6a1a2f46e66c1b63c
WebBulletin.aspx.designer.cs	2	d013a8c3b553cd2674faeab3d989470336e60da0dcda0faeb765176760a38902
WebCatalog.aspx	71	cbb88ddf9e2a61a3aa1d303525874d4019ca1713e78a893daf429ccb362138b5
WebCatalog.aspx.cs	387	4fbce4f137e669bf0757adc004ed6fc2b70bcb744b15cabed2dace886c893f44
WebCatalog.aspx.designer.cs	63	3f77d1833a14cea73e30bb9789e10e97469985f4c31d44251067a84b7f9f6efa
WebCategoryCompany.aspx	1	df6b8c8c0df0493a8c5562f7de7b644094202d90edc0c8c724056f8054b0de55
WebCategoryDocument.aspx	1	ae0818828021ac7db4bd1ca3baa2746b09c

		7387941c0ecd9d04f777f498c4945
WebCategoryReport.aspx	8	2f98b0cef5f9cd342fae980dd822192359ea780315a19ad34cc027a70f34629a
WebChangeDocName.aspx	7	4b813ac8667673be4abf7373ff63b5ed9feafcad35fa493f9b72a1718d20c7d3
WebCmpLogin.aspx	15	36cb858cd90dfedda34941c555abf4afd0ac26849bdd34c65d26656233245e96
WebCompanySel.aspx	13	87c037630068993646ef12ee107346520e5ca3f45823ec15cfcb65236deb9773
WebCreateStdDoc.aspx	7	86cb889d6e0059d024526d7858d367fa9f35a09150d52b54f3bc7b04f7490971
WebDataSelect.aspx	6	e06f0ca20108983306f89247468fec70be59ff4cdb1ddb3fdca5ea26984c733a
WebDeleteFiles.ashx.cs	2	6da8a8466ae4730b461384f827e5ec2392a01c195d6e9ebba3062a653abbb1af
WebDeptQueryCount.aspx	10	019d0d88a9cf8a2c4d21ff02c66979307687919280a93ea8cf10f2a45986e564
WebDocCount.aspx	31	083f60dc8ce22a7bcb5bc515334f261b4d7f99a46ffbbbd5316619fefc304b23
WebDocCountCompare.aspx	15	5a7d0874afbd7e31ea6bbf7aaa48f84cca237786d835419589ecbcf608bbb05b
WebDocPrint.aspx	1	9d25dac3cc1a82e0e5db8f1ed18e00be17e27b9620d894ed09429c3a84c3383a
WebDocPrint.aspx.cs	2	052bed87b8d4f1f2d9bd5380ef55053cc6f47da08051ff05884a6cceced8f3f2
WebDocPrint.aspx.designer.cs	1	47b4f6d2e728dadc60c5261375fbea9b0f136940fce8206ec63dc5e0d6378193
WebDocPrintA.aspx	1	ab40ac1118bf06111c64d0ed36438d03c597f4d59b3b8c7a751297ac5c1484f3
WebDocQuery.aspx	18	0322cabe12133b0329d7651dd5243f0076165b8705e86bb560252615b41b022b
WebDocRecoverMtn.aspx	19	48cd6cfff51093b24e15e9cbbfd1209efdd7a3f010c99691c3cff1a865ef9f86
WebDocReview.aspx	16	c9f17c069727afa4124a75aa16b0c00d337cbe269d7f5882ba7c5f9a47093d48
WebDocument.aspx	3	ddc060b802b45e6ad25876008954e7198c44943fc5f09c02b91144347a1dc3cc
WebDocument.aspx.cs	7	f8fb18c802587d6419e1dc20c006bfa011371265d279fb956d26c939a9869527
WebDocument.aspx.designer.cs	2	8f657a24c2dbcf32eaad49ed3ee08d2a52efa9a5318df10959fcdc46d9702a05
WebDocumentLog.aspx	5	fe7c6003adc13ff6b5040f1f041761b929af1cec90b10093be10ffe68dfa5128
WebDocumentLog.aspx.cs	6	f75c85021e76fb3415176457121b95454aef406a16d377b2ff861d164e4428fb
WebDocumentLog.aspx.designer.cs	2	7c4df58b97f771ff974a1e9da84bf1531cb59ac70f57e2e0dd0f782c663b5efb
WebDownloadFiles.aspx	2	3c398e3e5e827821e338314dbdb4f3ca3d6c54441fa1f449a0e36464563a6453
WebDownloadFiles.aspx.cs	15	72d0e441840c20e893ab0f891a9023e5c10

		48cfe1fdef727a2bdea9bc32f96c5
WebDownloadFiles.aspx.designer.cs	2	001e7391324726f4fc4e4bc98f2e07d96297e62ce1f4ecca0a5e3ecb0029a7f5
WebEditor.aspx	10	d61fec587057774fe23ee4691a81e0540418c710781dcee2829a26ae8f2f388a
WebEditor.aspx.cs	123	f01b4fead636666c03e4e71f352c9145adee9cfaa9434ff3d907bf08be13b483
WebEditor.aspx.designer.cs	10	a9f412228eb1f3c2dc83f864b2dd68d6d80ae22117dbbb1c46b77bbb3c381062
WebError.aspx	2	341627f8292f593f4dbd6be02fd4d432f045b21b5619b3984fb8c85df73866fe
WebError.aspx.cs	1	78484d9e63763d5a041c1244c7d769e99e9e05a959b8def23812e8fd3f8fde33
WebError.aspx.designer.cs	2	078db8c9d63f85595d7907a6160bcf5a0dad1c5b17c3d462e089fce58bfdb6f3
WebExportXML.aspx	1	042fff26d656a5dec10d42a3a954e3c70dbbf9f7c3b57d6dd691c462299527eb
WebExportXML.aspx.cs	3	4f59ad0e2470d5ec4f1857c73531b2d8fecf096c42da784fd4df84001b68bc62
WebExportXML.aspx.designer.cs	1	d0e0f36b75e5e5e5e39317ad20b052b39cc2115d07cdcfa07d28637349c355ba
WebFileUp.aspx	7	61f39399b00a5898507ff0c7a88482a0870474afe854c1665151a996a1c89660
WebGetGroupNo.ashx.cs	2	4024df4f40256fe7730cb22ad3548eabfb5e02926cb8191149200f446f53627a
WebGoAnyPage.ashx.cs	26	7299f443a473f09586e36653cd4581e22809e3bc0c72162179ff4715ba7e6bed
WebGoAnyPageLoad.ashx.cs	29	3c64d5ac366db91a919682a7edddb338a0573d1c7cde3292471423f2f1064757
WebListSelect.aspx	6	f0a64603c965f0d9cb08c5edbef1e167b67dd5b51a7ec29e77cff429b1fe7542
WebLogEvent.aspx	20	e5e8374ffeb4f174c615ca2c320354c72de610d344f8a684c2f69fa03189419c
WebLogon.aspx	15	436605f78f5a47fb591dae54ccabf067785342e3bc099742d89dc9204d65e476
WebLogon.aspx.cs	15	39c55f1eca0d49a9d2540952f7196500116f680d8a068d70bf6adb0ec87a345f
WebLogon.aspx.designer.cs	8	d1b2b578702afbef5e0d6621fe20ac5e5dac98022686b150c0a8da8f3c25f758
WebLogonBroker.Common.aspx	1	525319050897711d37360ddcee790804c505813df8f5b480878fa39dda88d50a
WebLogout.aspx	2	f57887683ea65cf856e0165d35bd72a2c2b9b64b4a46fd42e20129b4ade8dbdb
WebLogout.aspx.cs	4	c87529208f355234b1d1c18c16d88cd8751552bcb6c63139c9aef2fe81523c88
WebLogout.aspx.designer.cs	2	69386e7b6280e41a2015b76a4d630cd8dd39d4381c09d94c5abeabdf40b02607
WebLogQuery.aspx	14	ce4a87bc84fb4b57a8f577a51883df74ccf9e31e1b030ecad4b0b39248f664f6
WebMailA.aspx	1	7f552749056aa004cc32fb16716dcd98aca1

		ba2edd025766e46400ef72d21100
WebMailA.aspx.cs	10	8ea6243ef06028aa4b0c598fbd10176c3ac09cb4a9d7590422e09ab3b96ecb18
WebMailA.aspx.designer.cs	2	713ab2c51ee69934753d80e0e75635f7a0185d227a6e24c5f2900bb3f219ad91
WebMailTo.aspx	4	1ce32a770607c5daedad185d4775c877e3e3711ebcddc309ef5e610cdc7cb625
WebMailTo.aspx.cs	5	78bf511724ce87250adecac47660ed9bdbc3c114bd7d7c5724cbca28682710c4
WebMailTo.aspx.designer.cs	3	dcb557c55044b8ff177e3f12eff90eb8c4b9865e8b344c5f4950d0c7939fe437
WebMark.aspx	7	05676d67f69f73fd95fb785380b2db13157e07fc07cdb7db8974552955ec39d3
WebMark.aspx.cs	10	e935e861e5d84b91ac881313f99aff04efafc485cdd85209e51cd4b330640639
WebMark.aspx.designer.cs	5	f5bd5b783a3e418e0f26db337117fe22395574bde9ce58f071a83ec735ef1adc
WebMIS.aspx	1	15d07aa21d1405692546a45a9cc5797498b47bbee177746c649e705eebf88e29
WebMIS.aspx.cs	4	8891dd0e0a95ab6152cfd903e5821a802771f74f2b24df74df00e0661cbe3e9a
WebMIS.aspx.designer.cs	2	0cc2b464ef1e2a1bfbcdb35651da1154d7f2d7971a06080e51b360bc938d1c05
WebModify2022_1.aspx	4	f6c7e7320507bd290529ac57b69e6ee81ce93c21bb65f75acccd3573b34738ce
WebNotePad.aspx	4	f4f8a0c3b513b6f39948be29960b4d23c8b2cc357f04da7f1abef6e2bd6d1216
WebNotePad.aspx.cs	32	ee56000e68e2432a26fc3c4ef99f312db454af7fd4b95aff99f7dae40c55901d
WebNotePad.aspx.designer.cs	3	fe065848e86d0b59b11c631b7b03e68bc44ea42b93c0c0647a3f1e749ec4e3ca
WebOptionMultiNew.aspx	6	90619838cf355f9b8a1b76b13d8cb1d2cda1e570518eb87d3668eae52084f240
WebOptionSelect.aspx	6	130cb0dd810b6ea7449e19abb4444655b67072f560383a985d3e3ee15fd7503b
WebPrint.aspx	11	51ed8c11322d83feb9a99d037a4b395ae21dfa790bbd4e613383534a7def2baa
WebPrint.aspx.cs	38	df68998763a8665713964aed2c2a8f64d8f9aaba7a82caf011d9d6ec2cbf1fc7
WebPrint.aspx.designer.cs	13	330b3220fc8d3e136c78288812c0b0591864938ad6fb3294670ffca6be75d111
WebPrintAllData.aspx	1	65cd17c3f5e0a4dc21c9ea5e2fc41210ba3fa50280c12902a8be463430214d9e
WebPrintAllData.aspx.cs	24	e097f3997ec9de4e337f65d1c6c76e89c00307198b3ab1f910722f2919951ae0
WebPrintAllData.aspx.designer.cs	2	c943b030a54d5ab2775230d85054dde8c1b9acdbf142a04b6d1a280bc2377d39
WebRedirect.aspx	2	a371082f60ceda26220c1b973fded953d1f911a58cbc51a753ade47309d5d8ef
WebRedirect.aspx.cs	12	7871e4b50991b1c50738d46cbfc805fc567b

		6647dbbcd079e9100ac327fa889c
WebRedirect.aspx.designer.cs	2	5637c5d58fdd4212a7a3b6afaaf1409ffd050cd25e2d338eabfe12ca241f1a4e
WebReport.aspx	4	de89df4a657b317a546721c6be06016d671da04bc9e2aab4f39da0fda2a8db1f
WebReport2019_1.aspx	14	7aa512b30325415bba5f0304432355457ba53ce88019a033908128152f0ca300
WebReport2019_10.aspx	18	899dcaa27a9249b66661ac5a949b5b103b70ba24b10f5b0fa3acc1d9a94a8058
WebReport2019_11.aspx	17	29c34c9b60469f84f9b3f42a4607ca938089c6b0877555afdaf6843dc561c764
WebReport2019_12.aspx	17	3b0759489b6cf9d4cc0e106508c07b6c12ac8766315fc92596f085b87a40d1f0
WebReport2019_13.aspx	18	1b6f59dc8fcf3fb1452ee81175f52839b5929fccdd52f3be932680f99941c3f4
WebReport2019_2.aspx	17	04434afafb9369c95e9c02fc414884efbf518a0bbfb06abea59394f3a80a602c
WebReport2019_3.aspx	15	bc0a87dd5a248647055af1bd6625a1cf05190e605592f4681c249028240c6c82
WebReport2019_4.aspx	18	b49bce4b130dca614f1e1d6ffca14bef0110c131573880eaa33d6a5ac96a2e45
WebReport2019_5.aspx	16	933c454945a5024450c9fea73fc9edf4c723f53483579bb4100bfe5feb49d1d2
WebReport2019_6.aspx	18	111db909ac1d56e55942bfd116c976f33e2b64e35516ee497b5bd3aebd89a25e
WebReport2019_7.aspx	18	d1831f1201563cf3c07762f41243aaea72b6cf2d4635389f04f00abe11b582f1
WebReport2019_8.aspx	16	5771b1ea1d7269a36d6b93e1753c71a4b44ca054471c1ae1279ad7be89e62583
WebReport2019_9.aspx	18	d6ccf6f992d10c03c76da2d172a3f778b8929e9d0f77a730fc1ae6748474b682
WebReport2022_1.aspx	16	0d499d44b7693fe5caf007fa95cc5d2c579a23490f4929ff4e0e9ee62850bcc2
WebSaveParm1.ashx.cs	4	3eb3315f18caadde16e6e0da8be41c1fba13ba691e6552c26a643cbe9d48a5eb
WebSite.aspx	13	b24474a26070007d7812cff31ba380ce85caa74b0f208f17526f48d504a32e75
WebStdSelect.aspx	6	10971eaeaad030cdd6ed8e80938318ea4cb3a4acee88a4afa2a678b090fd2d4d
WebTreeCodeMtn.aspx	10	921ec7692c7ddf6e98d900b77f938935f68db1a8954e0cd681f459e5c651f1ad
WebTreeSelect.aspx	9	9d0e1c626d96600df851990de8cec15fa16357983e6fac7fe25f2d0580e8b7e6
WebUserIPEmail.aspx	12	9a907e546795ca2b4803c312baf4fb9ec5df17923cc356745d2e6cbeb701eab3
WebUserLogin.aspx	15	a075ddd6b696f931f782e26f0a6f2fff23ab98890257eac13ce2305192afa331

已掃描的查詢

查詢名稱	找到的問題
Heuristic SQL Injection	183
Heuristic Stored XSS	97
Heuristic Parameter Tampering	88
Exposure of Resource to Wrong Sphere	73
Insufficient Logging of Sensitive Operations	49
Heuristic CSRF	41
Heuristic DB Parameter Tampering	38
Pages Without Global Error Handler	23
Client Potential DOM Open Redirect	18
Client DOM Open Redirect	15
Improper Exception Handling	12
Information Exposure Through an Error Message	12
Client JQuery Deprecated Symbols	8
Hardcoded Absolute Path	6
Password in Configuration File	6
Client Side Only Validation	3
DebugEnabled	2
Information Exposure via Headers	2
Client Regex Injection	1
Missing Content Security Policy	1
Absolute Path Traversal	0
Angular Client DOM XSS	0
Angular Client Stored DOM XSS	0
Angular Deprecated API	0
Angular Usage of Unsafe DOM Sanitizer	0
AngularJS SCE Disabled	0
Blind SQL Injections	0
Buffer Overflow	0
CGI XSS	0
Cleartext Storage Of Sensitive Information	0
Client Cookies Inspection	0
Client Cross Frame Scripting Attack	0
Client Cross Session Contamination	0
Client CSS Injection	0
Client DB Parameter Tampering	0
Client DOM Code Injection	0
Client DOM Cookie Poisoning	0
Client DOM Stored Code Injection	0
Client DOM Stored XSS	0
Client DOM XSRF	0
Client DOM XSS	0
Client DoS By Sleep	0
Client Empty Password	0
Client Hardcoded Domain	0
Client Header Manipulation	0
Client Heuristic Poor XSS Validation	0
Client HTML5 Easy To Guess Database Name	0

Client HTML5 Heuristic Session Insecure Storage	0
Client HTML5 Information Exposure	0
Client HTML5 Insecure Storage	0
Client HTML5 Store Sensitive data In Web Storage	0
Client Insecure Randomness	0
Client Insufficient Key Size	0
Client Manual CSRF Token Handling	0
Client Null Password	0
Client Overly Permissive Message Posting	0
Client Password In Comment	0
Client Password Weak Encryption	0
Client Path Manipulation	0
Client Potential Code Injection	0
Client Potential ReDoS In Match	0
Client Potential ReDoS In Replace	0
Client Potential XSS	0
Client Privacy Violation	0
Client ReDoS From Regex Injection	0
Client ReDoS In Match	0
Client ReDos In RegExp	0
Client ReDoS In Replace	0
Client Reflected File Download	0
Client Remote File Inclusion	0
Client Resource Injection	0
Client Sandbox Allows Scripts With Same Origin	0
Client Second Order Sql Injection	0
Client Server Empty Password	0
Client SQL Injection	0
Client Untrusted Activex	0
Client Use Of Deprecated SQL Database	0
Client Use Of Iframe Without Sandbox	0
Client Use Of JQuery Deprecated Version	0
Client Weak Cryptographic Hash	0
Client Weak Encryption	0
Client Weak Password Authentication	0
Client XPATH Injection	0
Clipboard Information Leakage	0
Code Injection	0
Command Argument Injection	0
Command Injection	0
Connection String Injection	0
Cookie Injection	0
Cookie Poisoning	0
CookieLess Authentication	0
CookieLess Session State	0
Cookies Inspection	0
Cordova Code Injection	0
Cordova File Disclosure	0

Cordova File Manipulation	0
Cordova Insufficient Domain Whitelist	0
Cordova Missing Content Security Policy	0
Cordova Open Redirect	0
Cordova Permissive Content Security Policy	0
Cordova Privacy Violation	0
CSRF	0
CSV Injection	0
CustomError	0
Dangerous File Upload	0
Data Filter Injection	0
DB Parameter Tampering	0
Deserialization of Untrusted Data	0
Deserialization of Untrusted Data MSMQ	0
Directory Browse	0
DOM Code Injection	0
DOM Cookie Poisoning	0
DOM CSRF	0
DOM Open Redirect	0
DOM XSS	0
DoS by Sleep	0
Dynamic File Inclusion	0
Dynamic SQL Queries	0
Elmah Enabled	0
Excessive Data Exposure	0
Frameable Login Page	0
Hardcoded Connection String	0
Hardcoded password in Connection String	0
HardcodedCredentials	0
Heap Inspection	0
Heuristic 2nd Order SQL Injection	0
HTTP Response Splitting	0
HttpOnlyCookies	0
HttpOnlyCookies In Config	0
Impersonation Issue	0
Improper Encoding Of Output	0
Improper Locking	0
Improper Restriction of XXE Ref	0
Improper Session Management	0
Inappropriate Encoding for Output Context	0
Information Exposure Through Directory Listing	0
Information Exposure Through Log Files	0
Information Exposure Through Query Strings	0
Information Leak Through Persistent Cookies	0
Insecure Cookie	0
Insufficient Connection String Encryption	0
Insufficient Logging of Database Actions	0
Insufficient Logging of Exceptions	0

Insufficient Transport Layer Security	0
Insufficiently Protected Credentials	0
JavaScript Hijacking	0
Jelly Injection	0
Jelly XSS	0
JSON Hijacking	0
JWT Excessive Expiration Time	0
JWT Lack Of Expiration Time	0
JWT No Expiration Time Validation	0
JWT No NotBefore Validation	0
JWT No Signature Verification	0
JWT Sensitive Information Exposure	0
JWT Use Of Hardcoded Secret	0
JWT Use Of None Algorithm	0
Kony Code Injection	0
Kony Hardcoded EncryptionKey	0
Kony Information Leakage	0
Kony Path Injection	0
Kony Reflected XSS	0
Kony Second Order SQL Injection	0
Kony SQL Injection	0
Kony Stored Code Injection	0
Kony Stored XSS	0
Kony Unsecure Browser Configuration	0
Kony Unsecure iOSBrowser Configuration	0
Kony URL Injection	0
Kony Use WeakEncryption	0
Kony Use WeakHash	0
LDAP Injection	0
Leaving Temporary Files	0
Lightning DOM XSS	0
Lightning Stored XSS	0
Log Forging	0
Missing Column Encryption	0
Missing CSP Header	0
Missing Encryption of Sensitive Data	0
Missing Function Level Authorization	0
Missing HSTS Header	0
Missing Object Level Authorization	0
Missing X Frame Options	0
MongoDB NoSQL Injection	0
MVC View Injection	0
No Request Validation	0
NonUniqueFormName	0
Not Using a Random IV with CBC Mode	0
Open Redirect	0
Overly Permissive Cross Origin Resource Sharing Policy	0
Parameter Tampering	0

Password In Comment	0
Password Weak Encryption	0
Path Traversal	0
Permissive Content Security Policy	0
Persistent Connection String	0
Plaintext Storage of a Password	0
Poor Database Access Control	0
Potential ReDoS	0
Potential ReDoS By Injection	0
Potential ReDoS In Code	0
Potential ReDoS In Static Field	0
Potentially Vulnerable To Csrf	0
Privacy Violation	0
Race Condition within a Thread	0
React Deprecated	0
ReDoS By Regex Injection	0
ReDoS In Code	0
ReDOS in RegExp	0
ReDoS In Validation	0
Reflected XSS	0
Reflected XSS All Clients	0
Reflected XSS Specific Clients	0
Relative Path Traversal	0
RequireSSL	0
Resource Injection	0
SAPUI5 Deprecated Symbols	0
SAPUI5 Hardcoded UserId In Comments	0
SAPUI5 Potential Malicious File Upload	0
SAPUI5 Use Of Hardcoded URL	0
Second Order SQL Injection	0
Security Misconfiguration	0
Session Clearing Problems	0
Session Fixation	0
Session Poisoning	0
SlidingExpiration	0
SQL Injection	0
SQL Injection Evasion Attack	0
SSL Verification Bypass	0
Stored Code Injection	0
Stored Command Argument Injection	0
Stored Command Injection	0
Stored LDAP Injection	0
Stored Path Traversal	0
Stored XPath Injection	0
Stored XSS	0
TraceEnabled	0
Trust Boundary Violation in Session Variables	0
Unchecked Input For Loop Condition	0

Uncontrolled Format String	0
Unencrypted Sensitive Data Storage	0
Unencrypted Web Config File	0
Unprotected Cookie	0
Unrestricted File Upload	0
Unsafe Object Binding	0
Unsafe Reflection	0
Use of Broken or Risky Cryptographic Algorithm	0
Use Of Broken Or Risky Cryptographic Algorithm	0
Use Of Controlled Input On Sensitive Field	0
Use of Cryptographically Weak PRNG	0
Use of Deprecated or Obsolete Functions	0
Use of Hard coded Cryptographic Key	0
Use Of Hardcoded Password	0
Use Of HTTP Sensitive Data Exposure	0
Use of Insufficiently Random Values	0
Use of RSA Algorithm without OAEP	0
UTF7 XSS	0
VF Remoting Client Potential Code Injection	0
VF Remoting Client Potential XSRF	0
VF Remoting Client Potential XSS	0
Visible Pointers	0
Vue DOM XSS	0
Weak Password Authentication	0
XML External Entities XXE	0
XPath Injection	0
XS Code Injection	0
XS CSRF	0
XS Log Injection	0
XS Open Redirect	0
XS Parameter Tampering	0
XS Reflected XSS	0
XS Response Splitting	0
XS Second Order SQL Injection	0
XS SQL Injection	0
XS Stored Code Injection	0
XS Stored XSS	0
XS Unencrypted Data Transfer	0
XS Use Of Hardcoded URL	0
XSS Evasion Attack	0

掃描結果詳細資料

HttpOnlyCookies

查詢路徑:

CSharp\公司\CSharp Medium Threat\HttpOnlyCookies 版本:2

描述

HttpOnlyCookies\路徑 1:

嚴重程度：	中風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=1
狀態	反覆出現的問題
Detection Date	11/30/2022 4:05:54 PM

Web應用程式的 btnWord_Click 方法在 WebBatchPrint.aspx.cs 的 629 行中新增一個 cookie Write， 並在 response 中 return 這個 cookie。 但是， 應用程式並非配置為 : 會自動為Cookie 套用"httpOnly"屬性， 而且程式碼中也沒有明確的將"httpOnly"屬性加到 Cookie 中。

	來源	目的地
檔案	WebBatchPrint.aspx.cs	WebBatchPrint.aspx.cs
行	638	638
物件	Write	Write

代碼片斷

檔案名稱 WebBatchPrint.aspx.cs

方法 protected void btnWord_Click(object sender, System.EventArgs e)

```
....
638.         Response.Write("<html><head><meta charset=\"utf-8\" http-
equiv=\"Content-Type\" content=\"text/html\">");
```

Heuristic SQL Injection

查詢路徑:

CSharp\Cx\CSharp Heuristic\Heuristic SQL Injection 版本:1

類別

PCI DSS v3.2.1: PCI DSS (3.2.1) - 6.5.1 - Injection flaws - particularly SQL injection
OWASP Top 10 2013: A1-Injection
FISMA 2014: System And Information Integrity
NIST SP 800-53: SI-10 Information Input Validation (P1)
OWASP Top 10 2017: A1-Injection
ASD STIG 4.10: APSC-DV-002540 - CAT I The application must not be vulnerable to SQL Injection.
OWASP Top 10 2021: A3-Injection

描述

Heuristic SQL Injection\路徑 1:

嚴重程度：	低風險
結果狀態：	校驗

線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=153
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過ShowDocument.aspx.cs中的31之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 ShowDocument.aspx.cs 的 Page_Load 第 31 的使用者輸入 QueryString_DocID 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	ShowDocument.aspx.cs	ShowDocument.aspx.cs
行	46	116
物件	QueryString_DocID	Query

代碼片斷	
檔案名稱	ShowDocument.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
46.           strA =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g["DocID"], false));
47.           _sDocID = (strA == null) ? "" : strA.Trim();
....
116.           objReader1 = objDB1.Query("SELECT DocDefID, DocTitle,
DocXML FROM DocCatalog WITH(NOLOCK) WHERE DocID = '" + _sDocID + "'");
```

Heuristic SQL Injection\路徑 2:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=154
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過ShowLogDocument.aspx.cs中的21之Format執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 ShowLogDocument.aspx.cs 的 Page_Load 第 21 的使用者輸入 QueryString_ID 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	ShowLogDocument.aspx.cs	ShowLogDocument.aspx.cs
行	25	27

物件	QueryString_ID	Format

代碼片斷

檔案名稱 ShowLogDocument.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
25.         string strID =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g["ID"], false));
....
27.         string strSQL = String.Format("SELECT EvtNote, EvtDocID FROM
LogEvent WHERE EvtID ={0}", strID);
```

Heuristic SQL Injection\路徑 3:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=155
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebAttachLink.aspx.cs中的23之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebAttachLink.aspx.cs 的 GetRequest 第 96 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebAttachLink.aspx.cs	WebAttachLink.aspx.cs
行	100	67
物件	Form	Query

代碼片斷

檔案名稱 WebAttachLink.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
100.          strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
105.        return strResult.Trim();
```

檔案名稱 WebAttachLink.aspx.cs

方法 protected void Page_Load(object sender, EventArgs e)

```
....
28.         varDocID = GetRequest("DocID", "");
....
30.         varLogKey = GetRequest("LogKey", "\u0001");
....
67.           objDR0 = objDB0.Query("SELECT DocXML FROM DocCatalog
WITH(NOLOCK) WHERE DocID = '" + varDocID + "';");
```

Heuristic SQL Injection\路徑 4:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=156
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebAttachLink.aspx.cs中的23之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebAttachLink.aspx.cs 的 GetRequest 第 96 的使用者輸入 QueryString_strName 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebAttachLink.aspx.cs	WebAttachLink.aspx.cs
行	103	67
物件	QueryString_strName	Query

代碼片斷

檔案名稱 WebAttachLink.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
103.             strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
105.         return strResult.Trim();
```

檔案名稱 WebAttachLink.aspx.cs

方法 protected void Page_Load(object sender, EventArgs e)

```
....
28.         varDocID = GetRequest("DocID", "");
....
30.         varLogKey = GetRequest("LogKey", "\u0001");
....
67.           objDR0 = objDB0.Query("SELECT DocXML FROM DocCatalog
WITH(NOLOCK) WHERE DocID = '" + varDocID + "';");
```

Heuristic SQL Injection\路徑 5:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=157
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_LockedCompany透過WebCatalog.aspx.cs中的297之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	301
物件	Form	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_FolderMtn(int Action)

```
....
646.        string rtnPublicArea = this.Sub_GetRequest("rtnPublicArea",
"");
....
804.              DOC.SetFolderAuthority(strFolderID,
strPublicArea.Split(','), rtnPublicArea.Split(','));
805.              this.Sub_Debug("SetFolderAuthority", DOC.DebugSQL);
....
849.            this.Sub_Debug("strFolderID", strFolderID);
850.            this.Sub_Debug("strFolderName", strFolderName);
851.            this.Sub_Debug("strCompany", strCompany + ", " +
strCompanyCode);
852.            this.Sub_Debug("strDepartment", strDepartment + ", " +
strDepartmentCode);
853.            this.Sub_Debug("strPublish", strPublish);
854.            this.Sub_Debug("strPublicArea", strPublicArea);
855.            this.Sub_Debug("strORG", strORG);
856.            this.Sub_Debug("rtnPublicArea", rtnPublicArea);
....
874.            strTMP = this.Sub_Organization("ORG", 2, "?", "", false,
"選擇單位", string.Format("{0}/{1}", strCompanyCode, strDepartmentCode),
string.Format("{0}/{1}", strCompany, strDepartment), false,
this.Sub_LockedCompany() + strChangeFlag, out strOnClick);
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_LockedCompany()

```
....
301.        DR = this.DOC.Query(strSQL);
```

Heuristic SQL Injection\路徑 6:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=158
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_Category透過WebCatalog.aspx.cs中的358之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	396
物件	Form	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
396.               System.Data.SqlClient.SqlDataReader objDR1 =
DOC.Query(string.Format(strSQL1, strCategoryID));
```

Heuristic SQL Injection\路徑 7:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=159
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_FolderMtn透過WebCatalog.aspx.cs中的632之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	679
物件	Form	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_FolderMtn(int Action)

```
....
638.         string strFolderID = this.Sub_GetRequest("txtFolderID", "");
....
679.           objDR1 = DOC.Query("SELECT ID FROM DocFolder WITH(NOLOCK)
WHERE DocFolderID = '" + strFolderID + "';");
```

Heuristic SQL Injection\路徑 8:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=160
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_FolderMtn透過WebCatalog.aspx.cs中的632之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	903
物件	Form	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱	WebCatalog.aspx.cs
方法	private void Sub_FolderMtn(int Action)

```
....
638.         string strFolderID = this.Sub_GetRequest("txtFolderID", "");
....
892.             objDR1 = DOC.GetFolderAuthority(strFolderID);
893.             while(objDR1.Read())
....
895.               strCode = objDR1.IsDBNull(1) ? "" :
objDR1.GetString(1).Trim();
....
902.                   string[] aryTemp1 = strCode.Split('/'); strName =
"";
903.                   objDR2 = objDBA.Query("SELECT CompanyName FROM
AppCompany WITH(NOLOCK) WHERE CompanyNo = '" + aryTemp1[0] + "'");
```

Heuristic SQL Injection\路徑 9:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=161
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_FolderMtn透過WebCatalog.aspx.cs中的632之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	911
物件	Form	Query

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱	WebCatalog.aspx.cs

方法	private void Sub_FolderMtn(int Action)

```
....
638.        string strFolderID = this.Sub_GetRequest("txtFolderID", "");
....
892.            objDR1 = DOC.GetFolderAuthority(strFolderID);
893.            while(objDR1.Read())
....
895.              strCode = objDR1.IsDBNull(1) ? "" :
objDR1.GetString(1).Trim();
....
902.                  string[] aryTemp1 = strCode.Split('/'); strName =
"";
....
911.                    objDR2 = objDBA.Query("SELECT DepartmentName
FROM AppDepartment WITH(NOLOCK) WHERE CompanyNo = '" + aryTemp1[0] + "'
AND DepartmentNo = '" + aryTemp1[1] + "'");
```

Heuristic SQL Injection\路徑 10:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=162
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_FolderMtn透過WebCatalog.aspx.cs中的632之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	920
物件	Form	Query

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.        return strReq;
```

檔案名稱	WebCatalog.aspx.cs

方法	private void Sub_FolderMtn(int Action)

```
....
638.        string strFolderID = this.Sub_GetRequest("txtFolderID", "");
....
892.            objDR1 = DOC.GetFolderAuthority(strFolderID);
893.            while(objDR1.Read())
....
895.              strCode = objDR1.IsDBNull(1) ? "" :
objDR1.GetString(1).Trim();
....
902.                  string[] aryTemp1 = strCode.Split('/'); strName =
"";
....
920.                    objDR2 = objDBA.Query("SELECT Name FROM AppUser
WITH(NOLOCK) WHERE UID = '" + aryTemp1[2] + "'");
```

Heuristic SQL Injection\路徑 11:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=163
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_Organization透過WebCatalog.aspx.cs中的1007之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	1012
物件	Form	Query

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.        return strReq;
```

檔案名稱	WebCatalog.aspx.cs

方法	private void Sub_FolderMtn(int Action)

```
....
646.        string rtnPublicArea = this.Sub_GetRequest("rtnPublicArea",
"");
....
804.              DOC.SetFolderAuthority(strFolderID,
strPublicArea.Split(','), rtnPublicArea.Split(','));
805.              this.Sub_Debug("SetFolderAuthority", DOC.DebugSQL);
....
849.            this.Sub_Debug("strFolderID", strFolderID);
850.            this.Sub_Debug("strFolderName", strFolderName);
851.            this.Sub_Debug("strCompany", strCompany + ", " +
strCompanyCode);
852.            this.Sub_Debug("strDepartment", strDepartment + ", " +
strDepartmentCode);
853.            this.Sub_Debug("strPublish", strPublish);
854.            this.Sub_Debug("strPublicArea", strPublicArea);
855.            this.Sub_Debug("strORG", strORG);
856.            this.Sub_Debug("rtnPublicArea", rtnPublicArea);
....
874.            strTMP = this.Sub_Organization("ORG", 2, "?", "", false,
"選擇單位", string.Format("{0}/{1}", strCompanyCode, strDepartmentCode),
string.Format("{0}/{1}", strCompany, strDepartment), false,
this.Sub_LockedCompany() + strChangeFlag, out strOnClick);
```

檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_Organization(string Key, int OrgType, string Attribute, string ScriptName, bool UsingTextArea, string Title, string Codes, string Names, bool AddInput, string LockORG, out string strOnClick)

```
....
1012.        System.Data.SqlClient.SqlDataReader objDR1 =
DOC.Query(strSQL);
```

Heuristic SQL Injection\路徑 12:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=164
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs

行	1837	2654
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.         public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2654.            if(strValue == DR.GetValue(iStart).ToString())
```

Heuristic SQL Injection\路徑 13:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=165
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetFieldType執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2689
物件	Form	GetFieldType

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
```

Heuristic SQL Injection\路徑 14:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=166
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetFieldType執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2692
物件	Form	GetFieldType

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2692.                  if(DR.GetFieldType(i).ToString() ==
"System.DateTime")
```

Heuristic SQL Injection\路徑 15:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=167
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2697
物件	Form	GetValue

代碼片斷
檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.         public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2692.                  if(DR.GetFieldType(i).ToString() ==
"System.DateTime")
....
2697.
if(DR.GetValue(DataIndex[Index][0]).ToString() != "")
```

Heuristic SQL Injection\路徑 16:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=168
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的GetDataIndexData透過PageSetting.cs中的2599之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2605
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs
方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2668.                if(DR.GetValue(DataIndex[Index][0]).ToString() !=
"")
2669.                  strItem = string.Format(fmtHref,
string.Format(Href[Index], this.GetDataIndexData(DR, Index, DataIndex)),
DR.GetValue(iStart));
```

檔案名稱 PageSetting.cs

方法 private string[] GetDataIndexData(System.Data.SqlClient.SqlDataReader DR, int Index, int[][] DataIndex)

```
....
2599.         private string[]
GetDataIndexData(System.Data.SqlClient.SqlDataReader DR, int Index,
int[][] DataIndex)
....
2605.             aryDataIndexData[i] =
DR.GetValue(DataIndex[Index][i]).ToString();
```

Heuristic SQL Injection\路徑 17:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=169
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2700
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2692.                  if(DR.GetFieldType(i).ToString() ==
"System.DateTime")
....
2697.
if(DR.GetValue(DataIndex[Index][0]).ToString() != "")
....
2700.                        strItem = DR.GetValue(iStart).ToString();
```

Heuristic SQL Injection\路徑 18:

嚴重程度： 低風險
結果狀態： 校驗

線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=170
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之IsDBNull執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2704
物件	Form	IsDBNull

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2692.                  if(DR.GetFieldType(i).ToString() ==
"System.DateTime")
....
2704.                      if(DR.IsDBNull(i))
```

Heuristic SQL Injection\路徑 19:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=171
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2718
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.         public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2692.                  if(DR.GetFieldType(i).ToString() ==
"System.DateTime")
....
2718.
if(DR.GetValue(DataIndex[Index][0]).ToString() != "")
```

Heuristic SQL Injection\路徑 20:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=172
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2721
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs
方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2692.                  if(DR.GetFieldType(i).ToString() ==
"System.DateTime")
....
2718.
if(DR.GetValue(DataIndex[Index][0]).ToString() != "")
....
2721.                        strItem = DR.GetValue(i).ToString();
```

Heuristic SQL Injection\路徑 21:

嚴重程度： 低風險

結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=173
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2734
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法	public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.         public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱	PageSetting.cs
方法	private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.         private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.           while(DR.Read())
....
2651.             varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                 if(DR.GetFieldType(i).IsValueType)
....
2734.                     if(DR.GetValue(DataIndex[Index][0]).ToString()
!= "")
```

Heuristic SQL Injection\路徑 22:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=174
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2738
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.         public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2734.                    if(DR.GetValue(DataIndex[Index][0]).ToString()
!= "")
....
2737.                      string strTemp1 = DR.GetValue(i).ToString();
2738.                      if(RedColorFlag[Index] != 0 &&
DR.GetValue(RedColorFlag[Index]).ToString() != "0")
```

Heuristic SQL Injection\路徑 23:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=175
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2737
物件	Form	GetValue

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2734.                    if(DR.GetValue(DataIndex[Index][0]).ToString()
!= "")
....
2737.                      string strTemp1 = DR.GetValue(i).ToString();
```

Heuristic SQL Injection\路徑 24:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=1

	0300&pathid=176
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2743
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2734.                    if(DR.GetValue(DataIndex[Index][0]).ToString()
!= "")
....
2743.                      strItem = DR.GetValue(i).ToString();
```

Heuristic SQL Injection\路徑 25:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=177
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2746
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.         public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2746.                    strItem = DR.GetValue(i).ToString();
```

Heuristic SQL Injection\路徑 26:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=178
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2668
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs
方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs
方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2668.                if(DR.GetValue(DataIndex[Index][0]).ToString() !=
"")
```

Heuristic SQL Injection\路徑 27:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=179
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2678
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs
方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs
方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs
方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2674.                strItem = DR.GetValue(iStart).ToString();
....
2678.              strValue = DR.GetValue(iStart).ToString();
```

Heuristic SQL Injection\路徑 28:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=180
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2671
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.         public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2668.                if(DR.GetValue(DataIndex[Index][0]).ToString() !=
"")
....
2671.                  strItem = DR.GetValue(iStart).ToString();
```

Heuristic SQL Injection\路徑 29:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=181
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2674
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2674.                strItem = DR.GetValue(iStart).ToString();
```

Heuristic SQL Injection\路徑 30:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=182
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2651
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
```

Heuristic SQL Injection\路徑 31:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=183
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_UpdateNewsLetter透過WebCatalog.aspx.cs中的1863之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	1933
物件	Form	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_NewsLetter(int Action)

```
....
2158.         string strEMailB = this.Sub_GetRequest("txtEMailB", "");
....
2209.         strB = this.Sub_UpdateNewsLetter(strMode, "最新文件快訊",
strEMailB, "AppKMNewOrder", ref strOrderDoc) + "<br />";
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateNewsLetter(string Mode, string Name, string Mail, string Table, ref string strOrderDoc)

```
....
1863.      private string Sub_UpdateNewsLetter(string Mode, string Name,
string Mail, string Table, ref string strOrderDoc)
....
1928.              strSQL = string.Format(fmtSQL4a, Table, Mail);
....
1933.            DR = this.DOC.Query(strSQL);
```

Heuristic SQL Injection\路徑 32:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=184
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_UpdateChiefLetter透過WebCatalog.aspx.cs中的1955之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	2021

物件	Form	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_ChiefLetter(int Action)

```
....
2048.         string strEMailC = this.Sub_GetRequest("txtEMailC", "");
....
2080.         strA = this.Sub_UpdateChiefLetter(strMode, "首長電子報",
strEMailC, strTitleC);
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateChiefLetter(string Mode, string Name, string Mail, string Title)

```
....
1955.       private string Sub_UpdateChiefLetter(string Mode, string
Name, string Mail, string Title)
....
2009.               strWhere = "E_mail = '" + Mail + "'";
....
2019.             strSQL = fmtSQL4 + " Where " + strWhere;
....
2021.             DR = this.DOC.Query(strSQL);
```

Heuristic SQL Injection\路徑 33:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=185
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_MailList透過WebCatalog.aspx.cs中的2342之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。

根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	2360
物件	Form	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequestEmpty(string Name, string Default)

```
....
1850.        string strResult = Sub_GetRequest(Name, "");
....
1852.        return strResult;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_NewsLetter(int Action)

```
....
2253.          string strMailClass =
this.Sub_GetRequestEmpty("txtMailClass", "AppKMNewOrder");
....
2271.          DR = this.Sub_MailList(strMailClass);
```

檔案名稱 WebCatalog.aspx.cs

方法 private System.Data.SqlClient.SqlDataReader Sub_MailList(string MailClass)

```
....
2342.      private System.Data.SqlClient.SqlDataReader
Sub_MailList(string MailClass)
....
2351.            strSQL = string.Format(strSQL, MailClass);
....
2360.        return this.DOC.Query(strSQL);
```

Heuristic SQL Injection\路徑 34:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=186
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_PersonalSetting透過WebCatalog.aspx.cs中的3904之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	4260
物件	Form	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 public void Sub_PersonalSetting(int Action)

```
....
4007.             strEmail = this.Sub_GetRequest("txtEmail", "").Trim();
....
4016.               objDR1 = this.DOC.Query(string.Format("SELECT * FROM
AppUser WITH(NOLOCK) WHERE Email = '{0}' AND UID <> '{1}'",
strEmail.Replace("'", "''"), WhoAmI.UserID));
....
4027.                 WhoAmI.eMail = strEmail;
....
4260.         objDR1 = this.DOC.Query(string.Format(strSQL3,
WhoAmI.eMail, WhoAmI.UserID));
```

Heuristic SQL Injection\路徑 35:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=1

	0300&pathid=187
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_PersonalHideIframe透過WebCatalog.aspx.cs中的4306之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	4310
物件	Form	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs
方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs
方法 public void Sub_PersonalSetting(int Action)

```
....
3941.             Ary[11] = PSET.HideIframe =
Sub_PersonalHideIframe("NoUse", this.Sub_GetRequest("OrderIframes",
""));
```

檔案名稱 WebCatalog.aspx.cs
方法 private string Sub_PersonalHideIframe(string strIframeKey, string strIframe)

```
....
4306.      private string Sub_PersonalHideIframe(string strIframeKey,
string strIframe)
....
4310.         System.Data.SqlClient.SqlDataReader objDR1 =
DOC.Query(string.Format(strSQL, strIframeKey, strIframe.Replace(",",
"','")));
```

Heuristic SQL Injection\路徑 36:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=188
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_UpdateChiefLetter透過WebCatalog.aspx.cs中的1955之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	1972
物件	Form	Execute

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_ChiefLetter(int Action)

```
....
2048.         string strEMailC = this.Sub_GetRequest("txtEMailC", "");
....
2080.         strA = this.Sub_UpdateChiefLetter(strMode, "首長電子報",
strEMailC, strTitleC);
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateChiefLetter(string Mode, string Name, string Mail, string Title)

```
....
1955.      private string Sub_UpdateChiefLetter(string Mode, string
Name, string Mail, string Title)
....
1971.            strSQL = string.Format(fmtSQL2, Title, Mail,
WhoAmI.UserID, _sUporg);
1972.            if(this.DOC.Execute(strSQL) == 0)
```

Heuristic SQL Injection\路徑 37:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=189
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_PersonalSetting透過WebCatalog.aspx.cs中的3904之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	3948
物件	Form	Execute

代碼片斷

檔案名稱 WebCatalog.aspx.cs
方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs
方法 private string Sub_GetRequestEmpty(string Name, string Default)

```
....
1850.        string strResult = Sub_GetRequest(Name, "");
....
1852.        return strResult;
```

檔案名稱	WebCatalog.aspx.cs
方法	public void Sub_PersonalSetting(int Action)

```
....
3944.            Ary[14] = PSET.RightArea =
this.Sub_GetRequestEmpty("checkRightArea", "");
....
3946.            strSQL1 = string.Format(strSQL1, Ary);
....
3948.            this.DOC.Execute(strSQL1, strSQL2);
```

Heuristic SQL Injection\路徑 38:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=190
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_PersonalSetting透過WebCatalog.aspx.cs中的3904之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	3993
物件	Form	Execute

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱	WebCatalog.aspx.cs
方法	public void Sub_PersonalSetting(int Action)

```
....
3976.            strWordA = this.Sub_GetRequest("txtPasswordA",
"").Trim();
....
3987.              string strWord = objEncrypt.Encrypt(strWordA);
....
3989.              objSB1.Append("UPDATE AppUserPassword SET
Member_password = '" + strWord + "' WHERE Uid = '" + WhoAmI.UserID +
"';");
3990.              objSB1.Append("UPDATE AppUser WITH(READPAST) SET
LoginPWD = '" + strWordA.Replace("'", "''") + "' WHERE UID ='" +
WhoAmI.UserID + "';");
....
3993.                this.DOC.Execute(objSB1.ToString());
```

Heuristic SQL Injection\路徑 39:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=191
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_PersonalSetting透過WebCatalog.aspx.cs中的3904之Insert執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	4232
物件	Form	Insert

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 public void Sub_PersonalSetting(int Action)

```
....
3976.            strWordA = this.Sub_GetRequest("txtPasswordA",
"").Trim();
....
3987.              string strWord = objEncrypt.Encrypt(strWordA);
....
3989.              objSB1.Append("UPDATE AppUserPassword SET
Member_password = '" + strWord + "' WHERE Uid = '" + WhoAmI.UserID +
"';");
3990.              objSB1.Append("UPDATE AppUser WITH(READPAST) SET
LoginPWD = '" + strWordA.Replace("'", "''") + "' WHERE UID ='" +
WhoAmI.UserID + "';");
....
3993.                this.DOC.Execute(objSB1.ToString());
....
4232.        objSB1.Insert(0, "<SELECT NAME='listDocItems90'
ID='listDocItems90'
onkeypress=\"javascript:switch(window.event.keyCode){case
43:MoveDnOptionItemA('listDocItems90');break;case
45:MoveUpOptionItemA('listDocItems90');break;}try { event.preventDefault
? event.preventDefault() : (event.returnValue = false); } catch (err) {
return false; }\" SIZE=" + k.ToString() + ">");
```

Heuristic SQL Injection\路徑 40:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=192
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_PersonalSetting透過WebCatalog.aspx.cs中的3904之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	4026
物件	Form	Execute

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 public void Sub_PersonalSetting(int Action)

```
....
4007.             strEmail = this.Sub_GetRequest("txtEmail", "").Trim();
....
4016.               objDR1 = this.DOC.Query(string.Format("SELECT * FROM
AppUser WITH(NOLOCK) WHERE Email = '{0}' AND UID <> '{1}'",
strEmail.Replace("'", "''"), WhoAmI.UserID));
....
4026.                 this.DOC.Execute(string.Format("UPDATE AppUser SET
Email = '{0}' WHERE UID = '{1}'", strEmail, WhoAmI.UserID));
```

Heuristic SQL Injection\路徑 41:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=193
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_UpdateChiefLetter透過WebCatalog.aspx.cs中的1955之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	1976
物件	Form	Execute

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_ChiefLetter(int Action)

```
....
2048.        string strEMailC = this.Sub_GetRequest("txtEMailC", "");
....
2080.        strA = this.Sub_UpdateChiefLetter(strMode, "首長電子報",
strEMailC, strTitleC);
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateChiefLetter(string Mode, string Name, string Mail, string Title)

```
....
1955.      private string Sub_UpdateChiefLetter(string Mode, string
Name, string Mail, string Title)
....
1975.              strSQL = string.Format(fmtSQL1, Title, Mail,
WhoAmI.UserID, _sUporg);
1976.              if(this.DOC.Execute(strSQL) == 0)
```

Heuristic SQL Injection\路徑 42:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=194
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_UpdateNewsLetter透過WebCatalog.aspx.cs中的1863之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	1891
物件	Form	Execute

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_NewsLetter(int Action)

```
....
2158.         string strEMailB = this.Sub_GetRequest("txtEMailB", "");
....
2209.         strB = this.Sub_UpdateNewsLetter(strMode, "最新文件快訊",
strEMailB, "AppKMNewOrder", ref strOrderDoc) + "<br />";
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateNewsLetter(string Mode, string Name, string Mail, string Table, ref string strOrderDoc)

```
....
1863.       private string Sub_UpdateNewsLetter(string Mode, string Name,
string Mail, string Table, ref string strOrderDoc)
....
1889.                 strSQL = string.Format(fmtSQL3, Table, Mail);
....
1891.                 if(this.DOC.Execute(strSQL) == 0) strMSG =
"目前不在寄送名單中。"; else strMSG = "不會再寄送。";
```

Heuristic SQL Injection\路徑 43:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=195
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的SendAlert透過WebCatalog.aspx.cs中的2882之RegisterStartupScript執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	2886
物件	Form	RegisterStartupScript

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_FolderMtn(int Action)

```
....
639.         string strFolderName = this.Sub_GetRequest("txtFolder",
"").Replace(",", ", ");
....
692.               this.SendAlert(string.Format("開放區域：{0},\n成功刪除！",
strFolderName), "");
```

檔案名稱 WebCatalog.aspx.cs

方法 private void SendAlert(string strMsg, string AppCMD)

```
....
2882.      private void SendAlert(string strMsg, string AppCMD)
....
2884.         string strJava = "\n<script>window.alert(\"" +
strMsg.Replace("\"", "\\\"").TrimEnd('\n').Replace("\r",
"").Replace("\n", "\\n") + "\");" + AppCMD + "</script>\n";
....
2886.         this.ClientScript.RegisterStartupScript(this.GetType(),
"cds_Alert", strJava);
```

Heuristic SQL Injection\路徑 44:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=196
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_UpdateNewsLetter透過WebCatalog.aspx.cs中的1863之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	1922
物件	Form	Execute

代碼片斷

檔案名稱 WebCatalog.aspx.cs
方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs
方法 private void Sub_NewsLetter(int Action)

```
....
2158.         string strEMailB = this.Sub_GetRequest("txtEMailB", "");
....
2209.         strB = this.Sub_UpdateNewsLetter(strMode, "最新文件快訊",
strEMailB, "AppKMNewOrder", ref strOrderDoc) + "<br />";
```

檔案名稱 WebCatalog.aspx.cs
方法 private string Sub_UpdateNewsLetter(string Mode, string Name, string Mail, string Table, ref string strOrderDoc)

```
....
1863.      private string Sub_UpdateNewsLetter(string Mode, string Name,
string Mail, string Table, ref string strOrderDoc)
....
1920.             strSQL = string.Format(fmtSQL3, Table, Mail);
....
1922.             if(this.DOC.Execute(strSQL) == 0) strMSG =
"目前不在寄送名單中。"; else strMSG = "不會再寄送。";
```

Heuristic SQL Injection\路徑 45:

嚴重程度： 低風險

結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=197
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_UpdateChiefLetter透過WebCatalog.aspx.cs中的1955之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	1993
物件	Form	Execute

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_ChiefLetter(int Action)

```
....
2048.         string strEMailC = this.Sub_GetRequest("txtEMailC", "");
....
2080.         strA = this.Sub_UpdateChiefLetter(strMode, "首長電子報",
strEMailC, strTitleC);
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateChiefLetter(string Mode, string Name, string Mail, string Title)

```
....
1955.      private string Sub_UpdateChiefLetter(string Mode, string
Name, string Mail, string Title)
....
1991.            strSQL = string.Format(fmtSQL3, Mail);
....
1993.            if(this.DOC.Execute(strSQL) == 0)
```

Heuristic SQL Injection\路徑 46:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=198
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_LockedCompany透過WebCatalog.aspx.cs中的297之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	301
物件	QueryString_Name	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs
方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs
方法 private void Sub_FolderMtn(int Action)

```
....
646.        string rtnPublicArea = this.Sub_GetRequest("rtnPublicArea",
"");
....
804.              DOC.SetFolderAuthority(strFolderID,
strPublicArea.Split(','), rtnPublicArea.Split(','));
805.              this.Sub_Debug("SetFolderAuthority", DOC.DebugSQL);
....
849.            this.Sub_Debug("strFolderID", strFolderID);
850.            this.Sub_Debug("strFolderName", strFolderName);
851.            this.Sub_Debug("strCompany", strCompany + ", " +
strCompanyCode);
852.            this.Sub_Debug("strDepartment", strDepartment + ", " +
strDepartmentCode);
853.            this.Sub_Debug("strPublish", strPublish);
854.            this.Sub_Debug("strPublicArea", strPublicArea);
855.            this.Sub_Debug("strORG", strORG);
856.            this.Sub_Debug("rtnPublicArea", rtnPublicArea);
....
874.            strTMP = this.Sub_Organization("ORG", 2, "?", "", false,
"選擇單位", string.Format("{0}/{1}", strCompanyCode, strDepartmentCode),
string.Format("{0}/{1}", strCompany, strDepartment), false,
this.Sub_LockedCompany() + strChangeFlag, out strOnClick);
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_LockedCompany()

```
....
301.        DR = this.DOC.Query(strSQL);
```

Heuristic SQL Injection\路徑 47:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=199
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_Category透過WebCatalog.aspx.cs中的358之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	396
物件	QueryString_Name	Query

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_GetRequest(string Name, string Default)

```
....
1840.             strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.         return strReq;
```

檔案名稱	WebCatalog.aspx.cs
方法	private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
396.              System.Data.SqlClient.SqlDataReader objDR1 =
DOC.Query(string.Format(strSQL1, strCategoryID));
```

Heuristic SQL Injection\路徑 48:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=200
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_FolderMtn透過WebCatalog.aspx.cs中的632之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	679
物件	QueryString_Name	Query

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_FolderMtn(int Action)

```
....
638.        string strFolderID = this.Sub_GetRequest("txtFolderID", "");
....
679.          objDR1 = DOC.Query("SELECT ID FROM DocFolder WITH(NOLOCK)
WHERE DocFolderID = '" + strFolderID + "';");
```

Heuristic SQL Injection\路徑 49:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=201
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_FolderMtn透過WebCatalog.aspx.cs中的632之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	903
物件	QueryString_Name	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱	WebCatalog.aspx.cs
方法	private void Sub_FolderMtn(int Action)

```
....
638.         string strFolderID = this.Sub_GetRequest("txtFolderID", "");
....
892.             objDR1 = DOC.GetFolderAuthority(strFolderID);
893.             while(objDR1.Read())
....
895.               strCode = objDR1.IsDBNull(1) ? "" :
objDR1.GetString(1).Trim();
....
902.                   string[] aryTemp1 = strCode.Split('/'); strName =
"";
903.                   objDR2 = objDBA.Query("SELECT CompanyName FROM
AppCompany WITH(NOLOCK) WHERE CompanyNo = '" + aryTemp1[0] + "'");
```

Heuristic SQL Injection\路徑 50:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=202
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_FolderMtn透過WebCatalog.aspx.cs中的632之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	911
物件	QueryString_Name	Query

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_GetRequest(string Name, string Default)

```
....
1840.             strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.         return strReq;
```

檔案名稱	WebCatalog.aspx.cs

方法	private void Sub_FolderMtn(int Action)

```
....
638.        string strFolderID = this.Sub_GetRequest("txtFolderID", "");
....
892.            objDR1 = DOC.GetFolderAuthority(strFolderID);
893.            while(objDR1.Read())
....
895.              strCode = objDR1.IsDBNull(1) ? "" :
objDR1.GetString(1).Trim();
....
902.                  string[] aryTemp1 = strCode.Split('/'); strName =
"";
....
911.                    objDR2 = objDBA.Query("SELECT DepartmentName
FROM AppDepartment WITH(NOLOCK) WHERE CompanyNo = '" + aryTemp1[0] + "'
AND DepartmentNo = '" + aryTemp1[1] + "'");
```

Heuristic SQL Injection\路徑 51:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=203
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_FolderMtn透過WebCatalog.aspx.cs中的632之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	920
物件	QueryString_Name	Query

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱	WebCatalog.aspx.cs

方法	private void Sub_FolderMtn(int Action)

```
....
638.         string strFolderID = this.Sub_GetRequest("txtFolderID", "");
....
892.             objDR1 = DOC.GetFolderAuthority(strFolderID);
893.             while(objDR1.Read())
....
895.               strCode = objDR1.IsDBNull(1) ? "" :
objDR1.GetString(1).Trim();
....
902.                   string[] aryTemp1 = strCode.Split('/'); strName =
"";
....
920.                     objDR2 = objDBA.Query("SELECT Name FROM AppUser
WITH(NOLOCK) WHERE UID = '" + aryTemp1[2] + "'");
```

Heuristic SQL Injection\路徑 52:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=204
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_Organization透過WebCatalog.aspx.cs中的1007之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	1012
物件	QueryString_Name	Query

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_GetRequest(string Name, string Default)

```
....
1840.             strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.         return strReq;
```

檔案名稱	WebCatalog.aspx.cs

方法	private void Sub_FolderMtn(int Action)

```
....
646.        string rtnPublicArea = this.Sub_GetRequest("rtnPublicArea",
"");
....
804.              DOC.SetFolderAuthority(strFolderID,
strPublicArea.Split(','), rtnPublicArea.Split(','));
805.              this.Sub_Debug("SetFolderAuthority", DOC.DebugSQL);
....
849.            this.Sub_Debug("strFolderID", strFolderID);
850.            this.Sub_Debug("strFolderName", strFolderName);
851.            this.Sub_Debug("strCompany", strCompany + ", " +
strCompanyCode);
852.            this.Sub_Debug("strDepartment", strDepartment + ", " +
strDepartmentCode);
853.            this.Sub_Debug("strPublish", strPublish);
854.            this.Sub_Debug("strPublicArea", strPublicArea);
855.            this.Sub_Debug("strORG", strORG);
856.            this.Sub_Debug("rtnPublicArea", rtnPublicArea);
....
874.            strTMP = this.Sub_Organization("ORG", 2, "?", "", false,
"選擇單位", string.Format("{0}/{1}", strCompanyCode, strDepartmentCode),
string.Format("{0}/{1}", strCompany, strDepartment), false,
this.Sub_LockedCompany() + strChangeFlag, out strOnClick);
```

檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_Organization(string Key, int OrgType, string Attribute, string ScriptName, bool UsingTextArea, string Title, string Codes, string Names, bool AddInput, string LockORG, out string strOnClick)

```
....
1012.        System.Data.SqlClient.SqlDataReader objDR1 =
DOC.Query(strSQL);
```

Heuristic SQL Injection\路徑 53:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=205
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs

行	1840	2654
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2654.            if(strValue == DR.GetValue(iStart).ToString())
```

Heuristic SQL Injection\路徑 54:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=206
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetFieldType執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2689
物件	QueryString_Name	GetFieldType

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
```

Heuristic SQL Injection\路徑 55:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=207
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetFieldType執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2692
物件	QueryString_Name	GetFieldType

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.             strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2692.                  if(DR.GetFieldType(i).ToString() ==
"System.DateTime")
```

Heuristic SQL Injection\路徑 56:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=208
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2697
物件	QueryString_Name	GetValue

代碼片斷
檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2692.                  if(DR.GetFieldType(i).ToString() ==
"System.DateTime")
....
2697.
if(DR.GetValue(DataIndex[Index][0]).ToString() != "")
```

Heuristic SQL Injection\路徑 57:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=209
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的GetDataIndexData透過PageSetting.cs中的2599之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2605
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs
方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2668.                if(DR.GetValue(DataIndex[Index][0]).ToString() !=
"")
2669.                  strItem = string.Format(fmtHref,
string.Format(Href[Index], this.GetDataIndexData(DR, Index, DataIndex)),
DR.GetValue(iStart));
```

檔案名稱 PageSetting.cs

方法 private string[] GetDataIndexData(System.Data.SqlClient.SqlDataReader DR, int Index, int[][] DataIndex)

```
....
2599.        private string[]
GetDataIndexData(System.Data.SqlClient.SqlDataReader DR, int Index,
int[][] DataIndex)
....
2605.            aryDataIndexData[i] =
DR.GetValue(DataIndex[Index][i]).ToString();
```

Heuristic SQL Injection\路徑 58:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=210
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2700
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2692.                  if(DR.GetFieldType(i).ToString() ==
"System.DateTime")
....
2697.
if(DR.GetValue(DataIndex[Index][0]).ToString() != "")
....
2700.                        strItem = DR.GetValue(iStart).ToString();
```

Heuristic SQL Injection\路徑 59:

嚴重程度： 低風險
結果狀態： 校驗

線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=211
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之IsDBNull執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2704
物件	QueryString_Name	IsDBNull

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.             strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2692.                  if(DR.GetFieldType(i).ToString() ==
"System.DateTime")
....
2704.                      if(DR.IsDBNull(i))
```

Heuristic SQL Injection\路徑 60:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=212
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2718
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2692.                  if(DR.GetFieldType(i).ToString() ==
"System.DateTime")
....
2718.
if(DR.GetValue(DataIndex[Index][0]).ToString() != "")
```

Heuristic SQL Injection\路徑 61:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=213
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2721
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs
方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2692.                  if(DR.GetFieldType(i).ToString() ==
"System.DateTime")
....
2718.
if(DR.GetValue(DataIndex[Index][0]).ToString() != "")
....
2721.                        strItem = DR.GetValue(i).ToString();
```

Heuristic SQL Injection\路徑 62:

嚴重程度： 低風險

結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=214
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2734
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.             strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法	public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.         public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱	PageSetting.cs
方法	private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.         private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.           while(DR.Read())
....
2651.             varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                 if(DR.GetFieldType(i).IsValueType)
....
2734.                     if(DR.GetValue(DataIndex[Index][0]).ToString()
!= "")
```

Heuristic SQL Injection\路徑 63:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=215
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2738
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.             strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.         public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2734.                    if(DR.GetValue(DataIndex[Index][0]).ToString()
!= "")
....
2737.                      string strTemp1 = DR.GetValue(i).ToString();
2738.                      if(RedColorFlag[Index] != 0 &&
DR.GetValue(RedColorFlag[Index]).ToString() != "0")
```

Heuristic SQL Injection\路徑 64:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=216
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2737
物件	QueryString_Name	GetValue

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2734.                    if(DR.GetValue(DataIndex[Index][0]).ToString()
!= "")
....
2737.                      string strTemp1 = DR.GetValue(i).ToString();
```

Heuristic SQL Injection\路徑 65:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=1

	0300&pathid=217
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2743
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.             strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2734.                    if(DR.GetValue(DataIndex[Index][0]).ToString()
!= "")
....
2743.                      strItem = DR.GetValue(i).ToString();
```

Heuristic SQL Injection\路徑 66:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=218
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2746
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2746.                    strItem = DR.GetValue(i).ToString();
```

Heuristic SQL Injection\路徑 67:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=219
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2668
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs
方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs
方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2668.                if(DR.GetValue(DataIndex[Index][0]).ToString() !=
"")
```

Heuristic SQL Injection\路徑 68:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=220
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2678
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs
方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.             strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs
方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs
方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2674.                strItem = DR.GetValue(iStart).ToString();
....
2678.              strValue = DR.GetValue(iStart).ToString();
```

Heuristic SQL Injection\路徑 69:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=221
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2671
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2668.                if(DR.GetValue(DataIndex[Index][0]).ToString() !=
"")
....
2671.                  strItem = DR.GetValue(iStart).ToString();
```

Heuristic SQL Injection\路徑 70:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=222
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2674
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2674.                strItem = DR.GetValue(iStart).ToString();
```

Heuristic SQL Injection\路徑 71:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=223
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的AddItem透過PageSetting.cs中的2610之GetValue執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2651
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.             strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
```

Heuristic SQL Injection\路徑 72:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=224
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_UpdateNewsLetter透過WebCatalog.aspx.cs中的1863之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	1933
物件	QueryString_Name	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_NewsLetter(int Action)

```
....
2158.        string strEMailB = this.Sub_GetRequest("txtEMailB", "");
....
2209.        strB = this.Sub_UpdateNewsLetter(strMode, "最新文件快訊",
strEMailB, "AppKMNewOrder", ref strOrderDoc) + "<br />";
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateNewsLetter(string Mode, string Name, string Mail, string Table, ref string strOrderDoc)

```
....
1863.      private string Sub_UpdateNewsLetter(string Mode, string Name,
string Mail, string Table, ref string strOrderDoc)
....
1928.              strSQL = string.Format(fmtSQL4a, Table, Mail);
....
1933.            DR = this.DOC.Query(strSQL);
```

Heuristic SQL Injection\路徑 73:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=225
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_UpdateChiefLetter透過WebCatalog.aspx.cs中的1955之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	2021

物件	QueryString_Name	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_ChiefLetter(int Action)

```
....
2048.        string strEMailC = this.Sub_GetRequest("txtEMailC", "");
....
2080.        strA = this.Sub_UpdateChiefLetter(strMode, "首長電子報",
strEMailC, strTitleC);
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateChiefLetter(string Mode, string Name, string Mail, string Title)

```
....
1955.      private string Sub_UpdateChiefLetter(string Mode, string
Name, string Mail, string Title)
....
2009.              strWhere = "E_mail = '" + Mail + "'";
....
2019.            strSQL = fmtSQL4 + " Where " + strWhere;
....
2021.            DR = this.DOC.Query(strSQL);
```

Heuristic SQL Injection\路徑 74:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=226
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_MailList透過WebCatalog.aspx.cs中的2342之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。

根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	2360
物件	QueryString_Name	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.             strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequestEmpty(string Name, string Default)

```
....
1850.         string strResult = Sub_GetRequest(Name, "");
....
1852.         return strResult;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_NewsLetter(int Action)

```
....
2253.           string strMailClass =
this.Sub_GetRequestEmpty("txtMailClass", "AppKMNewOrder");
....
2271.           DR = this.Sub_MailList(strMailClass);
```

檔案名稱 WebCatalog.aspx.cs

方法 private System.Data.SqlClient.SqlDataReader Sub_MailList(string MailClass)

```
....
2342.       private System.Data.SqlClient.SqlDataReader
Sub_MailList(string MailClass)
....
2351.             strSQL = string.Format(strSQL, MailClass);
....
2360.         return this.DOC.Query(strSQL);
```

Heuristic SQL Injection\路徑 75:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=227
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_PersonalSetting透過WebCatalog.aspx.cs中的3904之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	4260
物件	QueryString_Name	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 public void Sub_PersonalSetting(int Action)

```
....
4007.            strEmail = this.Sub_GetRequest("txtEmail", "").Trim();
....
4016.              objDR1 = this.DOC.Query(string.Format("SELECT * FROM
AppUser WITH(NOLOCK) WHERE Email = '{0}' AND UID <> '{1}'",
strEmail.Replace("'", "''"), WhoAmI.UserID));
....
4027.                WhoAmI.eMail = strEmail;
....
4260.        objDR1 = this.DOC.Query(string.Format(strSQL3,
WhoAmI.eMail, WhoAmI.UserID));
```

Heuristic SQL Injection\路徑 76:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=1

	0300&pathid=228
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_PersonalHideIframe透過WebCatalog.aspx.cs中的4306之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	4310
物件	QueryString_Name	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs
方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.             strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs
方法 public void Sub_PersonalSetting(int Action)

```
....
3941.             Ary[11] = PSET.HideIframe =
Sub_PersonalHideIframe("NoUse", this.Sub_GetRequest("OrderIframes",
""));
```

檔案名稱 WebCatalog.aspx.cs
方法 private string Sub_PersonalHideIframe(string strIframeKey, string strIframe)

```
....
4306.      private string Sub_PersonalHideIframe(string strIframeKey,
string strIframe)
....
4310.         System.Data.SqlClient.SqlDataReader objDR1 =
DOC.Query(string.Format(strSQL, strIframeKey, strIframe.Replace(",",
"','")));
```

Heuristic SQL Injection\路徑 77:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=229
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_UpdateChiefLetter透過WebCatalog.aspx.cs中的1955之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	1972
物件	QueryString_Name	Execute

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.             strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_ChiefLetter(int Action)

```
....
2048.         string strEMailC = this.Sub_GetRequest("txtEMailC", "");
....
2080.         strA = this.Sub_UpdateChiefLetter(strMode, "首長電子報",
strEMailC, strTitleC);
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateChiefLetter(string Mode, string Name, string Mail, string Title)

```
....
1955.      private string Sub_UpdateChiefLetter(string Mode, string
Name, string Mail, string Title)
....
1971.            strSQL = string.Format(fmtSQL2, Title, Mail,
WhoAmI.UserID, _sUporg);
1972.            if(this.DOC.Execute(strSQL) == 0)
```

Heuristic SQL Injection\路徑 78:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=230
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_PersonalSetting透過WebCatalog.aspx.cs中的3904之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	3948
物件	QueryString_Name	Execute

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequestEmpty(string Name, string Default)

```
....
1850.        string strResult = Sub_GetRequest(Name, "");
....
1852.        return strResult;
```

檔案名稱	WebCatalog.aspx.cs
方法	public void Sub_PersonalSetting(int Action)

```
....
3944.            Ary[14] = PSET.RightArea =
this.Sub_GetRequestEmpty("checkRightArea", "");
....
3946.            strSQL1 = string.Format(strSQL1, Ary);
....
3948.            this.DOC.Execute(strSQL1, strSQL2);
```

Heuristic SQL Injection\路徑 79:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=231
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_PersonalSetting透過WebCatalog.aspx.cs中的3904之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	3993
物件	QueryString_Name	Execute

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱	WebCatalog.aspx.cs
方法	public void Sub_PersonalSetting(int Action)

```
....
3976.            strWordA = this.Sub_GetRequest("txtPasswordA",
"").Trim();
....
3987.              string strWord = objEncrypt.Encrypt(strWordA);
....
3989.              objSB1.Append("UPDATE AppUserPassword SET
Member_password = '" + strWord + "' WHERE Uid = '" + WhoAmI.UserID +
"';");
3990.              objSB1.Append("UPDATE AppUser WITH(READPAST) SET
LoginPWD = '" + strWordA.Replace("'", "''") + "' WHERE UID ='" +
WhoAmI.UserID + "';");
....
3993.                this.DOC.Execute(objSB1.ToString());
```

Heuristic SQL Injection\路徑 80:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=232
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_PersonalSetting透過WebCatalog.aspx.cs中的3904之Insert執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	4232
物件	QueryString_Name	Insert

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 public void Sub_PersonalSetting(int Action)

```
....
3976.            strWordA = this.Sub_GetRequest("txtPasswordA",
"").Trim();
....
3987.              string strWord = objEncrypt.Encrypt(strWordA);
....
3989.              objSB1.Append("UPDATE AppUserPassword SET
Member_password = '" + strWord + "' WHERE Uid = '" + WhoAmI.UserID +
"';");
3990.              objSB1.Append("UPDATE AppUser WITH(READPAST) SET
LoginPWD = '" + strWordA.Replace("'", "''") + "' WHERE UID ='" +
WhoAmI.UserID + "';");
....
3993.                this.DOC.Execute(objSB1.ToString());
....
4232.        objSB1.Insert(0, "<SELECT NAME='listDocItems90'
ID='listDocItems90'
onkeypress=\"javascript:switch(window.event.keyCode){case
43:MoveDnOptionItemA('listDocItems90');break;case
45:MoveUpOptionItemA('listDocItems90');break;}try { event.preventDefault
? event.preventDefault() : (event.returnValue = false); } catch (err) {
return false; }\" SIZE=" + k.ToString() + ">");
```

Heuristic SQL Injection\路徑 81:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=233
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_PersonalSetting透過WebCatalog.aspx.cs中的3904之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	4026
物件	QueryString_Name	Execute

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱	WebCatalog.aspx.cs
方法	public void Sub_PersonalSetting(int Action)

```
....
4007.            strEmail = this.Sub_GetRequest("txtEmail", "").Trim();
....
4016.              objDR1 = this.DOC.Query(string.Format("SELECT * FROM
AppUser WITH(NOLOCK) WHERE Email = '{0}' AND UID <> '{1}'",
strEmail.Replace("'", "''"), WhoAmI.UserID));
....
4026.                this.DOC.Execute(string.Format("UPDATE AppUser SET
Email = '{0}' WHERE UID = '{1}'", strEmail, WhoAmI.UserID));
```

Heuristic SQL Injection\路徑 82:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=234
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_UpdateChiefLetter透過WebCatalog.aspx.cs中的1955之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	1976
物件	QueryString_Name	Execute

代碼片斷

檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_ChiefLetter(int Action)

```
....
2048.        string strEMailC = this.Sub_GetRequest("txtEMailC", "");
....
2080.        strA = this.Sub_UpdateChiefLetter(strMode, "首長電子報",
strEMailC, strTitleC);
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateChiefLetter(string Mode, string Name, string Mail, string Title)

```
....
1955.      private string Sub_UpdateChiefLetter(string Mode, string
Name, string Mail, string Title)
....
1975.              strSQL = string.Format(fmtSQL1, Title, Mail,
WhoAmI.UserID, _sUporg);
1976.              if(this.DOC.Execute(strSQL) == 0)
```

Heuristic SQL Injection\路徑 83:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=235
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_UpdateNewsLetter透過WebCatalog.aspx.cs中的1863之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	1891
物件	QueryString_Name	Execute

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_NewsLetter(int Action)

```
....
2158.        string strEMailB = this.Sub_GetRequest("txtEMailB", "");
....
2209.        strB = this.Sub_UpdateNewsLetter(strMode, "最新文件快訊",
strEMailB, "AppKMNewOrder", ref strOrderDoc) + "<br />";
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateNewsLetter(string Mode, string Name, string Mail, string Table, ref string strOrderDoc)

```
....
1863.      private string Sub_UpdateNewsLetter(string Mode, string Name,
string Mail, string Table, ref string strOrderDoc)
....
1889.                strSQL = string.Format(fmtSQL3, Table, Mail);
....
1891.                if(this.DOC.Execute(strSQL) == 0) strMSG =
"目前不在寄送名單中。"; else strMSG = "不會再寄送。";
```

Heuristic SQL Injection\路徑 84:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=236
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的SendAlert透過WebCatalog.aspx.cs中的2882之RegisterStartupScript執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	2886
物件	QueryString_Name	RegisterStartupScript

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_FolderMtn(int Action)

```
....
639.        string strFolderName = this.Sub_GetRequest("txtFolder",
"").Replace(",", ", ");
....
692.             this.SendAlert(string.Format("開放區域：{0},\n成功刪除！",
strFolderName), "");
```

檔案名稱 WebCatalog.aspx.cs

方法 private void SendAlert(string strMsg, string AppCMD)

```
....
2882.      private void SendAlert(string strMsg, string AppCMD)
....
2884.        string strJava = "\n<script>window.alert(\"" +
strMsg.Replace("\"", "\\\"").TrimEnd('\n').Replace("\r",
"").Replace("\n", "\\n") + "\");" + AppCMD + "</script>\n";
....
2886.        this.ClientScript.RegisterStartupScript(this.GetType(),
"cds_Alert", strJava);
```

Heuristic SQL Injection\路徑 85:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=237
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_UpdateNewsLetter透過WebCatalog.aspx.cs中的1863之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	1922
物件	QueryString_Name	Execute

代碼片斷

檔案名稱 WebCatalog.aspx.cs
方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs
方法 private void Sub_NewsLetter(int Action)

```
....
2158.        string strEMailB = this.Sub_GetRequest("txtEMailB", "");
....
2209.        strB = this.Sub_UpdateNewsLetter(strMode, "最新文件快訊",
strEMailB, "AppKMNewOrder", ref strOrderDoc) + "<br />";
```

檔案名稱 WebCatalog.aspx.cs
方法 private string Sub_UpdateNewsLetter(string Mode, string Name, string Mail, string Table, ref string strOrderDoc)

```
....
1863.      private string Sub_UpdateNewsLetter(string Mode, string Name,
string Mail, string Table, ref string strOrderDoc)
....
1920.            strSQL = string.Format(fmtSQL3, Table, Mail);
....
1922.            if(this.DOC.Execute(strSQL) == 0) strMSG =
"目前不在寄送名單中。"; else strMSG = "不會再寄送。";
```

Heuristic SQL Injection\路徑 86:

嚴重程度： 低風險

結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=238
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_UpdateChiefLetter透過WebCatalog.aspx.cs中的1955之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_GetRequest 第 1829 的使用者輸入 QueryString_Name 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	1993
物件	QueryString_Name	Execute

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.             strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_ChiefLetter(int Action)

```
....
2048.         string strEMailC = this.Sub_GetRequest("txtEMailC", "");
....
2080.         strA = this.Sub_UpdateChiefLetter(strMode, "首長電子報",
strEMailC, strTitleC);
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateChiefLetter(string Mode, string Name, string Mail, string Title)

```
....
1955.      private string Sub_UpdateChiefLetter(string Mode, string
Name, string Mail, string Title)
....
1991.            strSQL = string.Format(fmtSQL3, Mail);
....
1993.            if(this.DOC.Execute(strSQL) == 0)
```

Heuristic SQL Injection\路徑 87:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=239
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_LockedCompany透過WebCatalog.aspx.cs中的297之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_FolderMtn 第 632 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	675	301
物件	Form	Query

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private void Sub_FolderMtn(int Action)

```
....
675.            strFolderName =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form["strFolder
Name"], false));
....
789.                    strFolderID = DOC.CreateFolder(strFolderName,
strCompanyCode, strDepartmentCode, WhoAmI.UserID, (strPublish == "True"
? true : false));
....
804.              DOC.SetFolderAuthority(strFolderID,
strPublicArea.Split(','), rtnPublicArea.Split(','));
805.              this.Sub_Debug("SetFolderAuthority", DOC.DebugSQL);
....
849.            this.Sub_Debug("strFolderID", strFolderID);
850.            this.Sub_Debug("strFolderName", strFolderName);
851.            this.Sub_Debug("strCompany", strCompany + ", " +
strCompanyCode);
852.            this.Sub_Debug("strDepartment", strDepartment + ", " +
strDepartmentCode);
853.            this.Sub_Debug("strPublish", strPublish);
854.            this.Sub_Debug("strPublicArea", strPublicArea);
855.            this.Sub_Debug("strORG", strORG);
856.            this.Sub_Debug("rtnPublicArea", rtnPublicArea);
....
874.            strTMP = this.Sub_Organization("ORG", 2, "?", "", false,
"選擇單位", string.Format("{0}/{1}", strCompanyCode, strDepartmentCode),
string.Format("{0}/{1}", strCompany, strDepartment), false,
this.Sub_LockedCompany() + strChangeFlag, out strOnClick);
```

檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_LockedCompany()

```
....
301.        DR = this.DOC.Query(strSQL);
```

Heuristic SQL Injection\路徑 88:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=240
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_FolderMtn透過WebCatalog.aspx.cs中的632之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_FolderMtn 第 632 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs

行	675	903
物件	Form	Query

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private void Sub_FolderMtn(int Action)

```
....
675.            strFolderName =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form["strFolder
Name"], false));
....
789.                    strFolderID = DOC.CreateFolder(strFolderName,
strCompanyCode, strDepartmentCode, WhoAmI.UserID, (strPublish == "True"
? true : false));
....
892.            objDR1 = DOC.GetFolderAuthority(strFolderID);
893.            while(objDR1.Read())
....
895.              strCode = objDR1.IsDBNull(1) ? "" :
objDR1.GetString(1).Trim();
....
902.                    string[] aryTemp1 = strCode.Split('/'); strName =
"";
903.                    objDR2 = objDBA.Query("SELECT CompanyName FROM
AppCompany WITH(NOLOCK) WHERE CompanyNo = '" + aryTemp1[0] + "'");
```

Heuristic SQL Injection\路徑 89:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=241
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_FolderMtn透過WebCatalog.aspx.cs中的632之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_FolderMtn 第 632 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	675	911
物件	Form	Query

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private void Sub_FolderMtn(int Action)

```
....
675.            strFolderName =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form["strFolder
Name"], false));
....
789.                    strFolderID = DOC.CreateFolder(strFolderName,
strCompanyCode, strDepartmentCode, WhoAmI.UserID, (strPublish == "True"
? true : false));
....
892.            objDR1 = DOC.GetFolderAuthority(strFolderID);
893.            while(objDR1.Read())
....
895.              strCode = objDR1.IsDBNull(1) ? "" :
objDR1.GetString(1).Trim();
....
902.                    string[] aryTemp1 = strCode.Split('/'); strName =
"";
....
911.                      objDR2 = objDBA.Query("SELECT DepartmentName
FROM AppDepartment WITH(NOLOCK) WHERE CompanyNo = '" + aryTemp1[0] + "'
AND DepartmentNo = '" + aryTemp1[1] + "'");
```

Heuristic SQL Injection\路徑 90:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=242
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_FolderMtn透過WebCatalog.aspx.cs中的632之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_FolderMtn 第 632 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	675	920
物件	Form	Query

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private void Sub_FolderMtn(int Action)

```
....
675.            strFolderName =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form["strFolder
Name"], false));
....
789.                    strFolderID = DOC.CreateFolder(strFolderName,
strCompanyCode, strDepartmentCode, WhoAmI.UserID, (strPublish == "True"
? true : false));
....
892.            objDR1 = DOC.GetFolderAuthority(strFolderID);
893.            while(objDR1.Read())
....
895.              strCode = objDR1.IsDBNull(1) ? "" :
objDR1.GetString(1).Trim();
....
902.                    string[] aryTemp1 = strCode.Split('/'); strName =
"";
....
920.                     objDR2 = objDBA.Query("SELECT Name FROM AppUser
WITH(NOLOCK) WHERE UID = '" + aryTemp1[2] + "'");
```

Heuristic SQL Injection\路徑 91:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=243
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_Organization透過WebCatalog.aspx.cs中的1007之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_FolderMtn 第 632 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	675	1012
物件	Form	Query

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private void Sub_FolderMtn(int Action)

```
....
675.            strFolderName =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form["strFolder
Name"], false));
....
789.                    strFolderID = DOC.CreateFolder(strFolderName,
strCompanyCode, strDepartmentCode, WhoAmI.UserID, (strPublish == "True"
? true : false));
....
804.              DOC.SetFolderAuthority(strFolderID,
strPublicArea.Split(','), rtnPublicArea.Split(','));
805.              this.Sub_Debug("SetFolderAuthority", DOC.DebugSQL);
....
849.            this.Sub_Debug("strFolderID", strFolderID);
850.            this.Sub_Debug("strFolderName", strFolderName);
851.            this.Sub_Debug("strCompany", strCompany + ", " +
strCompanyCode);
852.            this.Sub_Debug("strDepartment", strDepartment + ", " +
strDepartmentCode);
853.            this.Sub_Debug("strPublish", strPublish);
854.            this.Sub_Debug("strPublicArea", strPublicArea);
855.            this.Sub_Debug("strORG", strORG);
856.            this.Sub_Debug("rtnPublicArea", rtnPublicArea);
....
874.            strTMP = this.Sub_Organization("ORG", 2, "?", "", false,
"選擇單位", string.Format("{0}/{1}", strCompanyCode, strDepartmentCode),
string.Format("{0}/{1}", strCompany, strDepartment), false,
this.Sub_LockedCompany() + strChangeFlag, out strOnClick);
```

檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_Organization(string Key, int OrgType, string Attribute, string ScriptName, bool UsingTextArea, string Title, string Codes, string Names, bool AddInput, string LockORG, out string strOnClick)

```
....
1012.        System.Data.SqlClient.SqlDataReader objDR1 =
DOC.Query(strSQL);
```

Heuristic SQL Injection\路徑 92:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=244
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的SendAlert透過WebCatalog.aspx.cs中的2882之RegisterStartupScript執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_FolderMtn 第 632 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	675	2886
物件	Form	RegisterStartupScript

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_FolderMtn(int Action)

```
....
675.            strFolderName =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form["strFolder
Name"], false));
....
692.              this.SendAlert(string.Format("開放區域：{0},\n成功刪除！",
strFolderName), "");
```

檔案名稱 WebCatalog.aspx.cs

方法 private void SendAlert(string strMsg, string AppCMD)

```
....
2882.      private void SendAlert(string strMsg, string AppCMD)
....
2884.        string strJava = "\n<script>window.alert(\"" +
strMsg.Replace("\"", "\\\"").TrimEnd('\n').Replace("\r",
"").Replace("\n", "\\n") + "\");" + AppCMD + "</script>\n";
....
2886.        this.ClientScript.RegisterStartupScript(this.GetType(),
"cds_Alert", strJava);
```

Heuristic SQL Injection\路徑 93:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=245
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_FolderMtn透過WebCatalog.aspx.cs中的632之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_FolderMtn 第 632 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs

行	674	679
物件	Form	Query

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private void Sub_FolderMtn(int Action)

```
....
674.             strFolderID =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form["strFolder
ID"], false));
....
679.          objDR1 = DOC.Query("SELECT ID FROM DocFolder WITH(NOLOCK)
WHERE DocFolderID = '" + strFolderID + "';");
```

Heuristic SQL Injection\路徑 94:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=246
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:06 PM

應用程式中的Sub_FolderMtn透過WebCatalog.aspx.cs中的632之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_FolderMtn 第 632 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	674	903
物件	Form	Query

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private void Sub_FolderMtn(int Action)

```
....
674.            strFolderID =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form["strFolder
ID"], false));
....
892.            objDR1 = DOC.GetFolderAuthority(strFolderID);
893.            while(objDR1.Read())
....
895.              strCode = objDR1.IsDBNull(1) ? "" :
objDR1.GetString(1).Trim();
....
902.                  string[] aryTemp1 = strCode.Split('/'); strName =
"";
903.                  objDR2 = objDBA.Query("SELECT CompanyName FROM
AppCompany WITH(NOLOCK) WHERE CompanyNo = '" + aryTemp1[0] + "'");
```

Heuristic SQL Injection\路徑 95:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=247
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_FolderMtn透過WebCatalog.aspx.cs中的632之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_FolderMtn 第 632 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	674	911
物件	Form	Query

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private void Sub_FolderMtn(int Action)

```
....
674.            strFolderID =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form["strFolder
ID"], false));
....
892.            objDR1 = DOC.GetFolderAuthority(strFolderID);
893.            while(objDR1.Read())
....
895.              strCode = objDR1.IsDBNull(1) ? "" :
objDR1.GetString(1).Trim();
....
902.                  string[] aryTemp1 = strCode.Split('/'); strName =
"";
....
911.                    objDR2 = objDBA.Query("SELECT DepartmentName
FROM AppDepartment WITH(NOLOCK) WHERE CompanyNo = '" + aryTemp1[0] + "'
AND DepartmentNo = '" + aryTemp1[1] + "'");
```

Heuristic SQL Injection\路徑 96:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=248
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_FolderMtn透過WebCatalog.aspx.cs中的632之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebCatalog.aspx.cs 的 Sub_FolderMtn 第 632 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	674	920
物件	Form	Query

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private void Sub_FolderMtn(int Action)

```
....
674.            strFolderID =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form["strFolder
ID"], false));
....
892.            objDR1 = DOC.GetFolderAuthority(strFolderID);
893.            while(objDR1.Read())
....
895.              strCode = objDR1.IsDBNull(1) ? "" :
objDR1.GetString(1).Trim();
....
902.                  string[] aryTemp1 = strCode.Split('/'); strName =
"";
....
920.                    objDR2 = objDBA.Query("SELECT Name FROM AppUser
WITH(NOLOCK) WHERE UID = '" + aryTemp1[2] + "'");
```

Heuristic SQL Injection\路徑 97:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=249
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的BindData透過WebDocumentLog.aspx.cs中的52之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebDocumentLog.aspx.cs 的 Sub_GetRequest 第 137 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebDocumentLog.aspx.cs	WebDocumentLog.aspx.cs
行	141	65
物件	Form	Query

代碼片斷

檔案名稱 WebDocumentLog.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
141.          strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
146.        return strResult.Trim();
```

檔案名稱 WebDocumentLog.aspx.cs

方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
33.         _sDocID = Sub_GetRequest("DocID", "");
....
42.             if(!IsPostBack) BindData();
```

檔案名稱	WebDocumentLog.aspx.cs
方法	private void BindData()

```
....
65.         System.Data.DataTable tb = objDoc.Query(string.Format(fmtSQL,
_sDocID), "LogEvent");
```

Heuristic SQL Injection\路徑 98:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=250
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的BindData透過WebDocumentLog.aspx.cs中的52之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebDocumentLog.aspx.cs 的 Sub_GetRequest 第 137 的使用者輸入 QueryString_strName 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebDocumentLog.aspx.cs	WebDocumentLog.aspx.cs
行	144	65
物件	QueryString_strName	Query

代碼片斷	
檔案名稱	WebDocumentLog.aspx.cs
方法	private string Sub_GetRequest(string strName, string strDefault)

```
....
144.             strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
146.         return strResult.Trim();
```

檔案名稱	WebDocumentLog.aspx.cs

方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
33.         _sDocID = Sub_GetRequest("DocID", "");
....
42.             if(!IsPostBack) BindData();
```

檔案名稱	WebDocumentLog.aspx.cs
方法	private void BindData()

```
....
65.         System.Data.DataTable tb = objDoc.Query(string.Format(fmtSQL,
_sDocID), "LogEvent");
```

Heuristic SQL Injection\路徑 99:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=251
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebDownloadFiles.aspx.cs中的33之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebDownloadFiles.aspx.cs 的 GetRequest 第 163 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebDownloadFiles.aspx.cs	WebDownloadFiles.aspx.cs
行	167	81
物件	Form	Query

代碼片斷	
檔案名稱	WebDownloadFiles.aspx.cs
方法	private string GetRequest(string strName, string strDefault)

```
....
167.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
172.         return strResult.Trim();
```

檔案名稱	WebDownloadFiles.aspx.cs

方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
41.         varDocID = GetRequest("DocID", "");
....
43.         varLogKey = GetRequest("LogKey", "\u0001");
....
81.           objDR0 = objDB0.Query("SELECT DocXML FROM DocCatalog
WITH(NOLOCK) WHERE DocID = '" + varDocID + "';");
```

Heuristic SQL Injection\路徑 100:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=252
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebDownloadFiles.aspx.cs中的33之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebDownloadFiles.aspx.cs 的 GetRequest 第 163 的使用者輸入 QueryString_strName 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebDownloadFiles.aspx.cs	WebDownloadFiles.aspx.cs
行	170	81
物件	QueryString_strName	Query

代碼片斷	
檔案名稱	WebDownloadFiles.aspx.cs
方法	private string GetRequest(string strName, string strDefault)

```
....
170.            strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
172.        return strResult.Trim();
```

檔案名稱	WebDownloadFiles.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
41.         varDocID = GetRequest("DocID", "");
....
43.         varLogKey = GetRequest("LogKey", "\u0001");
....
81.           objDR0 = objDB0.Query("SELECT DocXML FROM DocCatalog
WITH(NOLOCK) WHERE DocID = '" + varDocID + "';");
```

Heuristic SQL Injection\路徑 101:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=253
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebEditor.aspx.cs中的115之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 GetRequest 第 88 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	90	283
物件	Form	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
90.         string strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.Form[strNam
e], false), false));
....
97.         return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 private string XSS(string strData, bool bUrl)

```
....
2841.      private string XSS(string strData, bool bUrl)
....
2843.        string strResult = HttpUtility.HtmlEncode(strData);
2844.        strResult = HttpUtility.HtmlDecode(strResult);
....
2846.        return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
149.        if(!IsPostBack) LogKey.Value = GetRequest("LogKey",
"\u0001\u0002\u0003");
....
217.        objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.          _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
283.          objDR1 = objDoc.Query(objSqlCommand, 30);
```

Heuristic SQL Injection\路徑 102:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=254
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebEditor.aspx.cs中的115之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 GetRequest 第 88 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	90	305
物件	Form	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
90.         string strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.Form[strNam
e], false), false));
....
97.         return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 private string XSS(string strData, bool bUrl)

```
....
2841.       private string XSS(string strData, bool bUrl)
....
2843.         string strResult = HttpUtility.HtmlEncode(strData);
2844.         strResult = HttpUtility.HtmlDecode(strResult);
....
2846.         return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
149.         if(!IsPostBack) LogKey.Value = GetRequest("LogKey",
"\u0001\u0002\u0003");
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.           _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
305.             objDR1 = objDoc.Query(objSqlCommand, 30);
```

Heuristic SQL Injection\路徑 103:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=255
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebEditor.aspx.cs中的115之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 GetRequest 第 88 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

來源	目的地

檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	90	328
物件	Form	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
90.         string strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.Form[strNam
e], false), false));
....
97.         return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 private string XSS(string strData, bool bUrl)

```
....
2841.       private string XSS(string strData, bool bUrl)
....
2843.         string strResult = HttpUtility.HtmlEncode(strData);
2844.         strResult = HttpUtility.HtmlDecode(strResult);
....
2846.         return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
149.         if(!IsPostBack) LogKey.Value = GetRequest("LogKey",
"\u0001\u0002\u0003");
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.           _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
328.                 objDR1 = objDoc.Query(string.Format(strSQL1,
CSS(txt_DocID.Value), objDoc.User.CompanyCode,
objDoc.User.DepartmentCode));
```

Heuristic SQL Injection\路徑 104:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=256
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebEditor.aspx.cs中的115之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 GetRequest 第 88 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	90	341
物件	Form	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
90.         string strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.Form[strNam
e], false), false));
....
97.         return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 private string XSS(string strData, bool bUrl)

```
....
2841.       private string XSS(string strData, bool bUrl)
....
2843.         string strResult = HttpUtility.HtmlEncode(strData);
2844.         strResult = HttpUtility.HtmlDecode(strResult);
....
2846.         return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
149.        if(!IsPostBack) LogKey.Value = GetRequest("LogKey",
"\u0001\u0002\u0003");
....
217.        objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.          _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
328.                objDR1 = objDoc.Query(string.Format(strSQL1,
CSS(txt_DocID.Value), objDoc.User.CompanyCode,
objDoc.User.DepartmentCode));
....
341.                objDR1 = objDoc.Query(string.Format(strSQL1,
CSS(txt_DocID.Value)));
```

Heuristic SQL Injection\路徑 105:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=257
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebEditor.aspx.cs中的115之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 GetRequest 第 88 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	90	380
物件	Form	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
90.        string strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.Form[strNam
e], false), false));
....
97.        return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 private string XSS(string strData, bool bUrl)

```
....
2841.      private string XSS(string strData, bool bUrl)
....
2843.        string strResult = HttpUtility.HtmlEncode(strData);
2844.        strResult = HttpUtility.HtmlDecode(strResult);
....
2846.        return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
257.            sDocID = txt_DocID.Value = GetRequest("DocID", "");
....
380.              objDR1 = objDoc.Query(string.Format(strSQL1, sDocID));
```

Heuristic SQL Injection\路徑 106:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=258
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的CopyAttachmentFiles透過WebEditor.aspx.cs中的633之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 GetRequest 第 88 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	90	650
物件	Form	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
90.        string strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.Form[strNam
e], false), false));
....
97.        return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 private string XSS(string strData, bool bUrl)

```
....
2841.      private string XSS(string strData, bool bUrl)
....
2843.        string strResult = HttpUtility.HtmlEncode(strData);
2844.        strResult = HttpUtility.HtmlDecode(strResult);
....
2846.        return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
149.        if(!IsPostBack) LogKey.Value = GetRequest("LogKey",
"\u0001\u0002\u0003");
....
217.        objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
243.          ExportXml();
```

檔案名稱 WebEditor.aspx.cs

方法 private void ExportXml()

```
....
794.                sXML = CopyAttachmentFiles(PathT(txt_DocID.Value),
PathT(txt_NewDocID.Value), sXML);
```

檔案名稱 WebEditor.aspx.cs

方法 private string CopyAttachmentFiles(string OldDocID, string DocID, string DocXML)

```
....
635.        OldDocID = PathT(OldDocID);
636.        DocID = PathT(DocID);
....
650.        objDR1 = objDoc.Query(string.Format(fmtSQL,
txt_NewDocID.Value));
```

Heuristic SQL Injection\路徑 107:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=259
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的ExportXml透過WebEditor.aspx.cs中的724之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 GetRequest 第 88 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	90	819
物件	Form	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs
方法 private string GetRequest(string strName, string strDefault)

```
....
90.         string strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.Form[strNam
e], false), false));
....
97.         return strResult;
```

檔案名稱 WebEditor.aspx.cs
方法 private string XSS(string strData, bool bUrl)

```
....
2841.      private string XSS(string strData, bool bUrl)
....
2843.        string strResult = HttpUtility.HtmlEncode(strData);
2844.        strResult = HttpUtility.HtmlDecode(strResult);
....
2846.        return strResult;
```

檔案名稱 WebEditor.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
149.         if(!IsPostBack) LogKey.Value = GetRequest("LogKey",
"\u0001\u0002\u0003");
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
243.           ExportXml();
```

檔案名稱 WebEditor.aspx.cs

方法 private void ExportXml()

```
....
819.              objDR1 = objDoc.Query("SELECT KmAction, OtherMessage,
DocXML FROM DocCatalog WITH(NOLOCK) WHERE DocID = '" + txt_DocID.Value +
"';");
```

Heuristic SQL Injection\路徑 108:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=260
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的GetDocParams透過WebEditor.aspx.cs中的1146之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 GetRequest 第 88 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	90	1150
物件	Form	Query

代碼片斷
檔案名稱 WebEditor.aspx.cs
方法 private string GetRequest(string strName, string strDefault)

```
....
90.         string strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.Form[strNam
e], false), false));
....
97.         return strResult;
```

檔案名稱 WebEditor.aspx.cs
方法 private string XSS(string strData, bool bUrl)

```
....
2841.       private string XSS(string strData, bool bUrl)
....
2843.         string strResult = HttpUtility.HtmlEncode(strData);
2844.         strResult = HttpUtility.HtmlDecode(strResult);
....
2846.         return strResult;
```

檔案名稱	WebEditor.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
149.        if(!IsPostBack) LogKey.Value = GetRequest("LogKey",
"\u0001\u0002\u0003");
....
217.        objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.          _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
441.                  Literal1.Text = XMLGenRef(ref oXml);
```

檔案名稱	WebEditor.aspx.cs
方法	private string XMLGenRef(ref System.Xml.XmlDocument oXml)

```
....
1198.            dr = objDoc.Apply(_sDocType);
....
1210.        aCategoryList = GetDocParams(sSQL);
```

檔案名稱	WebEditor.aspx.cs
方法	private Hashtable GetDocParams(string SQL)

```
....
1150.        SqlDataReader dr = objDoc.Query(SQL);
```

Heuristic SQL Injection\路徑 109:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=261
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebEditor.aspx.cs中的115之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 GetRequest 第 88 的使用者輸入 QueryString_strName 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs

行	93	283
物件	QueryString_strName	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
93.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.QueryString
[strName], false), false));
....
97.         return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 private string XSS(string strData, bool bUrl)

```
....
2841.      private string XSS(string strData, bool bUrl)
....
2843.         string strResult = HttpUtility.HtmlEncode(strData);
2844.         strResult = HttpUtility.HtmlDecode(strResult);
....
2846.         return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
149.         if(!IsPostBack) LogKey.Value = GetRequest("LogKey",
"\u0001\u0002\u0003");
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.           _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
283.           objDR1 = objDoc.Query(objSqlCommand, 30);
```

Heuristic SQL Injection\路徑 110:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=262
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebEditor.aspx.cs中的115之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 GetRequest 第 88 的使用者輸入 QueryString_strName 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	93	305
物件	QueryString_strName	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs
方法 private string GetRequest(string strName, string strDefault)

```
....
93.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.QueryString
[strName], false), false));
....
97.         return strResult;
```

檔案名稱 WebEditor.aspx.cs
方法 private string XSS(string strData, bool bUrl)

```
....
2841.      private string XSS(string strData, bool bUrl)
....
2843.        string strResult = HttpUtility.HtmlEncode(strData);
2844.        strResult = HttpUtility.HtmlDecode(strResult);
....
2846.        return strResult;
```

檔案名稱 WebEditor.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
149.         if(!IsPostBack) LogKey.Value = GetRequest("LogKey",
"\u0001\u0002\u0003");
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.           _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
305.             objDR1 = objDoc.Query(objSqlCommand, 30);
```

Heuristic SQL Injection\路徑 111:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=263
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebEditor.aspx.cs中的115之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 GetRequest 第 88 的使用者輸入 QueryString_strName 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	93	328
物件	QueryString_strName	Query

代碼片斷	
檔案名稱	WebEditor.aspx.cs
方法	private string GetRequest(string strName, string strDefault)

```
....
93.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.QueryString
[strName], false), false));
....
97.         return strResult;
```

檔案名稱	WebEditor.aspx.cs
方法	private string XSS(string strData, bool bUrl)

```
....
2841.      private string XSS(string strData, bool bUrl)
....
2843.        string strResult = HttpUtility.HtmlEncode(strData);
2844.        strResult = HttpUtility.HtmlDecode(strResult);
....
2846.        return strResult;
```

檔案名稱	WebEditor.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
149.        if(!IsPostBack) LogKey.Value = GetRequest("LogKey",
"\u0001\u0002\u0003");
....
217.        objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.          _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
328.                objDR1 = objDoc.Query(string.Format(strSQL1,
CSS(txt_DocID.Value), objDoc.User.CompanyCode,
objDoc.User.DepartmentCode));
```

Heuristic SQL Injection\路徑 112:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=264
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebEditor.aspx.cs中的115之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 GetRequest 第 88 的使用者輸入 QueryString_strName 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	93	341
物件	QueryString_strName	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
93.          strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.QueryString
[strName], false), false));
....
97.        return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 private string XSS(string strData, bool bUrl)

```
....
2841.      private string XSS(string strData, bool bUrl)
....
2843.        string strResult = HttpUtility.HtmlEncode(strData);
2844.        strResult = HttpUtility.HtmlDecode(strResult);
....
2846.        return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
149.        if(!IsPostBack) LogKey.Value = GetRequest("LogKey",
"\u0001\u0002\u0003");
....
217.        objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.          _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
328.                objDR1 = objDoc.Query(string.Format(strSQL1,
CSS(txt_DocID.Value), objDoc.User.CompanyCode,
objDoc.User.DepartmentCode));
....
341.                objDR1 = objDoc.Query(string.Format(strSQL1,
CSS(txt_DocID.Value)));
```

Heuristic SQL Injection\路徑 113:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=265
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebEditor.aspx.cs中的115之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 GetRequest 第 88 的使用者輸入 QueryString_strName 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	93	380
物件	QueryString_strName	Query

代碼片斷
檔案名稱 WebEditor.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
93.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.QueryString
[strName], false), false));
....
97.         return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 private string XSS(string strData, bool bUrl)

```
....
2841.       private string XSS(string strData, bool bUrl)
....
2843.         string strResult = HttpUtility.HtmlEncode(strData);
2844.         strResult = HttpUtility.HtmlDecode(strResult);
....
2846.         return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
257.            sDocID = txt_DocID.Value = GetRequest("DocID", "");
....
380.              objDR1 = objDoc.Query(string.Format(strSQL1, sDocID));
```

Heuristic SQL Injection\路徑 114:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=266
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的CopyAttachmentFiles透過WebEditor.aspx.cs中的633之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 GetRequest 第 88 的使用者輸入 QueryString_strName 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	93	650
物件	QueryString_strName	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
93.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.QueryString
[strName], false), false));
....
97.         return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 private string XSS(string strData, bool bUrl)

```
....
2841.       private string XSS(string strData, bool bUrl)
....
2843.         string strResult = HttpUtility.HtmlEncode(strData);
2844.         strResult = HttpUtility.HtmlDecode(strResult);
....
2846.         return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
149.         if(!IsPostBack) LogKey.Value = GetRequest("LogKey",
"\u0001\u0002\u0003");
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
243.           ExportXml();
```

檔案名稱 WebEditor.aspx.cs

方法 private void ExportXml()

```
....
794.                 sXML = CopyAttachmentFiles(PathT(txt_DocID.Value),
PathT(txt_NewDocID.Value), sXML);
```

檔案名稱 WebEditor.aspx.cs

方法 private string CopyAttachmentFiles(string OldDocID, string DocID, string DocXML)

```
....
635.         OldDocID = PathT(OldDocID);
636.         DocID = PathT(DocID);
....
650.         objDR1 = objDoc.Query(string.Format(fmtSQL,
txt_NewDocID.Value));
```

Heuristic SQL Injection\路徑 115:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=267
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的ExportXml透過WebEditor.aspx.cs中的724之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 GetRequest 第 88 的使用者輸入 QueryString_strName 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	93	819
物件	QueryString_strName	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
93.          strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.QueryString
[strName], false), false));
....
97.         return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 private string XSS(string strData, bool bUrl)

```
....
2841.      private string XSS(string strData, bool bUrl)
....
2843.        string strResult = HttpUtility.HtmlEncode(strData);
2844.        strResult = HttpUtility.HtmlDecode(strResult);
....
2846.        return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
149.        if(!IsPostBack) LogKey.Value = GetRequest("LogKey",
"\u0001\u0002\u0003");
....
217.        objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
243.          ExportXml();
```

檔案名稱 WebEditor.aspx.cs

方法 private void ExportXml()

```
....
819.              objDR1 = objDoc.Query("SELECT KmAction, OtherMessage,
DocXML FROM DocCatalog WITH(NOLOCK) WHERE DocID = '" + txt_DocID.Value +
"';");
```

Heuristic SQL Injection\路徑 116:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=268
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的GetDocParams透過WebEditor.aspx.cs中的1146之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 GetRequest 第 88 的使用者輸入 QueryString_strName 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	93	1150
物件	QueryString_strName	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
93.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.QueryString
[strName], false), false));
....
97.         return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 private string XSS(string strData, bool bUrl)

```
....
2841.       private string XSS(string strData, bool bUrl)
....
2843.         string strResult = HttpUtility.HtmlEncode(strData);
2844.         strResult = HttpUtility.HtmlDecode(strResult);
....
2846.         return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
149.         if(!IsPostBack) LogKey.Value = GetRequest("LogKey",
"\u0001\u0002\u0003");
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.           _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
441.                   Literal1.Text = XMLGenRef(ref oXml);
```

檔案名稱 WebEditor.aspx.cs

方法 private string XMLGenRef(ref System.Xml.XmlDocument oXml)

```
....
1198.             dr = objDoc.Apply(_sDocType);
....
1210.         aCategoryList = GetDocParams(sSQL);
```

檔案名稱 WebEditor.aspx.cs

方法 private Hashtable GetDocParams(string SQL)

```
....
1150.         SqlDataReader dr = objDoc.Query(SQL);
```

Heuristic SQL Injection\路徑 117:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=269
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebEditor.aspx.cs中的115之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 Page_Load 第 115 的使用者輸入 Value 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	150	283
物件	Value	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
150.         _sLogKey = CSS(LogKey.Value);
....
187.           + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.           _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
283.           objDR1 = objDoc.Query(objSqlCommand, 30);
```

檔案名稱 WebEditor.aspx.cs

方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic SQL Injection\路徑 118:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=270
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebEditor.aspx.cs中的115之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 Page_Load 第 115 的使用者輸入 Value 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	150	305
物件	Value	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
150.        _sLogKey = CSS(LogKey.Value);
....
187.          + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
217.        objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.          _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
305.            objDR1 = objDoc.Query(objSqlCommand, 30);
```

檔案名稱 WebEditor.aspx.cs

方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic SQL Injection\路徑 119:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=271
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebEditor.aspx.cs中的115之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 Page_Load 第 115 的使用者輸入 Value 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	150	328
物件	Value	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
150.        _sLogKey = CSS(LogKey.Value);
....
187.          + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
217.        objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.          _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
328.                objDR1 = objDoc.Query(string.Format(strSQL1,
CSS(txt_DocID.Value), objDoc.User.CompanyCode,
objDoc.User.DepartmentCode));
```

檔案名稱 WebEditor.aspx.cs

方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic SQL Injection\路徑 120:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=272
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebEditor.aspx.cs中的115之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 Page_Load 第 115 的使用者輸入 Value 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	150	341
物件	Value	Query

代碼片斷
檔案名稱 WebEditor.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
150.        _sLogKey = CSS(LogKey.Value);
....
187.          + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
217.        objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.          _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
328.                objDR1 = objDoc.Query(string.Format(strSQL1,
CSS(txt_DocID.Value), objDoc.User.CompanyCode,
objDoc.User.DepartmentCode));
....
341.                objDR1 = objDoc.Query(string.Format(strSQL1,
CSS(txt_DocID.Value)));
```

檔案名稱 WebEditor.aspx.cs

方法	private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic SQL Injection\路徑 121:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=273
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebEditor.aspx.cs中的115之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 Page_Load 第 115 的使用者輸入 Value 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	150	380
物件	Value	Query

代碼片斷	
檔案名稱	WebEditor.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
150.         _sLogKey = CSS(LogKey.Value);
....
187.           + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.           _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
380.               objDR1 = objDoc.Query(string.Format(strSQL1, sDocID));
```

檔案名稱	WebEditor.aspx.cs
方法	private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic SQL Injection\路徑 122:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=274
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的CopyAttachmentFiles透過WebEditor.aspx.cs中的633之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 Page_Load 第 115 的使用者輸入 Value 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	150	650
物件	Value	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
150.        _sLogKey = CSS(LogKey.Value);
....
187.          + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
217.        objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
243.          ExportXml();
```

檔案名稱 WebEditor.aspx.cs

方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

檔案名稱 WebEditor.aspx.cs

方法 private void ExportXml()

```
....
794.                sXML = CopyAttachmentFiles(PathT(txt_DocID.Value),
PathT(txt_NewDocID.Value), sXML);
```

檔案名稱 WebEditor.aspx.cs

方法 private string CopyAttachmentFiles(string OldDocID, string DocID, string DocXML)

```
....
635.        OldDocID = PathT(OldDocID);
636.        DocID = PathT(DocID);
....
650.        objDR1 = objDoc.Query(string.Format(fmtSQL,
txt_NewDocID.Value));
```

Heuristic SQL Injection\路徑 123:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=275
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的ExportXml透過WebEditor.aspx.cs中的724之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 Page_Load 第 115 的使用者輸入 Value 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	150	819
物件	Value	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
150.         _sLogKey = CSS(LogKey.Value);
....
187.           + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
243.           ExportXml();
```

檔案名稱	WebEditor.aspx.cs
方法	private string CSS(string strData)

```
....
2831.       private string CSS(string strData)
....
2833.         return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

檔案名稱	WebEditor.aspx.cs
方法	private void ExportXml()

```
....
819.               objDR1 = objDoc.Query("SELECT KmAction, OtherMessage,
DocXML FROM DocCatalog WITH(NOLOCK) WHERE DocID = '" + txt_DocID.Value +
"';");
```

Heuristic SQL Injection\路徑 124:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=276
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的GetDocParams透過WebEditor.aspx.cs中的1146之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 Page_Load 第 115 的使用者輸入 Value 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	150	1150

物件	Value	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
150.         _sLogKey = CSS(LogKey.Value);
....
187.           + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.           _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
441.                   Literal1.Text = XMLGenRef(ref oXml);
```

檔案名稱 WebEditor.aspx.cs

方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.         return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

檔案名稱 WebEditor.aspx.cs

方法 private string XMLGenRef(ref System.Xml.XmlDocument oXml)

```
....
1198.             dr = objDoc.Apply(_sDocType);
....
1210.         aCategoryList = GetDocParams(sSQL);
```

檔案名稱 WebEditor.aspx.cs

方法 private Hashtable GetDocParams(string SQL)

```
....
1150.         SqlDataReader dr = objDoc.Query(SQL);
```

Heuristic SQL Injection\路徑 125:

嚴重程度： 低風險

結果狀態： 校驗

線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=277

狀態 反覆出現的問題

Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebEditor.aspx.cs中的115之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 Page_Load 第 115 的使用者輸入 Value 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	187	283
物件	Value	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
187.           + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
185.         _sUploadFilePath =
Cdsys.KM.Utility.KMconfig.GetAppSetting("DataPATH", "C:/Temp",
"C:/Temp")
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.           _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
283.           objDR1 = objDoc.Query(objSqlCommand, 30);
```

檔案名稱 WebEditor.aspx.cs

方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic SQL Injection\路徑 126:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=278
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebEditor.aspx.cs中的115之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 Page_Load 第 115 的使用者輸入 Value 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	187	305
物件	Value	Query

代碼片斷
檔案名稱 WebEditor.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
187.           + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
185.         _sUploadFilePath =
Cdsys.KM.Utility.KMconfig.GetAppSetting("DataPATH", "C:/Temp",
"C:/Temp")
....
217.        objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.           _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
305.             objDR1 = objDoc.Query(objSqlCommand, 30);
```

檔案名稱 WebEditor.aspx.cs
方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.         return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic SQL Injection\路徑 127:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=279
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebEditor.aspx.cs中的115之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 Page_Load 第 115 的使用者輸入 Value 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	187	328
物件	Value	Query

代碼片斷
檔案名稱 WebEditor.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
187.           + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
185.         _sUploadFilePath =
Cdsys.KM.Utility.KMconfig.GetAppSetting("DataPATH", "C:/Temp",
"C:/Temp")
....
217.        objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.           _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
328.                objDR1 = objDoc.Query(string.Format(strSQL1,
CSS(txt_DocID.Value), objDoc.User.CompanyCode,
objDoc.User.DepartmentCode));
```

檔案名稱 WebEditor.aspx.cs
方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.         return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic SQL Injection\路徑 128:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=280
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebEditor.aspx.cs中的115之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 Page_Load 第 115 的使用者輸入 Value 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	187	341
物件	Value	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
187.           + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
185.         _sUploadFilePath =
Cdsys.KM.Utility.KMconfig.GetAppSetting("DataPATH", "C:/Temp",
"C:/Temp")
....
217.        objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.           _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
328.                 objDR1 = objDoc.Query(string.Format(strSQL1,
CSS(txt_DocID.Value), objDoc.User.CompanyCode,
objDoc.User.DepartmentCode));
....
341.                 objDR1 = objDoc.Query(string.Format(strSQL1,
CSS(txt_DocID.Value)));
```

檔案名稱 WebEditor.aspx.cs

方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.         return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic SQL Injection\路徑 129:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=281
狀態	反覆出現的問題

Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebEditor.aspx.cs中的115之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 Page_Load 第 115 的使用者輸入 Value 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	187	380
物件	Value	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
187.           + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
185.         _sUploadFilePath =
Cdsys.KM.Utility.KMconfig.GetAppSetting("DataPATH", "C:/Temp",
"C:/Temp")
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.           _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
380.               objDR1 = objDoc.Query(string.Format(strSQL1, sDocID));
```

檔案名稱 WebEditor.aspx.cs

方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.         return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic SQL Injection\路徑 130:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=282
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的CopyAttachmentFiles透過WebEditor.aspx.cs中的633之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 Page_Load 第 115 的使用者輸入 Value 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	187	650
物件	Value	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
187.           + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
185.         _sUploadFilePath =
Cdsys.KM.Utility.KMconfig.GetAppSetting("DataPATH", "C:/Temp",
"C:/Temp")
....
217.        objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
243.          ExportXml();
```

檔案名稱 WebEditor.aspx.cs
方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

檔案名稱 WebEditor.aspx.cs
方法 private void ExportXml()

```
....
794.                sXML = CopyAttachmentFiles(PathT(txt_DocID.Value),
PathT(txt_NewDocID.Value), sXML);
```

檔案名稱 WebEditor.aspx.cs
方法 private string CopyAttachmentFiles(string OldDocID, string DocID, string DocXML)

```
....
635.         OldDocID = PathT(OldDocID);
636.         DocID = PathT(DocID);
....
650.         objDR1 = objDoc.Query(string.Format(fmtSQL,
txt_NewDocID.Value));
```

Heuristic SQL Injection\路徑 131:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=283
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的ExportXml透過WebEditor.aspx.cs中的724之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 Page_Load 第 115 的使用者輸入 Value 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	187	819
物件	Value	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
187.           + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
185.         _sUploadFilePath =
Cdsys.KM.Utility.KMconfig.GetAppSetting("DataPATH", "C:/Temp",
"C:/Temp")
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
243.           ExportXml();
```

檔案名稱 WebEditor.aspx.cs

方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

檔案名稱 WebEditor.aspx.cs

方法 private void ExportXml()

```
....
819.              objDR1 = objDoc.Query("SELECT KmAction, OtherMessage,
DocXML FROM DocCatalog WITH(NOLOCK) WHERE DocID = '" + txt_DocID.Value +
"';");
```

Heuristic SQL Injection\路徑 132:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=284
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的GetDocParams透過WebEditor.aspx.cs中的1146之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 Page_Load 第 115 的使用者輸入 Value 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	187	1150
物件	Value	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
187.           + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
185.         _sUploadFilePath =
Cdsys.KM.Utility.KMconfig.GetAppSetting("DataPATH", "C:/Temp",
"C:/Temp")
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.           _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
441.                   Literal1.Text = XMLGenRef(ref oXml);
```

檔案名稱 WebEditor.aspx.cs

方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.         return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

檔案名稱 WebEditor.aspx.cs

方法 private string XMLGenRef(ref System.Xml.XmlDocument oXml)

```
....
1198.             dr = objDoc.Apply(_sDocType);
....
1210.         aCategoryList = GetDocParams(sSQL);
```

檔案名稱 WebEditor.aspx.cs

方法 private Hashtable GetDocParams(string SQL)

```
....
1150.         SqlDataReader dr = objDoc.Query(SQL);
```

Heuristic SQL Injection\路徑 133:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=285
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebEditor.aspx.cs中的115之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 Page_Load 第 115 的使用者輸入 Value 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	328	328
物件	Value	Query

代碼片斷
檔案名稱 WebEditor.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
328.                 objDR1 = objDoc.Query(string.Format(strSQL1,
CSS(txt_DocID.Value), objDoc.User.CompanyCode,
objDoc.User.DepartmentCode));
```

檔案名稱 WebEditor.aspx.cs
方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic SQL Injection\路徑 134:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=286
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebEditor.aspx.cs中的115之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 Page_Load 第 115 的使用者輸入 Value 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地

檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	341	341
物件	Value	Query

代碼片斷
檔案名稱 WebEditor.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
341.                objDR1 = objDoc.Query(string.Format(strSQL1,
CSS(txt_DocID.Value)));
```

檔案名稱 WebEditor.aspx.cs
方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic SQL Injection\路徑 135:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=287
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebEditor.aspx.cs中的115之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 Page_Load 第 115 的使用者輸入 Value 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	263	380
物件	Value	Query

代碼片斷
檔案名稱 WebEditor.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
263.            sDocID = CSS(txt_DocID.Value);
....
380.              objDR1 = objDoc.Query(string.Format(strSQL1, sDocID));
```

檔案名稱 WebEditor.aspx.cs

方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic SQL Injection\路徑 136:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=288
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的CopyAttachmentFiles透過WebEditor.aspx.cs中的633之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 CopyAttachmentFiles 第 633 的使用者輸入 Value 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	650	650
物件	Value	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private string CopyAttachmentFiles(string OldDocID, string DocID, string DocXML)

```
....
650.          objDR1 = objDoc.Query(string.Format(fmtSQL,
txt_NewDocID.Value));
```

Heuristic SQL Injection\路徑 137:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=1

	0300&pathid=289
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的ExportXml透過WebEditor.aspx.cs中的724之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebEditor.aspx.cs 的 ExportXml 第 724 的使用者輸入 Value 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	819	819
物件	Value	Query

代碼片斷
檔案名稱 WebEditor.aspx.cs
方法 private void ExportXml()

```
....
819.              objDR1 = objDoc.Query("SELECT KmAction, OtherMessage,
DocXML FROM DocCatalog WITH(NOLOCK) WHERE DocID = '" + txt_DocID.Value +
"';");
```

Heuristic SQL Injection\路徑 138:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=290
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的ProcessRequest透過WebGetGroupNo.ashx.cs中的21之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebGetGroupNo.ashx.cs 的 ProcessRequest 第 21 的使用者輸入 Params 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebGetGroupNo.ashx.cs	WebGetGroupNo.ashx.cs
行	25	29
物件	Params	Query

代碼片斷

檔案名稱	WebGetGroupNo.ashx.cs
方法	public void ProcessRequest(HttpContext context)

```
....
25.         string strGroupID =
HttpUtility.UrlDecode(context.Request.Params["ParmData"]);
....
29.         System.Data.SqlClient.SqlDataReader objDR1 =
objDB1.Query(string.Format("SELECT COUNT(*) FROM DocGroupCatalog
WITH(NOLOCK) WHERE GroupID = '{0}';", strGroupID));
```

Heuristic SQL Injection\路徑 139:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=291
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebMailA.aspx.cs中的20之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebMailA.aspx.cs 的 Page_Load 第 20 的使用者輸入 QueryString_DocID 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebMailA.aspx.cs	WebMailA.aspx.cs
行	38	59
物件	QueryString_DocID	Query

代碼片斷	
檔案名稱	WebMailA.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
38.         strDocID =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g["DocID"], false)); // Ū□□ URL □□ɕ□□
....
59.         objDR1 = objDB1.Query(string.Format(strSQL1, strDocID));
```

Heuristic SQL Injection\路徑 140:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=292
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebMailA.aspx.cs中的20之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebMailA.aspx.cs 的 Page_Load 第 20 的使用者輸入 QueryString_DocID 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebMailA.aspx.cs	WebMailA.aspx.cs
行	38	166
物件	QueryString_DocID	Execute

代碼片斷
檔案名稱 WebMailA.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
38.         strDocID =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g["DocID"], false)); // ū��  URL  ��□ƹ��
....
166.             objDB1.Execute(string.Format(strSQL2, strDocID)); //
2007/05/16
```

Heuristic SQL Injection\路徑 141:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=293
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的BuildBookList透過WebMark.aspx.cs中的76之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebMark.aspx.cs 的 Page_Load 第 26 的使用者輸入 QueryString_DocID 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebMark.aspx.cs	WebMark.aspx.cs
行	34	81
物件	QueryString_DocID	Query

代碼片斷

檔案名稱	WebMark.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
34.         varDocID =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g["DocID"], false)); // Ū�� URL ��ڿ��
....
55.               this.divGroupMark.Style["display"] = "none";
BuildBookList();
```

檔案名稱	WebMark.aspx.cs
方法	private void BuildBookList()

```
....
81.         objDR1 = objDB1.Query(string.Format("SELECT DocTitle FROM
DocCatalog WITH(NOLOCK) WHERE DocID = '{0}';", varDocID));
```

Heuristic SQL Injection\路徑 142:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=294
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的BuildGroupList透過WebMark.aspx.cs中的95之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebMark.aspx.cs 的 Page_Load 第 26 的使用者輸入 QueryString_DocID 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebMark.aspx.cs	WebMark.aspx.cs
行	34	100
物件	QueryString_DocID	Query

代碼片斷	
檔案名稱	WebMark.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
34.         varDocID =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g["DocID"], false)); // Ū�� URL ��ڿ��
....
59.               BuildGroupList();
```

檔案名稱 WebMark.aspx.cs

方法 private void BuildGroupList()

```
....
100.         objDR1 = objDB1.Query(string.Format("SELECT DocTitle FROM
DocCatalog WITH(NOLOCK) WHERE DocID = '{0}';", varDocID));
```

Heuristic SQL Injection\路徑 143:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=295
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的fGroupAdd_Click透過WebMark.aspx.cs中的192之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebMark.aspx.cs 的 Page_Load 第 26 的使用者輸入 QueryString_DocID 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebMark.aspx.cs	WebMark.aspx.cs
行	34	222
物件	QueryString_DocID	Execute

代碼片斷

檔案名稱 WebMark.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
34.         varDocID =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g["DocID"], false)); // Ū�� URL ��ڿ��
```

檔案名稱 WebMark.aspx

方法

```
....
127.
```

檔案名稱 WebMark.aspx.cs

方法 protected void fGroupAdd_Click(object sender, System.EventArgs e)

```
....
222.            objDB1.Execute(string.Format(fmtSQL2, aryID[i],
varDocID, objDoc.User.UserID),
```

Heuristic SQL Injection\路徑 144:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=296
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_SAY透過WebNotePad.aspx.cs中的190之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebNotePad.aspx.cs 的 Sub_GetRequest 第 628 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	632	239
物件	Form	Query

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
632.            strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
637.        return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
173.                    this.Sub_SAY(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtTitle", ""),
this.Sub_GetRequest("txtEssay", ""), "", "",
(objDoc.User.eMail.IndexOf("@") < 0 ? objDoc.User.Name :
objDoc.User.eMail), this.Sub_GetRequest("txtChk", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_SAY(string Action, string SysName, string ID, string Title, string Essay, string StartDate, string EndDate, string Commentator, string Chk)

```
....
190.      private void Sub_SAY(string Action, string SysName, string ID,
string Title, string Essay, string StartDate, string EndDate, string
Commentator, string Chk)
....
238.            strSQL = string.Format("SELECT MemoName, MemoContent,
CreateDate, CompanyNo, DepartmentNo, Account, Chk FROM GoodMemo WHERE ID
= '{0}'", ID);
239.            DR = this.objDoc.Query(strSQL);
```

Heuristic SQL Injection\路徑 145:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=297
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_SAY透過WebNotePad.aspx.cs中的190之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebNotePad.aspx.cs 的 Sub_GetRequest 第 628 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	632	262
物件	Form	Query

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
632.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
637.         return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
173.                   this.Sub_SAY(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtTitle", ""),
this.Sub_GetRequest("txtEssay", ""), "", "",
(objDoc.User.eMail.IndexOf("@") < 0 ? objDoc.User.Name :
objDoc.User.eMail), this.Sub_GetRequest("txtChk", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_SAY(string Action, string SysName, string ID, string Title, string Essay, string StartDate, string EndDate, string Commentator, string Chk)

```
....
190.       private void Sub_SAY(string Action, string SysName, string ID,
string Title, string Essay, string StartDate, string EndDate, string
Commentator, string Chk)
....
261.             strSQL = string.Format("SELECT MemoName, MemoContent,
CreateDate, CompanyNo, DepartmentNo, Account, Chk FROM GoodMemo WHERE ID
= '{0}'", ID);
262.             DR = this.objDoc.Query(strSQL);
```

Heuristic SQL Injection\路徑 146:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=298
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_GBK透過WebNotePad.aspx.cs中的287之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebNotePad.aspx.cs 的 Sub_GetRequest 第 628 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs

行	632	337
物件	Form	Query

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
632.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
637.         return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
147.                   this.Sub_GBK(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtBookName",
""), this.Sub_GetRequest("txtBookStore", ""),
this.Sub_GetRequest("txtBookSummary", ""),
this.Sub_GetRequest("txtStartDate", ""),
this.Sub_GetRequest("txtEndDate", ""), (objDoc.User.eMail.IndexOf("@") <
0 ? objDoc.User.Name : objDoc.User.eMail),
this.Sub_GetRequest("txtBookAuthor", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_GBK(string Action, string SysName, string ID, string BookName, string BookStore, string BookContent, string StartDate, string EndDate, string Commentator, string BookAuthor)

```
....
287.       private void Sub_GBK(string Action, string SysName, string ID,
string BookName, string BookStore, string BookContent, string StartDate,
string EndDate, string Commentator, string BookAuthor)
....
336.             strSQL = string.Format("SELECT BookName, BookStore,
BookContent, Sdate, Edate, CreateDate, CompanyNo, DepartmentNo, Account,
note1 FROM GoodBook WHERE ID = '{0}'", ID);
337.             DR = this.objDoc.Query(strSQL);
```

Heuristic SQL Injection\路徑 147:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=299
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_GBK透過WebNotePad.aspx.cs中的287之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebNotePad.aspx.cs 的 Sub_GetRequest 第 628 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	632	360
物件	Form	Query

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
632.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
637.         return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
147.                   this.Sub_GBK(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtBookName",
""), this.Sub_GetRequest("txtBookStore", ""),
this.Sub_GetRequest("txtBookSummary", ""),
this.Sub_GetRequest("txtStartDate", ""),
this.Sub_GetRequest("txtEndDate", ""), (objDoc.User.eMail.IndexOf("@") <
0 ? objDoc.User.Name : objDoc.User.eMail),
this.Sub_GetRequest("txtBookAuthor", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_GBK(string Action, string SysName, string ID, string BookName, string BookStore, string BookContent, string StartDate, string EndDate, string Commentator, string BookAuthor)

```
....
287.      private void Sub_GBK(string Action, string SysName, string ID,
string BookName, string BookStore, string BookContent, string StartDate,
string EndDate, string Commentator, string BookAuthor)
....
359.            strSQL = string.Format("SELECT BookName, BookStore,
BookContent, Sdate, Edate, CreateDate, CompanyNo, DepartmentNo, Account,
note1 FROM GoodBook WHERE ID = '{0}'", ID);
360.            DR = this.objDoc.Query(strSQL);
```

Heuristic SQL Injection\路徑 148:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=300
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_SAY透過WebNotePad.aspx.cs中的190之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebNotePad.aspx.cs 的 Sub_GetRequest 第 628 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	632	233
物件	Form	Execute

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
632.          strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
637.        return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
173.                    this.Sub_SAY(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtTitle", ""),
this.Sub_GetRequest("txtEssay", ""), "", "",
(objDoc.User.eMail.IndexOf("@") < 0 ? objDoc.User.Name :
objDoc.User.eMail), this.Sub_GetRequest("txtChk", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_SAY(string Action, string SysName, string ID, string Title, string Essay, string StartDate, string EndDate, string Commentator, string Chk)

```
....
190.      private void Sub_SAY(string Action, string SysName, string ID,
string Title, string Essay, string StartDate, string EndDate, string
Commentator, string Chk)
....
232.            strSQL += string.Format("UPDATE GoodMemo SET Chk =
'{6}', MemoName = N'{0}', MemoContent = N'{1}', CreateDate = GETDATE(),
CompanyNo = '{2}', DepartmentNo = '{3}', Account = N'{4}' WHERE (id =
{5});", Title, Essay, objDoc.User.CompanyCode,
objDoc.User.DepartmentCode, Commentator, ID, Chk);
233.            this.objDoc.Execute(strSQL);
```

Heuristic SQL Injection\路徑 149:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=301
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_GBK透過WebNotePad.aspx.cs中的287之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebNotePad.aspx.cs 的 Sub_GetRequest 第 628 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	632	331
物件	Form	Execute

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
632.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
637.         return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
147.                   this.Sub_GBK(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtBookName",
""), this.Sub_GetRequest("txtBookStore", ""),
this.Sub_GetRequest("txtBookSummary", ""),
this.Sub_GetRequest("txtStartDate", ""),
this.Sub_GetRequest("txtEndDate", ""), (objDoc.User.eMail.IndexOf("@") <
0 ? objDoc.User.Name : objDoc.User.eMail),
this.Sub_GetRequest("txtBookAuthor", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_GBK(string Action, string SysName, string ID, string BookName, string BookStore, string BookContent, string StartDate, string EndDate, string Commentator, string BookAuthor)

```
....
287.       private void Sub_GBK(string Action, string SysName, string ID,
string BookName, string BookStore, string BookContent, string StartDate,
string EndDate, string Commentator, string BookAuthor)
....
330.             strSQL = string.Format("UPDATE GoodBook SET BookName =
N'{1}', BookStore = N'{2}', BookContent = N'{3}', Sdate = '{4}', Edate =
'{5}', CreateDate = GETDATE(), CompanyNo = '{6}', DepartmentNo = '{7}',
Account = N'{8}', note1 = N'{9}' WHERE (id = {0})", ID, BookName,
BookStore, BookContent, StartDate, EndDate, objDoc.User.CompanyCode,
objDoc.User.DepartmentCode, Commentator, BookAuthor);
331.             this.objDoc.Execute(strSQL);
```

Heuristic SQL Injection\路徑 150:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=302
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_SAY透過WebNotePad.aspx.cs中的190之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。

因此，攻擊者能在 WebNotePad.aspx.cs 的 Sub_GetRequest 第 628 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	632	213
物件	Form	Execute

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
632.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
637.         return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
173.                   this.Sub_SAY(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtTitle", ""),
this.Sub_GetRequest("txtEssay", ""), "", "",
(objDoc.User.eMail.IndexOf("@") < 0 ? objDoc.User.Name :
objDoc.User.eMail), this.Sub_GetRequest("txtChk", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_SAY(string Action, string SysName, string ID, string Title, string Essay, string StartDate, string EndDate, string Commentator, string Chk)

```
....
190.       private void Sub_SAY(string Action, string SysName, string ID,
string Title, string Essay, string StartDate, string EndDate, string
Commentator, string Chk)
....
212.               strSQL = string.Format("INSERT INTO GoodMemo (Chk,
MemoName, MemoContent, CreateDate, CcompanyNo, DepartmentNo, Account)
VALUES ('{5}', N'{0}', N'{1}', GETDATE(), '{2}', '{3}', N'{4}')", Title,
Essay, objDoc.User.CompanyCode, objDoc.User.DepartmentCode, Commentator,
Chk);
213.               this.objDoc.Execute(strSQL);
```

Heuristic SQL Injection\路徑 151:

嚴重程度： 低風險
結果狀態： 校驗

線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=303
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_GBK透過WebNotePad.aspx.cs中的287之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebNotePad.aspx.cs 的 Sub_GetRequest 第 628 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	632	313
物件	Form	Execute

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
632.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
637.         return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
147.                   this.Sub_GBK(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtBookName",
""), this.Sub_GetRequest("txtBookStore", ""),
this.Sub_GetRequest("txtBookSummary", ""),
this.Sub_GetRequest("txtStartDate", ""),
this.Sub_GetRequest("txtEndDate", ""), (objDoc.User.eMail.IndexOf("@") <
0 ? objDoc.User.Name : objDoc.User.eMail),
this.Sub_GetRequest("txtBookAuthor", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_GBK(string Action, string SysName, string ID, string BookName, string BookStore, string BookContent, string StartDate, string EndDate, string Commentator, string BookAuthor)

```
....
287.      private void Sub_GBK(string Action, string SysName, string ID,
string BookName, string BookStore, string BookContent, string StartDate,
string EndDate, string Commentator, string BookAuthor)
....
312.              strSQL = string.Format("INSERT INTO GoodBook
(BookName, BookStore, BookContent, Sdate, Edate, CreateDate, CompanyNo,
DepartmentNo, Account, note1) VALUES (N'{0}', N'{1}', N'{2}', '{3}',
'{4}', GETDATE(), '{5}', '{6}', N'{7}', N'{8}')", BookName, BookStore,
BookContent, StartDate, EndDate, objDoc.User.CompanyCode,
objDoc.User.DepartmentCode, Commentator, BookAuthor);
313.              this.objDoc.Execute(strSQL);
```

Heuristic SQL Injection\路徑 152:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=304
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_SAY透過WebNotePad.aspx.cs中的190之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebNotePad.aspx.cs 的 Sub_GetRequest 第 628 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	632	221
物件	Form	Execute

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
632.          strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
637.        return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
173.                    this.Sub_SAY(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtTitle", ""),
this.Sub_GetRequest("txtEssay", ""), "", "",
(objDoc.User.eMail.IndexOf("@") < 0 ? objDoc.User.Name :
objDoc.User.eMail), this.Sub_GetRequest("txtChk", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_SAY(string Action, string SysName, string ID, string Title, string Essay, string StartDate, string EndDate, string Commentator, string Chk)

```
....
190.      private void Sub_SAY(string Action, string SysName, string ID,
string Title, string Essay, string StartDate, string EndDate, string
Commentator, string Chk)
....
220.            strSQL = string.Format("DELETE FROM GoodMemo WHERE ID =
'{0}'", ID);
221.            this.objDoc.Execute(strSQL);
```

Heuristic SQL Injection\路徑 153:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=305
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_GBK透過WebNotePad.aspx.cs中的287之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebNotePad.aspx.cs 的 Sub_GetRequest 第 628 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	632	321
物件	Form	Execute

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
632.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
637.         return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
147.                   this.Sub_GBK(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtBookName",
""), this.Sub_GetRequest("txtBookStore", ""),
this.Sub_GetRequest("txtBookSummary", ""),
this.Sub_GetRequest("txtStartDate", ""),
this.Sub_GetRequest("txtEndDate", ""), (objDoc.User.eMail.IndexOf("@") <
0 ? objDoc.User.Name : objDoc.User.eMail),
this.Sub_GetRequest("txtBookAuthor", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_GBK(string Action, string SysName, string ID, string BookName, string BookStore, string BookContent, string StartDate, string EndDate, string Commentator, string BookAuthor)

```
....
287.       private void Sub_GBK(string Action, string SysName, string ID,
string BookName, string BookStore, string BookContent, string StartDate,
string EndDate, string Commentator, string BookAuthor)
....
320.             strSQL = string.Format("DELETE FROM GoodBook WHERE ID =
'{0}'", ID);
321.             this.objDoc.Execute(strSQL);
```

Heuristic SQL Injection\路徑 154:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=306
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_SAY透過WebNotePad.aspx.cs中的190之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebNotePad.aspx.cs 的 Sub_GetRequest 第 628 的使用者輸入 QueryString_strName 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	635	239
物件	QueryString_strName	Query

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
635.            strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
637.        return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
173.                  this.Sub_SAY(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtTitle", ""),
this.Sub_GetRequest("txtEssay", ""), "", "",
(objDoc.User.eMail.IndexOf("@") < 0 ? objDoc.User.Name :
objDoc.User.eMail), this.Sub_GetRequest("txtChk", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_SAY(string Action, string SysName, string ID, string Title, string Essay, string StartDate, string EndDate, string Commentator, string Chk)

```
....
190.      private void Sub_SAY(string Action, string SysName, string ID,
string Title, string Essay, string StartDate, string EndDate, string
Commentator, string Chk)
....
238.            strSQL = string.Format("SELECT MemoName, MemoContent,
CreateDate, CompanyNo, DepartmentNo, Account, Chk FROM GoodMemo WHERE ID
= '{0}'", ID);
239.            DR = this.objDoc.Query(strSQL);
```

Heuristic SQL Injection\路徑 155:

嚴重程度： 低風險

結果狀態： 校驗

線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=307

狀態 反覆出現的問題

Detection Date 7/8/2022 3:05:06 PM

應用程式中的Sub_SAY透過WebNotePad.aspx.cs中的190之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebNotePad.aspx.cs 的 Sub_GetRequest 第 628 的使用者輸入 QueryString_strName 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	635	262
物件	QueryString_strName	Query

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
635.            strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
637.        return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
173.                  this.Sub_SAY(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtTitle", ""),
this.Sub_GetRequest("txtEssay", ""), "", "",
(objDoc.User.eMail.IndexOf("@") < 0 ? objDoc.User.Name :
objDoc.User.eMail), this.Sub_GetRequest("txtChk", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_SAY(string Action, string SysName, string ID, string Title, string Essay, string StartDate, string EndDate, string Commentator, string Chk)

```
....
190.      private void Sub_SAY(string Action, string SysName, string ID,
string Title, string Essay, string StartDate, string EndDate, string
Commentator, string Chk)
....
261.            strSQL = string.Format("SELECT MemoName, MemoContent,
CreateDate, CompanyNo, DepartmentNo, Account, Chk FROM GoodMemo WHERE ID
= '{0}'", ID);
262.            DR = this.objDoc.Query(strSQL);
```

Heuristic SQL Injection\路徑 156:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=308
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_GBK透過WebNotePad.aspx.cs中的287之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebNotePad.aspx.cs 的 Sub_GetRequest 第 628 的使用者輸入 QueryString_strName 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	635	337
物件	QueryString_strName	Query

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
635.             strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
637.         return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
147.                     this.Sub_GBK(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtBookName",
""), this.Sub_GetRequest("txtBookStore", ""),
this.Sub_GetRequest("txtBookSummary", ""),
this.Sub_GetRequest("txtStartDate", ""),
this.Sub_GetRequest("txtEndDate", ""), (objDoc.User.eMail.IndexOf("@") <
0 ? objDoc.User.Name : objDoc.User.eMail),
this.Sub_GetRequest("txtBookAuthor", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_GBK(string Action, string SysName, string ID, string BookName, string BookStore, string BookContent, string StartDate, string EndDate, string Commentator, string BookAuthor)

```
....
287.      private void Sub_GBK(string Action, string SysName, string ID,
string BookName, string BookStore, string BookContent, string StartDate,
string EndDate, string Commentator, string BookAuthor)
....
336.            strSQL = string.Format("SELECT BookName, BookStore,
BookContent, Sdate, Edate, CreateDate, CompanyNo, DepartmentNo, Account,
note1 FROM GoodBook WHERE ID = '{0}'", ID);
337.            DR = this.objDoc.Query(strSQL);
```

Heuristic SQL Injection\路徑 157:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=309
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_GBK透過WebNotePad.aspx.cs中的287之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebNotePad.aspx.cs 的 Sub_GetRequest 第 628 的使用者輸入 QueryString_strName 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	635	360
物件	QueryString_strName	Query

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
635.            strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
637.        return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
147.                    this.Sub_GBK(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtBookName",
""), this.Sub_GetRequest("txtBookStore", ""),
this.Sub_GetRequest("txtBookSummary", ""),
this.Sub_GetRequest("txtStartDate", ""),
this.Sub_GetRequest("txtEndDate", ""), (objDoc.User.eMail.IndexOf("@") <
0 ? objDoc.User.Name : objDoc.User.eMail),
this.Sub_GetRequest("txtBookAuthor", ""));
```

檔案名稱	WebNotePad.aspx.cs
方法	private void Sub_GBK(string Action, string SysName, string ID, string BookName, string BookStore, string BookContent, string StartDate, string EndDate, string Commentator, string BookAuthor)

```
....
287.      private void Sub_GBK(string Action, string SysName, string ID,
string BookName, string BookStore, string BookContent, string StartDate,
string EndDate, string Commentator, string BookAuthor)
....
359.            strSQL = string.Format("SELECT BookName, BookStore,
BookContent, Sdate, Edate, CreateDate, CompanyNo, DepartmentNo, Account,
note1 FROM GoodBook WHERE ID = '{0}'", ID);
360.            DR = this.objDoc.Query(strSQL);
```

Heuristic SQL Injection\路徑 158:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=310
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_SAY透過WebNotePad.aspx.cs中的190之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebNotePad.aspx.cs 的 Sub_GetRequest 第 628 的使用者輸入 QueryString_strName 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	635	233
物件	QueryString_strName	Execute

代碼片斷	
檔案名稱	WebNotePad.aspx.cs
方法	private string Sub_GetRequest(string strName, string strDefault)

```
....
635.            strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
637.        return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
173.                    this.Sub_SAY(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtTitle", ""),
this.Sub_GetRequest("txtEssay", ""), "", "",
(objDoc.User.eMail.IndexOf("@") < 0 ? objDoc.User.Name :
objDoc.User.eMail), this.Sub_GetRequest("txtChk", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_SAY(string Action, string SysName, string ID, string Title, string Essay, string StartDate, string EndDate, string Commentator, string Chk)

```
....
190.      private void Sub_SAY(string Action, string SysName, string ID,
string Title, string Essay, string StartDate, string EndDate, string
Commentator, string Chk)
....
232.            strSQL += string.Format("UPDATE GoodMemo SET Chk =
'{6}', MemoName = N'{0}', MemoContent = N'{1}', CreateDate = GETDATE(),
CompanyNo = '{2}', DepartmentNo = '{3}', Account = N'{4}' WHERE (id =
{5});", Title, Essay, objDoc.User.CompanyCode,
objDoc.User.DepartmentCode, Commentator, ID, Chk);
233.            this.objDoc.Execute(strSQL);
```

Heuristic SQL Injection\路徑 159:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=311
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_GBK透過WebNotePad.aspx.cs中的287之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebNotePad.aspx.cs 的 Sub_GetRequest 第 628 的使用者輸入 QueryString_strName 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

來源	目的地

檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	635	331
物件	QueryString_strName	Execute

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
635.            strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
637.        return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
147.                    this.Sub_GBK(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtBookName",
""), this.Sub_GetRequest("txtBookStore", ""),
this.Sub_GetRequest("txtBookSummary", ""),
this.Sub_GetRequest("txtStartDate", ""),
this.Sub_GetRequest("txtEndDate", ""), (objDoc.User.eMail.IndexOf("@") <
0 ? objDoc.User.Name : objDoc.User.eMail),
this.Sub_GetRequest("txtBookAuthor", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_GBK(string Action, string SysName, string ID, string BookName, string BookStore, string BookContent, string StartDate, string EndDate, string Commentator, string BookAuthor)

```
....
287.      private void Sub_GBK(string Action, string SysName, string ID,
string BookName, string BookStore, string BookContent, string StartDate,
string EndDate, string Commentator, string BookAuthor)
....
330.            strSQL = string.Format("UPDATE GoodBook SET BookName =
N'{1}', BookStore = N'{2}', BookContent = N'{3}', Sdate = '{4}', Edate =
'{5}', CreateDate = GETDATE(), CompanyNo = '{6}', DepartmentNo = '{7}',
Account = N'{8}', note1 = N'{9}' WHERE (id = {0})", ID, BookName,
BookStore, BookContent, StartDate, EndDate, objDoc.User.CompanyCode,
objDoc.User.DepartmentCode, Commentator, BookAuthor);
331.            this.objDoc.Execute(strSQL);
```

Heuristic SQL Injection\路徑 160:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=1

	0300&pathid=312
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_SAY透過WebNotePad.aspx.cs中的190之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebNotePad.aspx.cs 的 Sub_GetRequest 第 628 的使用者輸入 QueryString_strName 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	635	213
物件	QueryString_strName	Execute

代碼片斷

檔案名稱 WebNotePad.aspx.cs
方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
635.             strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
637.         return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
173.                   this.Sub_SAY(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtTitle", ""),
this.Sub_GetRequest("txtEssay", ""), "", "",
(objDoc.User.eMail.IndexOf("@") < 0 ? objDoc.User.Name :
objDoc.User.eMail), this.Sub_GetRequest("txtChk", ""));
```

檔案名稱 WebNotePad.aspx.cs
方法 private void Sub_SAY(string Action, string SysName, string ID, string Title, string Essay, string StartDate, string EndDate, string Commentator, string Chk)

```
....
190.      private void Sub_SAY(string Action, string SysName, string ID,
string Title, string Essay, string StartDate, string EndDate, string
Commentator, string Chk)
....
212.              strSQL = string.Format("INSERT INTO GoodMemo (Chk,
MemoName, MemoContent, CreateDate, CompanyNo, DepartmentNo, Account)
VALUES ('{5}', N'{0}', N'{1}', GETDATE(), '{2}', '{3}', N'{4}')", Title,
Essay, objDoc.User.CompanyCode, objDoc.User.DepartmentCode, Commentator,
Chk);
213.              this.objDoc.Execute(strSQL);
```

Heuristic SQL Injection\路徑 161:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=313
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_GBK透過WebNotePad.aspx.cs中的287之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebNotePad.aspx.cs 的 Sub_GetRequest 第 628 的使用者輸入 QueryString_strName 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	635	313
物件	QueryString_strName	Execute

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
635.              strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
637.          return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
147.                    this.Sub_GBK(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtBookName",
""), this.Sub_GetRequest("txtBookStore", ""),
this.Sub_GetRequest("txtBookSummary", ""),
this.Sub_GetRequest("txtStartDate", ""),
this.Sub_GetRequest("txtEndDate", ""), (objDoc.User.eMail.IndexOf("@") <
0 ? objDoc.User.Name : objDoc.User.eMail),
this.Sub_GetRequest("txtBookAuthor", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_GBK(string Action, string SysName, string ID, string BookName, string BookStore, string BookContent, string StartDate, string EndDate, string Commentator, string BookAuthor)

```
....
287.      private void Sub_GBK(string Action, string SysName, string ID,
string BookName, string BookStore, string BookContent, string StartDate,
string EndDate, string Commentator, string BookAuthor)
....
312.              strSQL = string.Format("INSERT INTO GoodBook
(BookName, BookStore, BookContent, Sdate, Edate, CreateDate, CompanyNo,
DepartmentNo, Account, note1) VALUES (N'{0}', N'{1}', N'{2}', '{3}',
'{4}', GETDATE(), '{5}', '{6}', N'{7}', N'{8}')", BookName, BookStore,
BookContent, StartDate, EndDate, objDoc.User.CompanyCode,
objDoc.User.DepartmentCode, Commentator, BookAuthor);
313.              this.objDoc.Execute(strSQL);
```

Heuristic SQL Injection\路徑 162:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=314
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_SAY透過WebNotePad.aspx.cs中的190之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebNotePad.aspx.cs 的 Sub_GetRequest 第 628 的使用者輸入 QueryString_strName 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	635	221
物件	QueryString_strName	Execute

代碼片斷

檔案名稱	WebNotePad.aspx.cs
方法	private string Sub_GetRequest(string strName, string strDefault)

```
....
635.            strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
637.        return strResult.Trim();
```

檔案名稱	WebNotePad.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
173.                    this.Sub_SAY(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtTitle", ""),
this.Sub_GetRequest("txtEssay", ""), "", "",
(objDoc.User.eMail.IndexOf("@") < 0 ? objDoc.User.Name :
objDoc.User.eMail), this.Sub_GetRequest("txtChk", ""));
```

檔案名稱	WebNotePad.aspx.cs
方法	private void Sub_SAY(string Action, string SysName, string ID, string Title, string Essay, string StartDate, string EndDate, string Commentator, string Chk)

```
....
190.      private void Sub_SAY(string Action, string SysName, string ID,
string Title, string Essay, string StartDate, string EndDate, string
Commentator, string Chk)
....
220.            strSQL = string.Format("DELETE FROM GoodMemo WHERE ID =
'{0}'", ID);
221.            this.objDoc.Execute(strSQL);
```

Heuristic SQL Injection\路徑 163:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=315
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_GBK透過WebNotePad.aspx.cs中的287之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebNotePad.aspx.cs 的 Sub_GetRequest 第 628 的使用者輸入 QueryString_strName 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

來源	目的地

檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	635	321
物件	QueryString_strName	Execute

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
635.            strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
637.        return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
147.                    this.Sub_GBK(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtBookName",
""), this.Sub_GetRequest("txtBookStore", ""),
this.Sub_GetRequest("txtBookSummary", ""),
this.Sub_GetRequest("txtStartDate", ""),
this.Sub_GetRequest("txtEndDate", ""), (objDoc.User.eMail.IndexOf("@") <
0 ? objDoc.User.Name : objDoc.User.eMail),
this.Sub_GetRequest("txtBookAuthor", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_GBK(string Action, string SysName, string ID, string BookName, string BookStore, string BookContent, string StartDate, string EndDate, string Commentator, string BookAuthor)

```
....
287.      private void Sub_GBK(string Action, string SysName, string ID,
string BookName, string BookStore, string BookContent, string StartDate,
string EndDate, string Commentator, string BookAuthor)
....
320.            strSQL = string.Format("DELETE FROM GoodBook WHERE ID =
'{0}'", ID);
321.            this.objDoc.Execute(strSQL);
```

Heuristic SQL Injection\路徑 164:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=316
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的BindData透過WebPrint.aspx.cs中的244之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebPrint.aspx.cs 的 PrintDoc 第 508 的使用者輸入 AbsoluteUri 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	514	251
物件	AbsoluteUri	Query

代碼片斷

檔案名稱 WebPrint.aspx.cs

方法 private void PrintDoc()

```
....
514.        string sUrl = CSS(Request.Url.AbsoluteUri);
515.        objDoc.URL = sUrl = sUrl.Substring(0, sUrl.LastIndexOf("/")
+ 1);
....
585.        objDoc.SaveData(strGUID, "WebDocPrintA", ref strHead, ref
strResult, true, DateTime.Now.ToString("yyyy/MM/dd HH:mm:ss"));
```

檔案名稱 WebPrint.aspx.cs

方法 private string CSS(string strData)

```
....
939.      private string CSS(string strData)
....
941.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

檔案名稱 WebPrint.aspx.cs

方法 protected void btn_Print_Click(object sender, System.EventArgs e)

```
....
417.        PrintDoc();
```

檔案名稱 WebPrint.aspx

方法

```
....
154.
```

檔案名稱 WebPrint.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
53.         LogKey .Value = _sLogKey = GetRequest("LogKey", ""); // Ū��
URL ��ɛ��
```

檔案名稱 WebPrint.aspx.cs

方法 protected void btn_SaveExcel_Click(object sender, System.EventArgs e)

```
....
891.         BindData(false);
```

檔案名稱 WebPrint.aspx.cs

方法 private void BindData(bool bBindData)

```
....
251.         System.Data.SqlClient.SqlDataReader objReader1 =
objDoc.Query(strSQL1);
```

Heuristic SQL Injection\路徑 165:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=317
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的PrintDoc透過WebPrint.aspx.cs中的508之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebPrint.aspx.cs 的 PrintDoc 第 508 的使用者輸入 AbsoluteUri 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	514	532

物件	AbsoluteUri	Query

代碼片斷

檔案名稱 WebPrint.aspx.cs

方法 private void PrintDoc()

```
....
514.         string sUrl = CSS(Request.Url.AbsoluteUri);
515.         objDoc.URL = sUrl = sUrl.Substring(0, sUrl.LastIndexOf("/")
+ 1);
....
525.           Cdsys.KM.Utility.QueryDocData objQDoc =
objDoc.QueryDocumentXML(aDocid[i]); // 95/04/27
....
532.             objDR = objDoc.Query("SELECT * FROM DocDefMain WHERE
DocDefID = '" + objQDoc.DocDefID + "'");
```

檔案名稱 WebPrint.aspx.cs

方法 private string CSS(string strData)

```
....
939.       private string CSS(string strData)
....
941.         return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic SQL Injection\路徑 166:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=318
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的BindData透過WebPrint.aspx.cs中的244之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebPrint.aspx.cs 的 SaveDocHTML 第 597 的使用者輸入 AbsoluteUri 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	617	251
物件	AbsoluteUri	Query

代碼片斷

檔案名稱 WebPrint.aspx.cs

方法 private void SaveDocHTML()

```
....
617.        string sUrl = CSS(Request.Url.AbsoluteUri);
618.        objDoc.URL = sUrl = sUrl.Substring(0, sUrl.LastIndexOf("/")
+ 1);
```

檔案名稱 WebPrint.aspx.cs

方法 private string CSS(string strData)

```
....
939.      private string CSS(string strData)
....
941.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

檔案名稱 WebPrint.aspx.cs

方法 protected void btn_SaveHTML_Click(object sender, System.EventArgs e)

```
....
469.        SaveDocHTML();
```

檔案名稱 WebPrint.aspx

方法

```
....
154.
```

檔案名稱 WebPrint.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
53.        LogKey .Value = _sLogKey = GetRequest("LogKey", ""); // Ū��
URL ��ɔ��
```

檔案名稱 WebPrint.aspx.cs

方法 protected void btn_SaveExcel_Click(object sender, System.EventArgs e)

```
....
891.        BindData(false);
```

檔案名稱	WebPrint.aspx.cs
方法	private void BindData(bool bBindData)

```
....
251.        System.Data.SqlClient.SqlDataReader objReader1 =
objDoc.Query(strSQL1);
```

Heuristic SQL Injection\路徑 167:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=319
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的SaveDocHTML透過WebPrint.aspx.cs中的597之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebPrint.aspx.cs 的 SaveDocHTML 第 597 的使用者輸入 AbsoluteUri 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	617	655
物件	AbsoluteUri	Query

代碼片斷

檔案名稱	WebPrint.aspx.cs
方法	private void SaveDocHTML()

```
....
617.        string sUrl = CSS(Request.Url.AbsoluteUri);
618.        objDoc.URL = sUrl = sUrl.Substring(0, sUrl.LastIndexOf("/")
+ 1);
....
628.          Cdsys.KM.Utility.QueryDocData objQDoc =
objDoc.QueryDocumentXML(aDocid[i]); // 95/04/27
....
655.            objDR = objDoc.Query("SELECT * FROM DocDefMain WHERE
DocDefID = '" + objQDoc.DocDefID + "'");
```

檔案名稱	WebPrint.aspx.cs
方法	private string CSS(string strData)

```
....
939.      private string CSS(string strData)
....
941.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic SQL Injection\路徑 168:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=320
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的PrintDoc透過WebPrint.aspx.cs中的508之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebPrint.aspx.cs 的 GetRequest 第 143 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	147	532
物件	Form	Query

代碼片斷

檔案名稱 WebPrint.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
147.          strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
152.        return strResult.Trim();
```

檔案名稱 WebPrint.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
53.        LogKey .Value = _sLogKey = GetRequest("LogKey", ""); // Ū��
URL ��ƪ��
```

檔案名稱 WebPrint.aspx

方法

```
....
154.
```

檔案名稱 WebPrint.aspx.cs

方法 protected void btn_Print_Click(object sender, System.EventArgs e)

```
....
414.         ChangeSelect();
....
416.         SaveSelect();
417.         PrintDoc();
```

檔案名稱 WebPrint.aspx.cs

方法 private void PrintDoc()

```
....
512.         string sDocIDs = objDoc.LoadData(_sLogKey, "PrintIDs");
513.         sDocIDs = sDocIDs.Replace("@", ",");
....
517.         string[] aDocid = sDocIDs.Split(',');
....
525.           Cdsys.KM.Utility.QueryDocData objQDoc =
objDoc.QueryDocumentXML(aDocid[i]); // 95/04/27
....
532.             objDR = objDoc.Query("SELECT * FROM DocDefMain WHERE
DocDefID = '" + objQDoc.DocDefID + "'");
```

Heuristic SQL Injection\路徑 169:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=321
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的SaveDocHTML透過WebPrint.aspx.cs中的597之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebPrint.aspx.cs 的 GetRequest 第 143 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	147	655

物件	Form	Query

代碼片斷

檔案名稱 WebPrint.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
147.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
152.         return strResult.Trim();
```

檔案名稱 WebPrint.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
53.         LogKey .Value = _sLogKey = GetRequest("LogKey", ""); // Ū��
URL ��ɛ��
```

檔案名稱 WebPrint.aspx

方法

```
....
154.
```

檔案名稱 WebPrint.aspx.cs

方法 protected void btn_SaveHTML_Click(object sender, System.EventArgs e)

```
....
466.         ChangeSelect();
....
468.         SaveSelect();
469.         SaveDocHTML();
```

檔案名稱 WebPrint.aspx.cs

方法 private void SaveDocHTML()

```
....
599.         UI(false);
....
615.         string sDocIDs = objDoc.LoadData(_sLogKey, "PrintIDs");
616.         sDocIDs = sDocIDs.Replace("@", ",");
....
620.         string[] aDocid = sDocIDs.Split(',');
....
628.           Cdsys.KM.Utility.QueryDocData objQDoc =
objDoc.QueryDocumentXML(aDocid[i]); // 95/04/27
....
655.             objDR = objDoc.Query("SELECT * FROM DocDefMain WHERE
DocDefID = '" + objQDoc.DocDefID + "'");
```

Heuristic SQL Injection\路徑 170:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=322
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的PrintDoc透過WebPrint.aspx.cs中的508之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebPrint.aspx.cs 的 GetRequest 第 143 的使用者輸入 QueryString_strName 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	150	532
物件	QueryString_strName	Query

代碼片斷

檔案名稱 WebPrint.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
150.             strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
152.         return strResult.Trim();
```

檔案名稱 WebPrint.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
53.         LogKey .Value = _sLogKey = GetRequest("LogKey", ""); // Ū��
URL ��□ع��
```

檔案名稱 WebPrint.aspx

方法

```
....
154.
```

檔案名稱 WebPrint.aspx.cs

方法 protected void btn_Print_Click(object sender, System.EventArgs e)

```
....
414.         ChangeSelect();
....
416.         SaveSelect();
417.         PrintDoc();
```

檔案名稱 WebPrint.aspx.cs

方法 private void PrintDoc()

```
....
512.         string sDocIDs = objDoc.LoadData(_sLogKey, "PrintIDs");
513.         sDocIDs = sDocIDs.Replace("@", ",");
....
517.         string[] aDocid = sDocIDs.Split(',');
....
525.           Cdsys.KM.Utility.QueryDocData objQDoc =
objDoc.QueryDocumentXML(aDocid[i]); // 95/04/27
....
532.             objDR = objDoc.Query("SELECT * FROM DocDefMain WHERE
DocDefID = '" + objQDoc.DocDefID + "'");
```

Heuristic SQL Injection\路徑 171:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=323
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的SaveDocHTML透過WebPrint.aspx.cs中的597之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。

因此，攻擊者能在 WebPrint.aspx.cs 的 GetRequest 第 143 的使用者輸入 QueryString_strName 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	150	655
物件	QueryString_strName	Query

代碼片斷

檔案名稱 WebPrint.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
150.            strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
152.        return strResult.Trim();
```

檔案名稱 WebPrint.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
53.        LogKey .Value = _sLogKey = GetRequest("LogKey", ""); // Ū��
URL ��ƪ��
```

檔案名稱 WebPrint.aspx

方法

```
....
154.
```

檔案名稱 WebPrint.aspx.cs

方法 protected void btn_SaveHTML_Click(object sender, System.EventArgs e)

```
....
466.        ChangeSelect();
....
468.        SaveSelect();
469.        SaveDocHTML();
```

檔案名稱 WebPrint.aspx.cs

方法	private void SaveDocHTML()

```
....
599.         UI(false);
....
615.         string sDocIDs = objDoc.LoadData(_sLogKey, "PrintIDs");
616.         sDocIDs = sDocIDs.Replace("@", ",");
....
620.         string[] aDocid = sDocIDs.Split(',');
....
628.           Cdsys.KM.Utility.QueryDocData objQDoc =
objDoc.QueryDocumentXML(aDocid[i]); // 95/04/27
....
655.             objDR = objDoc.Query("SELECT * FROM DocDefMain WHERE
DocDefID = '" + objQDoc.DocDefID + "'");
```

Heuristic SQL Injection\路徑 172:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=324
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的ProcessType_1透過WebPrintAllData.aspx.cs中的199之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebPrintAllData.aspx.cs 的 GetRequest 第 43 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebPrintAllData.aspx.cs	WebPrintAllData.aspx.cs
行	47	212
物件	Form	Query

代碼片斷	
檔案名稱	WebPrintAllData.aspx.cs
方法	private string GetRequest(string strName, string strDefault)

```
....
47.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
52.         return strResult.Trim();
```

檔案名稱	WebPrintAllData.aspx.cs
方法	private void ProcessType_1()

```
....
206.         string strDocID = GetRequest("DocID", "");
....
210.           strSQL = string.Format("SELECT ID FROM DocCatalog WHERE
DocID = '{0}';", strDocID);
....
212.         objDR1 = objDB1.Query(strSQL);
```

Heuristic SQL Injection\路徑 173:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=325
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的ProcessType_1透過WebPrintAllData.aspx.cs中的199之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebPrintAllData.aspx.cs 的 GetRequest 第 43 的使用者輸入 QueryString_strName 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebPrintAllData.aspx.cs	WebPrintAllData.aspx.cs
行	50	212
物件	QueryString_strName	Query

代碼片斷

檔案名稱 WebPrintAllData.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
50.            strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
52.        return strResult.Trim();
```

檔案名稱 WebPrintAllData.aspx.cs

方法 private void ProcessType_1()

```
....
206.         string strDocID = GetRequest("DocID", "");
....
210.           strSQL = string.Format("SELECT ID FROM DocCatalog WHERE
DocID = '{0}';", strDocID);
....
212.         objDR1 = objDB1.Query(strSQL);
```

Heuristic SQL Injection\路徑 174:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=326
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebRedirect.aspx.cs中的77之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebRedirect.aspx.cs 的 Sub_Request 第 140 的使用者輸入 QueryString_DocID 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebRedirect.aspx.cs	WebRedirect.aspx.cs
行	152	87
物件	QueryString_DocID	Query

代碼片斷

檔案名稱 WebRedirect.aspx.cs
方法 private void Sub_Request()

```
....
152.         strA =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g["DocID"], false));
153.         varDocID = (strA == null) ? "" : strA.Trim();
....
168.         varURL = Sub_CheckURL(strA, varDocID, varKey);
```

檔案名稱 WebRedirect.aspx.cs
方法 private string Sub_CheckURL(string URL, string DocID, string Key)

```
....
251.      private string Sub_CheckURL(string URL, string DocID, string
Key)
....
256.          URL = "ShowDocument.aspx?DocID=" + DocID + "&LogKey=" +
Key + "&SysFrom=" + varSysFrom; // 95/01/12
....
280.        return URL.Replace("http://http://", "http://");
```

檔案名稱 WebRedirect.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
79.        Sub_Request();
80.        Sub_VerifyUser();
....
82.        if(varURL.ToLower().StartsWith("http"))
....
84.          string[] aryTemp = varURL.Replace("\\", "/").Split('/');
85.          string strFileName = aryTemp[aryTemp.Length - 1];
....
87.          objDR1 = DBA.Query("SELECT TOP 1 Filename FROM
DocUploadFile WITH(NOLOCK) WHERE Rfilename = N'" + strFileName + "';");
```

檔案名稱 WebRedirect.aspx.cs

方法 public void Sub_VerifyUser()

```
....
136.        Sub_ConvertUrl();
```

Heuristic SQL Injection\路徑 175:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=327
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebRedirect.aspx.cs中的77之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebRedirect.aspx.cs 的 Sub_Request 第 140 的使用者輸入 QueryString_Key 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebRedirect.aspx.cs	WebRedirect.aspx.cs

行	149	87
物件	QueryString_Key	Query

代碼片斷

檔案名稱 WebRedirect.aspx.cs

方法 private void Sub_Request()

```
....
149.        strA =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g["Key"], false));
150.        varKey = (strA == null) ? "" : strA.Trim();
....
168.        varURL = Sub_CheckURL(strA, varDocID, varKey);
```

檔案名稱 WebRedirect.aspx.cs

方法 private string Sub_CheckURL(string URL, string DocID, string Key)

```
....
251.      private string Sub_CheckURL(string URL, string DocID, string
Key)
....
256.          URL = "ShowDocument.aspx?DocID=" + DocID + "&LogKey=" +
Key + "&SysFrom=" + varSysFrom; // 95/01/12
....
280.        return URL.Replace("http://http://", "http://");
```

檔案名稱 WebRedirect.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
79.        Sub_Request();
80.        Sub_VerifyUser();
....
82.        if(varURL.ToLower().StartsWith("http"))
....
84.          string[] aryTemp = varURL.Replace("\\", "/").Split('/');
85.          string strFileName = aryTemp[aryTemp.Length - 1];
....
87.          objDR1 = DBA.Query("SELECT TOP 1 Filename FROM
DocUploadFile WITH(NOLOCK) WHERE Rfilename = N'" + strFileName + "';");
```

檔案名稱 WebRedirect.aspx.cs

方法 public void Sub_VerifyUser()

```
....
136.        Sub_ConvertUrl();
```

Heuristic SQL Injection\路徑 176:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=328
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Sub_CheckKey透過WebRedirect.aspx.cs中的186之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebRedirect.aspx.cs 的 Sub_Request 第 140 的使用者輸入 QueryString_Key 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebRedirect.aspx.cs	WebRedirect.aspx.cs
行	149	191
物件	QueryString_Key	Query

代碼片斷	
檔案名稱	WebRedirect.aspx.cs
方法	private void Sub_Request()

```
....
149.         strA =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g["Key"], false));
150.         varKey = (strA == null) ? "" : strA.Trim();
....
168.         varURL = Sub_CheckURL(strA, varDocID, varKey);
....
180.           this.Sub_CheckKey(varKey);
```

檔案名稱	WebRedirect.aspx.cs
方法	private bool Sub_CheckKey(string Key)

```
....
186.      private bool Sub_CheckKey(string Key)
....
190.         string strSQL = "SELECT UID FROM UserProcess WHERE (LogKey =
'" + Key + "')";
191.         System.Data.SqlClient.SqlDataReader DR = DBA.Query(strSQL);
```

Heuristic SQL Injection\路徑 177:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=329

狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebRedirect.aspx.cs中的77之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebRedirect.aspx.cs 的 Sub_Request 第 140 的使用者輸入 QueryString_SysFrom 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebRedirect.aspx.cs	WebRedirect.aspx.cs
行	146	87
物件	QueryString_SysFrom	Query

代碼片斷

檔案名稱 WebRedirect.aspx.cs

方法 private void Sub_Request()

```
....
146.         strA =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g["SysFrom"], false));
147.         varSysFrom = (strA == null) ? "" : strA.Trim();
....
168.         varURL = Sub_CheckURL(strA, varDocID, varKey);
```

檔案名稱 WebRedirect.aspx.cs

方法 private string Sub_CheckURL(string URL, string DocID, string Key)

```
....
256.           URL = "ShowDocument.aspx?DocID=" + DocID + "&LogKey=" +
Key + "&SysFrom=" + varSysFrom; // 95/01/12
....
280.         return URL.Replace("http://http://", "http://");
```

檔案名稱 WebRedirect.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
79.         Sub_Request();
80.         Sub_VerifyUser();
....
82.         if(varURL.ToLower().StartsWith("http"))
....
84.           string[] aryTemp = varURL.Replace("\\", "/").Split('/');
85.           string strFileName = aryTemp[aryTemp.Length - 1];
....
87.           objDR1 = DBA.Query("SELECT TOP 1 Filename FROM
DocUploadFile WITH(NOLOCK) WHERE Rfilename = N'" + strFileName + "';");
```

檔案名稱	WebRedirect.aspx.cs
方法	public void Sub_VerifyUser()

```
....
136.         Sub_ConvertUrl();
```

Heuristic SQL Injection\路徑 178:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=330
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebRedirect.aspx.cs中的77之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebRedirect.aspx.cs 的 Sub_Request 第 140 的使用者輸入 QueryString_URL 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebRedirect.aspx.cs	WebRedirect.aspx.cs
行	155	87
物件	QueryString_URL	Query

代碼片斷	
檔案名稱	WebRedirect.aspx.cs
方法	private void Sub_Request()

```
....
155.         strA =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g["URL"], false));
156.         strA = (strA == null) ? "" : strA.Trim();
....
168.         varURL = Sub_CheckURL(strA, varDocID, varKey);
```

檔案名稱 WebRedirect.aspx.cs

方法 private string Sub_CheckURL(string URL, string DocID, string Key)

```
....
251.      private string Sub_CheckURL(string URL, string DocID, string
Key)
....
280.         return URL.Replace("http://http://", "http://");
```

檔案名稱 WebRedirect.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
79.         Sub_Request();
80.         Sub_VerifyUser();
....
82.         if(varURL.ToLower().StartsWith("http"))
....
84.           string[] aryTemp = varURL.Replace("\\", "/").Split('/');
85.           string strFileName = aryTemp[aryTemp.Length - 1];
....
87.           objDR1 = DBA.Query("SELECT TOP 1 Filename FROM
DocUploadFile WITH(NOLOCK) WHERE Rfilename = N'" + strFileName + "';");
```

檔案名稱 WebRedirect.aspx.cs

方法 public void Sub_VerifyUser()

```
....
136.         Sub_ConvertUrl();
```

Heuristic SQL Injection\路徑 179:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=331
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebRedirect.aspx.cs中的77之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebRedirect.aspx.cs 的 TransferFileToUrl 第 206 的使用者輸入 ToString 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebRedirect.aspx.cs	WebRedirect.aspx.cs
行	211	87
物件	ToString	Query

代碼片斷

檔案名稱 WebRedirect.aspx.cs
方法 private string TransferFileToUrl(string DocID, string strFileName)

```
....
211.        string varURL = CSS(Request.Url.ToString());
....
213.        string strURL = varURL.Substring(0, varURL.LastIndexOf("/")
+ 1); // ﷺtﷺ (/)
....
234.            strFileUpURL = strURL + aryTemp[aryTemp.Length - 1];
....
244.            if(System.IO.File.Exists(sPath)) return strFileUpURL +
"/" + strFileName;
```

檔案名稱 WebRedirect.aspx.cs
方法 private string CSS(string strData)

```
....
285.      private string CSS(string strData)
....
287.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

檔案名稱 WebRedirect.aspx.cs
方法 private string Sub_CheckURL(string URL, string DocID, string Key)

```
....
272.            URL = TransferFileToUrl(DocID, sFileName);
....
280.        return URL.Replace("http://http://", "http://");
```

檔案名稱 WebRedirect.aspx.cs
方法 private void Sub_Request()

```
....
168.         varURL = Sub_CheckURL(strA, varDocID, varKey);
```

檔案名稱 WebRedirect.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
79.         Sub_Request();
80.         Sub_VerifyUser();
....
82.         if(varURL.ToLower().StartsWith("http"))
....
84.           string[] aryTemp = varURL.Replace("\\", "/").Split('/');
85.           string strFileName = aryTemp[aryTemp.Length - 1];
....
87.           objDR1 = DBA.Query("SELECT TOP 1 Filename FROM
DocUploadFile WITH(NOLOCK) WHERE Rfilename = N'" + strFileName + "';");
```

檔案名稱 WebRedirect.aspx.cs

方法 public void Sub_VerifyUser()

```
....
136.         Sub_ConvertUrl();
```

Heuristic SQL Injection\路徑 180:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=332
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebRedirect.aspx.cs中的77之Query執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebRedirect.aspx.cs 的 TransferFileToUrl 第 206 的使用者輸入 Url 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebRedirect.aspx.cs	WebRedirect.aspx.cs
行	211	87
物件	Url	Query

代碼片斷

檔案名稱 WebRedirect.aspx.cs
方法 private string TransferFileToUrl(string DocID, string strFileName)

```
....
211.         string varURL = CSS(Request.Url.ToString());
....
213.         string strURL = varURL.Substring(0, varURL.LastIndexOf("/")
+ 1); // ﷺtﷺ (/)
....
234.             strFileUpURL = strURL + aryTemp[aryTemp.Length - 1];
....
244.             if(System.IO.File.Exists(sPath)) return strFileUpURL +
"/" + strFileName;
```

檔案名稱 WebRedirect.aspx.cs
方法 private string CSS(string strData)

```
....
285.      private string CSS(string strData)
....
287.         return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

檔案名稱 WebRedirect.aspx.cs
方法 private string Sub_CheckURL(string URL, string DocID, string Key)

```
....
272.             URL = TransferFileToUrl(DocID, sFileName);
....
280.         return URL.Replace("http://http://", "http://");
```

檔案名稱 WebRedirect.aspx.cs
方法 private void Sub_Request()

```
....
168.         varURL = Sub_CheckURL(strA, varDocID, varKey);
```

檔案名稱 WebRedirect.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
79.         Sub_Request();
80.         Sub_VerifyUser();
....
82.         if(varURL.ToLower().StartsWith("http"))
....
84.           string[] aryTemp = varURL.Replace("\\", "/").Split('/');
85.           string strFileName = aryTemp[aryTemp.Length - 1];
....
87.           objDR1 = DBA.Query("SELECT TOP 1 Filename FROM
DocUploadFile WITH(NOLOCK) WHERE Rfilename = N'" + strFileName + "';");
```

檔案名稱	WebRedirect.aspx.cs
方法	public void Sub_VerifyUser()

```
....
136.         Sub_ConvertUrl();
```

Heuristic SQL Injection\路徑 181:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=333
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebLogout.aspx.cs中的20之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebLogout.aspx.cs 的 Sub_GetRequest 第 50 的使用者輸入 Form 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebLogout.aspx.cs	WebLogout.aspx.cs
行	54	41
物件	Form	Execute

代碼片斷	
檔案名稱	WebLogout.aspx.cs
方法	private string Sub_GetRequest(string strName, string strDefault)

```
....
54.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
59.         return strResult.Trim();
```

檔案名稱 WebLogout.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
24.         string strLogKey = Sub_GetRequest("LogKey", ""); // Ū�� Form
�����
....
34.             strSQL += "Update LogUser Set LogoutDate = getdate()
WHERE (LogKey = '" + strLogKey.Trim() + "');\n";
35.             strSQL += "DELETE UserProcess WHERE LogKey = '" +
strLogKey.Trim() + "';\n";
....
41.           try { objDB.Execute(strSQL); }
```

Heuristic SQL Injection\路徑 182:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=334
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebLogout.aspx.cs中的20之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebLogout.aspx.cs 的 Sub_GetRequest 第 50 的使用者輸入 QueryString_strName 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebLogout.aspx.cs	WebLogout.aspx.cs
行	57	41
物件	QueryString_strName	Execute

代碼片斷

檔案名稱 WebLogout.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
57.             strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
59.         return strResult.Trim();
```

檔案名稱	WebLogout.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
24.         string strLogKey = Sub_GetRequest("LogKey", ""); // Ū◆◆ Form
◆◆◆◆◆
....
34.             strSQL += "Update LogUser Set LogoutDate = getdate()
WHERE (LogKey = '" + strLogKey.Trim() + "');\n";
35.             strSQL += "DELETE UserProcess WHERE LogKey = '" +
strLogKey.Trim() + "';\n";
....
41.           try { objDB.Execute(strSQL); }
```

Heuristic SQL Injection\路徑 183:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=335
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

應用程式中的Page_Load透過WebLogout.aspx.cs中的20之Execute執行SQL 查詢(Query)。此應用程式建構SQL查詢時，在未過濾的的情況下將不受信任的字串嵌入到查詢字串中。串接後的查詢字串被提交到資料庫，在資料庫中進行解析(parse)及執行(executed)。
這種明顯的資料庫存取看似是被封裝在外部元件或API中。
因此，攻擊者能在 WebLogout.aspx.cs 的 Page_Load 第 20 的使用者輸入 QueryString_LogKey 放入惡意資料。由於這些資料沒有進行消毒，因此會隨著程式流進入外部API，再從外部流入資料庫伺服器。
根據分析，這有機會造成 SQL Injection 攻擊。

	來源	目的地
檔案	WebLogout.aspx.cs	WebLogout.aspx.cs
行	25	41
物件	QueryString_LogKey	Execute

代碼片斷	
檔案名稱	WebLogout.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
25.         if(strLogKey == null) strLogKey =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g["LogKey"], false)); // Ū��  URL  ��□ƹ��
....
37.           strSQL += "DELETE UserCache WHERE LogKey = '" +
strLogKey.Trim() + "';\n";
....
41.           try { objDB.Execute(strSQL); }
```

Heuristic Stored XSS

查詢路徑:

CSharp\Cx\CSharp Heuristic\Heuristic Stored XSS 版本:1

類別

PCI DSS v3.2.1: PCI DSS (3.2.1) - 6.5.7 - Cross-site scripting (XSS)
OWASP Top 10 2013: A3-Cross-Site Scripting (XSS)
FISMA 2014: System And Information Integrity
NIST SP 800-53: SI-15 Information Output Filtering (P0)
OWASP Top 10 2017: A7-Cross-Site Scripting (XSS)
ASD STIG 4.10: APSC-DV-002490 - CAT I The application must protect from Cross-Site Scripting (XSS) vulnerabilities.
OWASP Top 10 2021: A3-Injection

描述

Heuristic Stored XSS\路徑 1:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=676
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Page_Load在ShowDocument.aspx.cs第31 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於ShowUserFormat在ShowDocument.aspx.cs第161行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	ShowDocument.aspx.cs	ShowDocument.aspx.cs
行	75	166
物件	Query	Text

代碼片斷	
檔案名稱	ShowDocument.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
75.                   objReader1 = objDB1.Query("SELECT GUID FROM AppUser
WITH(NOLOCK) WHERE ID = '" + aryUser[1] + "';");
....
81.                   objReader1.Close();
....
40.           objDoc = new Doc(_sConnectionString, this);
....
132.                   ShowUserFormat(strShowDef, strShowField,
strDocXML);
```

檔案名稱 ShowDocument.aspx

方法

```
....
57.
```

檔案名稱 ShowDocument.aspx.cs

方法 private void ShowUserFormat(string strShowDef, string strShowField, string strDocXML)

```
....
165.          string strData = objDoc.PringPrintDocument(strDocXML,
strShowDef, strShowField).Replace("<br>", "<p></p>");
166.          this.Literal_Text.Text = strData.Replace("<ﾟ˛dﾟX>",
strRep);
```

Heuristic Stored XSS\路徑 2:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=677
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Page_Load在ShowDocument.aspx.cs第31 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Page_Load在ShowDocument.aspx.cs第31行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	ShowDocument.aspx.cs	ShowDocument.aspx.cs
行	75	152
物件	Query	Text

代碼片斷

檔案名稱 ShowDocument.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
75.                   objReader1 = objDB1.Query("SELECT GUID FROM AppUser
WITH(NOLOCK) WHERE ID = '" + aryUser[1] + "';");
....
81.                   objReader1.Close();
....
40.           objDoc = new Doc(_sConnectionString, this);
....
152.           Literal_Content.Text = objDoc.PrintDocument(_sDocID,
objQDoc.DocXml, new string[0], "", true, "");
```

檔案名稱 ShowDocument.aspx

方法

```
....
57.
```

Heuristic Stored XSS\路徑 3:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=678
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Page_Load在ShowDocument.aspx.cs第31 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於ShowUserFormat在ShowDocument.aspx.cs第161行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	ShowDocument.aspx.cs	ShowDocument.aspx.cs
行	116	166
物件	Query	Text

代碼片斷

檔案名稱 ShowDocument.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
116.           objReader1 = objDB1.Query("SELECT DocDefID, DocTitle,
DocXML FROM DocCatalog WITH(NOLOCK) WHERE DocID = '" + _sDocID + "'");
....
121.             string strDocXML = objReader1.GetString(2);
....
132.                   ShowUserFormat(strShowDef, strShowField,
strDocXML);
```

檔案名稱 ShowDocument.aspx.cs

方法	private void ShowUserFormat(string strShowDef, string strShowField, string strDocXML)

```
....
161.      private void ShowUserFormat(string strShowDef, string
strShowField, string strDocXML)
....
165.          string strData = objDoc.PringPrintDocument(strDocXML,
strShowDef, strShowField).Replace("<br>", "<p></p>");
166.          this.Literal_Text.Text = strData.Replace("<ﾟ､dﾟX>",
strRep);
```

Heuristic Stored XSS\路徑 4:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=679
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Page_Load在ShowDocument.aspx.cs第31 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Page_Load在ShowDocument.aspx.cs第31行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	ShowDocument.aspx.cs	ShowDocument.aspx.cs
行	116	152
物件	Query	Text

代碼片斷	
檔案名稱	ShowDocument.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
116.           objReader1 = objDB1.Query("SELECT DocDefID, DocTitle,
DocXML FROM DocCatalog WITH(NOLOCK) WHERE DocID = '" + _sDocID + "'");
....
40.           objDoc = new Doc(_sConnectionString, this);
....
152.           Literal_Content.Text = objDoc.PrintDocument(_sDocID,
objQDoc.DocXml, new string[0], "", true, "");
```

檔案名稱	ShowDocument.aspx
方法	

```
....
57.
```

Heuristic Stored XSS\路徑 5:

嚴重程度：	低風險

結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=680
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Page_Load在ShowLogDocument.aspx.cs第21 行從資訊庫中獲取資訊，做為Format元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Page_Load在ShowLogDocument.aspx.cs第21行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	ShowLogDocument.aspx.cs	ShowLogDocument.aspx.cs
行	27	33
物件	Format	Text

代碼片斷	
檔案名稱	ShowLogDocument.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
27.         string strSQL = String.Format("SELECT EvtNote, EvtDocID FROM
LogEvent WHERE EvtID ={0}", strID);
28.         System.Data.SqlClient.SqlDataReader objDR1 =
objDoc.Query(strSQL);
....
33.           this.ShowData.Text =
objDoc.PrintDocument(objDR1.GetValue(1).ToString(), strXML, new
string[0], "", true, "");
```

Heuristic Stored XSS\路徑 6:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=681
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_MyFunction在WebCatalog.aspx.cs第159 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_MyFunction在WebCatalog.aspx.cs第159行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	220	292
物件	Query	Text

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private void Sub_MyFunction(string FunctionNames)

```
....
220.                objDR = DOC.Query(string.Format("SELECT
FunName,FunDescription FROM AppSubFunction WITH(NOLOCK) WHERE FunKey =
'{0}' AND Enabled = 1 ORDER BY FunSeqNo", aryData[3]));
221.                while(objDR.Read())
....
223.                  strName = objDR.IsDBNull(0) ? "" :
objDR.GetString(0).Trim();
....
233.                      strSubItem.Append(string.Format(fmtSubItem2,
strAction, strDescription, strName));
....
239.                strSubItem.Append(string.Format(fmtSubItem2, "",
"javascript:ShowMenu('divFunc', event);", "<font
style='color:#C71585;'><b>關閉</b></font>"));
240.                strSubMenu = string.Format(strSubMenu, strSubItem);
....
242.                strMyFunction.Append(string.Format("<td><A
class=Function_Web1 href='#' onclick=\"Javascript:ShowMenu('divFunc',
event)\" title='" + aryData[0] + "'> " + aryData[1] +
" </A>{0}</td>", strSubMenu));
....
292.        this.MyFunction.Text = string.Format(fmtTable,
strMyFunction.ToString());
```

Heuristic Stored XSS\路徑 7:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=682
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_MyFunction在WebCatalog.aspx.cs第159 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_MyFunction在WebCatalog.aspx.cs第159行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	254	292
物件	Query	Text

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private void Sub_MyFunction(string FunctionNames)

```
....
254.                objDR = DOC.Query(string.Format("SELECT
FunDescription FROM AppSubFunction WITH(NOLOCK) WHERE FunKey = '{0}' AND
Enabled = 1 ORDER BY FunSeqNo", aryData[3]));
....
257.                  string strFunDescription = objDR.IsDBNull(0) ? ""
: objDR.GetString(0).Trim();
....
261.                    strFunDescription =
WhoAmI.ReplaceParameter(strFunDescription);
262.                    strFunDescription =
strFunDescription.Replace("@GetAppRoleParam@",
varAppRoleParam).Replace("&&", "&").Replace("?&", "?");
....
264.
if(strFunDescription.ToLower().StartsWith("javascript:"))
265.                      strMyFunction.Append(string.Format("<td><A
class=Function_Web1 href='#' onclick=\"{0}\" title='" + aryData[0] +
"'> " + aryData[1] + " </A></td>", strFunDescription));
....
292.        this.MyFunction.Text = string.Format(fmtTable,
strMyFunction.ToString());
```

Heuristic Stored XSS\路徑 8:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=683
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_Category在WebCatalog.aspx.cs第358 行從資訊庫中獲取資訊，做為Format元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Category在WebCatalog.aspx.cs第358行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	524	565
物件	Format	Text

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
524.            sTable1 = String.Format(sTable1,
encrypt.Encrypt(sCategorySQL));
....
565.            this.X02Y02.Text += sTable1;
```

Heuristic Stored XSS\路徑 9:

嚴重程度：	低風險
結果狀態：	校驗

線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=684
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_FolderMtn在WebCatalog.aspx.cs第632 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Debug在WebCatalog.aspx.cs第3722行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	903	3733
物件	Query	Text

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_FolderMtn(int Action)

```
....
903.                    objDR2 = objDBA.Query("SELECT CompanyName FROM
AppCompany WITH(NOLOCK) WHERE CompanyNo = '" + aryTemp1[0] + "'");
....
906.                      strName += objDR2.IsDBNull(0) ? "" :
objDR2.GetString(0);
....
929.                  strPublicArea += strCode + ","; rtnPublicArea +=
strName + ",";
....
934.              rtnPublicArea = rtnPublicArea.TrimEnd(',');
....
937.              this.Sub_Debug(new string[] { "strPublicArea",
"rtnPublicArea" }, new string[] { strPublicArea, rtnPublicArea });
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Debug(string[] Name, string[] Value)

```
....
3722.       private void Sub_Debug(string[] Name, string[] Value)
....
3731.             strData.Append(string.Format(strItem, Name[i],
Value[i]));
....
3733.           this.Note.Text += string.Format(strTable,
strData.ToString().Replace("','", "', '"));
```

Heuristic Stored XSS\路徑 10:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=685
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_FolderMtn在WebCatalog.aspx.cs第632 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Debug在WebCatalog.aspx.cs第3722行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	911	3733
物件	Query	Text

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_FolderMtn(int Action)

```
....
911.                    objDR2 = objDBA.Query("SELECT DepartmentName
FROM AppDepartment WITH(NOLOCK) WHERE CompanyNo = '" + aryTemp1[0] + "'
AND DepartmentNo = '" + aryTemp1[1] + "'");
....
914.                      strName += "/" + (objDR2.IsDBNull(0) ? "" :
objDR2.GetString(0));
....
929.                strPublicArea += strCode + ","; rtnPublicArea +=
strName + ",";
....
934.            rtnPublicArea = rtnPublicArea.TrimEnd(',');
....
937.            this.Sub_Debug(new string[] { "strPublicArea",
"rtnPublicArea" }, new string[] { strPublicArea, rtnPublicArea });
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Debug(string[] Name, string[] Value)

```
....
3722.      private void Sub_Debug(string[] Name, string[] Value)
....
3731.            strData.Append(string.Format(strItem, Name[i],
Value[i]));
....
3733.          this.Note.Text += string.Format(strTable,
strData.ToString().Replace("','", "', '"));
```

Heuristic Stored XSS\路徑 11:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=686
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_FolderMtn在WebCatalog.aspx.cs第632 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Debug在WebCatalog.aspx.cs第3722行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	920	3733
物件	Query	Text

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_FolderMtn(int Action)

```
....
920.                    objDR2 = objDBA.Query("SELECT Name FROM AppUser
WITH(NOLOCK) WHERE UID = '" + aryTemp1[2] + "'");
....
923.                      strName += "/" + (objDR2.IsDBNull(0) ? "" :
objDR2.GetString(0));
....
929.                strPublicArea += strCode + ","; rtnPublicArea +=
strName + ",";
....
934.            rtnPublicArea = rtnPublicArea.TrimEnd(',');
....
937.            this.Sub_Debug(new string[] { "strPublicArea",
"rtnPublicArea" }, new string[] { strPublicArea, rtnPublicArea });
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Debug(string[] Name, string[] Value)

```
....
3722.      private void Sub_Debug(string[] Name, string[] Value)
....
3731.            strData.Append(string.Format(strItem, Name[i],
Value[i]));
....
3733.          this.Note.Text += string.Format(strTable,
strData.ToString().Replace("','", "', '"));
```

Heuristic Stored XSS\路徑 12:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=687
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_BuildGroupPage在WebCatalog.aspx.cs第1293 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_BuildGroupPage在WebCatalog.aspx.cs第1293行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1297	1300
物件	Query	Text

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private void Sub_BuildGroupPage()

```
....
1297.         System.Data.SqlClient.SqlDataReader objDR1 =
DOC.Query("SELECT paramValue FROM AppParameter WITH(NOLOCK) WHERE SysID
= '0' AND ModuleName = 'FirstGroup';");
....
1299.         if(objDR1.Read()) strFstData =
objDR1.GetValue(0).ToString();
1300.         FstData.Text = strFstData;
```

Heuristic Stored XSS\路徑 13:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=688
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_Setup在WebCatalog.aspx.cs第1433 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Setup在WebCatalog.aspx.cs第1433行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1451	1506
物件	Query	Text

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 public void Sub_Setup()

```
....
1451.         System.Data.SqlClient.SqlDataReader DR =
this.DOC.Query(string.Format(strSQL, WhoAmI.UserID));
....
1454.         while(DR.Read())
....
1456.           strGroupName = DR.GetValue(0).ToString().Trim();
....
1458.           strFuncName = DR.GetValue(2).ToString().Trim();
....
1475.                 LIST.Add("", strFuncName,
string.Format("Send('91919','系統管理','','','','','{0}','','','1','1','"
+ varOrderBy + "','')", strFuncID));
....
1495.         strList.Append(LIST.Generate());
....
1506.         this.X01Y07.Text = strList.ToString();
```

檔案名稱 PageSetting.cs

方法 public void Add(string IMG, string Name, string NavigateUrl)

```
....
1050.         public void Add(string IMG, string Name, string
NavigateUrl)
....
1061.               objList.Append(string.Format(fmtItem1, strIMG, Name,
NavigateUrl));
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
1079.             return string.Format(fmtTable, strTitle, objList);
```

Heuristic Stored XSS\路徑 14:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=689
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_SetReport在WebCatalog.aspx.cs第1509 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_SetReport在WebCatalog.aspx.cs第1509行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1523	1565

物件	Query	Text

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 public void Sub_SetReport()

```
....
1523.         System.Data.SqlClient.SqlDataReader DR =
this.DOC.Query(string.Format(strSQL, WhoAmI.UserID,
strDocDefIDs.Replace(",", "','")));
....
1534.         while(DR.Read())
....
1536.           strFuncID = DR.GetValue(0).ToString().Trim();
1537.           strFuncName = DR.GetValue(1).ToString().Trim();
....
1557.             objSB1.Append(string.Format(fmtTR, strTagA,
strFuncName));
....
1565.           this.X01Y05.Text = string.Format(fmtTB,
objSB1.ToString());
```

Heuristic Stored XSS\路徑 15:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=690
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法GetDataIndexData在PageSetting.cs第2599 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_2120在WebCatalog.aspx.cs第1688行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2605	1727
物件	GetValue	Text

代碼片斷
檔案名稱 PageSetting.cs
方法 private string[] GetDataIndexData(System.Data.SqlClient.SqlDataReader DR, int Index, int[][] DataIndex)

```
....
2605.             aryDataIndexData[i] =
DR.GetValue(DataIndex[Index][i]).ToString();
....
2607.         return aryDataIndexData;
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2740.                       strItem = string.Format(fmtHref,
string.Format(Href[Index], this.GetDataIndexData(DR, Index, DataIndex)),
strTemp1);
....
2749.                   strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), ItemAlign + strTdAppAttribute, ""));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.           return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex, int[] RedColorFlag)

```
....
2596.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, aryDataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex)

```
....
2586.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, DataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[] HrefShowDataIndex)

```
....
2040.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_2120()

```
....
1725.         PRT.Add3(DOC.Query(strSQL), true, "1,4", aryHref,
aryShowIndex, aryHrefIndex);
....
1727.         this.X02Y01.Text = PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.           return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 16:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=691
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法GetDataIndexData在PageSetting.cs第2599 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_2110在WebCatalog.aspx.cs第1735行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2605	1771
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string[] GetDataIndexData(System.Data.SqlClient.SqlDataReader DR, int Index, int[][] DataIndex)

```
....
2605.             aryDataIndexData[i] =
DR.GetValue(DataIndex[Index][i]).ToString();
....
2607.           return aryDataIndexData;
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2740.                       strItem = string.Format(fmtHref,
string.Format(Href[Index], this.GetDataIndexData(DR, Index, DataIndex)),
strTemp1);
....
2749.                   strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), ItemAlign + strTdAppAttribute, ""));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.           return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex, int[] RedColorFlag)

```
....
2596.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, aryDataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex)

```
....
2586.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, DataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[] HrefShowDataIndex)

```
....
2040.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_2110()

```
....
1770.         PRT.Add3(DOC.Bulletin(varTitle, ""), true, "1,4,5,6",
aryHref, aryShowIndex, aryHrefIndex);
1771.         this.X02Y01.Text = PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.          return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 17:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=692
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法GetDataIndexData在PageSetting.cs第2599 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_2130在WebCatalog.aspx.cs第1778行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2605	1815
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string[] GetDataIndexData(System.Data.SqlClient.SqlDataReader DR, int Index, int[][] DataIndex)

```
....
2605.              aryDataIndexData[i] =
DR.GetValue(DataIndex[Index][i]).ToString();
....
2607.          return aryDataIndexData;
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2740.                       strItem = string.Format(fmtHref,
string.Format(Href[Index], this.GetDataIndexData(DR, Index, DataIndex)),
strTemp1);
....
2749.                   strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), ItemAlign + strTdAppAttribute, ""));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.           return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex, int[] RedColorFlag)

```
....
2596.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, aryDataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex)

```
....
2586.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, DataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[] HrefShowDataIndex)

```
....
2040.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_2130()

```
....
1813.         PRT.Add3(DOC.Query(strSQL), true, "4", aryHref,
aryShowIndex, aryHrefIndex);
....
1815.         this.X02Y01.Text = PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.           return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 18:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=693
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法GetDataIndexData在PageSetting.cs第2599 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Category在WebCatalog.aspx.cs第358行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2605	564
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string[] GetDataIndexData(System.Data.SqlClient.SqlDataReader DR, int Index, int[][] DataIndex)

```
....
2605.             aryDataIndexData[i] =
DR.GetValue(DataIndex[Index][i]).ToString();
....
2607.           return aryDataIndexData;
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2740.                       strItem = string.Format(fmtHref,
string.Format(Href[Index], this.GetDataIndexData(DR, Index, DataIndex)),
strTemp1);
....
2749.                   strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), ItemAlign + strTdAppAttribute, ""));
....
2754.            strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.            strText.Replace("@rowspan@", "");
....
2765.          return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09

....
564.            this.X02Y02.Text += PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.          return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 19:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=694

狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法GetDataIndexData在PageSetting.cs第2599 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Category在WebCatalog.aspx.cs第358行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2605	565
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string[] GetDataIndexData(System.Data.SqlClient.SqlDataReader DR, int Index, int[][] DataIndex)

```
....
2605.             aryDataIndexData[i] =
DR.GetValue(DataIndex[Index][i]).ToString();
....
2607.          return aryDataIndexData;
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2740.                       strItem = string.Format(fmtHref,
string.Format(Href[Index], this.GetDataIndexData(DR, Index, DataIndex)),
strTemp1);
....
2749.                   strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), ItemAlign + strTdAppAttribute, ""));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.          return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09

....
564.            this.X02Y02.Text += PRT.Generate();
565.            this.X02Y02.Text += sTable1;
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.          return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 20:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=695
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_2120在WebCatalog.aspx.cs第1688行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2700	1727
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2700.                        strItem = DR.GetValue(iStart).ToString();
....
2710.                    strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), "text-align:center;" + strTdAppAttribute,
""));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.          return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex, int[] RedColorFlag)

```
....
2596.          return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, aryDataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex)

```
....
2586.          return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, DataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[] HrefShowDataIndex)

```
....
2040.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_2120()

```
....
1725.        PRT.Add3(DOC.Query(strSQL), true, "1,4", aryHref,
aryShowIndex, aryHrefIndex);
....
1727.        this.X02Y01.Text = PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.          return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 21:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=696
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_2110在WebCatalog.aspx.cs第1735行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2700	1771
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2700.                        strItem = DR.GetValue(iStart).ToString();
....
2710.                    strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), "text-align:center;" + strTdAppAttribute,
""));
....
2754.            strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.            strText.Replace("@rowspan@", "");
....
2765.          return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex, int[] RedColorFlag)

```
....
2596.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, aryDataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex)

```
....
2586.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, DataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[] HrefShowDataIndex)

```
....
2040.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_2110()

```
....
1770.         PRT.Add3(DOC.Bulletin(varTitle, ""), true, "1,4,5,6",
aryHref, aryShowIndex, aryHrefIndex);
1771.         this.X02Y01.Text = PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.           return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 22:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=697
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_2130在WebCatalog.aspx.cs第1778行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2700	1815
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2700.                        strItem = DR.GetValue(iStart).ToString();
....
2710.                    strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), "text-align:center;" + strTdAppAttribute,
""));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.          return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex, int[] RedColorFlag)

```
....
2596.          return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, aryDataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex)

```
....
2586.          return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, DataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[] HrefShowDataIndex)

```
....
2040.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_2130()

```
....
1813.         PRT.Add3(DOC.Query(strSQL), true, "4", aryHref,
aryShowIndex, aryHrefIndex);
....
1815.         this.X02Y01.Text = PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.          return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 23:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=698
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Category在WebCatalog.aspx.cs第358行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2700	564
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2700.                        strItem = DR.GetValue(iStart).ToString();
....
2710.                    strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), "text-align:center;" + strTdAppAttribute,
""));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.          return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09

....
564.            this.X02Y02.Text += PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.          return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 24:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=699
狀態 反覆出現的問題

Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Category在WebCatalog.aspx.cs第358行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2700	565
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2700.                        strItem = DR.GetValue(iStart).ToString();
....
2710.                    strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), "text-align:center;" + strTdAppAttribute,
""));
....
2754.            strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.            strText.Replace("@rowspan@", "");
....
2765.          return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09

....
564.            this.X02Y02.Text += PRT.Generate();
565.            this.X02Y02.Text += sTable1;
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.          return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 25:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=700
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_2120在WebCatalog.aspx.cs第1688行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2721	1727
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2721.                        strItem = DR.GetValue(i).ToString();
....
2726.                    strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), "text-align:right;" + strTdAppAttribute, ""));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.          return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex, int[] RedColorFlag)

```
....
2596.          return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, aryDataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex)

```
....
2586.          return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, DataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[] HrefShowDataIndex)

```
....
2040.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_2120()

```
....
1725.        PRT.Add3(DOC.Query(strSQL), true, "1,4", aryHref,
aryShowIndex, aryHrefIndex);
....
1727.        this.X02Y01.Text = PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.          return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 26:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=701
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_2110在WebCatalog.aspx.cs第1735行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2721	1771
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2721.                          strItem = DR.GetValue(i).ToString();
....
2726.                      strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), "text-align:right;" + strTdAppAttribute, ""));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.           return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex, int[] RedColorFlag)

```
....
2596.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, aryDataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex)

```
....
2586.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, DataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[] HrefShowDataIndex)

```
....
2040.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_2110()

```
....
1770.         PRT.Add3(DOC.Bulletin(varTitle, ""), true, "1,4,5,6",
aryHref, aryShowIndex, aryHrefIndex);
1771.         this.X02Y01.Text = PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.           return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 27:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=702
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_2130在WebCatalog.aspx.cs第1778行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2721	1815
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2721.                        strItem = DR.GetValue(i).ToString();
....
2726.                    strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), "text-align:right;" + strTdAppAttribute, ""));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.          return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex, int[] RedColorFlag)

```
....
2596.          return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, aryDataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex)

```
....
2586.          return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, DataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[] HrefShowDataIndex)

```
....
2040.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_2130()

```
....
1813.         PRT.Add3(DOC.Query(strSQL), true, "4", aryHref,
aryShowIndex, aryHrefIndex);
....
1815.         this.X02Y01.Text = PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.          return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 28:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=703
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Category在WebCatalog.aspx.cs第358行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2721	564
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2721.                        strItem = DR.GetValue(i).ToString();
....
2726.                    strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), "text-align:right;" + strTdAppAttribute, ""));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.          return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09

....
564.            this.X02Y02.Text += PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.          return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 29:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=704
狀態 反覆出現的問題

Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Category在WebCatalog.aspx.cs第358行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2721	565
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2721.                          strItem = DR.GetValue(i).ToString();
....
2726.                      strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), "text-align:right;" + strTdAppAttribute, ""));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.           return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2056.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09

....
564.            this.X02Y02.Text += PRT.Generate();
565.            this.X02Y02.Text += sTable1;
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.          return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 30:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=705
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_2120在WebCatalog.aspx.cs第1688行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2737	1727
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2737.                    string strTemp1 = DR.GetValue(i).ToString();
....
2740.                    strItem = string.Format(fmtHref,
string.Format(Href[Index], this.GetDataIndexData(DR, Index, DataIndex)),
strTemp1);
....
2749.                strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), ItemAlign + strTdAppAttribute, ""));
....
2754.            strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.            strText.Replace("@rowspan@", "");
....
2765.          return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex, int[] RedColorFlag)

```
....
2596.          return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, aryDataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex)

```
....
2586.          return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, DataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[] HrefShowDataIndex)

```
....
2040.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_2120()

```
....
1725.         PRT.Add3(DOC.Query(strSQL), true, "1,4", aryHref,
aryShowIndex, aryHrefIndex);
....
1727.         this.X02Y01.Text = PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.           return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 31:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=706
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_2110在WebCatalog.aspx.cs第1735行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2737	1771
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2737.                       string strTemp1 = DR.GetValue(i).ToString();
....
2740.                       strItem = string.Format(fmtHref,
string.Format(Href[Index], this.GetDataIndexData(DR, Index, DataIndex)),
strTemp1);
....
2749.                   strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), ItemAlign + strTdAppAttribute, ""));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.           return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex, int[] RedColorFlag)

```
....
2596.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, aryDataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex)

```
....
2586.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, DataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[] HrefShowDataIndex)

```
....
2040.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_2110()

```
....
1770.         PRT.Add3(DOC.Bulletin(varTitle, ""), true, "1,4,5,6",
aryHref, aryShowIndex, aryHrefIndex);
1771.         this.X02Y01.Text = PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.           return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 32:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=707
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_2130在WebCatalog.aspx.cs第1778行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2737	1815
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2737.                       string strTemp1 = DR.GetValue(i).ToString();
....
2740.                       strItem = string.Format(fmtHref,
string.Format(Href[Index], this.GetDataIndexData(DR, Index, DataIndex)),
strTemp1);
....
2749.                   strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), ItemAlign + strTdAppAttribute, ""));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.           return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex, int[] RedColorFlag)

```
....
2596.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, aryDataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex)

```
....
2586.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, DataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[] HrefShowDataIndex)

```
....
2040.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_2130()

```
....
1813.         PRT.Add3(DOC.Query(strSQL), true, "4", aryHref,
aryShowIndex, aryHrefIndex);
....
1815.         this.X02Y01.Text = PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.           return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 33:

嚴重程度： 低風險
結果狀態： 校驗

線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=708
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Category在WebCatalog.aspx.cs第358行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2737	564
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2737.                       string strTemp1 = DR.GetValue(i).ToString();
....
2740.                       strItem = string.Format(fmtHref,
string.Format(Href[Index], this.GetDataIndexData(DR, Index, DataIndex)),
strTemp1);
....
2749.                   strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), ItemAlign + strTdAppAttribute, ""));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.           return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2056.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09

....
564.            this.X02Y02.Text += PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.          return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 34:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=709
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Category在WebCatalog.aspx.cs第358行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2737	565
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2737.                       string strTemp1 = DR.GetValue(i).ToString();
....
2740.                       strItem = string.Format(fmtHref,
string.Format(Href[Index], this.GetDataIndexData(DR, Index, DataIndex)),
strTemp1);
....
2749.                   strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), ItemAlign + strTdAppAttribute, ""));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.           return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2056.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09

....
564.             this.X02Y02.Text += PRT.Generate();
565.             this.X02Y02.Text += sTable1;
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.           return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 35:

嚴重程度： 低風險
結果狀態： 校驗

線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=710
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_2120在WebCatalog.aspx.cs第1688行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2743	1727
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2743.                       strItem = DR.GetValue(i).ToString();
....
2749.                   strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), ItemAlign + strTdAppAttribute, ""));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.           return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex, int[] RedColorFlag)

```
....
2596.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, aryDataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex)

```
....
2586.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, DataIndex, RedColorFlag);
```

檔案名稱	PageSetting.cs
方法	public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[] HrefShowDataIndex)

```
....
2040.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex));
```

檔案名稱	WebCatalog.aspx.cs
方法	private void Sub_2120()

```
....
1725.         PRT.Add3(DOC.Query(strSQL), true, "1,4", aryHref,
aryShowIndex, aryHrefIndex);
....
1727.         this.X02Y01.Text = PRT.Generate();
```

檔案名稱	PageSetting.cs
方法	public string Generate()

```
....
2783.           return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 36:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=711
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_2110在WebCatalog.aspx.cs第1735行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2743	1771
物件	GetValue	Text

代碼片斷	
檔案名稱	PageSetting.cs
方法	private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2743.                       strItem = DR.GetValue(i).ToString();
....
2749.                   strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), ItemAlign + strTdAppAttribute, ""));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.          return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex, int[] RedColorFlag)

```
....
2596.          return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, aryDataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex)

```
....
2586.          return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, DataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[] HrefShowDataIndex)

```
....
2040.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_2110()

```
....
1770.        PRT.Add3(DOC.Bulletin(varTitle, ""), true, "1,4,5,6",
aryHref, aryShowIndex, aryHrefIndex);
1771.        this.X02Y01.Text = PRT.Generate();
```

檔案名稱	PageSetting.cs
方法	public string Generate()

```
....
2783.          return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 37:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=712
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_2130在WebCatalog.aspx.cs第1778行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2743	1815
物件	GetValue	Text

代碼片斷

檔案名稱	PageSetting.cs
方法	private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2743.                      strItem = DR.GetValue(i).ToString();
....
2749.                  strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), ItemAlign + strTdAppAttribute, ""));
....
2754.            strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.            strText.Replace("@rowspan@", "");
....
2765.          return string.Format(fmtTable, strText.ToString());
```

檔案名稱	PageSetting.cs
方法	private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex, int[] RedColorFlag)

```
....
2596.          return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, aryDataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex)

```
....
2586.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, DataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[] HrefShowDataIndex)

```
....
2040.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_2130()

```
....
1813.         PRT.Add3(DOC.Query(strSQL), true, "4", aryHref,
aryShowIndex, aryHrefIndex);
....
1815.         this.X02Y01.Text = PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.           return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 38:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=713
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Category在WebCatalog.aspx.cs第358行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

來源	目的地

檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2743	564
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2743.                       strItem = DR.GetValue(i).ToString();
....
2749.                   strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), ItemAlign + strTdAppAttribute, ""));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.           return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2056.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09

....
564.             this.X02Y02.Text += PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.          return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 39:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=714
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Category在WebCatalog.aspx.cs第358行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2743	565
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2743.                      strItem = DR.GetValue(i).ToString();
....
2749.                  strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), ItemAlign + strTdAppAttribute, ""));
....
2754.            strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.            strText.Replace("@rowspan@", "");
....
2765.          return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09

....
564.            this.X02Y02.Text += PRT.Generate();
565.            this.X02Y02.Text += sTable1;
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.           return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 40:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=715
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_2120在WebCatalog.aspx.cs第1688行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2746	1727
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2746.                    strItem = DR.GetValue(i).ToString();
....
2749.                  strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), ItemAlign + strTdAppAttribute, ""));
....
2754.            strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.            strText.Replace("@rowspan@", "");
....
2765.          return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex, int[] RedColorFlag)

```
....
2596.          return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, aryDataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex)

```
....
2586.          return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, DataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[] HrefShowDataIndex)

```
....
2040.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_2120()

```
....
1725.        PRT.Add3(DOC.Query(strSQL), true, "1,4", aryHref,
aryShowIndex, aryHrefIndex);
....
1727.        this.X02Y01.Text = PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.          return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 41:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=716
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_2110在WebCatalog.aspx.cs第1735行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2746	1771
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2746.                    strItem = DR.GetValue(i).ToString();
....
2749.                  strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), ItemAlign + strTdAppAttribute, ""));
....
2754.            strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.            strText.Replace("@rowspan@", "");
....
2765.          return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex, int[] RedColorFlag)

```
....
2596.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, aryDataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex)

```
....
2586.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, DataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[] HrefShowDataIndex)

```
....
2040.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_2110()

```
....
1770.         PRT.Add3(DOC.Bulletin(varTitle, ""), true, "1,4,5,6",
aryHref, aryShowIndex, aryHrefIndex);
1771.         this.X02Y01.Text = PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.           return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 42:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=717
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_2130在WebCatalog.aspx.cs第1778行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2746	1815
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2746.                    strItem = DR.GetValue(i).ToString();
....
2749.                  strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), ItemAlign + strTdAppAttribute, ""));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.          return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex, int[] RedColorFlag)

```
....
2596.          return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, aryDataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex)

```
....
2586.          return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, DataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[] HrefShowDataIndex)

```
....
2040.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_2130()

```
....
1813.         PRT.Add3(DOC.Query(strSQL), true, "4", aryHref,
aryShowIndex, aryHrefIndex);
....
1815.         this.X02Y01.Text = PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.          return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 43:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=718
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Category在WebCatalog.aspx.cs第358行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2746	564
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2746.                    strItem = DR.GetValue(i).ToString();
....
2749.                  strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), ItemAlign + strTdAppAttribute, ""));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.          return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09

....
564.            this.X02Y02.Text += PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.          return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 44:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=719
狀態 反覆出現的問題

Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Category在WebCatalog.aspx.cs第358行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2746	565
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2746.                    strItem = DR.GetValue(i).ToString();
....
2749.                  strData.Append(string.Format(fmtTd, (strItem ==
"" ? " " : strItem), ItemAlign + strTdAppAttribute, ""));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.          return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09

....
564.            this.X02Y02.Text += PRT.Generate();
565.            this.X02Y02.Text += sTable1;
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.          return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 45:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=720
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_2120在WebCatalog.aspx.cs第1688行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2671	1727
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2671.                   strItem = DR.GetValue(iStart).ToString();
....
2676.               strData.Append(string.Format(fmtTdSpan, (strItem ==
"" ? " " : strItem), ItemAlign + FirstBgColor));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.           return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex, int[] RedColorFlag)

```
....
2596.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, aryDataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex)

```
....
2586.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, DataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[] HrefShowDataIndex)

```
....
2040.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_2120()

```
....
1725.         PRT.Add3(DOC.Query(strSQL), true, "1,4", aryHref,
aryShowIndex, aryHrefIndex);
....
1727.         this.X02Y01.Text = PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.          return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 46:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=721
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_2110在WebCatalog.aspx.cs第1735行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2671	1771
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2671.                    strItem = DR.GetValue(iStart).ToString();
....
2676.                strData.Append(string.Format(fmtTdSpan, (strItem ==
"" ? " " : strItem), ItemAlign + FirstBgColor));
....
2754.              strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.              strText.Replace("@rowspan@", "");
....
2765.            return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex, int[] RedColorFlag)

```
....
2596.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, aryDataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex)

```
....
2586.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, DataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[] HrefShowDataIndex)

```
....
2040.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_2110()

```
....
1770.         PRT.Add3(DOC.Bulletin(varTitle, ""), true, "1,4,5,6",
aryHref, aryShowIndex, aryHrefIndex);
1771.         this.X02Y01.Text = PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.           return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 47:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=722
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_2130在WebCatalog.aspx.cs第1778行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2671	1815
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2671.                   strItem = DR.GetValue(iStart).ToString();
....
2676.               strData.Append(string.Format(fmtTdSpan, (strItem ==
"" ? " " : strItem), ItemAlign + FirstBgColor));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.           return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex, int[] RedColorFlag)

```
....
2596.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, aryDataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex)

```
....
2586.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, DataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[] HrefShowDataIndex)

```
....
2040.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_2130()

```
....
1813.         PRT.Add3(DOC.Query(strSQL), true, "4", aryHref,
aryShowIndex, aryHrefIndex);
....
1815.         this.X02Y01.Text = PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.          return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 48:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=723
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Category在WebCatalog.aspx.cs第358行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2671	564
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2671.                   strItem = DR.GetValue(iStart).ToString();
....
2676.                strData.Append(string.Format(fmtTdSpan, (strItem ==
"" ? " " : strItem), ItemAlign + FirstBgColor));
....
2754.              strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.              strText.Replace("@rowspan@", "");
....
2765.           return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2056.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09

....
564.             this.X02Y02.Text += PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.           return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 49:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=724
狀態 反覆出現的問題

Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Category在WebCatalog.aspx.cs第358行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2671	565
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2671.                   strItem = DR.GetValue(iStart).ToString();
....
2676.               strData.Append(string.Format(fmtTdSpan, (strItem ==
"" ? " " : strItem), ItemAlign + FirstBgColor));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.           return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2056.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09

....
564.            this.X02Y02.Text += PRT.Generate();
565.            this.X02Y02.Text += sTable1;
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.          return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 50:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=725
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_2120在WebCatalog.aspx.cs第1688行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2674	1727
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2674.                strItem = DR.GetValue(iStart).ToString();
....
2676.              strData.Append(string.Format(fmtTdSpan, (strItem ==
"" ? " " : strItem), ItemAlign + FirstBgColor));
....
2754.            strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.            strText.Replace("@rowspan@", "");
....
2765.          return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex, int[] RedColorFlag)

```
....
2596.          return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, aryDataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex)

```
....
2586.          return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, DataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[] HrefShowDataIndex)

```
....
2040.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_2120()

```
....
1725.        PRT.Add3(DOC.Query(strSQL), true, "1,4", aryHref,
aryShowIndex, aryHrefIndex);
....
1727.        this.X02Y01.Text = PRT.Generate();
```

檔案名稱	PageSetting.cs
方法	public string Generate()

```
....
2783.          return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 51:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=726
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_2110在WebCatalog.aspx.cs第1735行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2674	1771
物件	GetValue	Text

代碼片斷

檔案名稱	PageSetting.cs
方法	private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2674.                 strItem = DR.GetValue(iStart).ToString();
....
2676.               strData.Append(string.Format(fmtTdSpan, (strItem ==
"" ? " " : strItem), ItemAlign + FirstBgColor));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.           return string.Format(fmtTable, strText.ToString());
```

檔案名稱	PageSetting.cs
方法	private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex, int[] RedColorFlag)

```
....
2596.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, aryDataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex)

```
....
2586.           return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, DataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[] HrefShowDataIndex)

```
....
2040.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_2110()

```
....
1770.         PRT.Add3(DOC.Bulletin(varTitle, ""), true, "1,4,5,6",
aryHref, aryShowIndex, aryHrefIndex);
1771.         this.X02Y01.Text = PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.           return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 52:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=727
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_2130在WebCatalog.aspx.cs第1778行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2674	1815
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2674.                strItem = DR.GetValue(iStart).ToString();
....
2676.              strData.Append(string.Format(fmtTdSpan, (strItem ==
"" ? " " : strItem), ItemAlign + FirstBgColor));
....
2754.            strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.            strText.Replace("@rowspan@", "");
....
2765.          return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex, int[] RedColorFlag)

```
....
2596.          return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, aryDataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[] DataIndex)

```
....
2586.          return AddItem(DR, WithColumnHead, HiddenIndexs, Href,
HrefIndex, DataIndex, RedColorFlag);
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[] HrefShowDataIndex)

```
....
2040.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_2130()

```
....
1813.         PRT.Add3(DOC.Query(strSQL), true, "4", aryHref,
aryShowIndex, aryHrefIndex);
....
1815.         this.X02Y01.Text = PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.          return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 53:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=728
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Category在WebCatalog.aspx.cs第358行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2674	564
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2674.                strItem = DR.GetValue(iStart).ToString();
....
2676.              strData.Append(string.Format(fmtTdSpan, (strItem ==
"" ? " " : strItem), ItemAlign + FirstBgColor));
....
2754.             strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.             strText.Replace("@rowspan@", "");
....
2765.          return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09

....
564.             this.X02Y02.Text += PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.          return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 54:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=729
狀態 反覆出現的問題

Detection Date	7/8/2022 3:05:10 PM

方法AddItem在PageSetting.cs第2610 行從資訊庫中獲取資訊，做為GetValue元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Category在WebCatalog.aspx.cs第358行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	PageSetting.cs	WebCatalog.aspx.cs
行	2674	565
物件	GetValue	Text

代碼片斷

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2674.                strItem = DR.GetValue(iStart).ToString();
....
2676.              strData.Append(string.Format(fmtTdSpan, (strItem ==
"" ? " " : strItem), ItemAlign + FirstBgColor));
....
2754.            strText.Append(string.Format(fmtTr, ItemBgColor,
strData));
....
2760.            strText.Replace("@rowspan@", "");
....
2765.          return string.Format(fmtTable, strText.ToString());
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09

....
564.            this.X02Y02.Text += PRT.Generate();
565.            this.X02Y02.Text += sTable1;
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.          return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 55:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=730
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_UpdateNewsLetter在WebCatalog.aspx.cs第1863 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Debug在WebCatalog.aspx.cs第3737行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1933	3746
物件	Query	Text

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateNewsLetter(string Mode, string Name, string Mail, string Table, ref string strOrderDoc)

```
....
1933.            DR = this.DOC.Query(strSQL);
....
1936.              strMSG = string.Format("已於 {0} 訂閱。",
DR.GetDateTime(3).ToString("yyyy/MM/dd HH:mm:ss"));
....
1951.        return strMSG;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_NewsLetter(int Action)

```
....
2209.         strB = this.Sub_UpdateNewsLetter(strMode, "最新文件快訊",
strEMailB, "AppKMNewOrder", ref strOrderDoc) + "<br />";
2210.         this.Sub_Debug("strB", strB);
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Debug(string Name, string Value)

```
....
3737.      private void Sub_Debug(string Name, string Value)
....
3744.           strData.Append(string.Format(strItem, Name, Value));
....
3746.             this.Effect.Text += string.Format(strTable,
strData.ToString().Replace("','", "', '"));
```

Heuristic Stored XSS\路徑 56:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=731
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_UpdateNewsLetter在WebCatalog.aspx.cs第1863 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Debug在WebCatalog.aspx.cs第3737行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1933	3748
物件	Query	Text

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateNewsLetter(string Mode, string Name, string Mail, string Table, ref string strOrderDoc)

```
....
1933.            DR = this.DOC.Query(strSQL);
....
1936.              strMSG = string.Format("已於 {0} 訂閱。",
DR.GetDateTime(3).ToString("yyyy/MM/dd HH:mm:ss"));
....
1951.        return strMSG;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_NewsLetter(int Action)

```
....
2209.         strB = this.Sub_UpdateNewsLetter(strMode, "最新文件快訊",
strEMailB, "AppKMNewOrder", ref strOrderDoc) + "<br />";
2210.         this.Sub_Debug("strB", strB);
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Debug(string Name, string Value)

```
....
3737.      private void Sub_Debug(string Name, string Value)
....
3744.           strData.Append(string.Format(strItem, Name, Value));
....
3748.             this.Note.Text += string.Format(strTable,
strData.ToString().Replace("'",'"', "'", '"'));
```

Heuristic Stored XSS\路徑 57:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=732
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_UpdateChiefLetter在WebCatalog.aspx.cs第1955 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Debug在WebCatalog.aspx.cs第3737行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	2021	3746
物件	Query	Text

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateChiefLetter(string Mode, string Name, string Mail, string Title)

```
....
2021.            DR = this.DOC.Query(strSQL);
....
2023.              strMSG = string.Format("已於 {0} 訂閱。",
DR.GetDateTime(3).ToString()) + "," + DR.GetString(1);
....
2029.        return strMSG;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_ChiefLetter(int Action)

```
....
2080.         strA = this.Sub_UpdateChiefLetter(strMode, "首長電子報",
strEMailC, strTitleC);
2081.         this.Sub_Debug("strA", strA);
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Debug(string Name, string Value)

```
....
3737.      private void Sub_Debug(string Name, string Value)
....
3744.           strData.Append(string.Format(strItem, Name, Value));
....
3746.             this.Effect.Text += string.Format(strTable,
strData.ToString().Replace("'",'"', "'", '"'));
```

Heuristic Stored XSS\路徑 58:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=733
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_UpdateChiefLetter在WebCatalog.aspx.cs第1955 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Debug在WebCatalog.aspx.cs第3737行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	2021	3748
物件	Query	Text

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateChiefLetter(string Mode, string Name, string Mail, string Title)

```
....
2021.             DR = this.DOC.Query(strSQL);
....
2023.               strMSG = string.Format("已於 {0} 訂閱。",
DR.GetDateTime(3).ToString()) + "," + DR.GetString(1);
....
2029.         return strMSG;
```

檔案名稱	WebCatalog.aspx.cs
方法	private void Sub_ChiefLetter(int Action)

```
....
2080.         strA = this.Sub_UpdateChiefLetter(strMode, "首長電子報",
strEMailC, strTitleC);
2081.         this.Sub_Debug("strA", strA);
```

檔案名稱	WebCatalog.aspx.cs
方法	private void Sub_Debug(string Name, string Value)

```
....
3737.      private void Sub_Debug(string Name, string Value)
....
3744.           strData.Append(string.Format(strItem, Name, Value));
....
3748.             this.Note.Text += string.Format(strTable,
strData.ToString().Replace("'",'"', "'", '"'));
```

Heuristic Stored XSS\路徑 59:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=734
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_GetKMOrder在WebCatalog.aspx.cs第2295 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Debug在WebCatalog.aspx.cs第3737行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	2305	3746
物件	Query	Text

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_GetKMOrder(string strHideDocDefID)

```
....
2305.         System.Data.SqlClient.SqlDataReader objDR1 =
DOC.Query(strSQL);
2306.         while(objDR1.Read())
....
2308.           strDocDefID = objDR1.GetValue(0).ToString();
2309.           strDocDefName = objDR1.GetValue(1).ToString();
....
2311.           objSB1.Append(string.Format(fmtTR, strDocDefID, strCheck,
strDocDefName));
....
2325.         return string.Format(fmtTB,
objSB1.ToString().Replace("@font2@", DOC._aryFontSize[2]));
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateNewsLetter(string Mode, string Name, string Mail, string Table, ref string strOrderDoc)

```
....
1892.                 strOrderDoc = this.Sub_GetKMOrder("");
....
1897.               strHideDoc = this.Sub_SetKMOrder(strOrderDoc);
....
1912.               strSQL = string.Format(fmtIns, Table, Mail,
WhoAmI.UserID, strDateTime, strHideDoc);
1913.               this.Sub_Debug("Sub_UpdateNewsLetter", strSQL);
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_SetKMOrder(string strDocDefID)

```
....
2328.       private string Sub_SetKMOrder(string strDocDefID)
....
2332.         System.Data.SqlClient.SqlDataReader objDR1 =
DOC.Query(string.Format(strSQL, strDocDefID.Replace(",", "','")));
2333.         while(objDR1.Read())
....
2335.           objSB1.Append(objDR1.GetValue(0).ToString() + ",");
....
2339.         return objSB1.ToString().TrimEnd(',');
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Debug(string Name, string Value)

```
....
3737.       private void Sub_Debug(string Name, string Value)
....
3744.           strData.Append(string.Format(strItem, Name, Value));
....
3746.             this.Effect.Text += string.Format(strTable,
strData.ToString().Replace("','", "', '"));
```

Heuristic Stored XSS\路徑 60:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=735
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_GetKMOrder在WebCatalog.aspx.cs第2295 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Debug在WebCatalog.aspx.cs第3737行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	2305	3748
物件	Query	Text

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetKMOrder(string strHideDocDefID)

```
....
2305.         System.Data.SqlClient.SqlDataReader objDR1 =
DOC.Query(strSQL);
2306.         while(objDR1.Read())
....
2308.           strDocDefID = objDR1.GetValue(0).ToString();
2309.           strDocDefName = objDR1.GetValue(1).ToString();
....
2311.           objSB1.Append(string.Format(fmtTR, strDocDefID, strCheck,
strDocDefName));
....
2325.         return string.Format(fmtTB,
objSB1.ToString().Replace("@font2@", DOC._aryFontSize[2]));
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateNewsLetter(string Mode, string Name, string Mail, string Table, ref string strOrderDoc)

```
....
1892.                 strOrderDoc = this.Sub_GetKMOrder("");
....
1897.               strHideDoc = this.Sub_SetKMOrder(strOrderDoc);
....
1912.               strSQL = string.Format(fmtIns, Table, Mail,
WhoAmI.UserID, strDateTime, strHideDoc);
1913.               this.Sub_Debug("Sub_UpdateNewsLetter", strSQL);
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_SetKMOrder(string strDocDefID)

```
....
2328.      private string Sub_SetKMOrder(string strDocDefID)
....
2332.        System.Data.SqlClient.SqlDataReader objDR1 =
DOC.Query(string.Format(strSQL, strDocDefID.Replace(",", "','")));
2333.        while(objDR1.Read())
....
2335.          objSB1.Append(objDR1.GetValue(0).ToString() + ",");
....
2339.        return objSB1.ToString().TrimEnd(',');
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Debug(string Name, string Value)

```
....
3737.      private void Sub_Debug(string Name, string Value)
....
3744.          strData.Append(string.Format(strItem, Name, Value));
....
3748.            this.Note.Text += string.Format(strTable,
strData.ToString().Replace("','", "', '"));
```

Heuristic Stored XSS\路徑 61:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=736
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_MailList在WebCatalog.aspx.cs第2342 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_NewsLetter在WebCatalog.aspx.cs第2139行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	2360	2289
物件	Query	Text

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private System.Data.SqlClient.SqlDataReader Sub_MailList(string MailClass)

```
....
2360.        return this.DOC.Query(strSQL);
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_NewsLetter(int Action)

```
....
2271.           DR = this.Sub_MailList(strMailClass);
2272.           while(DR.Read())
....
2274.             Ary[i] = string.Format(fmtCell,
DR.GetValue(0).ToString(), DR.GetValue(2).ToString());
....
2277.               PRT.AddItem(Ary, "text-align:left;", 0, j % 2 == 0 ?
"background-color:#f6f6f6;" : "background-color:#ffffd0;");
....
2289.           this.X02Y02.Text = PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public void AddItem(string[] Item, string strAlign, int colLength, string TrBgcolor)

```
....
2116.         public void AddItem(string[] Item, string strAlign, int
colLength, string TrBgcolor)
....
2121.           AddItem(Item, aryAlign, colLength, TrBgcolor);
```

檔案名稱 PageSetting.cs

方法 public void AddItem(string[] Item, string[] aryAlign, int colLength, string TrBgcolor)

```
....
2124.         public void AddItem(string[] Item, string[] aryAlign, int
colLength, string TrBgcolor)
....
2135.             strData.Append(string.Format(fmtTd, (Item[i] == "" ?
" " : Item[i]), aryAlign[i], ""));
....
2142.           varText.Append(string.Format(fmtTr, TrBgcolor, strData));
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.           return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Stored XSS\路徑 62:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=737
狀態 反覆出現的問題

Detection Date	7/8/2022 3:05:10 PM

方法Sub_200在WebCatalog.aspx.cs第2631 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_200在WebCatalog.aspx.cs第2631行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	2700	2788
物件	Query	Text

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_200(bool IsTopLine, bool bOnlySelect)

```
....
2700.           DR = DOC.Query(fmtSQL);
....
2703.             string strTableName = DR.GetValue(0).ToString();
....
2708.               if(AppendCMD[i] == null) { AppendCMD[i] =
this.Sub_BuildAppendSql(strSearchCond, strTableName); break; }
....
2721.           DR = DOC.DocumentList(PSET.RecordsPerPage, DocFields,
strDocNames, aryClass, RequestQuery.Keywords, AttachmentSearchOK ? "" :
(RequestQuery.SearchWords + "," + RequestQuery.HotWords),
RequestQuery.SearchMore, RequestQuery.Companies, RequestQuery.DocIDs,
RequestQuery.StartDate, RequestQuery.EndDate, RequestQuery.Folders,
true, true, (DocOrder)RequestQuery.OrderBy, RequestQuery.LastIndex,
aryClassAnd, AttachCMD, AppendCMD);
....
2788.         this.X02Y05.Text =
DOC.ReplaceWord(DOC.PrintDataReader(fmtListTable, aryFormat, fmtTrColor,
DR, FieldIndex, 0, "", true, 1, "8,12", ref varAttachmentIDs, ref
varAttachmentResults, RequestQuery.PageNumber),
this.RequestQuery.ReplaceWords, "Red", false, false);
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_BuildAppendSql(string strSearchCond, string strTableName)

```
....
6629.       private string Sub_BuildAppendSql(string strSearchCond,
string strTableName)
....
6636.             + "SELECT DocID FROM " + strTableName + " WITH(NOLOCK)
WHERE "
....
6635.           objSQL.Append((objSQL.Length > 0 ? " AND " : "") +
"ZA.DocID " + _objWhereAction[i] + " ("
....
6640.         return objSQL.ToString();
```

Heuristic Stored XSS\路徑 63:

嚴重程度：	低風險

結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=738
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_200在WebCatalog.aspx.cs第2631 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_200在WebCatalog.aspx.cs第2631行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	2700	2789
物件	Query	Text

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_200(bool IsTopLine, bool bOnlySelect)

```
....
2700.           DR = DOC.Query(fmtSQL);
....
2703.             string strTableName = DR.GetValue(0).ToString();
....
2708.               if(AppendCMD[i] == null) { AppendCMD[i] =
this.Sub_BuildAppendSql(strSearchCond, strTableName); break; }
....
2721.           DR = DOC.DocumentList(PSET.RecordsPerPage, DocFields,
strDocNames, aryClass, RequestQuery.Keywords, AttachmentSearchOK ? "" :
(RequestQuery.SearchWords + "," + RequestQuery.HotWords),
RequestQuery.SearchMore, RequestQuery.Companies, RequestQuery.DocIDs,
RequestQuery.StartDate, RequestQuery.EndDate, RequestQuery.Folders,
true, true, (DocOrder)RequestQuery.OrderBy, RequestQuery.LastIndex,
aryClassAnd, AttachCMD, AppendCMD);
....
2788.         this.X02Y05.Text =
DOC.ReplaceWord(DOC.PrintDataReader(fmtListTable, aryFormat, fmtTrColor,
DR, FieldIndex, 0, "", true, 1, "8,12", ref varAttachmentIDs, ref
varAttachmentResults, RequestQuery.PageNumber),
this.RequestQuery.ReplaceWords, "Red", false, false);
2789.         this.X02Y05.Text += (WhoAmI.SerNo == "" ? "" : "<span
style='text-align:left;font-size:smaller;width:100%'>" +
System.DateTime.Now.ToShortDateString() + " " + WhoAmI.SerNo +
"</span>"); // 94/09/27
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_BuildAppendSql(string strSearchCond, string strTableName)

```
....
6629.      private string Sub_BuildAppendSql(string strSearchCond,
string strTableName)
....
6636.             + "SELECT DocID FROM " + strTableName + " WITH(NOLOCK)
WHERE "
....
6635.           objSQL.Append((objSQL.Length > 0 ? " AND " : "") +
"ZA.DocID " + _objWhereAction[i] + " ("
....
6640.         return objSQL.ToString();
```

Heuristic Stored XSS\路徑 64:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=739
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_CountVisit在WebCatalog.aspx.cs第3275 行從資訊庫中獲取資訊，做為Format元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Debug在WebCatalog.aspx.cs第3737行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	3402	3746
物件	Format	Text

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_CountVisit(string SysName, DateTime Date)

```
....
3402.         strSQL = String.Format("SELECT STR(DATEPART(M, LogDate), 2)
+ '.' + STR(DATEPART(YY, LogDate), 4) AS MM, Max(Total) AS Expr1 FROM
LogDateTime WITH(NOLOCK) WHERE (SysName = '" + SysName + "') AND
(datediff(m,LogDate , DATEADD(year, - 2, '{0}'))<0) AND (LogDate <=
'{0}') GROUP BY STR(DATEPART(M, LogDate), 2) + '.' + STR(DATEPART(YY,
LogDate), 4) ORDER BY MM", Date.ToString("yyyy/M/dd"));
....
3406.         this.Sub_Debug("近一年與去年同期月份流量比較", strSQL);
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Debug(string Name, string Value)

```
....
3737.      private void Sub_Debug(string Name, string Value)
....
3744.          strData.Append(string.Format(strItem, Name, Value));
....
3746.            this.Effect.Text += string.Format(strTable,
strData.ToString().Replace("','", "', '"));
```

Heuristic Stored XSS\路徑 65:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=740
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_CountVisit在WebCatalog.aspx.cs第3275 行從資訊庫中獲取資訊，做為Format元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Debug在WebCatalog.aspx.cs第3737行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	3402	3748
物件	Format	Text

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_CountVisit(string SysName, DateTime Date)

```
....
3402.        strSQL = String.Format("SELECT STR(DATEPART(M, LogDate), 2)
+ '.' + STR(DATEPART(YY, LogDate), 4) AS MM, Max(Total) AS Expr1 FROM
LogDateTime WITH(NOLOCK) WHERE (SysName = '" + SysName + "') AND
(datediff(m,LogDate , DATEADD(year, - 2, '{0}'))<0) AND (LogDate <=
'{0}') GROUP BY STR(DATEPART(M, LogDate), 2) + '.' + STR(DATEPART(YY,
LogDate), 4) ORDER BY MM", Date.ToString("yyyy/M/dd"));
....
3406.        this.Sub_Debug("近一年與去年同期月份流量比較", strSQL);
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Debug(string Name, string Value)

```
....
3737.      private void Sub_Debug(string Name, string Value)
....
3744.          strData.Append(string.Format(strItem, Name, Value));
....
3748.            this.Note.Text += string.Format(strTable,
strData.ToString().Replace("','", "', '"));
```

Heuristic Stored XSS\路徑 66:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=741
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_PersonalSetting在WebCatalog.aspx.cs第3904 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_PersonalSetting在WebCatalog.aspx.cs第3904行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	4260	4270
物件	Query	Text

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 public void Sub_PersonalSetting(int Action)

```
....
4260.         objDR1 = this.DOC.Query(string.Format(strSQL3,
WhoAmI.eMail, WhoAmI.UserID));
4261.         while(objDR1.Read())
....
4263.           strLoginDate = objDR1.IsDBNull(0) ? " " :
objDR1.GetDateTime(0).ToString("yyyy/MM/dd HH:mm:ss");
....
4266.           objSB1.Append(string.Format(fmtTr1, strLoginDate,
strLogoutDate, strLogIP));
....
4270.         this.X02Y01.Text += "<br />" + string.Format(fmtTable1,
objSB1.ToString());
```

Heuristic Stored XSS\路徑 67:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=742
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_VerifyUser在WebCatalog.aspx.cs第4323 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_VerifyUser在WebCatalog.aspx.cs第4323行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	4383	4385

物件	Query	Text

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 public void Sub_VerifyUser()

```
....
4383.         System.Data.SqlClient.SqlDataReader objDR1 =
DOC.Query("SELECT TOP 1 LoginDate FROM LogUser WITH(NOLOCK) WHERE UID =
'" + WhoAmI.UserID + "' AND LogKey <> '" + WhoAmI.LogKey + "' ORDER BY
LoginDate DESC;");
....
4385.           LoginMessage.Text += ", <span
class='LastLogin'>上次登入時間：" +
objDR1.GetDateTime(0).ToString("yyyy/MM/dd HH:mm:ss") + "</span>";
```

Heuristic Stored XSS\路徑 68:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=743
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_CheckReview在WebCatalog.aspx.cs第4696 行從資訊庫中獲取資訊，做為Sub_SqlDraft元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Debug在WebCatalog.aspx.cs第3737行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	4704	3746
物件	Sub_SqlDraft	Text

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private void Sub_CheckReview()

```
....
4704.         strSQL = this.Sub_SqlDraft("3");
....
4721.         this.Sub_Debug("草稿文件", strSQL);
```

檔案名稱 WebCatalog.aspx.cs
方法 private void Sub_Debug(string Name, string Value)

```
....
3737.      private void Sub_Debug(string Name, string Value)
....
3744.          strData.Append(string.Format(strItem, Name, Value));
....
3746.            this.Effect.Text += string.Format(strTable,
strData.ToString().Replace("'",'"', "'", '"'));
```

Heuristic Stored XSS\路徑 69:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=744
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_CheckReview在WebCatalog.aspx.cs第4696 行從資訊庫中獲取資訊，做為Sub_SqlDraft元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Debug在WebCatalog.aspx.cs第3737行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	4704	3748
物件	Sub_SqlDraft	Text

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_CheckReview()

```
....
4704.        strSQL = this.Sub_SqlDraft("3");
....
4721.        this.Sub_Debug("草稿文件", strSQL);
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Debug(string Name, string Value)

```
....
3737.      private void Sub_Debug(string Name, string Value)
....
3744.          strData.Append(string.Format(strItem, Name, Value));
....
3748.            this.Note.Text += string.Format(strTable,
strData.ToString().Replace("'",'"', "'", '"'));
```

Heuristic Stored XSS\路徑 70:

嚴重程度：	低風險
結果狀態：	校驗

線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=745
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_CheckReview在WebCatalog.aspx.cs第4696 行從資訊庫中獲取資訊，做為Sub_SqlDraft元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Debug在WebCatalog.aspx.cs第3737行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	4723	3746
物件	Sub_SqlDraft	Text

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_CheckReview()

```
....
4723.         strSQL = this.Sub_SqlDraft("1");
....
4740.         this.Sub_Debug("陳核中文件", strSQL);
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Debug(string Name, string Value)

```
....
3737.      private void Sub_Debug(string Name, string Value)
....
3744.          strData.Append(string.Format(strItem, Name, Value));
....
3746.            this.Effect.Text += string.Format(strTable,
strData.ToString().Replace("','", "', '"));
```

Heuristic Stored XSS\路徑 71:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=746
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_CheckReview在WebCatalog.aspx.cs第4696 行從資訊庫中獲取資訊，做為Sub_SqlDraft元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_Debug在WebCatalog.aspx.cs第3737行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs

行	4723	3748
物件	Sub_SqlDraft	Text

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_CheckReview()

```
....
4723.         strSQL = this.Sub_SqlDraft("1");
....
4740.         this.Sub_Debug("陳核中文件", strSQL);
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Debug(string Name, string Value)

```
....
3737.       private void Sub_Debug(string Name, string Value)
....
3744.           strData.Append(string.Format(strItem, Name, Value));
....
3748.             this.Note.Text += string.Format(strTable,
strData.ToString().Replace("','", "', '"));
```

Heuristic Stored XSS\路徑 72:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=747
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Build_CondGrid在WebCatalog.aspx.cs第6466 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Build_CondGrid在WebCatalog.aspx.cs第6466行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	6471	6621
物件	Query	Text

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private bool Build_CondGrid(string strDocDefName)

```
....
6471.         System.Data.SqlClient.SqlDataReader objDR1 =
objDB1.Query(fmtSQL);
....
6477.           strCond = objDR1.GetValue(1).ToString().Replace(" ", "");
....
6489.         aryCond = strCond.Replace("||", "\u0001").Split('\u0001');
....
6512.           aryField = aryCond[i].Split(',');
....
6617.             + "</div>", strCol, aryField[3]));
....
6615.           objSB1.Append(string.Format("<div id='CnDivArea_{0}'
class='divFloat CnDivArea' style='text-align: {1};'>"
....
6621.         CnCondData.Text = objSB1.ToString();
```

Heuristic Stored XSS\路徑 73:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=748
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法BindData在WebDocumentLog.aspx.cs第52 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於BindData在WebDocumentLog.aspx.cs第52行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebDocumentLog.aspx.cs	WebDocumentLog.aspx.cs
行	65	67
物件	Query	DataSource

代碼片斷

檔案名稱 WebDocumentLog.aspx.cs

方法 private void BindData()

```
....
65.         System.Data.DataTable tb = objDoc.Query(string.Format(fmtSQL,
_sDocID), "LogEvent");
....
67.         DataGrid1.DataSource = tb;
```

Heuristic Stored XSS\路徑 74:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=749
狀態	反覆出現的問題
Detection Date	11/21/2022 1:34:42 PM

方法Page_Load在WebDownloadFiles.aspx.cs第33 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Page_Load在WebDownloadFiles.aspx.cs第33行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebDownloadFiles.aspx.cs	WebDownloadFiles.aspx.cs
行	81	94
物件	Query	Text

代碼片斷

檔案名稱 WebDownloadFiles.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
81.           objDR0 = objDB0.Query("SELECT DocXML FROM DocCatalog
WITH(NOLOCK) WHERE DocID = '" + varDocID + "';");
....
41.         varDocID = GetRequest("DocID", "");
....
43.         varLogKey = GetRequest("LogKey", "\u0001");
....
76.           objDoc = new Cdsys.KM.Utility.Doc(varConnection, this);
....
92.                   + "window.location.href=\"" +
objDoc.cdsUrlEncode(strURL1) + "\";\n"
....
91.               strURL2 = "<script>\n"
....
94.               RunScript.Text = strURL2;
```

檔案名稱 WebDownloadFiles.aspx

方法

```
....
34.
```

Heuristic Stored XSS\路徑 75:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=750
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Page_Load在WebEditor.aspx.cs第115 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Page_Load在WebEditor.aspx.cs第115行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs

行	283	587
物件	Query	Text

代碼片斷
檔案名稱 WebEditor.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
283.          objDR1 = objDoc.Query(objSqlCommand, 30);
....
288.            strKmEditorRemark =
objDR1.GetValue(2).ToString().Trim();
....
587.            TitleRemark.Text = strKmEditorRemark;
```

Heuristic Stored XSS\路徑 76:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=751
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

方法Page_Load在WebEditor.aspx.cs第115 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Page_Load在WebEditor.aspx.cs第115行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	283	581
物件	Query	strJava

代碼片斷
檔案名稱 WebEditor.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
283.          objDR1 = objDoc.Query(objSqlCommand, 30);
....
287.            strInitFunc = objDR1.GetValue(1).ToString().Trim();
....
580.          string strJava = "<script>function InitialFuntion(){try{"
+ strInitFunc + "}catch(err){}}</script>";
581.          this.ClientScript.RegisterStartupScript(this.GetType(),
"InitialFunction", strJava);
```

Heuristic Stored XSS\路徑 77:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=1

	0300&pathid=752
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Page_Load在WebEditor.aspx.cs第115 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Page_Load在WebEditor.aspx.cs第115行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	305	425
物件	Query	Text

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
305.            objDR1 = objDoc.Query(objSqlCommand, 30);
....
308.              _sDocType = objDR1.GetValue(0).ToString();
....
425.              Literal1.Text = XMLGenRef(ref oXml);
```

檔案名稱 WebEditor.aspx.cs

方法 private string XMLGenRef(ref System.Xml.XmlDocument oXml)

```
....
1210.         aCategoryList = GetDocParams(sSQL);
....
1213.         aFloderList = GetDocParams(sSQL);
....
1215.         XmlList list1 = new XmlList("XmlList1", "95%", "",
"style='font-size:16px;'", "bgColor=\"#ccff99\"", _sDocType,
objDoc.User);
....
1294.         return list1.Html();
```

Heuristic Stored XSS\路徑 78:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=753
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Page_Load在WebEditor.aspx.cs第115 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Page_Load在WebEditor.aspx.cs第115行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	305	482
物件	Query	Text

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
305.            objDR1 = objDoc.Query(objSqlCommand, 30);
....
308.              _sDocType = objDR1.GetValue(0).ToString();
....
472.                  Literal1.Text = ListAutoGen();
....
482.          Literal1.Text += "<div style='font-size: 0px;
height:5px;'></div>";
```

檔案名稱 WebEditor.aspx.cs

方法 private string ListAutoGen()

```
....
1108.         XmlList list1 = new XmlList("XmlList1", "95%", "",
"style='font-size:16px;'", "bgColor=\"#ccff99\"", _sDocType,
objDoc.User);
....
1142.         return list1.Html();
```

Heuristic Stored XSS\路徑 79:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=754
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Page_Load在WebEditor.aspx.cs第115 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Page_Load在WebEditor.aspx.cs第115行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	305	441
物件	Query	Text

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
305.            objDR1 = objDoc.Query(objSqlCommand, 30);
....
308.              _sDocType = objDR1.GetValue(0).ToString();
....
431.                string sTemp = GetRequest("XML", "");
....
441.                  Literal1.Text = XMLGenRef(ref oXml);
```

檔案名稱	WebEditor.aspx.cs
方法	private string GetRequest(string strName, string strDefault)

```
....
90.         string strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.Form[strNam
e], false), false));
```

檔案名稱	WebEditor.aspx.cs
方法	private string XMLGenRef(ref System.Xml.XmlDocument oXml)

```
....
1210.         aCategoryList = GetDocParams(sSQL);
....
1213.         aFloderList = GetDocParams(sSQL);
....
1215.         XmlList list1 = new XmlList("XmlList1", "95%", "",
"style='font-size:16px;'", "bgColor=\"#ccff99\"", _sDocType,
objDoc.User);
....
1294.         return list1.Html();
```

Heuristic Stored XSS\路徑 80:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=755
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Page_Load在WebEditor.aspx.cs第115 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Page_Load在WebEditor.aspx.cs第115行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	305	445
物件	Query	Text

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
305.            objDR1 = objDoc.Query(objSqlCommand, 30);
....
308.              _sDocType = objDR1.GetValue(0).ToString();
....
431.                string sTemp = GetRequest("XML", "");
....
445.                  Literal1.Text = ListAutoGen();
```

檔案名稱 WebEditor.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
90.         string strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.Form[strNam
e], false), false));
```

檔案名稱 WebEditor.aspx.cs

方法 private string ListAutoGen()

```
....
1108.         XmlList list1 = new XmlList("XmlList1", "95%", "",
"style='font-size:16px;'", "bgColor=\"#ccff99\"", _sDocType,
objDoc.User);
....
1142.         return list1.Html();
```

Heuristic Stored XSS\路徑 81:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=756
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Page_Load在WebEditor.aspx.cs第115 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Page_Load在WebEditor.aspx.cs第115行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	305	468
物件	Query	Text

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
305.            objDR1 = objDoc.Query(objSqlCommand, 30);
....
308.              _sDocType = objDR1.GetValue(0).ToString();
....
468.                  Literal1.Text = XMLGenRef(ref oXml);
```

檔案名稱 WebEditor.aspx.cs

方法 private string XMLGenRef(ref System.Xml.XmlDocument oXml)

```
....
1210.         aCategoryList = GetDocParams(sSQL);
....
1213.         aFloderList = GetDocParams(sSQL);
....
1215.         XmlList list1 = new XmlList("XmlList1", "95%", "",
"style='font-size:16px;'", "bgColor=\"#ccff99\"", _sDocType,
objDoc.User);
....
1294.         return list1.Html();
```

Heuristic Stored XSS\路徑 82:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=757
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Page_Load在WebEditor.aspx.cs第115 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Page_Load在WebEditor.aspx.cs第115行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	305	472
物件	Query	Text

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
305.            objDR1 = objDoc.Query(objSqlCommand, 30);
....
308.              _sDocType = objDR1.GetValue(0).ToString();
....
472.                  Literal1.Text = ListAutoGen();
```

檔案名稱 WebEditor.aspx.cs

方法 private string ListAutoGen()

```
....
1108.         XmlList list1 = new XmlList("XmlList1", "95%", "",
"style='font-size:16px;'", "bgColor=\"#ccff99\"", _sDocType,
objDoc.User);
....
1142.         return list1.Html();
```

Heuristic Stored XSS\路徑 83:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=758
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Page_Load在WebEditor.aspx.cs第115 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Page_Load在WebEditor.aspx.cs第115行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	305	479
物件	Query	Text

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
305.            objDR1 = objDoc.Query(objSqlCommand, 30);
....
308.              _sDocType = objDR1.GetValue(0).ToString();
....
479.            Literal1.Text += ListAutoGen();
```

檔案名稱 WebEditor.aspx.cs

方法 private string ListAutoGen()

```
....
1108.         XmlList list1 = new XmlList("XmlList1", "95%", "",
"style='font-size:16px;'", "bgColor=\"#ccff99\"", _sDocType,
objDoc.User);
....
1142.         return list1.Html();
```

Heuristic Stored XSS\路徑 84:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=759
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Page_Load在WebEditor.aspx.cs第115 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Page_Load在WebEditor.aspx.cs第115行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	328	368
物件	Query	strJavaData

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
328.                 objDR1 = objDoc.Query(string.Format(strSQL1,
CSS(txt_DocID.Value), objDoc.User.CompanyCode,
objDoc.User.DepartmentCode));
....
331.                   strMsg =
objDR1.GetDateTime(0).ToString("yyyy/MM/dd HH:mm:ss");
332.                   strMsg = "�Q��촧b " + strMsg + "
�2s�L,�O_�~��2s�H";
....
335.                   strConfirm += "ret=window.confirm('" + strMsg +
"'); if(ret != true){break;}\n";
....
359.                   strJavaData += strConfirm;
....
368.
this.ClientScript.RegisterStartupScript(this.GetType(), "CheckYesNo",
strJavaData);
```

Heuristic Stored XSS\路徑 85:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=1

	0300&pathid=760
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_GoodMemo在WebLogon.aspx.cs第296 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_GoodMemo在WebLogon.aspx.cs第296行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebLogon.aspx.cs	WebLogon.aspx.cs
行	302	312
物件	Query	Text

代碼片斷	
檔案名稱	WebLogon.aspx.cs
方法	private void Sub_GoodMemo()

```
....
302.         System.Data.SqlClient.SqlDataReader objDR1 =
objDoc.Query(strSQL1);
....
306.           strMemoName = objDR1.IsDBNull(0) ? "" :
objDR1.GetString(0).Trim();
....
311.           string strTemp1 = fmtHead + string.Format(fmtData,
strMemoName, strMemoContent);
312.           X02Y01.Text = string.Format(fmtTable, strTemp1);
```

Heuristic Stored XSS\路徑 86:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=761
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Page_Load在WebMailA.aspx.cs第20 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Page_Load在WebMailA.aspx.cs第20行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebMailA.aspx.cs	WebMailA.aspx.cs
行	59	183
物件	Query	strJava

代碼片斷	
檔案名稱	WebMailA.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
59.         objDR1 = objDB1.Query(string.Format(strSQL1, strDocID));
....
63.           string strEmailFields = objDR1.GetValue(1).ToString();
....
78.                strBody = objDoc.PrintDocument(strDocID, strDocXML, new
string[0], "", true, "", strEmailFields);
....
156.             strErrMsg = objDB1.SendMail(strSmtpServer, strMailFrom,
strMailTo, strSubject, strBody, true);
....
162.               objDB1.Execute(string.Format(fmtSQL, strMailFrom,
strMailTo.Replace("'", "''"), "SendMsgError", strErrMsg.Replace("'",
"''")));
....
182.         strJava = "<script>" + strJava + "window.alert('" +
strErrMsg.Replace("'", "\\'") + "');window.opener = null;
window.open('', '_self'); window.close();</script>";
183.         this.ClientScript.RegisterClientScriptBlock(this.GetType(),
"cdsAlert", strJava);
```

Heuristic Stored XSS\路徑 87:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=762
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Page_Load在WebMailA.aspx.cs第20 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Page_Load在WebMailA.aspx.cs第20行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebMailA.aspx.cs	WebMailA.aspx.cs
行	131	183
物件	Query	strJava

代碼片斷	
檔案名稱	WebMailA.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
131.                objDR1 = objDB1.Query(string.Format(strSQL1,
aryCompany[i]));
....
134.                  string[] aryTemp1 =
objDR1.GetValue(0).ToString().Replace("&#65533;A", ",").Split(',');
....
137.                    string strTemp1 = aryTemp1[iAddrPos].Trim();
138.                    if(strTemp1 != "" && strTemp1.IndexOf("@") >= 0)
strMailTo += strTemp1 + ";";
....
144.             strMailTo = strMailTo.Trim(';');
....
156.           strErrMsg = objDB1.SendMail(strSmtpServer, strMailFrom,
strMailTo, strSubject, strBody, true);
....
162.             objDB1.Execute(string.Format(fmtSQL, strMailFrom,
strMailTo.Replace("'", "''"), "SendMsgError", strErrMsg.Replace("'",
"''")));
....
182.        strJava = "<script>" + strJava + "window.alert('" +
strErrMsg.Replace("'", "\\'") + "');window.opener = null;
window.open('', '_self'); window.close();</script>";
183.        this.ClientScript.RegisterClientScriptBlock(this.GetType(),
"cdsAlert", strJava);
```

Heuristic Stored XSS\路徑 88:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=763
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法BuildBookList在WebMark.aspx.cs第76 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於BuildBookList在WebMark.aspx.cs第76行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebMark.aspx.cs	WebMark.aspx.cs
行	81	82
物件	Query	Text

代碼片斷

檔案名稱 WebMark.aspx.cs

方法 private void BuildBookList()

```
....
81.         objDR1 = objDB1.Query(string.Format("SELECT DocTitle FROM
DocCatalog WITH(NOLOCK) WHERE DocID = '{0}';", varDocID));
82.         if(objDR1.Read()) fBookTitle.Text =
objDR1.GetValue(0).ToString();
```

Heuristic Stored XSS\路徑 89:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=764
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法BuildGroupList在WebMark.aspx.cs第95 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於BuildGroupList在WebMark.aspx.cs第95行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebMark.aspx.cs	WebMark.aspx.cs
行	100	101
物件	Query	Text

代碼片斷	
檔案名稱	WebMark.aspx.cs
方法	private void BuildGroupList()

```
....
100.         objDR1 = objDB1.Query(string.Format("SELECT DocTitle FROM
DocCatalog WITH(NOLOCK) WHERE DocID = '{0}';", varDocID));
101.         if(objDR1.Read()) fGroupTitle.Text =
objDR1.GetValue(0).ToString();
```

Heuristic Stored XSS\路徑 90:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=765
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_SAY在WebNotePad.aspx.cs第190 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_SAY在WebNotePad.aspx.cs第190行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	239	242
物件	Query	Text

代碼片斷	
檔案名稱	WebNotePad.aspx.cs
方法	private void Sub_SAY(string Action, string SysName, string ID, string Title, string Essay, string StartDate, string EndDate, string Commentator, string Chk)

```
....
239.            DR = this.objDoc.Query(strSQL);
....
242.              this.Note.Text = this.Sub_ShowSAY(DR.GetString(0),
DR.GetString(1), DR.GetString(5), "", "", false,
DR.GetValue(6).ToString());
```

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_ShowSAY(string Title, string Essay, string Commentator, string StartDate, string EndDate, bool ReadOnly, string Chk)

```
....
522.      private string Sub_ShowSAY(string Title, string Essay, string
Commentator, string StartDate, string EndDate, bool ReadOnly, string
Chk)
....
534.        strData.Append(string.Format(fmtItemA, "�i�K��",
"txtCommentator", Commentator, "ReadOnly"));
....
541.        return string.Format(fmtTable, "�p�~�Y��", strData);
```

Heuristic Stored XSS\路徑 91:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=766
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_SAY在WebNotePad.aspx.cs第190 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_SAY在WebNotePad.aspx.cs第190行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	262	265
物件	Query	Text

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_SAY(string Action, string SysName, string ID, string Title, string Essay, string StartDate, string EndDate, string Commentator, string Chk)

```
....
262.            DR = this.objDoc.Query(strSQL);
....
265.              this.Note.Text = this.Sub_ShowSAY(DR.GetString(0),
DR.GetString(1), DR.GetString(5), "", "", true,
DR.GetValue(6).ToString());
```

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_ShowSAY(string Title, string Essay, string Commentator, string StartDate, string EndDate, bool ReadOnly, string Chk)

```
....
522.      private string Sub_ShowSAY(string Title, string Essay, string
Commentator, string StartDate, string EndDate, bool ReadOnly, string
Chk)
....
534.        strData.Append(string.Format(fmtItemA, "�i�K��",
"txtCommentator", Commentator, "ReadOnly"));
....
541.        return string.Format(fmtTable, "�p�~�Y��", strData);
```

Heuristic Stored XSS\路徑 92:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=767
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_GBK在WebNotePad.aspx.cs第287 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_GBK在WebNotePad.aspx.cs第287行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	337	340
物件	Query	Text

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_GBK(string Action, string SysName, string ID, string BookName, string BookStore, string BookContent, string StartDate, string EndDate, string Commentator, string BookAuthor)

```
....
337.            DR = this.objDoc.Query(strSQL);
....
340.              this.Note.Text = this.Sub_ShowGBK(DR.GetString(0),
DR.GetString(1), DR.GetString(2), DR.GetString(8),
DR.GetDateTime(3).ToString("yyyy/MM/dd"),
DR.GetDateTime(4).ToString("yyyy/MM/dd"), DR.IsDBNull(9) ? "" :
DR.GetString(9), false);
```

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_ShowGBK(string Name, string Store, string Summary, string Commentator, string StartDate, string EndDate, string Athor, bool ReadOnly)

```
....
544.      private string Sub_ShowGBK(string Name, string Store, string
Summary, string Commentator, string StartDate, string EndDate, string
Athor, bool ReadOnly)
....
559.        strData.Append(string.Format(fmtItemC, "���G����",
"txtStartDate", StartDate, strReadOnly, "txtEndDate", EndDate));
....
561.        return string.Format(fmtTable, "�n�ψ���", strData);
```

Heuristic Stored XSS\路徑 93:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=768
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_GBK在WebNotePad.aspx.cs第287 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_GBK在WebNotePad.aspx.cs第287行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	360	363
物件	Query	Text

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_GBK(string Action, string SysName, string ID, string BookName, string BookStore, string BookContent, string StartDate, string EndDate, string Commentator, string BookAuthor)

```
....
360.            DR = this.objDoc.Query(strSQL);
....
363.              this.Note.Text = this.Sub_ShowGBK(DR.GetString(0),
DR.GetString(1), DR.GetString(2), DR.GetString(8),
DR.GetDateTime(3).ToString("yyyy/MM/dd"),
DR.GetDateTime(4).ToString("yyyy/MM/dd"), DR.IsDBNull(9) ? "" :
DR.GetString(9), true);
```

檔案名稱	WebNotePad.aspx.cs
方法	private string Sub_ShowGBK(string Name, string Store, string Summary, string Commentator, string StartDate, string EndDate, string Athor, bool ReadOnly)

```
....
544.      private string Sub_ShowGBK(string Name, string Store, string
Summary, string Commentator, string StartDate, string EndDate, string
Athor, bool ReadOnly)
....
559.        strData.Append(string.Format(fmtItemC, "���G����",
"txtStartDate", StartDate, strReadOnly, "txtEndDate", EndDate));
....
561.        return string.Format(fmtTable, "�n�ψ���", strData);
```

Heuristic Stored XSS\路徑 94:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=769
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法BindData在WebPrint.aspx.cs第244 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於btn_SelAll_Click在WebPrint.aspx.cs第840行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	251	845
物件	Query	DataSource

代碼片斷	
檔案名稱	WebPrint.aspx.cs
方法	private void BindData(bool bBindData)

```
....
251.        System.Data.SqlClient.SqlDataReader objReader1 =
objDoc.Query(strSQL1);
252.        while(objReader1.Read())
....
254.          if(objReader1.IsDBNull(0) || objReader1.IsDBNull(1)) { }
....
257.            strDocDefID = objReader1.GetString(0);
....
285.          string strXmlSql = objDoc.SearchDocsSql(_sQueryXml, out
strDocDefID);
....
287.          strAppCommand += " AND (ZA.DocDefID = '" + strDocDefID +
"')";
....
298.          string strWhere = " WHERE ZA.DocState_i = '1'" +
(strXmlSql == "" ? "" : " AND " + strXmlSql) + strAppCommand + " ORDER
BY " + _sOrderField;
....
303.          sSQL = "SELECT " + strFields + strFrom + strWhere;
....
309.        tb = objDoc.Query(sSQL, "PrintCatalog");
....
311.        tb1 = tb.Clone(); // 2s@MM tb P c table
(]t)
....
366.            DataRow myRowNew = tb1.NewRow();
....
374.        myView = new DataView(tb1);
```

檔案名稱 WebPrint.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
103.            BindData(true);
....
106.          Initial();
```

檔案名稱 WebPrint.aspx.cs

方法 private void Initial()

```
....
203.        LoadSelect();
```

檔案名稱 WebPrint.aspx

方法

```
....
154.
```

檔案名稱	WebPrint.aspx.cs
方法	protected void btn_SelAll_Click(object sender, System.EventArgs e)

```
....
845.         DataGrid2.DataSource = myView;
```

Heuristic Stored XSS\路徑 95:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=770
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法BindData在WebPrint.aspx.cs第244 行從資訊庫中獲取資訊，做為Query元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於BindData在WebPrint.aspx.cs第244行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	251	378
物件	Query	DataSource

代碼片斷	
檔案名稱	WebPrint.aspx.cs
方法	private void BindData(bool bBindData)

```
....
251.         System.Data.SqlClient.SqlDataReader objReader1 =
objDoc.Query(strSQL1);
252.         while(objReader1.Read())
....
254.           if(objReader1.IsDBNull(0) || objReader1.IsDBNull(1)) { }
....
257.             strDocDefID = objReader1.GetString(0);
....
285.           string strXmlSql = objDoc.SearchDocsSql(_sQueryXml, out
strDocDefID);
....
287.           strAppCommand += " AND (ZA.DocDefID = '" + strDocDefID +
"')";
....
298.           string strWhere = " WHERE ZA.DocState_i = '1'" +
(strXmlSql == "" ? "" : " AND " + strXmlSql) + strAppCommand + " ORDER
BY " + _sOrderField;
....
303.           sSQL = "SELECT " + strFields + strFrom + strWhere;
....
309.         tb = objDoc.Query(sSQL, "PrintCatalog");
....
311.         tb1 = tb.Clone(); // ?2s?@???M tb ?'P???c?? table
(???]?t???)
....
374.         myView = new DataView(tb1);
....
378.           DataGrid1.DataSource = myView;
```

Heuristic Stored XSS\路徑 96:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=771
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_PersonalSetting在WebCatalog.aspx.cs第3904 行從資訊庫中獲取資訊，做為Insert元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_PersonalSetting在WebCatalog.aspx.cs第3904行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	4232	4270
物件	Insert	Text

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	public void Sub_PersonalSetting(int Action)

```
....
4232.        objSB1.Insert(0, "<SELECT NAME='listDocItems90'
ID='listDocItems90'
onkeypress=\"javascript:switch(window.event.keyCode){case
43:MoveDnOptionItemA('listDocItems90');break;case
45:MoveUpOptionItemA('listDocItems90');break;}try { event.preventDefault
? event.preventDefault() : (event.returnValue = false); } catch (err) {
return false; }\" SIZE=" + k.ToString() + ">");
4233.        objSB1.Append("</SELECT>");
4234.        objSB1.Append("<INPUT type='button' style='cursor:pointer;'
name='FilesUp_1' id ='FilesUp_1' value='上移(-)'
ondblclick=this.onclick()
onclick=\"javascript:MoveUpOptionItemA('listDocItems90');\" />");
4235.        objSB1.Append("<INPUT type='button' style='cursor:pointer;'
name='FilesDn_1' id ='FilesDn_1' value='下移(+)'
ondblclick=this.onclick()
onclick=\"javascript:MoveDnOptionItemA('listDocItems90');\" />");
4236.        objSB1.Append("<INPUT type='hidden' name='txtDocItem90'
id='txtDocItem90' value='' />");
4237.        PRT.AddItem(new string[] { "文件類型順序：",
objSB1.ToString(), "" }, aryAlign, 0, "background-color:#e8e8ff;");
....
4257.        objSB1.Append(string.Format(fmtTr1, "登入時間", "登出時間",
"IP"));
....
4270.        this.X02Y01.Text += "<br />" + string.Format(fmtTable1,
objSB1.ToString());
```

Heuristic Stored XSS\路徑 97:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=772
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

方法Sub_PersonalSetting在WebCatalog.aspx.cs第3904 行從資訊庫中獲取資訊，做為Insert元素值。而程式流程中沒有被正確地過濾(Filter)或編碼(Encode)，並最終於Sub_PersonalSetting在WebCatalog.aspx.cs第3904行顯示在使用者端。這可以引發儲存的跨站腳本(Store Cross-Site-Scripting)攻擊。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	4232	4243
物件	Insert	Text

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	public void Sub_PersonalSetting(int Action)

```
....
4232.        objSB1.Insert(0, "<SELECT NAME='listDocItems90'
ID='listDocItems90'
onkeypress=\"javascript:switch(window.event.keyCode){case
43:MoveDnOptionItemA('listDocItems90');break;case
45:MoveUpOptionItemA('listDocItems90');break;}try { event.preventDefault
? event.preventDefault() : (event.returnValue = false); } catch (err) {
return false; }\" SIZE=" + k.ToString() + ">");
4233.        objSB1.Append("</SELECT>");
4234.        objSB1.Append("<INPUT type='button' style='cursor:pointer;'
name='FilesUp_1' id ='FilesUp_1' value='上移(-)'
ondblclick=this.onclick()
onclick=\"javascript:MoveUpOptionItemA('listDocItems90');\" />");
4235.        objSB1.Append("<INPUT type='button' style='cursor:pointer;'
name='FilesDn_1' id ='FilesDn_1' value='下移(+)'
ondblclick=this.onclick()
onclick=\"javascript:MoveDnOptionItemA('listDocItems90');\" />");
4236.        objSB1.Append("<INPUT type='hidden' name='txtDocItem90'
id='txtDocItem90' value='' />");
4237.        PRT.AddItem(new string[] { "文件類型順序：",
objSB1.ToString(), "" }, aryAlign, 0, "background-color:#e8e8ff;");
....
4239.        PRT.AddFoot("<input type=\"button\" name=\"btnSave\"
class=PRT_Button style='color:#0000cc' value=\"儲存\"
onclick=\"javascript:document.WebCatalog.Action.value=710;GetListData('l
istDocItems90','txtDocItem90');document.WebCatalog.submit();\" />"
....
4243.        this.X02Y01.Text += PRT.Generate();
```

檔案名稱 PageSetting.cs

方法 public void AddItem(string[] Item, string[] aryAlign, int colLength, string TrBgcolor)

```
....
2124.        public void AddItem(string[] Item, string[] aryAlign, int
colLength, string TrBgcolor)
....
2135.            strData.Append(string.Format(fmtTd, (Item[i] == "" ?
" " : Item[i]), aryAlign[i], ""));
....
2142.          varText.Append(string.Format(fmtTr, TrBgcolor, strData));
```

檔案名稱 PageSetting.cs

方法 public string Generate()

```
....
2783.          return string.Format(fmtTable, varText.Replace("@font2@",
objDoc._aryFontSize[2]));
```

Heuristic Parameter Tampering

查詢路徑:

CSharp\Cx\CSharp Heuristic\Heuristic Parameter Tampering 版本:1

類別

PCI DSS v3.2.1: PCI DSS (3.2.1) - 6.5.1 - Injection flaws - particularly SQL injection
OWASP Top 10 2013: A4-Insecure Direct Object References
OWASP Top 10 2017: A5-Broken Access Control
OWASP Top 10 2021: A4-Insecure Design

描述

Heuristic Parameter Tampering\路徑 1:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=347
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

ShowDocument.aspx.cs 中第 31 行的 Page_Load 方法從 QueryString_DocID 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 ShowDocument.aspx.cs 中第 31 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	ShowDocument.aspx.cs	ShowDocument.aspx.cs
行	46	116
物件	QueryString_DocID	Query

代碼片斷

檔案名稱 ShowDocument.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
46.           strA =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g["DocID"], false));
47.           _sDocID = (strA == null) ? "" : strA.Trim();
....
116.           objReader1 = objDB1.Query("SELECT DocDefID, DocTitle,
DocXML FROM DocCatalog WITH(NOLOCK) WHERE DocID = '" + _sDocID + "'");
```

Heuristic Parameter Tampering\路徑 2:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=348
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

ShowLogDocument.aspx.cs 中第 21 行的 Page_Load 方法從 QueryString_ID 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 ShowLogDocument.aspx.cs 中第 21 行的 Page_Load 方法用來與 Format 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	ShowLogDocument.aspx.cs	ShowLogDocument.aspx.cs
行	25	27
物件	QueryString_ID	Format

代碼片斷
檔案名稱 ShowLogDocument.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
25.         string strID =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g["ID"], false));
....
27.         string strSQL = String.Format("SELECT EvtNote, EvtDocID FROM
LogEvent WHERE EvtID ={0}", strID);
```

Heuristic Parameter Tampering\路徑 3:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=349
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebAttachLink.aspx.cs 中第 96 行的 GetRequest 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebAttachLink.aspx.cs 中第 23 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebAttachLink.aspx.cs	WebAttachLink.aspx.cs
行	100	67
物件	Form	Query

代碼片斷
檔案名稱 WebAttachLink.aspx.cs
方法 private string GetRequest(string strName, string strDefault)

```
....
100.          strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
105.        return strResult.Trim();
```

檔案名稱	WebAttachLink.aspx.cs
方法	protected void Page_Load(object sender, EventArgs e)

```
....
28.         varDocID = GetRequest("DocID", "");
....
30.         varLogKey = GetRequest("LogKey", "\u0001");
....
67.           objDR0 = objDB0.Query("SELECT DocXML FROM DocCatalog
WITH(NOLOCK) WHERE DocID = '" + varDocID + "';");
```

Heuristic Parameter Tampering\路徑 4:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=350
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebAttachLink.aspx.cs 中第 96 行的 GetRequest 方法從 QueryString_strName 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebAttachLink.aspx.cs 中第 23 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebAttachLink.aspx.cs	WebAttachLink.aspx.cs
行	103	67
物件	QueryString_strName	Query

代碼片斷

檔案名稱	WebAttachLink.aspx.cs
方法	private string GetRequest(string strName, string strDefault)

```
....
103.            strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
105.         return strResult.Trim();
```

檔案名稱	WebAttachLink.aspx.cs
方法	protected void Page_Load(object sender, EventArgs e)

```
....
28.         varDocID = GetRequest("DocID", "");
....
30.         varLogKey = GetRequest("LogKey", "\u0001");
....
67.           objDR0 = objDB0.Query("SELECT DocXML FROM DocCatalog
WITH(NOLOCK) WHERE DocID = '" + varDocID + "';");
```

Heuristic Parameter Tampering\路徑 5:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=351
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebBatchPrint.aspx.cs 中第 129 行的 GetRequest 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebBatchPrint.aspx.cs 中第 61 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebBatchPrint.aspx.cs	WebBatchPrint.aspx.cs
行	133	86
物件	Form	Query

代碼片斷

檔案名稱 WebBatchPrint.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
133.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
138.         return DFI(strResult.Trim());
```

檔案名稱 WebBatchPrint.aspx.cs

方法 private string DFI(string strData)

```
....
1124.       private string DFI(string strData)
....
1126.         strData = strData.Replace("\u0027", "\u0001");
1127.         strData = strData.Replace("'", "");
1128.         return strData.Replace("\u0001", "\u0027");
```

檔案名稱	WebBatchPrint.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
85.           string strGUID = GetRequest("GUID", ""); // Ū�� URL
�□ع��
86.           objDR1 = objDB1.Query("SELECT UID FROM AppUser WITH(NOLOCK)
WHERE GUID = '" + strGUID + "';");
```

Heuristic Parameter Tampering\路徑 6:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=352
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebBatchPrint.aspx.cs 中第 129 行的 GetRequest 方法從 QueryString_strName 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebBatchPrint.aspx.cs 中第 61 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebBatchPrint.aspx.cs	WebBatchPrint.aspx.cs
行	136	86
物件	QueryString_strName	Query

代碼片斷

檔案名稱	WebBatchPrint.aspx.cs
方法	private string GetRequest(string strName, string strDefault)

```
....
136.             strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
138.         return DFI(strResult.Trim());
```

檔案名稱	WebBatchPrint.aspx.cs
方法	private string DFI(string strData)

```
....
1124.      private string DFI(string strData)
....
1126.        strData = strData.Replace("\u0027", "\u0001");
1127.        strData = strData.Replace("'", "");
1128.        return strData.Replace("\u0001", "\u0027");
```

檔案名稱 WebBatchPrint.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
85.           string strGUID = GetRequest("GUID", ""); // Ū�� URL
�□ع��
86.           objDR1 = objDB1.Query("SELECT UID FROM AppUser WITH(NOLOCK)
WHERE GUID = '" + strGUID + "';");
```

Heuristic Parameter Tampering\路徑 7:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=353
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebBatchPrint.aspx.cs 中第 129 行的 GetRequest 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebBatchPrint.aspx.cs 中第 61 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebBatchPrint.aspx.cs	WebBatchPrint.aspx.cs
行	133	92
物件	Form	Query

代碼片斷

檔案名稱 WebBatchPrint.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
133.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
138.         return DFI(strResult.Trim());
```

檔案名稱 WebBatchPrint.aspx.cs

方法 private string DFI(string strData)

```
....
1124.      private string DFI(string strData)
....
1126.        strData = strData.Replace("\u0027", "\u0001");
1127.        strData = strData.Replace("'", "");
1128.        return strData.Replace("\u0001", "\u0027");
```

檔案名稱 WebBatchPrint.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
74.        varPKEY = GetRequest("PKEY", ""); // Ū�� URL ��ȡ��
75.        varFieldName = GetRequest("FieldName", "ID");
....
85.          string strGUID = GetRequest("GUID", ""); // Ū�� URL
��ȡ��
....
94.            + " WHERE A.LogKey = '" + varPKEY + "';");
....
92.          objDR1 = objDB1.Query("SELECT A.Name, B.DocDefName FROM
UserCache A WITH(NOLOCK)"
```

Heuristic Parameter Tampering\路徑 8:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=354
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebBatchPrint.aspx.cs 中第 129 行的 GetRequest 方法從 QueryString_strName 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebBatchPrint.aspx.cs 中第 61 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebBatchPrint.aspx.cs	WebBatchPrint.aspx.cs
行	136	92
物件	QueryString_strName	Query

代碼片斷

檔案名稱 WebBatchPrint.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
136.            strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
138.        return DFI(strResult.Trim());
```

檔案名稱 WebBatchPrint.aspx.cs

方法 private string DFI(string strData)

```
....
1124.      private string DFI(string strData)
....
1126.        strData = strData.Replace("\u0027", "\u0001");
1127.        strData = strData.Replace("'", "");
1128.        return strData.Replace("\u0001", "\u0027");
```

檔案名稱 WebBatchPrint.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
74.        varPKEY = GetRequest("PKEY", ""); // Ū�� URL ��ɀ��
75.        varFieldName = GetRequest("FieldName", "ID");
....
85.          string strGUID = GetRequest("GUID", ""); // Ū�� URL
��ɀ��
....
94.            + " WHERE A.LogKey = '" + varPKEY + "';");
....
92.          objDR1 = objDB1.Query("SELECT A.Name, B.DocDefName FROM
UserCache A WITH(NOLOCK)"
```

Heuristic Parameter Tampering\路徑 9:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=355
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebCatalog.aspx.cs 中第 1829 行的 Sub_GetRequest 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebCatalog.aspx.cs 中第 358 行的 Sub_Category 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs

行	1837	396
物件	Form	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
396.               System.Data.SqlClient.SqlDataReader objDR1 =
DOC.Query(string.Format(strSQL1, strCategoryID));
```

Heuristic Parameter Tampering\路徑 10:

嚴重程度： 低風險

結果狀態： 校驗

線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=356

狀態 反覆出現的問題

Detection Date 7/8/2022 3:05:06 PM

WebCatalog.aspx.cs 中第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebCatalog.aspx.cs 中第 358 行的 Sub_Category 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	396
物件	QueryString_Name	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
396.             System.Data.SqlClient.SqlDataReader objDR1 =
DOC.Query(string.Format(strSQL1, strCategoryID));
```

Heuristic Parameter Tampering\路徑 11:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=357
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebCatalog.aspx.cs 中第 1829 行的 Sub_GetRequest 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebCatalog.aspx.cs 中第 632 行的 Sub_FolderMtn 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	679
物件	Form	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_FolderMtn(int Action)

```
....
638.        string strFolderID = this.Sub_GetRequest("txtFolderID", "");
....
679.          objDR1 = DOC.Query("SELECT ID FROM DocFolder WITH(NOLOCK)
WHERE DocFolderID = '" + strFolderID + "';");
```

Heuristic Parameter Tampering\路徑 12:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=358
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebCatalog.aspx.cs 中第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebCatalog.aspx.cs 中第 632 行的 Sub_FolderMtn 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	679
物件	QueryString_Name	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_FolderMtn(int Action)

```
....
638.        string strFolderID = this.Sub_GetRequest("txtFolderID", "");
....
679.          objDR1 = DOC.Query("SELECT ID FROM DocFolder WITH(NOLOCK)
WHERE DocFolderID = '" + strFolderID + "';");
```

Heuristic Parameter Tampering\路徑 13:

嚴重程度：	低風險

結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=359
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebCatalog.aspx.cs 中第 632 行的 Sub_FolderMtn 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebCatalog.aspx.cs 中第 632 行的 Sub_FolderMtn 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	674	679
物件	Form	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_FolderMtn(int Action)

```
....
674.             strFolderID =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form["strFolder
ID"], false));
....
679.          objDR1 = DOC.Query("SELECT ID FROM DocFolder WITH(NOLOCK)
WHERE DocFolderID = '" + strFolderID + "';");
```

Heuristic Parameter Tampering\路徑 14:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=360
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebCatalog.aspx.cs 中第 1829 行的 Sub_GetRequest 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebCatalog.aspx.cs 中第 632 行的 Sub_FolderMtn 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	903
物件	Form	Query

代碼片斷

檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱	WebCatalog.aspx.cs
方法	private void Sub_FolderMtn(int Action)

```
....
638.        string strFolderID = this.Sub_GetRequest("txtFolderID", "");
....
892.            objDR1 = DOC.GetFolderAuthority(strFolderID);
893.            while(objDR1.Read())
....
895.              strCode = objDR1.IsDBNull(1) ? "" :
objDR1.GetString(1).Trim();
....
902.                  string[] aryTemp1 = strCode.Split('/'); strName =
"";
903.                  objDR2 = objDBA.Query("SELECT CompanyName FROM
AppCompany WITH(NOLOCK) WHERE CompanyNo = '" + aryTemp1[0] + "'");
```

Heuristic Parameter Tampering\路徑 15:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=361
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebCatalog.aspx.cs 中第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebCatalog.aspx.cs 中第 632 行的 Sub_FolderMtn 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	903
物件	QueryString_Name	Query

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_FolderMtn(int Action)

```
....
638.        string strFolderID = this.Sub_GetRequest("txtFolderID", "");
....
892.            objDR1 = DOC.GetFolderAuthority(strFolderID);
893.            while(objDR1.Read())
....
895.              strCode = objDR1.IsDBNull(1) ? "" :
objDR1.GetString(1).Trim();
....
902.                  string[] aryTemp1 = strCode.Split('/'); strName =
"";
903.                  objDR2 = objDBA.Query("SELECT CompanyName FROM
AppCompany WITH(NOLOCK) WHERE CompanyNo = '" + aryTemp1[0] + "'");
```

Heuristic Parameter Tampering\路徑 16:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=362
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebCatalog.aspx.cs 中第 632 行的 Sub_FolderMtn 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebCatalog.aspx.cs 中第 632 行的 Sub_FolderMtn 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	674	903
物件	Form	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_FolderMtn(int Action)

```
....
674.            strFolderID =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form["strFolder
ID"], false));
....
892.            objDR1 = DOC.GetFolderAuthority(strFolderID);
893.            while(objDR1.Read())
....
895.              strCode = objDR1.IsDBNull(1) ? "" :
objDR1.GetString(1).Trim();
....
902.                  string[] aryTemp1 = strCode.Split('/'); strName =
"";
903.                  objDR2 = objDBA.Query("SELECT CompanyName FROM
AppCompany WITH(NOLOCK) WHERE CompanyNo = '" + aryTemp1[0] + "'");
```

Heuristic Parameter Tampering\路徑 17:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=363
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebCatalog.aspx.cs 中第 632 行的 Sub_FolderMtn 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebCatalog.aspx.cs 中第 632 行的 Sub_FolderMtn 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	675	903
物件	Form	Query

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private void Sub_FolderMtn(int Action)

```
....
675.            strFolderName =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form["strFolder
Name"], false));
....
789.                    strFolderID = DOC.CreateFolder(strFolderName,
strCompanyCode, strDepartmentCode, WhoAmI.UserID, (strPublish == "True"
? true : false));
....
892.            objDR1 = DOC.GetFolderAuthority(strFolderID);
893.            while(objDR1.Read())
....
895.              strCode = objDR1.IsDBNull(1) ? "" :
objDR1.GetString(1).Trim();
....
902.                    string[] aryTemp1 = strCode.Split('/'); strName =
"";
903.                    objDR2 = objDBA.Query("SELECT CompanyName FROM
AppCompany WITH(NOLOCK) WHERE CompanyNo = '" + aryTemp1[0] + "'");
```

Heuristic Parameter Tampering\路徑 18:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=364
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebCatalog.aspx.cs 中第 1829 行的 Sub_GetRequest 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebCatalog.aspx.cs 中第 632 行的 Sub_FolderMtn 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	920
物件	Form	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_FolderMtn(int Action)

```
....
638.        string strFolderID = this.Sub_GetRequest("txtFolderID", "");
....
892.            objDR1 = DOC.GetFolderAuthority(strFolderID);
893.            while(objDR1.Read())
....
895.              strCode = objDR1.IsDBNull(1) ? "" :
objDR1.GetString(1).Trim();
....
902.                  string[] aryTemp1 = strCode.Split('/'); strName =
"";
....
920.                    objDR2 = objDBA.Query("SELECT Name FROM AppUser
WITH(NOLOCK) WHERE UID = '" + aryTemp1[2] + "'");
```

Heuristic Parameter Tampering\路徑 19:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=365
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebCatalog.aspx.cs 中第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebCatalog.aspx.cs 中第 632 行的 Sub_FolderMtn 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	920
物件	QueryString_Name	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_FolderMtn(int Action)

```
....
638.        string strFolderID = this.Sub_GetRequest("txtFolderID", "");
....
892.            objDR1 = DOC.GetFolderAuthority(strFolderID);
893.            while(objDR1.Read())
....
895.              strCode = objDR1.IsDBNull(1) ? "" :
objDR1.GetString(1).Trim();
....
902.                  string[] aryTemp1 = strCode.Split('/'); strName =
"";
....
920.                    objDR2 = objDBA.Query("SELECT Name FROM AppUser
WITH(NOLOCK) WHERE UID = '" + aryTemp1[2] + "'");
```

Heuristic Parameter Tampering\路徑 20:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=366
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebCatalog.aspx.cs 中第 632 行的 Sub_FolderMtn 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebCatalog.aspx.cs 中第 632 行的 Sub_FolderMtn 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	674	920
物件	Form	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_FolderMtn(int Action)

```
....
674.            strFolderID =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form["strFolder
ID"], false));
....
892.            objDR1 = DOC.GetFolderAuthority(strFolderID);
893.            while(objDR1.Read())
....
895.              strCode = objDR1.IsDBNull(1) ? "" :
objDR1.GetString(1).Trim();
....
902.                  string[] aryTemp1 = strCode.Split('/'); strName =
"";
....
920.                    objDR2 = objDBA.Query("SELECT Name FROM AppUser
WITH(NOLOCK) WHERE UID = '" + aryTemp1[2] + "'");
```

Heuristic Parameter Tampering\路徑 21:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=367
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebCatalog.aspx.cs 中第 632 行的 Sub_FolderMtn 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebCatalog.aspx.cs 中第 632 行的 Sub_FolderMtn 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	675	920
物件	Form	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_FolderMtn(int Action)

```
....
675.            strFolderName =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form["strFolder
Name"], false));
....
789.                  strFolderID = DOC.CreateFolder(strFolderName,
strCompanyCode, strDepartmentCode, WhoAmI.UserID, (strPublish == "True"
? true : false));
....
892.            objDR1 = DOC.GetFolderAuthority(strFolderID);
893.            while(objDR1.Read())
....
895.              strCode = objDR1.IsDBNull(1) ? "" :
objDR1.GetString(1).Trim();
....
902.                  string[] aryTemp1 = strCode.Split('/'); strName =
"";
....
920.                    objDR2 = objDBA.Query("SELECT Name FROM AppUser
WITH(NOLOCK) WHERE UID = '" + aryTemp1[2] + "'");
```

Heuristic Parameter Tampering\路徑 22:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=368
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebCatalog.aspx.cs 中第 632 行的 Sub_FolderMtn 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebCatalog.aspx.cs 中第 1007 行的 Sub_Organization 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	675	1012
物件	Form	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_FolderMtn(int Action)

```
....
675.            strFolderName =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form["strFolder
Name"], false));
....
789.                    strFolderID = DOC.CreateFolder(strFolderName,
strCompanyCode, strDepartmentCode, WhoAmI.UserID, (strPublish == "True"
? true : false));
....
804.              DOC.SetFolderAuthority(strFolderID,
strPublicArea.Split(','), rtnPublicArea.Split(','));
805.              this.Sub_Debug("SetFolderAuthority", DOC.DebugSQL);
....
849.            this.Sub_Debug("strFolderID", strFolderID);
850.            this.Sub_Debug("strFolderName", strFolderName);
851.            this.Sub_Debug("strCompany", strCompany + ", " +
strCompanyCode);
852.            this.Sub_Debug("strDepartment", strDepartment + ", " +
strDepartmentCode);
853.            this.Sub_Debug("strPublish", strPublish);
854.            this.Sub_Debug("strPublicArea", strPublicArea);
855.            this.Sub_Debug("strORG", strORG);
856.            this.Sub_Debug("rtnPublicArea", rtnPublicArea);
....
874.            strTMP = this.Sub_Organization("ORG", 2, "?", "", false,
"選擇單位", string.Format("{0}/{1}", strCompanyCode, strDepartmentCode),
string.Format("{0}/{1}", strCompany, strDepartment), false,
this.Sub_LockedCompany() + strChangeFlag, out strOnClick);
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_Organization(string Key, int OrgType, string Attribute, string ScriptName, bool UsingTextArea, string Title, string Codes, string Names, bool AddInput, string LockORG, out string strOnClick)

```
....
1012.        System.Data.SqlClient.SqlDataReader objDR1 =
DOC.Query(strSQL);
```

Heuristic Parameter Tampering\路徑 23:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=369
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebCatalog.aspx.cs 中第 1829 行的 Sub_GetRequest 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebCatalog.aspx.cs 中第 1007 行的 Sub_Organization 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	1012
物件	Form	Query

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱	WebCatalog.aspx.cs
方法	private void Sub_FolderMtn(int Action)

```
....
646.        string rtnPublicArea = this.Sub_GetRequest("rtnPublicArea",
"");
....
804.              DOC.SetFolderAuthority(strFolderID,
strPublicArea.Split(','), rtnPublicArea.Split(','));
805.              this.Sub_Debug("SetFolderAuthority", DOC.DebugSQL);
....
849.            this.Sub_Debug("strFolderID", strFolderID);
850.            this.Sub_Debug("strFolderName", strFolderName);
851.            this.Sub_Debug("strCompany", strCompany + ", " +
strCompanyCode);
852.            this.Sub_Debug("strDepartment", strDepartment + ", " +
strDepartmentCode);
853.            this.Sub_Debug("strPublish", strPublish);
854.            this.Sub_Debug("strPublicArea", strPublicArea);
855.            this.Sub_Debug("strORG", strORG);
856.            this.Sub_Debug("rtnPublicArea", rtnPublicArea);
....
874.            strTMP = this.Sub_Organization("ORG", 2, "?", "", false,
"選擇單位", string.Format("{0}/{1}", strCompanyCode, strDepartmentCode),
string.Format("{0}/{1}", strCompany, strDepartment), false,
this.Sub_LockedCompany() + strChangeFlag, out strOnClick);
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_Organization(string Key, int OrgType, string Attribute, string ScriptName, bool UsingTextArea, string Title, string Codes, string Names, bool AddInput, string LockORG, out string strOnClick)

```
....
1012.        System.Data.SqlClient.SqlDataReader objDR1 =
DOC.Query(strSQL);
```

Heuristic Parameter Tampering\路徑 24:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=370
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebCatalog.aspx.cs 中第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebCatalog.aspx.cs 中第 1007 行的 Sub_Organization 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	1012
物件	QueryString_Name	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.             strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_FolderMtn(int Action)

```
....
646.         string rtnPublicArea = this.Sub_GetRequest("rtnPublicArea",
"");
....
804.              DOC.SetFolderAuthority(strFolderID,
strPublicArea.Split(','), rtnPublicArea.Split(','));
805.              this.Sub_Debug("SetFolderAuthority", DOC.DebugSQL);
....
849.            this.Sub_Debug("strFolderID", strFolderID);
850.            this.Sub_Debug("strFolderName", strFolderName);
851.            this.Sub_Debug("strCompany", strCompany + ", " +
strCompanyCode);
852.            this.Sub_Debug("strDepartment", strDepartment + ", " +
strDepartmentCode);
853.            this.Sub_Debug("strPublish", strPublish);
854.            this.Sub_Debug("strPublicArea", strPublicArea);
855.            this.Sub_Debug("strORG", strORG);
856.            this.Sub_Debug("rtnPublicArea", rtnPublicArea);
....
874.            strTMP = this.Sub_Organization("ORG", 2, "?", "", false,
"選擇單位", string.Format("{0}/{1}", strCompanyCode, strDepartmentCode),
string.Format("{0}/{1}", strCompany, strDepartment), false,
this.Sub_LockedCompany() + strChangeFlag, out strOnClick);
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_Organization(string Key, int OrgType, string Attribute, string ScriptName, bool UsingTextArea, string Title, string Codes, string Names, bool AddInput, string LockORG, out string strOnClick)

```
....
1012.         System.Data.SqlClient.SqlDataReader objDR1 =
DOC.Query(strSQL);
```

Heuristic Parameter Tampering\路徑 25:

嚴重程度： 低風險

結果狀態： 校驗

線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=371

狀態 反覆出現的問題

Detection Date	7/8/2022 3:05:06 PM

WebCatalog.aspx.cs 中第 1829 行的 Sub_GetRequest 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebCatalog.aspx.cs 中第 1863 行的 Sub_UpdateNewsLetter 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	1933
物件	Form	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_NewsLetter(int Action)

```
....
2158.         string strEMailB = this.Sub_GetRequest("txtEMailB", "");
....
2209.         strB = this.Sub_UpdateNewsLetter(strMode, "最新文件快訊",
strEMailB, "AppKMNewOrder", ref strOrderDoc) + "<br />";
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateNewsLetter(string Mode, string Name, string Mail, string Table, ref string strOrderDoc)

```
....
1863.      private string Sub_UpdateNewsLetter(string Mode, string Name,
string Mail, string Table, ref string strOrderDoc)
....
1928.               strSQL = string.Format(fmtSQL4a, Table, Mail);
....
1933.             DR = this.DOC.Query(strSQL);
```

Heuristic Parameter Tampering\路徑 26:

嚴重程度：	低風險
結果狀態：	校驗

線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=372
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebCatalog.aspx.cs 中第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebCatalog.aspx.cs 中第 1863 行的 Sub_UpdateNewsLetter 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	1933
物件	QueryString_Name	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_NewsLetter(int Action)

```
....
2158.        string strEMailB = this.Sub_GetRequest("txtEMailB", "");
....
2209.        strB = this.Sub_UpdateNewsLetter(strMode, "最新文件快訊",
strEMailB, "AppKMNewOrder", ref strOrderDoc) + "<br />";
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateNewsLetter(string Mode, string Name, string Mail, string Table, ref string strOrderDoc)

```
....
1863.      private string Sub_UpdateNewsLetter(string Mode, string Name,
string Mail, string Table, ref string strOrderDoc)
....
1928.              strSQL = string.Format(fmtSQL4a, Table, Mail);
....
1933.            DR = this.DOC.Query(strSQL);
```

Heuristic Parameter Tampering\路徑 27:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=373
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebCatalog.aspx.cs 中第 1829 行的 Sub_GetRequest 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebCatalog.aspx.cs 中第 2342 行的 Sub_MailList 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	2360
物件	Form	Query

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_GetRequestEmpty(string Name, string Default)

```
....
1850.         string strResult = Sub_GetRequest(Name, "");
....
1852.         return strResult;
```

檔案名稱	WebCatalog.aspx.cs
方法	private void Sub_NewsLetter(int Action)

```
....
2253.           string strMailClass =
this.Sub_GetRequestEmpty("txtMailClass", "AppKMNewOrder");
....
2271.           DR = this.Sub_MailList(strMailClass);
```

檔案名稱	WebCatalog.aspx.cs
方法	private System.Data.SqlClient.SqlDataReader Sub_MailList(string MailClass)

```
....
2342.      private System.Data.SqlClient.SqlDataReader
Sub_MailList(string MailClass)
....
2351.            strSQL = string.Format(strSQL, MailClass);
....
2360.        return this.DOC.Query(strSQL);
```

Heuristic Parameter Tampering\路徑 28:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=374
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebCatalog.aspx.cs 中第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebCatalog.aspx.cs 中第 2342 行的 Sub_MailList 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	2360
物件	QueryString_Name	Query

代碼片斷

檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_GetRequestEmpty(string Name, string Default)

```
....
1850.        string strResult = Sub_GetRequest(Name, "");
....
1852.        return strResult;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_NewsLetter(int Action)

```
....
2253.          string strMailClass =
this.Sub_GetRequestEmpty("txtMailClass", "AppKMNewOrder");
....
2271.          DR = this.Sub_MailList(strMailClass);
```

檔案名稱 WebCatalog.aspx.cs

方法 private System.Data.SqlClient.SqlDataReader Sub_MailList(string MailClass)

```
....
2342.      private System.Data.SqlClient.SqlDataReader
Sub_MailList(string MailClass)
....
2351.            strSQL = string.Format(strSQL, MailClass);
....
2360.        return this.DOC.Query(strSQL);
```

Heuristic Parameter Tampering\路徑 29:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=375
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebCatalog.aspx.cs 中第 1829 行的 Sub_GetRequest 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebCatalog.aspx.cs 中第 3904 行的 Sub_PersonalSetting 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	4260
物件	Form	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 public void Sub_PersonalSetting(int Action)

```
....
4007.             strEmail = this.Sub_GetRequest("txtEmail", "").Trim();
....
4016.               objDR1 = this.DOC.Query(string.Format("SELECT * FROM
AppUser WITH(NOLOCK) WHERE Email = '{0}' AND UID <> '{1}'",
strEmail.Replace("'", "''"), WhoAmI.UserID));
....
4027.                 WhoAmI.eMail = strEmail;
....
4260.         objDR1 = this.DOC.Query(string.Format(strSQL3,
WhoAmI.eMail, WhoAmI.UserID));
```

Heuristic Parameter Tampering\路徑 30:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=376
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebCatalog.aspx.cs 中第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebCatalog.aspx.cs 中第 3904 行的 Sub_PersonalSetting 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	4260
物件	QueryString_Name	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 public void Sub_PersonalSetting(int Action)

```
....
4007.            strEmail = this.Sub_GetRequest("txtEmail", "").Trim();
....
4016.              objDR1 = this.DOC.Query(string.Format("SELECT * FROM
AppUser WITH(NOLOCK) WHERE Email = '{0}' AND UID <> '{1}'",
strEmail.Replace("'", "''"), WhoAmI.UserID));
....
4027.                WhoAmI.eMail = strEmail;
....
4260.        objDR1 = this.DOC.Query(string.Format(strSQL3,
WhoAmI.eMail, WhoAmI.UserID));
```

Heuristic Parameter Tampering\路徑 31:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=377
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebDownloadFiles.aspx.cs 中第 163 行的 GetRequest 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebDownloadFiles.aspx.cs 中第 33 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebDownloadFiles.aspx.cs	WebDownloadFiles.aspx.cs
行	167	81
物件	Form	Query

代碼片斷

檔案名稱 WebDownloadFiles.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
167.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
172.         return strResult.Trim();
```

檔案名稱 WebDownloadFiles.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
41.         varDocID = GetRequest("DocID", "");
....
43.         varLogKey = GetRequest("LogKey", "\u0001");
....
81.           objDR0 = objDB0.Query("SELECT DocXML FROM DocCatalog
WITH(NOLOCK) WHERE DocID = '" + varDocID + "';");
```

Heuristic Parameter Tampering\路徑 32:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=378
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebDownloadFiles.aspx.cs 中第 163 行的 GetRequest 方法從 QueryString_strName 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebDownloadFiles.aspx.cs 中第 33 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebDownloadFiles.aspx.cs	WebDownloadFiles.aspx.cs
行	170	81
物件	QueryString_strName	Query

代碼片斷

檔案名稱 WebDownloadFiles.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
170.             strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
172.         return strResult.Trim();
```

檔案名稱	WebDownloadFiles.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
41.         varDocID = GetRequest("DocID", "");
....
43.         varLogKey = GetRequest("LogKey", "\u0001");
....
81.           objDR0 = objDB0.Query("SELECT DocXML FROM DocCatalog
WITH(NOLOCK) WHERE DocID = '" + varDocID + "';");
```

Heuristic Parameter Tampering\路徑 33:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=379
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 88 行的 GetRequest 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 115 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	90	283
物件	Form	Query

代碼片斷

檔案名稱	WebEditor.aspx.cs
方法	private string GetRequest(string strName, string strDefault)

```
....
90.         string strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.Form[strNam
e], false), false));
....
97.         return strResult;
```

檔案名稱	WebEditor.aspx.cs
方法	private string XSS(string strData, bool bUrl)

```
....
2841.      private string XSS(string strData, bool bUrl)
....
2843.        string strResult = HttpUtility.HtmlEncode(strData);
2844.        strResult = HttpUtility.HtmlDecode(strResult);
....
2846.        return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
149.        if(!IsPostBack) LogKey.Value = GetRequest("LogKey",
"\u0001\u0002\u0003");
....
217.        objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.          _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
283.          objDR1 = objDoc.Query(objSqlCommand, 30);
```

Heuristic Parameter Tampering\路徑 34:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=380
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 88 行的 GetRequest 方法從 QueryString_strName 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 115 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	93	283
物件	QueryString_strName	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
93.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.QueryString
[strName], false), false));
....
97.         return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 private string XSS(string strData, bool bUrl)

```
....
2841.      private string XSS(string strData, bool bUrl)
....
2843.        string strResult = HttpUtility.HtmlEncode(strData);
2844.        strResult = HttpUtility.HtmlDecode(strResult);
....
2846.        return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
149.         if(!IsPostBack) LogKey.Value = GetRequest("LogKey",
"\u0001\u0002\u0003");
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.           _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
283.           objDR1 = objDoc.Query(objSqlCommand, 30);
```

Heuristic Parameter Tampering\路徑 35:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=381
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 115 行的 Page_Load 方法從 Value 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 115 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	150	283

物件	Value	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
150.         _sLogKey = CSS(LogKey.Value);
....
187.           + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.           _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
283.           objDR1 = objDoc.Query(objSqlCommand, 30);
```

檔案名稱 WebEditor.aspx.cs

方法 private string CSS(string strData)

```
....
2831.       private string CSS(string strData)
....
2833.         return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic Parameter Tampering\路徑 36:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=382
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 115 行的 Page_Load 方法從 Value 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 115 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	187	283
物件	Value	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
187.           + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
185.         _sUploadFilePath =
Cdsys.KM.Utility.KMconfig.GetAppSetting("DataPATH", "C:/Temp",
"C:/Temp")
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.           _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
283.           objDR1 = objDoc.Query(objSqlCommand, 30);
```

檔案名稱 WebEditor.aspx.cs

方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic Parameter Tampering\路徑 37:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=383
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 88 行的 GetRequest 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 115 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	90	305
物件	Form	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
90.         string strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.Form[strNam
e], false), false));
....
97.         return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 private string XSS(string strData, bool bUrl)

```
....
2841.       private string XSS(string strData, bool bUrl)
....
2843.         string strResult = HttpUtility.HtmlEncode(strData);
2844.         strResult = HttpUtility.HtmlDecode(strResult);
....
2846.         return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
257.            sDocID = txt_DocID.Value = GetRequest("DocID", "");
....
304.            objSqlCommand.Parameters["@DocID"].Value = sDocID;
305.            objDR1 = objDoc.Query(objSqlCommand, 30);
```

Heuristic Parameter Tampering\路徑 38:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=384
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 88 行的 GetRequest 方法從 QueryString_strName 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 115 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	93	305
物件	QueryString_strName	Query

代碼片斷

檔案名稱	WebEditor.aspx.cs
方法	private string GetRequest(string strName, string strDefault)

```
....
93.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.QueryString
[strName], false), false));
....
97.         return strResult;
```

檔案名稱	WebEditor.aspx.cs
方法	private string XSS(string strData, bool bUrl)

```
....
2841.      private string XSS(string strData, bool bUrl)
....
2843.        string strResult = HttpUtility.HtmlEncode(strData);
2844.        strResult = HttpUtility.HtmlDecode(strResult);
....
2846.        return strResult;
```

檔案名稱	WebEditor.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
257.            sDocID = txt_DocID.Value = GetRequest("DocID", "");
....
304.            objSqlCommand.Parameters["@DocID"].Value = sDocID;
305.            objDR1 = objDoc.Query(objSqlCommand, 30);
```

Heuristic Parameter Tampering\路徑 39:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=385
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 115 行的 Page_Load 方法從 Value 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 115 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	150	305
物件	Value	Query

代碼片斷
檔案名稱 WebEditor.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
150.         _sLogKey = CSS(LogKey.Value);
....
187.           + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.           _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
305.             objDR1 = objDoc.Query(objSqlCommand, 30);
```

檔案名稱 WebEditor.aspx.cs
方法 private string CSS(string strData)

```
....
2831.       private string CSS(string strData)
....
2833.         return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic Parameter Tampering\路徑 40:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=386
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 115 行的 Page_Load 方法從 Value 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 115 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	187	305
物件	Value	Query

代碼片斷
檔案名稱 WebEditor.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
187.           + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
185.         _sUploadFilePath =
Cdsys.KM.Utility.KMconfig.GetAppSetting("DataPATH", "C:/Temp",
"C:/Temp")
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.           _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
305.             objDR1 = objDoc.Query(objSqlCommand, 30);
```

檔案名稱 WebEditor.aspx.cs

方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic Parameter Tampering\路徑 41:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=387
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 115 行的 Page_Load 方法從 Value 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 115 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	263	305
物件	Value	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
263.            sDocID = CSS(txt_DocID.Value);
....
304.            objSqlCommand.Parameters["@DocID"].Value = sDocID;
305.            objDR1 = objDoc.Query(objSqlCommand, 30);
```

檔案名稱 WebEditor.aspx.cs

方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic Parameter Tampering\路徑 42:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=388
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 115 行的 Page_Load 方法從 Value 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 115 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	341	341
物件	Value	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
341.                objDR1 = objDoc.Query(string.Format(strSQL1,
CSS(txt_DocID.Value)));
```

檔案名稱 WebEditor.aspx.cs

方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic Parameter Tampering\路徑 43:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=389
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 88 行的 GetRequest 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 115 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	90	341
物件	Form	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
90.        string strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.Form[strNam
e], false), false));
....
97.        return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 private string XSS(string strData, bool bUrl)

```
....
2841.      private string XSS(string strData, bool bUrl)
....
2843.        string strResult = HttpUtility.HtmlEncode(strData);
2844.        strResult = HttpUtility.HtmlDecode(strResult);
....
2846.        return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
149.        if(!IsPostBack) LogKey.Value = GetRequest("LogKey",
"\u0001\u0002\u0003");
....
217.        objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.          _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
328.                objDR1 = objDoc.Query(string.Format(strSQL1,
CSS(txt_DocID.Value), objDoc.User.CompanyCode,
objDoc.User.DepartmentCode));
....
341.                objDR1 = objDoc.Query(string.Format(strSQL1,
CSS(txt_DocID.Value)));
```

Heuristic Parameter Tampering\路徑 44:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=390
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 88 行的 GetRequest 方法從 QueryString_strName 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 115 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	93	341
物件	QueryString_strName	Query

代碼片斷	
檔案名稱	WebEditor.aspx.cs
方法	private string GetRequest(string strName, string strDefault)

```
....
93.          strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.QueryString
[strName], false), false));
....
97.        return strResult;
```

檔案名稱	WebEditor.aspx.cs
方法	private string XSS(string strData, bool bUrl)

```
....
2841.      private string XSS(string strData, bool bUrl)
....
2843.        string strResult = HttpUtility.HtmlEncode(strData);
2844.        strResult = HttpUtility.HtmlDecode(strResult);
....
2846.        return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
149.        if(!IsPostBack) LogKey.Value = GetRequest("LogKey",
"\u0001\u0002\u0003");
....
217.        objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.          _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
328.                objDR1 = objDoc.Query(string.Format(strSQL1,
CSS(txt_DocID.Value), objDoc.User.CompanyCode,
objDoc.User.DepartmentCode));
....
341.                objDR1 = objDoc.Query(string.Format(strSQL1,
CSS(txt_DocID.Value)));
```

Heuristic Parameter Tampering\路徑 45:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=391
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 115 行的 Page_Load 方法從 Value 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 115 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	150	341
物件	Value	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
150.         _sLogKey = CSS(LogKey.Value);
....
187.           + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.           _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
328.                 objDR1 = objDoc.Query(string.Format(strSQL1,
CSS(txt_DocID.Value), objDoc.User.CompanyCode,
objDoc.User.DepartmentCode));
....
341.                 objDR1 = objDoc.Query(string.Format(strSQL1,
CSS(txt_DocID.Value)));
```

檔案名稱 WebEditor.aspx.cs

方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.         return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic Parameter Tampering\路徑 46:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=392
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 115 行的 Page_Load 方法從 Value 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 115 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	187	341
物件	Value	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
187.           + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
185.         _sUploadFilePath =
Cdsys.KM.Utility.KMconfig.GetAppSetting("DataPATH", "C:/Temp",
"C:/Temp")
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.           _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
328.                 objDR1 = objDoc.Query(string.Format(strSQL1,
CSS(txt_DocID.Value), objDoc.User.CompanyCode,
objDoc.User.DepartmentCode));
....
341.                 objDR1 = objDoc.Query(string.Format(strSQL1,
CSS(txt_DocID.Value)));
```

檔案名稱 WebEditor.aspx.cs

方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic Parameter Tampering\路徑 47:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=393
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 115 行的 Page_Load 方法從 Value 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 115 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	263	380
物件	Value	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
263.            sDocID = CSS(txt_DocID.Value);
....
380.              objDR1 = objDoc.Query(string.Format(strSQL1, sDocID));
```

檔案名稱 WebEditor.aspx.cs

方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic Parameter Tampering\路徑 48:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=394
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 88 行的 GetRequest 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 115 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	90	380
物件	Form	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
90.         string strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.Form[strNam
e], false), false));
....
97.         return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 private string XSS(string strData, bool bUrl)

```
....
2841.      private string XSS(string strData, bool bUrl)
....
2843.        string strResult = HttpUtility.HtmlEncode(strData);
2844.        strResult = HttpUtility.HtmlDecode(strResult);
....
2846.        return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
257.            sDocID = txt_DocID.Value = GetRequest("DocID", "");
....
380.              objDR1 = objDoc.Query(string.Format(strSQL1, sDocID));
```

Heuristic Parameter Tampering\路徑 49:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=395
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 88 行的 GetRequest 方法從 QueryString_strName 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 115 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	93	380
物件	QueryString_strName	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
93.          strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.QueryString
[strName], false), false));
....
97.        return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法	private string XSS(string strData, bool bUrl)

```
....
2841.      private string XSS(string strData, bool bUrl)
....
2843.        string strResult = HttpUtility.HtmlEncode(strData);
2844.        strResult = HttpUtility.HtmlDecode(strResult);
....
2846.        return strResult;
```

檔案名稱	WebEditor.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
257.            sDocID = txt_DocID.Value = GetRequest("DocID", "");
....
380.              objDR1 = objDoc.Query(string.Format(strSQL1, sDocID));
```

Heuristic Parameter Tampering\路徑 50:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=396
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 115 行的 Page_Load 方法從 Value 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 115 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	150	380
物件	Value	Query

代碼片斷	
檔案名稱	WebEditor.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
150.         _sLogKey = CSS(LogKey.Value);
....
187.           + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.           _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
380.               objDR1 = objDoc.Query(string.Format(strSQL1, sDocID));
```

檔案名稱 WebEditor.aspx.cs

方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic Parameter Tampering\路徑 51:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=397
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 115 行的 Page_Load 方法從 Value 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 115 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	187	380
物件	Value	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
187.          + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
185.         _sUploadFilePath =
Cdsys.KM.Utility.KMconfig.GetAppSetting("DataPATH", "C:/Temp",
"C:/Temp")
....
217.        objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
274.          _sDocShowType = sDocType; _sDocType =
objDoc.ShowNameToDocDefName(sDocType);
....
380.              objDR1 = objDoc.Query(string.Format(strSQL1, sDocID));
```

檔案名稱 WebEditor.aspx.cs

方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic Parameter Tampering\路徑 52:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=398
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 633 行的 CopyAttachmentFiles 方法從 Value 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 633 行的 CopyAttachmentFiles 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	650	650
物件	Value	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private string CopyAttachmentFiles(string OldDocID, string DocID, string DocXML)

```
....
650.         objDR1 = objDoc.Query(string.Format(fmtSQL,
txt_NewDocID.Value));
```

Heuristic Parameter Tampering\路徑 53:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=399
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 88 行的 GetRequest 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 633 行的 CopyAttachmentFiles 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	90	650
物件	Form	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
90.         string strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.Form[strNam
e], false), false));
....
97.         return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 private string XSS(string strData, bool bUrl)

```
....
2841.       private string XSS(string strData, bool bUrl)
....
2843.         string strResult = HttpUtility.HtmlEncode(strData);
2844.         strResult = HttpUtility.HtmlDecode(strResult);
....
2846.         return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
149.        if(!IsPostBack) LogKey.Value = GetRequest("LogKey",
"\u0001\u0002\u0003");
....
217.        objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
243.          ExportXml();
```

檔案名稱 WebEditor.aspx.cs

方法 private void ExportXml()

```
....
794.                sXML = CopyAttachmentFiles(PathT(txt_DocID.Value),
PathT(txt_NewDocID.Value), sXML);
```

檔案名稱 WebEditor.aspx.cs

方法 private string CopyAttachmentFiles(string OldDocID, string DocID, string DocXML)

```
....
635.        OldDocID = PathT(OldDocID);
636.        DocID = PathT(DocID);
....
650.        objDR1 = objDoc.Query(string.Format(fmtSQL,
txt_NewDocID.Value));
```

Heuristic Parameter Tampering\路徑 54:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=400
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 88 行的 GetRequest 方法從 QueryString_strName 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 633 行的 CopyAttachmentFiles 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	93	650
物件	QueryString_strName	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
93.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.QueryString
[strName], false), false));
....
97.         return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 private string XSS(string strData, bool bUrl)

```
....
2841.       private string XSS(string strData, bool bUrl)
....
2843.         string strResult = HttpUtility.HtmlEncode(strData);
2844.         strResult = HttpUtility.HtmlDecode(strResult);
....
2846.         return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
149.         if(!IsPostBack) LogKey.Value = GetRequest("LogKey",
"\u0001\u0002\u0003");
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
243.           ExportXml();
```

檔案名稱 WebEditor.aspx.cs

方法 private void ExportXml()

```
....
794.                 sXML = CopyAttachmentFiles(PathT(txt_DocID.Value),
PathT(txt_NewDocID.Value), sXML);
```

檔案名稱 WebEditor.aspx.cs

方法 private string CopyAttachmentFiles(string OldDocID, string DocID, string DocXML)

```
....
635.         OldDocID = PathT(OldDocID);
636.         DocID = PathT(DocID);
....
650.         objDR1 = objDoc.Query(string.Format(fmtSQL,
txt_NewDocID.Value));
```

Heuristic Parameter Tampering\路徑 55:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=401
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 115 行的 Page_Load 方法從 Value 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 633 行的 CopyAttachmentFiles 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	150	650
物件	Value	Query

代碼片斷	
檔案名稱	WebEditor.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
150.         _sLogKey = CSS(LogKey.Value);
....
187.           + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
243.           ExportXml();
```

檔案名稱	WebEditor.aspx.cs
方法	private string CSS(string strData)

```
....
2831.       private string CSS(string strData)
....
2833.         return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

檔案名稱	WebEditor.aspx.cs
方法	private void ExportXml()

```
....
794.                 sXML = CopyAttachmentFiles(PathT(txt_DocID.Value),
PathT(txt_NewDocID.Value), sXML);
```

檔案名稱 WebEditor.aspx.cs

方法 private string CopyAttachmentFiles(string OldDocID, string DocID, string DocXML)

```
....
635.         OldDocID = PathT(OldDocID);
636.         DocID = PathT(DocID);
....
650.         objDR1 = objDoc.Query(string.Format(fmtSQL,
txt_NewDocID.Value));
```

Heuristic Parameter Tampering\路徑 56:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=402
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 115 行的 Page_Load 方法從 Value 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 633 行的 CopyAttachmentFiles 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	187	650
物件	Value	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
187.           + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
185.         _sUploadFilePath =
Cdsys.KM.Utility.KMconfig.GetAppSetting("DataPATH", "C:/Temp",
"C:/Temp")
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
243.           ExportXml();
```

檔案名稱 WebEditor.aspx.cs

方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

檔案名稱 WebEditor.aspx.cs

方法 private void ExportXml()

```
....
794.                sXML = CopyAttachmentFiles(PathT(txt_DocID.Value),
PathT(txt_NewDocID.Value), sXML);
```

檔案名稱 WebEditor.aspx.cs

方法 private string CopyAttachmentFiles(string OldDocID, string DocID, string DocXML)

```
....
635.        OldDocID = PathT(OldDocID);
636.        DocID = PathT(DocID);
....
650.        objDR1 = objDoc.Query(string.Format(fmtSQL,
txt_NewDocID.Value));
```

Heuristic Parameter Tampering\路徑 57:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=403
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 724 行的 ExportXml 方法從 Value 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 724 行的 ExportXml 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	819	819
物件	Value	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private void ExportXml()

```
....
819.              objDR1 = objDoc.Query("SELECT KmAction, OtherMessage,
DocXML FROM DocCatalog WITH(NOLOCK) WHERE DocID = '" + txt_DocID.Value +
"';");
```

Heuristic Parameter Tampering\路徑 58:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=404
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 88 行的 GetRequest 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 724 行的 ExportXml 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	90	819
物件	Form	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
90.         string strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.Form[strNam
e], false), false));
....
97.         return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 private string XSS(string strData, bool bUrl)

```
....
2841.       private string XSS(string strData, bool bUrl)
....
2843.         string strResult = HttpUtility.HtmlEncode(strData);
2844.         strResult = HttpUtility.HtmlDecode(strResult);
....
2846.         return strResult;
```

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
149.        if(!IsPostBack) LogKey.Value = GetRequest("LogKey",
"\u0001\u0002\u0003");
....
217.        objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
243.          ExportXml();
```

檔案名稱 WebEditor.aspx.cs

方法 private void ExportXml()

```
....
819.              objDR1 = objDoc.Query("SELECT KmAction, OtherMessage,
DocXML FROM DocCatalog WITH(NOLOCK) WHERE DocID = '" + txt_DocID.Value +
"';");
```

Heuristic Parameter Tampering\路徑 59:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=405
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 88 行的 GetRequest 方法從 QueryString_strName 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 724 行的 ExportXml 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	93	819
物件	QueryString_strName	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
93.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(XSS(Request.QueryString
[strName], false), false));
....
97.         return strResult;
```

檔案名稱	WebEditor.aspx.cs
方法	private string XSS(string strData, bool bUrl)

```
....
2841.      private string XSS(string strData, bool bUrl)
....
2843.        string strResult = HttpUtility.HtmlEncode(strData);
2844.        strResult = HttpUtility.HtmlDecode(strResult);
....
2846.        return strResult;
```

檔案名稱	WebEditor.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
149.        if(!IsPostBack) LogKey.Value = GetRequest("LogKey",
"\u0001\u0002\u0003");
....
217.        objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
243.          ExportXml();
```

檔案名稱	WebEditor.aspx.cs
方法	private void ExportXml()

```
....
819.              objDR1 = objDoc.Query("SELECT KmAction, OtherMessage,
DocXML FROM DocCatalog WITH(NOLOCK) WHERE DocID = '" + txt_DocID.Value +
"';");
```

Heuristic Parameter Tampering\路徑 60:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=406
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 115 行的 Page_Load 方法從 Value 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 724 行的 ExportXml 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	150	819
物件	Value	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
150.         _sLogKey = CSS(LogKey.Value);
....
187.           + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
243.           ExportXml();
```

檔案名稱 WebEditor.aspx.cs

方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.         return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

檔案名稱 WebEditor.aspx.cs

方法 private void ExportXml()

```
....
819.               objDR1 = objDoc.Query("SELECT KmAction, OtherMessage,
DocXML FROM DocCatalog WITH(NOLOCK) WHERE DocID = '" + txt_DocID.Value +
"';");
```

Heuristic Parameter Tampering\路徑 61:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=407
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebEditor.aspx.cs 中第 115 行的 Page_Load 方法從 Value 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebEditor.aspx.cs 中第 724 行的 ExportXml 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	187	819

物件	Value	Query

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
187.           + CSS(UploadFilePath.Value).Replace("/", "\\").Trim('\\')
+ "\\";
....
185.         _sUploadFilePath =
Cdsys.KM.Utility.KMconfig.GetAppSetting("DataPATH", "C:/Temp",
"C:/Temp")
....
217.         objDoc = new Cdsys.KM.Utility.Doc(_sConnection, this);
....
243.           ExportXml();
```

檔案名稱 WebEditor.aspx.cs

方法 private string CSS(string strData)

```
....
2831.      private string CSS(string strData)
....
2833.         return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

檔案名稱 WebEditor.aspx.cs

方法 private void ExportXml()

```
....
819.               objDR1 = objDoc.Query("SELECT KmAction, OtherMessage,
DocXML FROM DocCatalog WITH(NOLOCK) WHERE DocID = '" + txt_DocID.Value +
"';");
```

Heuristic Parameter Tampering\路徑 62:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=408
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebGetGroupNo.ashx.cs 中第 21 行的 ProcessRequest 方法從 Params 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebGetGroupNo.ashx.cs 中第 21 行的 ProcessRequest 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

來源	目的地

檔案	WebGetGroupNo.ashx.cs	WebGetGroupNo.ashx.cs
行	25	29
物件	Params	Query

代碼片斷
檔案名稱 WebGetGroupNo.ashx.cs
方法 public void ProcessRequest(HttpContext context)

```
....
25.         string strGroupID =
HttpUtility.UrlDecode(context.Request.Params["ParmData"]);
....
29.         System.Data.SqlClient.SqlDataReader objDR1 =
objDB1.Query(string.Format("SELECT COUNT(*) FROM DocGroupCatalog
WITH(NOLOCK) WHERE GroupID = '{0}';", strGroupID));
```

Heuristic Parameter Tampering\路徑 63:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=409
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebMailA.aspx.cs 中第 20 行的 Page_Load 方法從 QueryString_DocID 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebMailA.aspx.cs 中第 20 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebMailA.aspx.cs	WebMailA.aspx.cs
行	38	59
物件	QueryString_DocID	Query

代碼片斷
檔案名稱 WebMailA.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
38.         strDocID =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g["DocID"], false)); // Ū�� URL ��ꌀ
....
59.         objDR1 = objDB1.Query(string.Format(strSQL1, strDocID));
```

Heuristic Parameter Tampering\路徑 64:

嚴重程度：	低風險

結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=410
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebMark.aspx.cs 中第 26 行的 Page_Load 方法從 QueryString_DocID 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebMark.aspx.cs 中第 76 行的 BuildBookList 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebMark.aspx.cs	WebMark.aspx.cs
行	34	81
物件	QueryString_DocID	Query

代碼片斷

檔案名稱 WebMark.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
34.         varDocID =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g["DocID"], false)); // Ū�� URL ��ɽ��
....
55.               this.divGroupMark.Style["display"] = "none";
BuildBookList();
```

檔案名稱 WebMark.aspx.cs

方法 private void BuildBookList()

```
....
81.         objDR1 = objDB1.Query(string.Format("SELECT DocTitle FROM
DocCatalog WITH(NOLOCK) WHERE DocID = '{0}';", varDocID));
```

Heuristic Parameter Tampering\路徑 65:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=411
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebMark.aspx.cs 中第 26 行的 Page_Load 方法從 QueryString_DocID 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebMark.aspx.cs 中第 95 行的 BuildGroupList 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebMark.aspx.cs	WebMark.aspx.cs
行	34	100
物件	QueryString_DocID	Query

代碼片斷

檔案名稱 WebMark.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
34.         varDocID =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g["DocID"], false)); // Ū��  URL  ��□ƪ��
....
59.                 BuildGroupList();
```

檔案名稱 WebMark.aspx.cs

方法 private void BuildGroupList()

```
....
100.         objDR1 = objDB1.Query(string.Format("SELECT DocTitle FROM
DocCatalog WITH(NOLOCK) WHERE DocID = '{0}';", varDocID));
```

Heuristic Parameter Tampering\路徑 66:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=412
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebNotePad.aspx.cs 中第 628 行的 Sub_GetRequest 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebNotePad.aspx.cs 中第 190 行的 Sub_SAY 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	632	239
物件	Form	Query

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
632.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
637.         return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
173.                   this.Sub_SAY(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtTitle", ""),
this.Sub_GetRequest("txtEssay", ""), "", "",
(objDoc.User.eMail.IndexOf("@") < 0 ? objDoc.User.Name :
objDoc.User.eMail), this.Sub_GetRequest("txtChk", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_SAY(string Action, string SysName, string ID, string Title, string Essay, string StartDate, string EndDate, string Commentator, string Chk)

```
....
190.       private void Sub_SAY(string Action, string SysName, string ID,
string Title, string Essay, string StartDate, string EndDate, string
Commentator, string Chk)
....
238.             strSQL = string.Format("SELECT MemoName, MemoContent,
CreateDate, CompanyNo, DepartmentNo, Account, Chk FROM GoodMemo WHERE ID
= '{0}'", ID);
239.             DR = this.objDoc.Query(strSQL);
```

Heuristic Parameter Tampering\路徑 67:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=413
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebNotePad.aspx.cs 中第 628 行的 Sub_GetRequest 方法從 QueryString_strName 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebNotePad.aspx.cs 中第 190 行的 Sub_SAY 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	635	239

物件	QueryString_strName	Query

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
635.            strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
637.        return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
173.                    this.Sub_SAY(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtTitle", ""),
this.Sub_GetRequest("txtEssay", ""), "", "",
(objDoc.User.eMail.IndexOf("@") < 0 ? objDoc.User.Name :
objDoc.User.eMail), this.Sub_GetRequest("txtChk", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_SAY(string Action, string SysName, string ID, string Title, string Essay, string StartDate, string EndDate, string Commentator, string Chk)

```
....
190.      private void Sub_SAY(string Action, string SysName, string ID,
string Title, string Essay, string StartDate, string EndDate, string
Commentator, string Chk)
....
238.            strSQL = string.Format("SELECT MemoName, MemoContent,
CreateDate, CompanyNo, DepartmentNo, Account, Chk FROM GoodMemo WHERE ID
= '{0}'", ID);
239.            DR = this.objDoc.Query(strSQL);
```

Heuristic Parameter Tampering\路徑 68:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=414
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebNotePad.aspx.cs 中第 628 行的 Sub_GetRequest 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebNotePad.aspx.cs 中第 190 行的 Sub_SAY 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	632	262
物件	Form	Query

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
632.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
637.         return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
173.                   this.Sub_SAY(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtTitle", ""),
this.Sub_GetRequest("txtEssay", ""), "", "",
(objDoc.User.eMail.IndexOf("@") < 0 ? objDoc.User.Name :
objDoc.User.eMail), this.Sub_GetRequest("txtChk", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_SAY(string Action, string SysName, string ID, string Title, string Essay, string StartDate, string EndDate, string Commentator, string Chk)

```
....
190.       private void Sub_SAY(string Action, string SysName, string ID,
string Title, string Essay, string StartDate, string EndDate, string
Commentator, string Chk)
....
261.             strSQL = string.Format("SELECT MemoName, MemoContent,
CreateDate, CompanyNo, DepartmentNo, Account, Chk FROM GoodMemo WHERE ID
= '{0}'", ID);
262.             DR = this.objDoc.Query(strSQL);
```

Heuristic Parameter Tampering\路徑 69:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=415
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebNotePad.aspx.cs 中第 628 行的 Sub_GetRequest 方法從 QueryString_strName 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebNotePad.aspx.cs 中第 190 行的 Sub_SAY 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	635	262
物件	QueryString_strName	Query

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
635.            strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
637.        return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
173.                  this.Sub_SAY(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtTitle", ""),
this.Sub_GetRequest("txtEssay", ""), "", "",
(objDoc.User.eMail.IndexOf("@") < 0 ? objDoc.User.Name :
objDoc.User.eMail), this.Sub_GetRequest("txtChk", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_SAY(string Action, string SysName, string ID, string Title, string Essay, string StartDate, string EndDate, string Commentator, string Chk)

```
....
190.      private void Sub_SAY(string Action, string SysName, string ID,
string Title, string Essay, string StartDate, string EndDate, string
Commentator, string Chk)
....
261.            strSQL = string.Format("SELECT MemoName, MemoContent,
CreateDate, CompanyNo, DepartmentNo, Account, Chk FROM GoodMemo WHERE ID
= '{0}'", ID);
262.            DR = this.objDoc.Query(strSQL);
```

Heuristic Parameter Tampering\路徑 70:

嚴重程度： 低風險
結果狀態： 校驗

線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=416
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebNotePad.aspx.cs 中第 628 行的 Sub_GetRequest 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebNotePad.aspx.cs 中第 287 行的 Sub_GBK 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	632	337
物件	Form	Query

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
632.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
637.         return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
147.                   this.Sub_GBK(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtBookName",
""), this.Sub_GetRequest("txtBookStore", ""),
this.Sub_GetRequest("txtBookSummary", ""),
this.Sub_GetRequest("txtStartDate", ""),
this.Sub_GetRequest("txtEndDate", ""), (objDoc.User.eMail.IndexOf("@") <
0 ? objDoc.User.Name : objDoc.User.eMail),
this.Sub_GetRequest("txtBookAuthor", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_GBK(string Action, string SysName, string ID, string BookName, string BookStore, string BookContent, string StartDate, string EndDate, string Commentator, string BookAuthor)

```
....
287.      private void Sub_GBK(string Action, string SysName, string ID,
string BookName, string BookStore, string BookContent, string StartDate,
string EndDate, string Commentator, string BookAuthor)
....
336.            strSQL = string.Format("SELECT BookName, BookStore,
BookContent, Sdate, Edate, CreateDate, CompanyNo, DepartmentNo, Account,
note1 FROM GoodBook WHERE ID = '{0}'", ID);
337.            DR = this.objDoc.Query(strSQL);
```

Heuristic Parameter Tampering\路徑 71:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=417
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebNotePad.aspx.cs 中第 628 行的 Sub_GetRequest 方法從 QueryString_strName 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebNotePad.aspx.cs 中第 287 行的 Sub_GBK 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	635	337
物件	QueryString_strName	Query

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
635.            strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
637.        return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
147.                    this.Sub_GBK(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtBookName",
""), this.Sub_GetRequest("txtBookStore", ""),
this.Sub_GetRequest("txtBookSummary", ""),
this.Sub_GetRequest("txtStartDate", ""),
this.Sub_GetRequest("txtEndDate", ""), (objDoc.User.eMail.IndexOf("@") <
0 ? objDoc.User.Name : objDoc.User.eMail),
this.Sub_GetRequest("txtBookAuthor", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_GBK(string Action, string SysName, string ID, string BookName, string BookStore, string BookContent, string StartDate, string EndDate, string Commentator, string BookAuthor)

```
....
287.      private void Sub_GBK(string Action, string SysName, string ID,
string BookName, string BookStore, string BookContent, string StartDate,
string EndDate, string Commentator, string BookAuthor)
....
336.            strSQL = string.Format("SELECT BookName, BookStore,
BookContent, Sdate, Edate, CreateDate, CompanyNo, DepartmentNo, Account,
note1 FROM GoodBook WHERE ID = '{0}'", ID);
337.            DR = this.objDoc.Query(strSQL);
```

Heuristic Parameter Tampering\路徑 72:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=418
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebNotePad.aspx.cs 中第 628 行的 Sub_GetRequest 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebNotePad.aspx.cs 中第 287 行的 Sub_GBK 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	632	360
物件	Form	Query

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
632.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
637.         return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
147.                   this.Sub_GBK(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtBookName",
""), this.Sub_GetRequest("txtBookStore", ""),
this.Sub_GetRequest("txtBookSummary", ""),
this.Sub_GetRequest("txtStartDate", ""),
this.Sub_GetRequest("txtEndDate", ""), (objDoc.User.eMail.IndexOf("@") <
0 ? objDoc.User.Name : objDoc.User.eMail),
this.Sub_GetRequest("txtBookAuthor", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_GBK(string Action, string SysName, string ID, string BookName, string BookStore, string BookContent, string StartDate, string EndDate, string Commentator, string BookAuthor)

```
....
287.       private void Sub_GBK(string Action, string SysName, string ID,
string BookName, string BookStore, string BookContent, string StartDate,
string EndDate, string Commentator, string BookAuthor)
....
359.             strSQL = string.Format("SELECT BookName, BookStore,
BookContent, Sdate, Edate, CreateDate, CompanyNo, DepartmentNo, Account,
note1 FROM GoodBook WHERE ID = '{0}'", ID);
360.             DR = this.objDoc.Query(strSQL);
```

Heuristic Parameter Tampering\路徑 73:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=419
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebNotePad.aspx.cs 中第 628 行的 Sub_GetRequest 方法從 QueryString_strName 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebNotePad.aspx.cs 中第 287 行的 Sub_GBK 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

來源	目的地

檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	635	360
物件	QueryString_strName	Query

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
635.            strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
637.        return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
147.                    this.Sub_GBK(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtBookName",
""), this.Sub_GetRequest("txtBookStore", ""),
this.Sub_GetRequest("txtBookSummary", ""),
this.Sub_GetRequest("txtStartDate", ""),
this.Sub_GetRequest("txtEndDate", ""), (objDoc.User.eMail.IndexOf("@") <
0 ? objDoc.User.Name : objDoc.User.eMail),
this.Sub_GetRequest("txtBookAuthor", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_GBK(string Action, string SysName, string ID, string BookName, string BookStore, string BookContent, string StartDate, string EndDate, string Commentator, string BookAuthor)

```
....
287.      private void Sub_GBK(string Action, string SysName, string ID,
string BookName, string BookStore, string BookContent, string StartDate,
string EndDate, string Commentator, string BookAuthor)
....
359.            strSQL = string.Format("SELECT BookName, BookStore,
BookContent, Sdate, Edate, CreateDate, CompanyNo, DepartmentNo, Account,
note1 FROM GoodBook WHERE ID = '{0}'", ID);
360.            DR = this.objDoc.Query(strSQL);
```

Heuristic Parameter Tampering\路徑 74:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=420
狀態	反覆出現的問題

Detection Date	7/8/2022 3:05:06 PM

WebPrint.aspx.cs 中第 143 行的 GetRequest 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebPrint.aspx.cs 中第 508 行的 PrintDoc 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	147	532
物件	Form	Query

代碼片斷

檔案名稱 WebPrint.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
147.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
152.         return strResult.Trim();
```

檔案名稱 WebPrint.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
53.         LogKey .Value = _sLogKey = GetRequest("LogKey", ""); // Ū��
URL ��ƪ��
```

檔案名稱 WebPrint.aspx

方法

```
....
154.
```

檔案名稱 WebPrint.aspx.cs

方法 protected void btn_Print_Click(object sender, System.EventArgs e)

```
....
414.         ChangeSelect();
....
416.         SaveSelect();
417.         PrintDoc();
```

檔案名稱 WebPrint.aspx.cs

方法 private void PrintDoc()

```
....
512.         string sDocIDs = objDoc.LoadData(_sLogKey, "PrintIDs");
513.         sDocIDs = sDocIDs.Replace("@", ",");
....
517.         string[] aDocid = sDocIDs.Split(',');
....
525.           Cdsys.KM.Utility.QueryDocData objQDoc =
objDoc.QueryDocumentXML(aDocid[i]); // 95/04/27
....
532.             objDR = objDoc.Query("SELECT * FROM DocDefMain WHERE
DocDefID = '" + objQDoc.DocDefID + "'");
```

Heuristic Parameter Tampering\路徑 75:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=421
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebPrint.aspx.cs 中第 143 行的 GetRequest 方法從 QueryString_strName 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebPrint.aspx.cs 中第 508 行的 PrintDoc 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	150	532
物件	QueryString_strName	Query

代碼片斷

檔案名稱 WebPrint.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
150.             strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
152.         return strResult.Trim();
```

檔案名稱 WebPrint.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
53.         LogKey .Value = _sLogKey = GetRequest("LogKey", ""); // Ū��
URL ��□ع��
```

檔案名稱 WebPrint.aspx

方法

```
....
154.
```

檔案名稱 WebPrint.aspx.cs

方法 protected void btn_Print_Click(object sender, System.EventArgs e)

```
....
414.         ChangeSelect();
....
416.         SaveSelect();
417.         PrintDoc();
```

檔案名稱 WebPrint.aspx.cs

方法 private void PrintDoc()

```
....
512.         string sDocIDs = objDoc.LoadData(_sLogKey, "PrintIDs");
513.         sDocIDs = sDocIDs.Replace("@", ",");
....
517.         string[] aDocid = sDocIDs.Split(',');
....
525.           Cdsys.KM.Utility.QueryDocData objQDoc =
objDoc.QueryDocumentXML(aDocid[i]); // 95/04/27
....
532.             objDR = objDoc.Query("SELECT * FROM DocDefMain WHERE
DocDefID = '" + objQDoc.DocDefID + "'");
```

Heuristic Parameter Tampering\路徑 76:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=422
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebPrint.aspx.cs 中第 508 行的 PrintDoc 方法從 AbsoluteUri 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebPrint.aspx.cs 中第 508 行的 PrintDoc 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	514	532
物件	AbsoluteUri	Query

代碼片斷

檔案名稱 WebPrint.aspx.cs

方法 private void PrintDoc()

```
....
514.         string sUrl = CSS(Request.Url.AbsoluteUri);
515.         objDoc.URL = sUrl = sUrl.Substring(0, sUrl.LastIndexOf("/")
+ 1);
....
525.           Cdsys.KM.Utility.QueryDocData objQDoc =
objDoc.QueryDocumentXML(aDocid[i]); // 95/04/27
....
532.             objDR = objDoc.Query("SELECT * FROM DocDefMain WHERE
DocDefID = '" + objQDoc.DocDefID + "'");
```

檔案名稱 WebPrint.aspx.cs

方法 private string CSS(string strData)

```
....
939.       private string CSS(string strData)
....
941.         return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic Parameter Tampering\路徑 77:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=423
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebPrint.aspx.cs 中第 143 行的 GetRequest 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebPrint.aspx.cs 中第 597 行的 SaveDocHTML 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	147	655
物件	Form	Query

代碼片斷

檔案名稱 WebPrint.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
147.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
152.         return strResult.Trim();
```

檔案名稱 WebPrint.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
53.         LogKey .Value = _sLogKey = GetRequest("LogKey", ""); // Ū��
URL ��ع��
```

檔案名稱 WebPrint.aspx

方法

```
....
154.
```

檔案名稱 WebPrint.aspx.cs

方法 protected void btn_SaveHTML_Click(object sender, System.EventArgs e)

```
....
466.         ChangeSelect();
....
468.         SaveSelect();
469.         SaveDocHTML();
```

檔案名稱 WebPrint.aspx.cs

方法 private void SaveDocHTML()

```
....
599.         UI(false);
....
615.         string sDocIDs = objDoc.LoadData(_sLogKey, "PrintIDs");
616.         sDocIDs = sDocIDs.Replace("@", ",");
....
620.         string[] aDocid = sDocIDs.Split(',');
....
628.           Cdsys.KM.Utility.QueryDocData objQDoc =
objDoc.QueryDocumentXML(aDocid[i]); // 95/04/27
....
655.             objDR = objDoc.Query("SELECT * FROM DocDefMain WHERE
DocDefID = '" + objQDoc.DocDefID + "'");
```

Heuristic Parameter Tampering\路徑 78:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=424
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebPrint.aspx.cs 中第 143 行的 GetRequest 方法從 QueryString_strName 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebPrint.aspx.cs 中第 597 行的 SaveDocHTML 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	150	655
物件	QueryString_strName	Query

代碼片斷

檔案名稱 WebPrint.aspx.cs

方法 private string GetRequest(string strName, string strDefault)

```
....
150.             strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
152.         return strResult.Trim();
```

檔案名稱 WebPrint.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
53.         LogKey .Value = _sLogKey = GetRequest("LogKey", ""); // Ū��
URL ��□ع��
```

檔案名稱 WebPrint.aspx

方法

```
....
154.
```

檔案名稱 WebPrint.aspx.cs

方法 protected void btn_SaveHTML_Click(object sender, System.EventArgs e)

```
....
466.         ChangeSelect();
....
468.         SaveSelect();
469.         SaveDocHTML();
```

檔案名稱 WebPrint.aspx.cs

方法 private void SaveDocHTML()

```
....
599.         UI(false);
....
615.         string sDocIDs = objDoc.LoadData(_sLogKey, "PrintIDs");
616.         sDocIDs = sDocIDs.Replace("@", ",");
....
620.         string[] aDocid = sDocIDs.Split(',');
....
628.           Cdsys.KM.Utility.QueryDocData objQDoc =
objDoc.QueryDocumentXML(aDocid[i]); // 95/04/27
....
655.             objDR = objDoc.Query("SELECT * FROM DocDefMain WHERE
DocDefID = '" + objQDoc.DocDefID + "'");
```

Heuristic Parameter Tampering\路徑 79:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=425
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebPrint.aspx.cs 中第 597 行的 SaveDocHTML 方法從 AbsoluteUri 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾

的情下被 WebPrint.aspx.cs 中第 597 行的 SaveDocHTML 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	617	655
物件	AbsoluteUri	Query

代碼片斷

檔案名稱 WebPrint.aspx.cs

方法 private void SaveDocHTML()

```
....
617.         string sUrl = CSS(Request.Url.AbsoluteUri);
618.         objDoc.URL = sUrl = sUrl.Substring(0, sUrl.LastIndexOf("/")
+ 1);
....
628.           Cdsys.KM.Utility.QueryDocData objQDoc =
objDoc.QueryDocumentXML(aDocid[i]); // 95/04/27
....
655.             objDR = objDoc.Query("SELECT * FROM DocDefMain WHERE
DocDefID = '" + objQDoc.DocDefID + "'");
```

檔案名稱 WebPrint.aspx.cs

方法 private string CSS(string strData)

```
....
939.      private string CSS(string strData)
....
941.         return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

Heuristic Parameter Tampering\路徑 80:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=426
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebRedirect.aspx.cs 中第 140 行的 Sub_Request 方法從 QueryString_DocID 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebRedirect.aspx.cs 中第 77 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebRedirect.aspx.cs	WebRedirect.aspx.cs

行	152	87
物件	QueryString_DocID	Query

代碼片斷

檔案名稱 WebRedirect.aspx.cs

方法 private void Sub_Request()

```
....
152.         strA =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g["DocID"], false));
153.         varDocID = (strA == null) ? "" : strA.Trim();
....
168.         varURL = Sub_CheckURL(strA, varDocID, varKey);
```

檔案名稱 WebRedirect.aspx.cs

方法 private string Sub_CheckURL(string URL, string DocID, string Key)

```
....
251.      private string Sub_CheckURL(string URL, string DocID, string
Key)
....
256.           URL = "ShowDocument.aspx?DocID=" + DocID + "&LogKey=" +
Key + "&SysFrom=" + varSysFrom; // 95/01/12
....
280.         return URL.Replace("http://http://", "http://");
```

檔案名稱 WebRedirect.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
79.         Sub_Request();
80.         Sub_VerifyUser();
....
82.         if(varURL.ToLower().StartsWith("http"))
....
84.           string[] aryTemp = varURL.Replace("\\", "/").Split('/');
85.           string strFileName = aryTemp[aryTemp.Length - 1];
....
87.           objDR1 = DBA.Query("SELECT TOP 1 Filename FROM
DocUploadFile WITH(NOLOCK) WHERE Rfilename = N'" + strFileName + "';");
```

檔案名稱 WebRedirect.aspx.cs

方法 public void Sub_VerifyUser()

```
....
136.         Sub_ConvertUrl();
```

Heuristic Parameter Tampering\路徑 81:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=427
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebRedirect.aspx.cs 中第 140 行的 Sub_Request 方法從 QueryString_Key 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebRedirect.aspx.cs 中第 77 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebRedirect.aspx.cs	WebRedirect.aspx.cs
行	149	87
物件	QueryString_Key	Query

代碼片斷	
檔案名稱	WebRedirect.aspx.cs
方法	private void Sub_Request()

```
....
149.         strA =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g["Key"], false));
150.         varKey = (strA == null) ? "" : strA.Trim();
....
168.         varURL = Sub_CheckURL(strA, varDocID, varKey);
```

檔案名稱	WebRedirect.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
79.         Sub_Request();
80.         Sub_VerifyUser();
....
82.         if(varURL.ToLower().StartsWith("http"))
....
84.           string[] aryTemp = varURL.Replace("\\", "/").Split('/');
85.           string strFileName = aryTemp[aryTemp.Length - 1];
....
87.           objDR1 = DBA.Query("SELECT TOP 1 Filename FROM
DocUploadFile WITH(NOLOCK) WHERE Rfilename = N'" + strFileName + "';");
```

檔案名稱	WebRedirect.aspx.cs
方法	public void Sub_VerifyUser()

```
....
136.        Sub_ConvertUrl();
```

檔案名稱 WebRedirect.aspx.cs

方法 private void Sub_ConvertUrl()

```
....
120.        sTemp = sTemp.Replace("@logkey", varKey);
121.        varURL = Server.UrlEncode(sTemp);
```

Heuristic Parameter Tampering\路徑 82:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=428
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebRedirect.aspx.cs 中第 140 行的 Sub_Request 方法從 QueryString_SysFrom 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebRedirect.aspx.cs 中第 77 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebRedirect.aspx.cs	WebRedirect.aspx.cs
行	146	87
物件	QueryString_SysFrom	Query

代碼片斷

檔案名稱 WebRedirect.aspx.cs

方法 private void Sub_Request()

```
....
146.        strA =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g["SysFrom"], false));
147.        varSysFrom = (strA == null) ? "" : strA.Trim();
....
168.        varURL = Sub_CheckURL(strA, varDocID, varKey);
```

檔案名稱 WebRedirect.aspx.cs

方法 private string Sub_CheckURL(string URL, string DocID, string Key)

```
....
256.          URL = "ShowDocument.aspx?DocID=" + DocID + "&LogKey=" +
Key + "&SysFrom=" + varSysFrom; // 95/01/12
....
280.         return URL.Replace("http://http://", "http://");
```

檔案名稱 WebRedirect.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
79.         Sub_Request();
80.         Sub_VerifyUser();
....
82.         if(varURL.ToLower().StartsWith("http"))
....
84.           string[] aryTemp = varURL.Replace("\\", "/").Split('/');
85.           string strFileName = aryTemp[aryTemp.Length - 1];
....
87.           objDR1 = DBA.Query("SELECT TOP 1 Filename FROM
DocUploadFile WITH(NOLOCK) WHERE Rfilename = N'" + strFileName + "';");
```

檔案名稱 WebRedirect.aspx.cs

方法 public void Sub_VerifyUser()

```
....
136.         Sub_ConvertUrl();
```

Heuristic Parameter Tampering\路徑 83:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=429
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebRedirect.aspx.cs 中第 140 行的 Sub_Request 方法從 QueryString_URL 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebRedirect.aspx.cs 中第 77 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebRedirect.aspx.cs	WebRedirect.aspx.cs
行	155	87
物件	QueryString_URL	Query

代碼片斷

檔案名稱 WebRedirect.aspx.cs
方法 private void Sub_Request()

```
....
155.         strA =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g["URL"], false));
156.         strA = (strA == null) ? "" : strA.Trim();
....
168.         varURL = Sub_CheckURL(strA, varDocID, varKey);
```

檔案名稱 WebRedirect.aspx.cs
方法 private string Sub_CheckURL(string URL, string DocID, string Key)

```
....
251.      private string Sub_CheckURL(string URL, string DocID, string
Key)
....
280.         return URL.Replace("http://http://", "http://");
```

檔案名稱 WebRedirect.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
79.         Sub_Request();
80.         Sub_VerifyUser();
....
82.         if(varURL.ToLower().StartsWith("http"))
....
84.           string[] aryTemp = varURL.Replace("\\", "/").Split('/');
85.           string strFileName = aryTemp[aryTemp.Length - 1];
....
87.           objDR1 = DBA.Query("SELECT TOP 1 Filename FROM
DocUploadFile WITH(NOLOCK) WHERE Rfilename = N'" + strFileName + "';");
```

檔案名稱 WebRedirect.aspx.cs
方法 public void Sub_VerifyUser()

```
....
136.         Sub_ConvertUrl();
```

Heuristic Parameter Tampering\路徑 84:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=430
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebRedirect.aspx.cs 中第 206 行的 TransferFileToUrl 方法從 ToString 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebRedirect.aspx.cs 中第 77 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebRedirect.aspx.cs	WebRedirect.aspx.cs
行	211	87
物件	ToString	Query

代碼片斷

檔案名稱 WebRedirect.aspx.cs

方法 private string TransferFileToUrl(string DocID, string strFileName)

```
....
211.        string varURL = CSS(Request.Url.ToString());
....
213.        string strURL = varURL.Substring(0, varURL.LastIndexOf("/")
+ 1); // ﷺtﷺ (/)
....
234.            strFileUpURL = strURL + aryTemp[aryTemp.Length - 1];
....
244.            if(System.IO.File.Exists(sPath)) return strFileUpURL +
"/" + strFileName;
```

檔案名稱 WebRedirect.aspx.cs

方法 private string CSS(string strData)

```
....
285.      private string CSS(string strData)
....
287.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

檔案名稱 WebRedirect.aspx.cs

方法 private string Sub_CheckURL(string URL, string DocID, string Key)

```
....
272.            URL = TransferFileToUrl(DocID, sFileName);
....
280.        return URL.Replace("http://http://", "http://");
```

檔案名稱 WebRedirect.aspx.cs

方法 private void Sub_Request()

```
....
168.         varURL = Sub_CheckURL(strA, varDocID, varKey);
```

檔案名稱 WebRedirect.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
79.         Sub_Request();
80.         Sub_VerifyUser();
....
82.         if(varURL.ToLower().StartsWith("http"))
....
84.           string[] aryTemp = varURL.Replace("\\", "/").Split('/');
85.           string strFileName = aryTemp[aryTemp.Length - 1];
....
87.           objDR1 = DBA.Query("SELECT TOP 1 Filename FROM
DocUploadFile WITH(NOLOCK) WHERE Rfilename = N'" + strFileName + "';");
```

檔案名稱 WebRedirect.aspx.cs

方法 public void Sub_VerifyUser()

```
....
136.         Sub_ConvertUrl();
```

Heuristic Parameter Tampering\路徑 85:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=431
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebRedirect.aspx.cs 中第 206 行的 TransferFileToUrl 方法從 Url 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebRedirect.aspx.cs 中第 77 行的 Page_Load 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebRedirect.aspx.cs	WebRedirect.aspx.cs
行	211	87
物件	Url	Query

代碼片斷

檔案名稱 WebRedirect.aspx.cs

方法 private string TransferFileToUrl(string DocID, string strFileName)

```
....
211.        string varURL = CSS(Request.Url.ToString());
....
213.        string strURL = varURL.Substring(0, varURL.LastIndexOf("/")
+ 1); // ﷺtﷺ (/)
....
234.            strFileUpURL = strURL + aryTemp[aryTemp.Length - 1];
....
244.            if(System.IO.File.Exists(sPath)) return strFileUpURL +
"/" + strFileName;
```

檔案名稱 WebRedirect.aspx.cs

方法 private string CSS(string strData)

```
....
285.      private string CSS(string strData)
....
287.        return
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(strData, false));
```

檔案名稱 WebRedirect.aspx.cs

方法 private string Sub_CheckURL(string URL, string DocID, string Key)

```
....
272.            URL = TransferFileToUrl(DocID, sFileName);
....
280.        return URL.Replace("http://http://", "http://");
```

檔案名稱 WebRedirect.aspx.cs

方法 private void Sub_Request()

```
....
168.        varURL = Sub_CheckURL(strA, varDocID, varKey);
```

檔案名稱 WebRedirect.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
79.         Sub_Request();
80.         Sub_VerifyUser();
....
82.         if(varURL.ToLower().StartsWith("http"))
....
84.           string[] aryTemp = varURL.Replace("\\", "/").Split('/');
85.           string strFileName = aryTemp[aryTemp.Length - 1];
....
87.           objDR1 = DBA.Query("SELECT TOP 1 Filename FROM
DocUploadFile WITH(NOLOCK) WHERE Rfilename = N'" + strFileName + "';");
```

檔案名稱	WebRedirect.aspx.cs
方法	public void Sub_VerifyUser()

```
....
136.         Sub_ConvertUrl();
```

Heuristic Parameter Tampering\路徑 86:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=432
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebRedirect.aspx.cs 中第 140 行的 Sub_Request 方法從 QueryString_Key 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebRedirect.aspx.cs 中第 186 行的 Sub_CheckKey 方法用來與 Query 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebRedirect.aspx.cs	WebRedirect.aspx.cs
行	149	191
物件	QueryString_Key	Query

代碼片斷

檔案名稱	WebRedirect.aspx.cs
方法	private void Sub_Request()

```
....
149.         strA =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g["Key"], false));
150.         varKey = (strA == null) ? "" : strA.Trim();
....
168.         varURL = Sub_CheckURL(strA, varDocID, varKey);
....
180.           this.Sub_CheckKey(varKey);
```

檔案名稱	WebRedirect.aspx.cs
方法	private bool Sub_CheckKey(string Key)

```
....
186.       private bool Sub_CheckKey(string Key)
....
190.         string strSQL = "SELECT UID FROM UserProcess WHERE (LogKey =
'" + Key + "')";
191.         System.Data.SqlClient.SqlDataReader DR = DBA.Query(strSQL);
```

Heuristic Parameter Tampering\路徑 87:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=433
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:06 PM

WebCatalog.aspx.cs 中第 1829 行的 Sub_GetRequest 方法從 Form 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebCatalog.aspx.cs 中第 3904 行的 Sub_PersonalSetting 方法用來與 Insert 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	4232
物件	Form	Insert

代碼片斷

檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 public void Sub_PersonalSetting(int Action)

```
....
3976.            strWordA = this.Sub_GetRequest("txtPasswordA",
"").Trim();
....
3990.              objSB1.Append("UPDATE AppUser WITH(READPAST) SET
LoginPWD = '" + strWordA.Replace("'", "''") + "' WHERE UID ='" +
WhoAmI.UserID + "';");
....
3993.                this.DOC.Execute(objSB1.ToString());
....
4232.        objSB1.Insert(0, "<SELECT NAME='listDocItems90'
ID='listDocItems90'
onkeypress=\"javascript:switch(window.event.keyCode){case
43:MoveDnOptionItemA('listDocItems90');break;case
45:MoveUpOptionItemA('listDocItems90');break;}try { event.preventDefault
? event.preventDefault() : (event.returnValue = false); } catch (err) {
return false; }\" SIZE=" + k.ToString() + ">");
```

Heuristic Parameter Tampering\路徑 88:

嚴重程度： 低風險

結果狀態： 校驗

線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=434

狀態 反覆出現的問題

Detection Date 7/8/2022 3:05:06 PM

WebCatalog.aspx.cs 中第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 接收使用者輸入變數，這個輸入變數之後在沒有驗證的狀況下被應用程式直接與其他字串連接成 SQL 指定字串變數，這個字串接著在沒有經過資料庫過濾的情下被 WebCatalog.aspx.cs 中第 3904 行的 Sub_PersonalSetting 方法用來與 Insert 資料庫進行查詢，這會讓能夠使用者篡改過濾參數。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	4232
物件	QueryString_Name	Insert

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱	WebCatalog.aspx.cs
方法	public void Sub_PersonalSetting(int Action)

```
....
3976.            strWordA = this.Sub_GetRequest("txtPasswordA",
"").Trim();
....
3990.              objSB1.Append("UPDATE AppUser WITH(READPAST) SET
LoginPWD = '" + strWordA.Replace("'", "''") + "' WHERE UID ='" +
WhoAmI.UserID + "';");
....
3993.                this.DOC.Execute(objSB1.ToString());
....
4232.        objSB1.Insert(0, "<SELECT NAME='listDocItems90'
ID='listDocItems90'
onkeypress=\"javascript:switch(window.event.keyCode){case
43:MoveDnOptionItemA('listDocItems90');break;case
45:MoveUpOptionItemA('listDocItems90');break;}try { event.preventDefault
? event.preventDefault() : (event.returnValue = false); } catch (err) {
return false; }\" SIZE=" + k.ToString() + ">");
```

Heuristic CSRF

查詢路徑:

CSharp\Cx\CSharp Heuristic\Heuristic CSRF 版本:4

類別

PCI DSS v3.2.1: PCI DSS (3.2.1) - 6.5.9 - Cross-site request forgery
OWASP Top 10 2013: A8-Cross-Site Request Forgery (CSRF)
NIST SP 800-53: SC-23 Session Authenticity (P1)
ASD STIG 4.10: APSC-DV-002500 - CAT II The application must protect from Cross-Site Request Forgery (CSRF) vulnerabilities.
OWASP Top 10 2021: A1-Broken Access Control

描述

Heuristic CSRF\路徑 1:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=40
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebCatalog.aspx.cs第1829行，Sub_GetRequest 方法從Form 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	1891
物件	Form	Execute

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_NewsLetter(int Action)

```
....
2158.         string strEMailB = this.Sub_GetRequest("txtEMailB", "");
....
2209.         strB = this.Sub_UpdateNewsLetter(strMode, "最新文件快訊",
strEMailB, "AppKMNewOrder", ref strOrderDoc) + "<br />";
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateNewsLetter(string Mode, string Name, string Mail, string Table, ref string strOrderDoc)

```
....
1863.      private string Sub_UpdateNewsLetter(string Mode, string Name,
string Mail, string Table, ref string strOrderDoc)
....
1889.                 strSQL = string.Format(fmtSQL3, Table, Mail);
....
1891.                 if(this.DOC.Execute(strSQL) == 0) strMSG =
"目前不在寄送名單中。"; else strMSG = "不會再寄送。";
```

Heuristic CSRF\路徑 2:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=41
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebCatalog.aspx.cs第1829行，Sub_GetRequest 方法從QueryString_Name 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	1891

物件	QueryString_Name	Execute

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_NewsLetter(int Action)

```
....
2158.        string strEMailB = this.Sub_GetRequest("txtEMailB", "");
....
2209.        strB = this.Sub_UpdateNewsLetter(strMode, "最新文件快訊",
strEMailB, "AppKMNewOrder", ref strOrderDoc) + "<br />";
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateNewsLetter(string Mode, string Name, string Mail, string Table, ref string strOrderDoc)

```
....
1863.      private string Sub_UpdateNewsLetter(string Mode, string Name,
string Mail, string Table, ref string strOrderDoc)
....
1889.                strSQL = string.Format(fmtSQL3, Table, Mail);
....
1891.                if(this.DOC.Execute(strSQL) == 0) strMSG =
"目前不在寄送名單中。"; else strMSG = "不會再寄送。";
```

Heuristic CSRF\路徑 3:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=42
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebCatalog.aspx.cs第1169行，Sub_Group 方法從Value 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs

行	1230	2886
物件	Value	RegisterStartupScript

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Group(int Action)

```
....
1230.                    string[] aryTmp1 =
this.StackHiddenData.Value.Replace("|,|", "\u0001").Split('\u0001');
1231.                    this.StackHiddenData.Value = string.Join("|,|",
aryTmp1, 1, aryTmp1.Length - 1);
....
1239.                varAction = 400; this.Sub_400();
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_400()

```
....
2904.           this.SendAlert("此篇文件不存在。", "window.history.go(-
1);");
```

檔案名稱 WebCatalog.aspx.cs

方法 private void SendAlert(string strMsg, string AppCMD)

```
....
2886.         this.ClientScript.RegisterStartupScript(this.GetType(),
"cds_Alert", strJava);
```

Heuristic CSRF\路徑 4:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=43
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebCatalog.aspx.cs第1829行，Sub_GetRequest 方法從Form 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	2886
物件	Form	RegisterStartupScript

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_FolderMtn(int Action)

```
....
639.         string strFolderName = this.Sub_GetRequest("txtFolder",
"").Replace(",", ", ");
....
692.               this.SendAlert(string.Format("開放區域：{0},\n成功刪除！",
strFolderName), "");
```

檔案名稱 WebCatalog.aspx.cs

方法 private void SendAlert(string strMsg, string AppCMD)

```
....
2882.       private void SendAlert(string strMsg, string AppCMD)
....
2884.         string strJava = "\n<script>window.alert(\"" +
strMsg.Replace("\"", "\\\"").TrimEnd('\n').Replace("\r",
"").Replace("\n", "\\n") + "\");" + AppCMD + "</script>\n";
....
2886.         this.ClientScript.RegisterStartupScript(this.GetType(),
"cds_Alert", strJava);
```

Heuristic CSRF\路徑 5:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=44
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebCatalog.aspx.cs第1829行，Sub_GetRequest 方法從QueryString_Name 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	2886

物件	QueryString_Name	RegisterStartupScript

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_FolderMtn(int Action)

```
....
639.        string strFolderName = this.Sub_GetRequest("txtFolder",
"").Replace(",", ", ");
....
692.             this.SendAlert(string.Format("開放區域：{0},\n成功刪除！",
strFolderName), "");
```

檔案名稱 WebCatalog.aspx.cs

方法 private void SendAlert(string strMsg, string AppCMD)

```
....
2882.      private void SendAlert(string strMsg, string AppCMD)
....
2884.        string strJava = "\n<script>window.alert(\"" +
strMsg.Replace("\"", "\\\"").TrimEnd('\n').Replace("\r",
"").Replace("\n", "\\n") + "\");" + AppCMD + "</script>\n";
....
2886.        this.ClientScript.RegisterStartupScript(this.GetType(),
"cds_Alert", strJava);
```

Heuristic CSRF\路徑 6:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=45
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebCatalog.aspx.cs第632行，Sub_FolderMtn 方法從Form 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

來源	目的地

檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	675	2886
物件	Form	RegisterStartupScript

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_FolderMtn(int Action)

```
....
675.            strFolderName =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form["strFolder
Name"], false));
....
692.              this.SendAlert(string.Format("開放區域：{0},\n成功刪除！",
strFolderName), "");
```

檔案名稱 WebCatalog.aspx.cs

方法 private void SendAlert(string strMsg, string AppCMD)

```
....
2882.      private void SendAlert(string strMsg, string AppCMD)
....
2884.        string strJava = "\n<script>window.alert(\"" +
strMsg.Replace("\"", "\\\"").TrimEnd('\n').Replace("\r",
"").Replace("\n", "\\n") + "\");" + AppCMD + "</script>\n";
....
2886.        this.ClientScript.RegisterStartupScript(this.GetType(),
"cds_Alert", strJava);
```

Heuristic CSRF\路徑 7:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=46
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebCatalog.aspx.cs第1829行，Sub_GetRequest 方法從Form 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	1922
物件	Form	Execute

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法	private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱	WebCatalog.aspx.cs
方法	private void Sub_NewsLetter(int Action)

```
....
2158.         string strEMailB = this.Sub_GetRequest("txtEMailB", "");
....
2209.         strB = this.Sub_UpdateNewsLetter(strMode, "最新文件快訊",
strEMailB, "AppKMNewOrder", ref strOrderDoc) + "<br />";
```

檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_UpdateNewsLetter(string Mode, string Name, string Mail, string Table, ref string strOrderDoc)

```
....
1863.      private string Sub_UpdateNewsLetter(string Mode, string Name,
string Mail, string Table, ref string strOrderDoc)
....
1920.             strSQL = string.Format(fmtSQL3, Table, Mail);
....
1922.             if(this.DOC.Execute(strSQL) == 0) strMSG =
"目前不在寄送名單中。"; else strMSG = "不會再寄送。";
```

Heuristic CSRF\路徑 8:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=47
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebCatalog.aspx.cs第1829行，Sub_GetRequest 方法從QueryString_Name 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	1922
物件	QueryString_Name	Execute

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_NewsLetter(int Action)

```
....
2158.        string strEMailB = this.Sub_GetRequest("txtEMailB", "");
....
2209.        strB = this.Sub_UpdateNewsLetter(strMode, "最新文件快訊",
strEMailB, "AppKMNewOrder", ref strOrderDoc) + "<br />";
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateNewsLetter(string Mode, string Name, string Mail, string Table, ref string strOrderDoc)

```
....
1863.      private string Sub_UpdateNewsLetter(string Mode, string Name,
string Mail, string Table, ref string strOrderDoc)
....
1920.            strSQL = string.Format(fmtSQL3, Table, Mail);
....
1922.            if(this.DOC.Execute(strSQL) == 0) strMSG =
"目前不在寄送名單中。"; else strMSG = "不會再寄送。";
```

Heuristic CSRF\路徑 9:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=48
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebCatalog.aspx.cs第1829行，Sub_GetRequest 方法從Form 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	1976

物件	Form	Execute

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_ChiefLetter(int Action)

```
....
2048.         string strEMailC = this.Sub_GetRequest("txtEMailC", "");
....
2080.         strA = this.Sub_UpdateChiefLetter(strMode, "首長電子報",
strEMailC, strTitleC);
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateChiefLetter(string Mode, string Name, string Mail, string Title)

```
....
1955.       private string Sub_UpdateChiefLetter(string Mode, string
Name, string Mail, string Title)
....
1975.               strSQL = string.Format(fmtSQL1, Title, Mail,
WhoAmI.UserID, _sUporg);
1976.               if(this.DOC.Execute(strSQL) == 0)
```

Heuristic CSRF\路徑 10:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=49
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebCatalog.aspx.cs第1829行，Sub_GetRequest 方法從QueryString_Name 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	1976

物件	QueryString_Name	Execute

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_ChiefLetter(int Action)

```
....
2048.        string strEMailC = this.Sub_GetRequest("txtEMailC", "");
....
2080.        strA = this.Sub_UpdateChiefLetter(strMode, "首長電子報",
strEMailC, strTitleC);
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateChiefLetter(string Mode, string Name, string Mail, string Title)

```
....
1955.      private string Sub_UpdateChiefLetter(string Mode, string
Name, string Mail, string Title)
....
1975.              strSQL = string.Format(fmtSQL1, Title, Mail,
WhoAmI.UserID, _sUporg);
1976.              if(this.DOC.Execute(strSQL) == 0)
```

Heuristic CSRF\路徑 11:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=50
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebCatalog.aspx.cs第1829行，Sub_GetRequest 方法從Form 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	1972

物件	Form	Execute

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_ChiefLetter(int Action)

```
....
2048.        string strEMailC = this.Sub_GetRequest("txtEMailC", "");
....
2080.        strA = this.Sub_UpdateChiefLetter(strMode, "首長電子報",
strEMailC, strTitleC);
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateChiefLetter(string Mode, string Name, string Mail, string Title)

```
....
1955.      private string Sub_UpdateChiefLetter(string Mode, string
Name, string Mail, string Title)
....
1971.            strSQL = string.Format(fmtSQL2, Title, Mail,
WhoAmI.UserID, _sUporg);
1972.            if(this.DOC.Execute(strSQL) == 0)
```

Heuristic CSRF\路徑 12:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=51
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebCatalog.aspx.cs第1829行，Sub_GetRequest 方法從QueryString_Name 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	1972

物件	QueryString_Name	Execute

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_ChiefLetter(int Action)

```
....
2048.        string strEMailC = this.Sub_GetRequest("txtEMailC", "");
....
2080.        strA = this.Sub_UpdateChiefLetter(strMode, "首長電子報",
strEMailC, strTitleC);
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateChiefLetter(string Mode, string Name, string Mail, string Title)

```
....
1955.      private string Sub_UpdateChiefLetter(string Mode, string
Name, string Mail, string Title)
....
1971.            strSQL = string.Format(fmtSQL2, Title, Mail,
WhoAmI.UserID, _sUporg);
1972.            if(this.DOC.Execute(strSQL) == 0)
```

Heuristic CSRF\路徑 13:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=52
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebCatalog.aspx.cs第1829行，Sub_GetRequest 方法從Form 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	1993

物件	Form	Execute

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_ChiefLetter(int Action)

```
....
2048.        string strEMailC = this.Sub_GetRequest("txtEMailC", "");
....
2080.        strA = this.Sub_UpdateChiefLetter(strMode, "首長電子報",
strEMailC, strTitleC);
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateChiefLetter(string Mode, string Name, string Mail, string Title)

```
....
1955.      private string Sub_UpdateChiefLetter(string Mode, string
Name, string Mail, string Title)
....
1991.            strSQL = string.Format(fmtSQL3, Mail);
....
1993.            if(this.DOC.Execute(strSQL) == 0)
```

Heuristic CSRF\路徑 14:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=53
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebCatalog.aspx.cs第1829行，Sub_GetRequest 方法從QueryString_Name 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	1993

物件	QueryString_Name	Execute

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_ChiefLetter(int Action)

```
....
2048.        string strEMailC = this.Sub_GetRequest("txtEMailC", "");
....
2080.        strA = this.Sub_UpdateChiefLetter(strMode, "首長電子報",
strEMailC, strTitleC);
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_UpdateChiefLetter(string Mode, string Name, string Mail, string Title)

```
....
1955.      private string Sub_UpdateChiefLetter(string Mode, string
Name, string Mail, string Title)
....
1991.            strSQL = string.Format(fmtSQL3, Mail);
....
1993.            if(this.DOC.Execute(strSQL) == 0)
```

Heuristic CSRF\路徑 15:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=54
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebCatalog.aspx.cs第1829行，Sub_GetRequest 方法從Form 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	2360

物件	Form	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequestEmpty(string Name, string Default)

```
....
1850.        string strResult = Sub_GetRequest(Name, "");
....
1852.        return strResult;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_NewsLetter(int Action)

```
....
2253.          string strMailClass =
this.Sub_GetRequestEmpty("txtMailClass", "AppKMNewOrder");
....
2271.          DR = this.Sub_MailList(strMailClass);
```

檔案名稱 WebCatalog.aspx.cs

方法 private System.Data.SqlClient.SqlDataReader Sub_MailList(string MailClass)

```
....
2342.      private System.Data.SqlClient.SqlDataReader
Sub_MailList(string MailClass)
....
2351.            strSQL = string.Format(strSQL, MailClass);
....
2360.        return this.DOC.Query(strSQL);
```

Heuristic CSRF\路徑 16:

嚴重程度： 低風險

結果狀態： 校驗

線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=55

狀態 反覆出現的問題

Detection Date	7/8/2022 3:04:50 PM

在WebCatalog.aspx.cs第1829行，Sub_GetRequest 方法從QueryString_Name 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	2360
物件	QueryString_Name	Query

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequestEmpty(string Name, string Default)

```
....
1850.        string strResult = Sub_GetRequest(Name, "");
....
1852.        return strResult;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_NewsLetter(int Action)

```
....
2253.          string strMailClass =
this.Sub_GetRequestEmpty("txtMailClass", "AppKMNewOrder");
....
2271.          DR = this.Sub_MailList(strMailClass);
```

檔案名稱 WebCatalog.aspx.cs

方法 private System.Data.SqlClient.SqlDataReader Sub_MailList(string MailClass)

```
....
2342.      private System.Data.SqlClient.SqlDataReader
Sub_MailList(string MailClass)
....
2351.            strSQL = string.Format(strSQL, MailClass);
....
2360.        return this.DOC.Query(strSQL);
```

Heuristic CSRF\路徑 17:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=56
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebCatalog.aspx.cs第1829行，Sub_GetRequest 方法從Form 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	3948
物件	Form	Execute

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequestEmpty(string Name, string Default)

```
....
1850.        string strResult = Sub_GetRequest(Name, "");
....
1852.        return strResult;
```

檔案名稱 WebCatalog.aspx.cs

方法 public void Sub_PersonalSetting(int Action)

```
....
3944.            Ary[14] = PSET.RightArea =
this.Sub_GetRequestEmpty("checkRightArea", "");
....
3946.            strSQL1 = string.Format(strSQL1, Ary);
....
3948.            this.DOC.Execute(strSQL1, strSQL2);
```

Heuristic CSRF\路徑 18:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=57
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebCatalog.aspx.cs第1829行，Sub_GetRequest 方法從QueryString_Name 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	3948
物件	QueryString_Name	Execute

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequestEmpty(string Name, string Default)

```
....
1850.        string strResult = Sub_GetRequest(Name, "");
....
1852.        return strResult;
```

檔案名稱 WebCatalog.aspx.cs

方法 public void Sub_PersonalSetting(int Action)

```
....
3944.            Ary[14] = PSET.RightArea =
this.Sub_GetRequestEmpty("checkRightArea", "");
....
3946.            strSQL1 = string.Format(strSQL1, Ary);
....
3948.            this.DOC.Execute(strSQL1, strSQL2);
```

Heuristic CSRF\路徑 19:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=58
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebCatalog.aspx.cs第1829行，Sub_GetRequest 方法從Form 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	3993
物件	Form	Execute

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 public void Sub_PersonalSetting(int Action)

```
....
3976.            strWordA = this.Sub_GetRequest("txtPasswordA",
"").Trim();
....
3990.              objSB1.Append("UPDATE AppUser WITH(READPAST) SET
LoginPWD = '" + strWordA.Replace("'", "''") + "' WHERE UID ='" +
WhoAmI.UserID + "';");
....
3993.                this.DOC.Execute(objSB1.ToString());
```

Heuristic CSRF\路徑 20:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=59
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebCatalog.aspx.cs第1829行，Sub_GetRequest 方法從QueryString_Name 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	3993
物件	QueryString_Name	Execute

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_GetRequest(string Name, string Default)

```
....
1840.             strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.         return strReq;
```

檔案名稱	WebCatalog.aspx.cs
方法	public void Sub_PersonalSetting(int Action)

```
....
3976.             strWordA = this.Sub_GetRequest("txtPasswordA",
"").Trim();
....
3990.               objSB1.Append("UPDATE AppUser WITH(READPAST) SET
LoginPWD = '" + strWordA.Replace("'", "''") + "' WHERE UID ='" +
WhoAmI.UserID + "';");
....
3993.                 this.DOC.Execute(objSB1.ToString());
```

Heuristic CSRF\路徑 21:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=60
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebCatalog.aspx.cs第1829行，Sub_GetRequest 方法從Form 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1837	4232
物件	Form	Insert

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 public void Sub_PersonalSetting(int Action)

```
....
3976.             strWordA = this.Sub_GetRequest("txtPasswordA",
"").Trim();
....
3990.               objSB1.Append("UPDATE AppUser WITH(READPAST) SET
LoginPWD = '" + strWordA.Replace("'", "''") + "' WHERE UID ='" +
WhoAmI.UserID + "';");
....
3993.                 this.DOC.Execute(objSB1.ToString());
....
4232.         objSB1.Insert(0, "<SELECT NAME='listDocItems90'
ID='listDocItems90'
onkeypress=\"javascript:switch(window.event.keyCode){case
43:MoveDnOptionItemA('listDocItems90');break;case
45:MoveUpOptionItemA('listDocItems90');break;}try { event.preventDefault
? event.preventDefault() : (event.returnValue = false); } catch (err) {
return false; }\" SIZE=" + k.ToString() + ">");
```

Heuristic CSRF\路徑 22:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=61
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebCatalog.aspx.cs第1829行，Sub_GetRequest 方法從QueryString_Name 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	4232
物件	QueryString_Name	Insert

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 public void Sub_PersonalSetting(int Action)

```
....
3976.            strWordA = this.Sub_GetRequest("txtPasswordA",
"").Trim();
....
3990.              objSB1.Append("UPDATE AppUser WITH(READPAST) SET
LoginPWD = '" + strWordA.Replace("'", "''") + "' WHERE UID ='" +
WhoAmI.UserID + "';");
....
3993.                this.DOC.Execute(objSB1.ToString());
....
4232.        objSB1.Insert(0, "<SELECT NAME='listDocItems90'
ID='listDocItems90'
onkeypress=\"javascript:switch(window.event.keyCode){case
43:MoveDnOptionItemA('listDocItems90');break;case
45:MoveUpOptionItemA('listDocItems90');break;}try { event.preventDefault
? event.preventDefault() : (event.returnValue = false); } catch (err) {
return false; }\" SIZE=" + k.ToString() + ">");
```

Heuristic CSRF\路徑 23:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=62
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebCatalog.aspx.cs第1829行，Sub_GetRequest 方法從Form 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs

行	1837	4026
物件	Form	Execute

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 public void Sub_PersonalSetting(int Action)

```
....
4007.             strEmail = this.Sub_GetRequest("txtEmail", "").Trim();
....
4016.               objDR1 = this.DOC.Query(string.Format("SELECT * FROM
AppUser WITH(NOLOCK) WHERE Email = '{0}' AND UID <> '{1}'",
strEmail.Replace("'", "''"), WhoAmI.UserID));
....
4026.                 this.DOC.Execute(string.Format("UPDATE AppUser SET
Email = '{0}' WHERE UID = '{1}'", strEmail, WhoAmI.UserID));
```

Heuristic CSRF\路徑 24:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=63
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebCatalog.aspx.cs第1829行，Sub_GetRequest 方法從QueryString_Name 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1840	4026
物件	QueryString_Name	Execute

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 public void Sub_PersonalSetting(int Action)

```
....
4007.            strEmail = this.Sub_GetRequest("txtEmail", "").Trim();
....
4016.              objDR1 = this.DOC.Query(string.Format("SELECT * FROM
AppUser WITH(NOLOCK) WHERE Email = '{0}' AND UID <> '{1}'",
strEmail.Replace("'", "''"), WhoAmI.UserID));
....
4026.                this.DOC.Execute(string.Format("UPDATE AppUser SET
Email = '{0}' WHERE UID = '{1}'", strEmail, WhoAmI.UserID));
```

Heuristic CSRF\路徑 25:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=64
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebLogout.aspx.cs第50行，Sub_GetRequest 方法從Form 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebLogout.aspx.cs	WebLogout.aspx.cs
行	54	41
物件	Form	Execute

代碼片斷

檔案名稱 WebLogout.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
54.            strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
59.          return strResult.Trim();
```

檔案名稱	WebLogout.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
24.         string strLogKey = Sub_GetRequest("LogKey", ""); // Ū�� Form
�����
....
34.             strSQL += "Update LogUser Set LogoutDate = getdate()
WHERE (LogKey = '" + strLogKey.Trim() + "');\n";
35.             strSQL += "DELETE UserProcess WHERE LogKey = '" +
strLogKey.Trim() + "';\n";
....
41.           try { objDB.Execute(strSQL); }
```

Heuristic CSRF\路徑 26:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=65
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebLogout.aspx.cs第50行，Sub_GetRequest 方法從QueryString_strName 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebLogout.aspx.cs	WebLogout.aspx.cs
行	57	41
物件	QueryString_strName	Execute

代碼片斷

檔案名稱	WebLogout.aspx.cs
方法	private string Sub_GetRequest(string strName, string strDefault)

```
....
57.             strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
59.         return strResult.Trim();
```

檔案名稱	WebLogout.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
24.         string strLogKey = Sub_GetRequest("LogKey", ""); // Ū�� Form
�����
....
34.             strSQL += "Update LogUser Set LogoutDate = getdate()
WHERE (LogKey = '" + strLogKey.Trim() + "');\n";
35.             strSQL += "DELETE UserProcess WHERE LogKey = '" +
strLogKey.Trim() + "';\n";
....
41.           try { objDB.Execute(strSQL); }
```

Heuristic CSRF\路徑 27:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=66
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebLogout.aspx.cs第20行，Page_Load 方法從QueryString_LogKey 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebLogout.aspx.cs	WebLogout.aspx.cs
行	25	41
物件	QueryString_LogKey	Execute

代碼片斷

檔案名稱 WebLogout.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
25.         if(strLogKey == null) strLogKey =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g["LogKey"], false)); // Ū�� URL ��ƪ��
....
37.           strSQL += "DELETE UserCache WHERE LogKey = '" +
strLogKey.Trim() + "';\n";
....
41.           try { objDB.Execute(strSQL); }
```

Heuristic CSRF\路徑 28:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=67
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebMailA.aspx.cs第20行，Page_Load 方法從QueryString_DocID 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebMailA.aspx.cs	WebMailA.aspx.cs
行	38	166
物件	QueryString_DocID	Execute

代碼片斷

檔案名稱 WebMailA.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
38.         strDocID =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g["DocID"], false)); // Ū��  URL  ��□ع��
....
166.             objDB1.Execute(string.Format(strSQL2, strDocID)); //
2007/05/16
```

Heuristic CSRF\路徑 29:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=68
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebMark.aspx.cs第26行，Page_Load 方法從QueryString_DocID 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebMark.aspx.cs	WebMark.aspx.cs
行	34	222
物件	QueryString_DocID	Execute

代碼片斷

檔案名稱 WebMark.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
34.         varDocID =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g["DocID"], false)); // Ū��  URL  ��□ع��
```

檔案名稱	WebMark.aspx
方法	

```
....
127.
```

檔案名稱	WebMark.aspx.cs
方法	protected void fGroupAdd_Click(object sender, System.EventArgs e)

```
....
222.            objDB1.Execute(string.Format(fmtSQL2, aryID[i],
varDocID, objDoc.User.UserID),
```

Heuristic CSRF\路徑 30:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=69
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebNotePad.aspx.cs第628行，Sub_GetRequest 方法從Form 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	632	213
物件	Form	Execute

代碼片斷

檔案名稱	WebNotePad.aspx.cs
方法	private string Sub_GetRequest(string strName, string strDefault)

```
....
632.          strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
637.        return strResult.Trim();
```

檔案名稱	WebNotePad.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
173.                    this.Sub_SAY(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtTitle", ""),
this.Sub_GetRequest("txtEssay", ""), "", "",
(objDoc.User.eMail.IndexOf("@") < 0 ? objDoc.User.Name :
objDoc.User.eMail), this.Sub_GetRequest("txtChk", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_SAY(string Action, string SysName, string ID, string Title, string Essay, string StartDate, string EndDate, string Commentator, string Chk)

```
....
190.      private void Sub_SAY(string Action, string SysName, string ID,
string Title, string Essay, string StartDate, string EndDate, string
Commentator, string Chk)
....
212.              strSQL = string.Format("INSERT INTO GoodMemo (Chk,
MemoName, MemoContent, CreateDate, CompanyNo, DepartmentNo, Account)
VALUES ('{5}', N'{0}', N'{1}', GETDATE(), '{2}', '{3}', N'{4}')", Title,
Essay, objDoc.User.CompanyCode, objDoc.User.DepartmentCode, Commentator,
Chk);
213.              this.objDoc.Execute(strSQL);
```

Heuristic CSRF\路徑 31:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=70
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebNotePad.aspx.cs第628行，Sub_GetRequest 方法從QueryString_strName 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	635	213
物件	QueryString_strName	Execute

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
635.            strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
637.        return strResult.Trim();
```

檔案名稱	WebNotePad.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
173.                    this.Sub_SAY(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtTitle", ""),
this.Sub_GetRequest("txtEssay", ""), "", "",
(objDoc.User.eMail.IndexOf("@") < 0 ? objDoc.User.Name :
objDoc.User.eMail), this.Sub_GetRequest("txtChk", ""));
```

檔案名稱	WebNotePad.aspx.cs
方法	private void Sub_SAY(string Action, string SysName, string ID, string Title, string Essay, string StartDate, string EndDate, string Commentator, string Chk)

```
....
190.      private void Sub_SAY(string Action, string SysName, string ID,
string Title, string Essay, string StartDate, string EndDate, string
Commentator, string Chk)
....
212.              strSQL = string.Format("INSERT INTO GoodMemo (Chk,
MemoName, MemoContent, CreateDate, CompanyNo, DepartmentNo, Account)
VALUES ('{5}', N'{0}', N'{1}', GETDATE(), '{2}', '{3}', N'{4}')", Title,
Essay, objDoc.User.CompanyCode, objDoc.User.DepartmentCode, Commentator,
Chk);
213.              this.objDoc.Execute(strSQL);
```

Heuristic CSRF\路徑 32:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=71
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebNotePad.aspx.cs第628行，Sub_GetRequest 方法從Form 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	632	221
物件	Form	Execute

代碼片斷	
檔案名稱	WebNotePad.aspx.cs
方法	private string Sub_GetRequest(string strName, string strDefault)

```
....
632.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
637.         return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
173.                   this.Sub_SAY(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtTitle", ""),
this.Sub_GetRequest("txtEssay", ""), "", "",
(objDoc.User.eMail.IndexOf("@") < 0 ? objDoc.User.Name :
objDoc.User.eMail), this.Sub_GetRequest("txtChk", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_SAY(string Action, string SysName, string ID, string Title, string Essay, string StartDate, string EndDate, string Commentator, string Chk)

```
....
190.       private void Sub_SAY(string Action, string SysName, string ID,
string Title, string Essay, string StartDate, string EndDate, string
Commentator, string Chk)
....
220.             strSQL = string.Format("DELETE FROM GoodMemo WHERE ID =
'{0}'", ID);
221.             this.objDoc.Execute(strSQL);
```

Heuristic CSRF\路徑 33:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=72
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebNotePad.aspx.cs第628行，Sub_GetRequest 方法從QueryString_strName 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	635	221
物件	QueryString_strName	Execute

代碼片斷

檔案名稱	WebNotePad.aspx.cs
方法	private string Sub_GetRequest(string strName, string strDefault)

```
....
635.            strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
637.        return strResult.Trim();
```

檔案名稱	WebNotePad.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
173.                  this.Sub_SAY(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtTitle", ""),
this.Sub_GetRequest("txtEssay", ""), "", "",
(objDoc.User.eMail.IndexOf("@") < 0 ? objDoc.User.Name :
objDoc.User.eMail), this.Sub_GetRequest("txtChk", ""));
```

檔案名稱	WebNotePad.aspx.cs
方法	private void Sub_SAY(string Action, string SysName, string ID, string Title, string Essay, string StartDate, string EndDate, string Commentator, string Chk)

```
....
190.      private void Sub_SAY(string Action, string SysName, string ID,
string Title, string Essay, string StartDate, string EndDate, string
Commentator, string Chk)
....
220.            strSQL = string.Format("DELETE FROM GoodMemo WHERE ID =
'{0}'", ID);
221.            this.objDoc.Execute(strSQL);
```

Heuristic CSRF\路徑 34:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=73
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebNotePad.aspx.cs第628行，Sub_GetRequest 方法從Form 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	632	233
物件	Form	Execute

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
632.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
637.         return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
173.                   this.Sub_SAY(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtTitle", ""),
this.Sub_GetRequest("txtEssay", ""), "", "",
(objDoc.User.eMail.IndexOf("@") < 0 ? objDoc.User.Name :
objDoc.User.eMail), this.Sub_GetRequest("txtChk", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_SAY(string Action, string SysName, string ID, string Title, string Essay, string StartDate, string EndDate, string Commentator, string Chk)

```
....
190.       private void Sub_SAY(string Action, string SysName, string ID,
string Title, string Essay, string StartDate, string EndDate, string
Commentator, string Chk)
....
232.             strSQL += string.Format("UPDATE GoodMemo SET Chk =
'{6}', MemoName = N'{0}', MemoContent = N'{1}', CreateDate = GETDATE(),
CompanyNo = '{2}', DepartmentNo = '{3}', Account = N'{4}' WHERE (id =
{5});", Title, Essay, objDoc.User.CompanyCode,
objDoc.User.DepartmentCode, Commentator, ID, Chk);
233.             this.objDoc.Execute(strSQL);
```

Heuristic CSRF\路徑 35:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=74
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebNotePad.aspx.cs第628行，Sub_GetRequest 方法從QueryString_strName 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

來源	目的地

檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	635	233
物件	QueryString_strName	Execute

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
635.            strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
637.        return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
173.                    this.Sub_SAY(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtTitle", ""),
this.Sub_GetRequest("txtEssay", ""), "", "",
(objDoc.User.eMail.IndexOf("@") < 0 ? objDoc.User.Name :
objDoc.User.eMail), this.Sub_GetRequest("txtChk", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_SAY(string Action, string SysName, string ID, string Title, string Essay, string StartDate, string EndDate, string Commentator, string Chk)

```
....
190.      private void Sub_SAY(string Action, string SysName, string ID,
string Title, string Essay, string StartDate, string EndDate, string
Commentator, string Chk)
....
232.            strSQL += string.Format("UPDATE GoodMemo SET Chk =
'{6}', MemoName = N'{0}', MemoContent = N'{1}', CreateDate = GETDATE(),
CompanyNo = '{2}', DepartmentNo = '{3}', Account = N'{4}' WHERE (id =
{5});", Title, Essay, objDoc.User.CompanyCode,
objDoc.User.DepartmentCode, Commentator, ID, Chk);
233.            this.objDoc.Execute(strSQL);
```

Heuristic CSRF\路徑 36:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=75
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebNotePad.aspx.cs第628行，Sub_GetRequest 方法從Form 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	632	313
物件	Form	Execute

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
632.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
637.         return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
147.                   this.Sub_GBK(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtBookName",
""), this.Sub_GetRequest("txtBookStore", ""),
this.Sub_GetRequest("txtBookSummary", ""),
this.Sub_GetRequest("txtStartDate", ""),
this.Sub_GetRequest("txtEndDate", ""), (objDoc.User.eMail.IndexOf("@") <
0 ? objDoc.User.Name : objDoc.User.eMail),
this.Sub_GetRequest("txtBookAuthor", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_GBK(string Action, string SysName, string ID, string BookName, string BookStore, string BookContent, string StartDate, string EndDate, string Commentator, string BookAuthor)

```
....
287.       private void Sub_GBK(string Action, string SysName, string ID,
string BookName, string BookStore, string BookContent, string StartDate,
string EndDate, string Commentator, string BookAuthor)
....
312.               strSQL = string.Format("INSERT INTO GoodBook
(BookName, BookStore, BookContent, Sdate, Edate, CreateDate, CompanyNo,
DepartmentNo, Account, note1) VALUES (N'{0}', N'{1}', N'{2}', '{3}',
'{4}', GETDATE(), '{5}', '{6}', N'{7}', N'{8}')", BookName, BookStore,
BookContent, StartDate, EndDate, objDoc.User.CompanyCode,
objDoc.User.DepartmentCode, Commentator, BookAuthor);
313.               this.objDoc.Execute(strSQL);
```

Heuristic CSRF\路徑 37:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=76
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebNotePad.aspx.cs第628行，Sub_GetRequest 方法從QueryString_strName 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	635	313
物件	QueryString_strName	Execute

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
635.             strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
637.         return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
147.                   this.Sub_GBK(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtBookName",
""), this.Sub_GetRequest("txtBookStore", ""),
this.Sub_GetRequest("txtBookSummary", ""),
this.Sub_GetRequest("txtStartDate", ""),
this.Sub_GetRequest("txtEndDate", ""), (objDoc.User.eMail.IndexOf("@") <
0 ? objDoc.User.Name : objDoc.User.eMail),
this.Sub_GetRequest("txtBookAuthor", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_GBK(string Action, string SysName, string ID, string BookName, string BookStore, string BookContent, string StartDate, string EndDate, string Commentator, string BookAuthor)

```
....
287.      private void Sub_GBK(string Action, string SysName, string ID,
string BookName, string BookStore, string BookContent, string StartDate,
string EndDate, string Commentator, string BookAuthor)
....
312.              strSQL = string.Format("INSERT INTO GoodBook
(BookName, BookStore, BookContent, Sdate, Edate, CreateDate, CompanyNo,
DepartmentNo, Account, note1) VALUES (N'{0}', N'{1}', N'{2}', '{3}',
'{4}', GETDATE(), '{5}', '{6}', N'{7}', N'{8}')", BookName, BookStore,
BookContent, StartDate, EndDate, objDoc.User.CompanyCode,
objDoc.User.DepartmentCode, Commentator, BookAuthor);
313.              this.objDoc.Execute(strSQL);
```

Heuristic CSRF\路徑 38:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=77
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebNotePad.aspx.cs第628行，Sub_GetRequest 方法從Form 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	632	321
物件	Form	Execute

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
632.          strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
637.        return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
147.                    this.Sub_GBK(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtBookName",
""), this.Sub_GetRequest("txtBookStore", ""),
this.Sub_GetRequest("txtBookSummary", ""),
this.Sub_GetRequest("txtStartDate", ""),
this.Sub_GetRequest("txtEndDate", ""), (objDoc.User.eMail.IndexOf("@") <
0 ? objDoc.User.Name : objDoc.User.eMail),
this.Sub_GetRequest("txtBookAuthor", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_GBK(string Action, string SysName, string ID, string BookName, string BookStore, string BookContent, string StartDate, string EndDate, string Commentator, string BookAuthor)

```
....
287.      private void Sub_GBK(string Action, string SysName, string ID,
string BookName, string BookStore, string BookContent, string StartDate,
string EndDate, string Commentator, string BookAuthor)
....
320.              strSQL = string.Format("DELETE FROM GoodBook WHERE ID =
'{0}'", ID);
321.              this.objDoc.Execute(strSQL);
```

Heuristic CSRF\路徑 39:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=78
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebNotePad.aspx.cs第628行，Sub_GetRequest 方法從QueryString_strName 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	635	321
物件	QueryString_strName	Execute

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
635.            strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
637.        return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
147.                    this.Sub_GBK(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtBookName",
""), this.Sub_GetRequest("txtBookStore", ""),
this.Sub_GetRequest("txtBookSummary", ""),
this.Sub_GetRequest("txtStartDate", ""),
this.Sub_GetRequest("txtEndDate", ""), (objDoc.User.eMail.IndexOf("@") <
0 ? objDoc.User.Name : objDoc.User.eMail),
this.Sub_GetRequest("txtBookAuthor", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_GBK(string Action, string SysName, string ID, string BookName, string BookStore, string BookContent, string StartDate, string EndDate, string Commentator, string BookAuthor)

```
....
287.      private void Sub_GBK(string Action, string SysName, string ID,
string BookName, string BookStore, string BookContent, string StartDate,
string EndDate, string Commentator, string BookAuthor)
....
320.            strSQL = string.Format("DELETE FROM GoodBook WHERE ID =
'{0}'", ID);
321.            this.objDoc.Execute(strSQL);
```

Heuristic CSRF\路徑 40:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=79
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebNotePad.aspx.cs第628行，Sub_GetRequest 方法從Form 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	632	331

物件	Form	Execute

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
632.           strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.Form[strNa
me], false));
....
637.         return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
147.                   this.Sub_GBK(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtBookName",
""), this.Sub_GetRequest("txtBookStore", ""),
this.Sub_GetRequest("txtBookSummary", ""),
this.Sub_GetRequest("txtStartDate", ""),
this.Sub_GetRequest("txtEndDate", ""), (objDoc.User.eMail.IndexOf("@") <
0 ? objDoc.User.Name : objDoc.User.eMail),
this.Sub_GetRequest("txtBookAuthor", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_GBK(string Action, string SysName, string ID, string BookName, string BookStore, string BookContent, string StartDate, string EndDate, string Commentator, string BookAuthor)

```
....
287.       private void Sub_GBK(string Action, string SysName, string ID,
string BookName, string BookStore, string BookContent, string StartDate,
string EndDate, string Commentator, string BookAuthor)
....
330.             strSQL = string.Format("UPDATE GoodBook SET BookName =
N'{1}', BookStore = N'{2}', BookContent = N'{3}', Sdate = '{4}', Edate =
'{5}', CreateDate = GETDATE(), CompanyNo = '{6}', DepartmentNo = '{7}',
Account = N'{8}', note1 = N'{9}' WHERE (id = {0})", ID, BookName,
BookStore, BookContent, StartDate, EndDate, objDoc.User.CompanyCode,
objDoc.User.DepartmentCode, Commentator, BookAuthor);
331.             this.objDoc.Execute(strSQL);
```

Heuristic CSRF\路徑 41:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=80
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:50 PM

在WebNotePad.aspx.cs第628行，Sub_GetRequest 方法從QueryString_strName 元素取得用戶請求URL的參數。這個參數資料經過程式碼後最終被用來修改資料庫的內容。應用程式在過程中並沒有重新對用戶進行身份驗證。這可能引發跨站請求偽造（Cross-Site Request Forgery, 簡稱為XSRF或CSRF）。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	635	331
物件	QueryString_strName	Execute

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private string Sub_GetRequest(string strName, string strDefault)

```
....
635.             strResult =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(this.Request.QueryStrin
g[strName], false));
....
637.         return strResult.Trim();
```

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
147.                   this.Sub_GBK(varAction, varTitle,
this.Sub_GetRequest("ItemID", ""), this.Sub_GetRequest("txtBookName",
""), this.Sub_GetRequest("txtBookStore", ""),
this.Sub_GetRequest("txtBookSummary", ""),
this.Sub_GetRequest("txtStartDate", ""),
this.Sub_GetRequest("txtEndDate", ""), (objDoc.User.eMail.IndexOf("@") <
0 ? objDoc.User.Name : objDoc.User.eMail),
this.Sub_GetRequest("txtBookAuthor", ""));
```

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_GBK(string Action, string SysName, string ID, string BookName, string BookStore, string BookContent, string StartDate, string EndDate, string Commentator, string BookAuthor)

```
....
287.      private void Sub_GBK(string Action, string SysName, string ID,
string BookName, string BookStore, string BookContent, string StartDate,
string EndDate, string Commentator, string BookAuthor)
....
330.            strSQL = string.Format("UPDATE GoodBook SET BookName =
N'{1}', BookStore = N'{2}', BookContent = N'{3}', Sdate = '{4}', Edate =
'{5}', CreateDate = GETDATE(), CompanyNo = '{6}', DepartmentNo = '{7}',
Account = N'{8}', note1 = N'{9}' WHERE (id = {0})", ID, BookName,
BookStore, BookContent, StartDate, EndDate, objDoc.User.CompanyCode,
objDoc.User.DepartmentCode, Commentator, BookAuthor);
331.            this.objDoc.Execute(strSQL);
```

Heuristic DB Parameter Tampering

查詢路徑:

CSharp\Cx\CSharp Heuristic\Heuristic DB Parameter Tampering 版本:2

類別

PCI DSS v3.2.1: PCI DSS (3.2.1) - 6.5.1 - Injection flaws - particularly SQL injection
OWASP Top 10 2013: A4-Insecure Direct Object References
FISMA 2014: Access Control
NIST SP 800-53: AC-3 Access Enforcement (P1)
OWASP Top 10 2017: A5-Broken Access Control
ASD STIG 4.10: APSC-DV-002530 - CAT II The application must validate all input.
OWASP Top 10 API: API1-Broken Object Level Authorization
OWASP Top 10 2021: A1-Broken Access Control

描述

Heuristic DB Parameter Tampering\路徑 1:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=2
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 Form 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2654
物件	Form	GetValue

代碼片斷

檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.         public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2654.            if(strValue == DR.GetValue(iStart).ToString())
```

Heuristic DB Parameter Tampering\路徑 2:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=3
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 Form 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetFieldType 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2689
物件	Form	GetFieldType

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
```

Heuristic DB Parameter Tampering\路徑 3:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=4
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 Form 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetFieldType 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2692
物件	Form	GetFieldType

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.         public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.         private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.           while(DR.Read())
....
2651.             varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                 if(DR.GetFieldType(i).IsValueType)
....
2692.                   if(DR.GetFieldType(i).ToString() ==
"System.DateTime")
```

Heuristic DB Parameter Tampering\路徑 4:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=5
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 Form 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2697
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2692.                  if(DR.GetFieldType(i).ToString() ==
"System.DateTime")
....
2697.
if(DR.GetValue(DataIndex[Index][0]).ToString() != "")
```

Heuristic DB Parameter Tampering\路徑 5:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=6

狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 Form 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2599 行的 GetDataIndexData 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2605
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2668.                if(DR.GetValue(DataIndex[Index][0]).ToString() !=
"")
2669.                  strItem = string.Format(fmtHref,
string.Format(Href[Index], this.GetDataIndexData(DR, Index, DataIndex)),
DR.GetValue(iStart));
```

檔案名稱 PageSetting.cs

方法 private string[] GetDataIndexData(System.Data.SqlClient.SqlDataReader DR, int Index, int[][] DataIndex)

```
....
2599.        private string[]
GetDataIndexData(System.Data.SqlClient.SqlDataReader DR, int Index,
int[][] DataIndex)
....
2605.            aryDataIndexData[i] =
DR.GetValue(DataIndex[Index][i]).ToString();
```

Heuristic DB Parameter Tampering\路徑 6:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=7
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 Form 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的

AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2700
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.         public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱	PageSetting.cs
方法	private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.         private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.           while(DR.Read())
....
2651.             varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                 if(DR.GetFieldType(i).IsValueType)
....
2692.                   if(DR.GetFieldType(i).ToString() ==
"System.DateTime")
....
2697.
if(DR.GetValue(DataIndex[Index][0]).ToString() != "")
....
2700.                         strItem = DR.GetValue(iStart).ToString();
```

Heuristic DB Parameter Tampering\路徑 7:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=8
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 Form 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 IsDBNull 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2704
物件	Form	IsDBNull

代碼片斷

檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2692.                  if(DR.GetFieldType(i).ToString() ==
"System.DateTime")
....
2704.                    if(DR.IsDBNull(i))
```

Heuristic DB Parameter Tampering\路徑 8:

嚴重程度： 低風險
結果狀態： 校驗

線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=9
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 Form 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2718
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2692.                  if(DR.GetFieldType(i).ToString() ==
"System.DateTime")
....
2718.
if(DR.GetValue(DataIndex[Index][0]).ToString() != "")
```

Heuristic DB Parameter Tampering\路徑 9:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=10
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 Form 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2721
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.         public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2692.                  if(DR.GetFieldType(i).ToString() ==
"System.DateTime")
....
2718.
if(DR.GetValue(DataIndex[Index][0]).ToString() != "")
....
2721.                        strItem = DR.GetValue(i).ToString();
```

Heuristic DB Parameter Tampering\路徑 10:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=11
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 Form 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2734
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2734.                    if(DR.GetValue(DataIndex[Index][0]).ToString()
!= "")
```

Heuristic DB Parameter Tampering\路徑 11:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=12
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 Form 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2738
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2734.                    if(DR.GetValue(DataIndex[Index][0]).ToString()
!= "")
....
2737.                      string strTemp1 = DR.GetValue(i).ToString();
2738.                      if(RedColorFlag[Index] != 0 &&
DR.GetValue(RedColorFlag[Index]).ToString() != "0")
```

Heuristic DB Parameter Tampering\路徑 12:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=13
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 Form 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2737
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2734.                    if(DR.GetValue(DataIndex[Index][0]).ToString()
!= "")
....
2737.                      string strTemp1 = DR.GetValue(i).ToString();
```

Heuristic DB Parameter Tampering\路徑 13:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=14
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 Form 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2743
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2734.                    if(DR.GetValue(DataIndex[Index][0]).ToString()
!= "")
....
2743.                      strItem = DR.GetValue(i).ToString();
```

Heuristic DB Parameter Tampering\路徑 14:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=15
狀態 反覆出現的問題

Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 Form 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2746
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2746.                    strItem = DR.GetValue(i).ToString();
```

Heuristic DB Parameter Tampering\路徑 15:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=16
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 Form 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2668
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.         public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2668.                if(DR.GetValue(DataIndex[Index][0]).ToString() !=
"")
```

Heuristic DB Parameter Tampering\路徑 16:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=17
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 Form 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2678
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2674.                strItem = DR.GetValue(iStart).ToString();
....
2678.              strValue = DR.GetValue(iStart).ToString();
```

Heuristic DB Parameter Tampering\路徑 17:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=18
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 Form 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2671
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.         public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.         private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.           while(DR.Read())
....
2651.             varDataValues += DR.GetValue(0).ToString() + ",";
....
2668.                 if(DR.GetValue(DataIndex[Index][0]).ToString() !=
"")
....
2671.                   strItem = DR.GetValue(iStart).ToString();
```

Heuristic DB Parameter Tampering\路徑 18:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=19
狀態 反覆出現的問題
Detection Date 7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 Form 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2674
物件	Form	GetValue

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.           strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2674.                strItem = DR.GetValue(iStart).ToString();
```

Heuristic DB Parameter Tampering\路徑 19:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=20
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 Form 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1837	2651
物件	Form	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1837.          strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.Form[Name],
false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.         public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.         private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.           while(DR.Read())
....
2651.             varDataValues += DR.GetValue(0).ToString() + ",";
```

Heuristic DB Parameter Tampering\路徑 20:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=21
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2654
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.             strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2654.            if(strValue == DR.GetValue(iStart).ToString())
```

Heuristic DB Parameter Tampering\路徑 21:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=22
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetFieldType 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2689
物件	QueryString_Name	GetFieldType

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.             strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.         public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱	PageSetting.cs
方法	private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.         private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.           while(DR.Read())
....
2651.             varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                 if(DR.GetFieldType(i).IsValueType)
```

Heuristic DB Parameter Tampering\路徑 22:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=23
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetFieldType 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2692
物件	QueryString_Name	GetFieldType

代碼片斷

檔案名稱	WebCatalog.aspx.cs
方法	private string Sub_GetRequest(string Name, string Default)

```
....
1840.             strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.         return strReq;
```

檔案名稱	WebCatalog.aspx.cs
方法	private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2692.                  if(DR.GetFieldType(i).ToString() ==
"System.DateTime")
```

Heuristic DB Parameter Tampering\路徑 23:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=24
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2697
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.             strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2692.                  if(DR.GetFieldType(i).ToString() ==
"System.DateTime")
....
2697.
if(DR.GetValue(DataIndex[Index][0]).ToString() != "")
```

Heuristic DB Parameter Tampering\路徑 24:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=25
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2599 行的 GetDataIndexData 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2605
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2668.                if(DR.GetValue(DataIndex[Index][0]).ToString() !=
"")
2669.                  strItem = string.Format(fmtHref,
string.Format(Href[Index], this.GetDataIndexData(DR, Index, DataIndex)),
DR.GetValue(iStart));
```

檔案名稱 PageSetting.cs

方法 private string[] GetDataIndexData(System.Data.SqlClient.SqlDataReader DR, int Index, int[][] DataIndex)

```
....
2599.        private string[]
GetDataIndexData(System.Data.SqlClient.SqlDataReader DR, int Index,
int[][] DataIndex)
....
2605.            aryDataIndexData[i] =
DR.GetValue(DataIndex[Index][i]).ToString();
```

Heuristic DB Parameter Tampering\路徑 25:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=26
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2700
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2692.                  if(DR.GetFieldType(i).ToString() ==
"System.DateTime")
....
2697.
if(DR.GetValue(DataIndex[Index][0]).ToString() != "")
....
2700.                        strItem = DR.GetValue(iStart).ToString();
```

Heuristic DB Parameter Tampering\路徑 26:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=27
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 IsDBNull 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2704
物件	QueryString_Name	IsDBNull

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2692.                  if(DR.GetFieldType(i).ToString() ==
"System.DateTime")
....
2704.                    if(DR.IsDBNull(i))
```

Heuristic DB Parameter Tampering\路徑 27:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=28
狀態 反覆出現的問題

Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2718
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.             strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2692.                  if(DR.GetFieldType(i).ToString() ==
"System.DateTime")
....
2718.
if(DR.GetValue(DataIndex[Index][0]).ToString() != "")
```

Heuristic DB Parameter Tampering\路徑 28:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=29
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2721
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2692.                  if(DR.GetFieldType(i).ToString() ==
"System.DateTime")
....
2718.
if(DR.GetValue(DataIndex[Index][0]).ToString() != "")
....
2721.                        strItem = DR.GetValue(i).ToString();
```

Heuristic DB Parameter Tampering\路徑 29:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=30
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2734
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2734.                    if(DR.GetValue(DataIndex[Index][0]).ToString()
!= "")
```

Heuristic DB Parameter Tampering\路徑 30:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=31
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2738
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.             strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2734.                    if(DR.GetValue(DataIndex[Index][0]).ToString()
!= "")
....
2737.                      string strTemp1 = DR.GetValue(i).ToString();
2738.                      if(RedColorFlag[Index] != 0 &&
DR.GetValue(RedColorFlag[Index]).ToString() != "0")
```

Heuristic DB Parameter Tampering\路徑 31:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=32
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2737
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2734.                    if(DR.GetValue(DataIndex[Index][0]).ToString()
!= "")
....
2737.                      string strTemp1 = DR.GetValue(i).ToString();
```

Heuristic DB Parameter Tampering\路徑 32:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=33
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2743
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2734.                    if(DR.GetValue(DataIndex[Index][0]).ToString()
!= "")
....
2743.                      strItem = DR.GetValue(i).ToString();
```

Heuristic DB Parameter Tampering\路徑 33:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=34
狀態 反覆出現的問題

Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2746
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.             strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2689.                if(DR.GetFieldType(i).IsValueType)
....
2746.                    strItem = DR.GetValue(i).ToString();
```

Heuristic DB Parameter Tampering\路徑 34:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=35
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2668
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2668.                if(DR.GetValue(DataIndex[Index][0]).ToString() !=
"")
```

Heuristic DB Parameter Tampering\路徑 35:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=36
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2678
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.            strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.        return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2674.                strItem = DR.GetValue(iStart).ToString();
....
2678.              strValue = DR.GetValue(iStart).ToString();
```

Heuristic DB Parameter Tampering\路徑 36:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=37
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2671
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.             strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.         public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.         private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.           while(DR.Read())
....
2651.             varDataValues += DR.GetValue(0).ToString() + ",";
....
2668.                 if(DR.GetValue(DataIndex[Index][0]).ToString() !=
"")
....
2671.                   strItem = DR.GetValue(iStart).ToString();
```

Heuristic DB Parameter Tampering\路徑 37:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=38
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2674
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.             strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.        string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.            DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.            PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.        public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.          varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱 PageSetting.cs

方法 private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.        private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.          while(DR.Read())
....
2651.            varDataValues += DR.GetValue(0).ToString() + ",";
....
2674.                strItem = DR.GetValue(iStart).ToString();
```

Heuristic DB Parameter Tampering\路徑 38:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=39
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:47 PM

WebCatalog.aspx.cs 第 1829 行的 Sub_GetRequest 方法從 QueryString_Name 元素接收了使用者輸入變數，這個輸入變數之後在沒有驗證下被應用程式用來過濾敏感性資料庫中個人資料紀錄的資料表，PageSetting.cs 第 2610 行的 AddItem 方法在沒有加入其他額外資料庫過濾的情況下向 GetValue 資料庫進行查詢， 這會讓使用者能夠單純使用不同的 id 選擇到不同資料。

	來源	目的地
檔案	WebCatalog.aspx.cs	PageSetting.cs
行	1840	2651
物件	QueryString_Name	GetValue

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_GetRequest(string Name, string Default)

```
....
1840.             strReq =
HttpUtility.HtmlDecode(AntiXssEncoder.HtmlEncode(Request.QueryString[Nam
e], false));
....
1845.         return strReq;
```

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
360.         string strCategoryID = this.Sub_GetRequest("txtCategoryID",
"");
....
512.             DR = this.DOC.QueryCategoryChilds(strCategoryID, "'上移',
'下移', '修改', '搬移', '刪除', '隱藏'"); // 95/06/09
....
561.             PRT.Add3(DR, true, "0,1,4,11", aryHref, new int[] { 3,
5, 6, 7, 8, 9, 10 }, new int[7][] { new int[] { 0 }, new int[] { 0 },
new int[] { 0 }, new int[] { 0 }, new int[] { 0, 3 }, new int[] { 0 },
new int[] { 0 } }, new int[] { 0, 0, 0, 0, 0, 0, 11 }); // 95/06/09
```

檔案名稱 PageSetting.cs

方法 public void Add3(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[] HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)

```
....
2053.         public void Add3(System.Data.SqlClient.SqlDataReader DR,
bool WithColumnHead, string HiddenIndexs, string[] HrefStyle, int[]
HrefLinkDataIndex, int[][] HrefShowDataIndex, int[] RedColorFlag)
....
2056.           varText.Append(this.AddItem(DR, WithColumnHead,
HiddenIndexs, HrefStyle, HrefLinkDataIndex, HrefShowDataIndex,
RedColorFlag));
```

檔案名稱	PageSetting.cs
方法	private string AddItem(System.Data.SqlClient.SqlDataReader DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[] HrefIndex, int[][] DataIndex, int[] RedColorFlag)

```
....
2610.         private string AddItem(System.Data.SqlClient.SqlDataReader
DR, bool WithColumnHead, string HiddenIndexs, string[] Href, int[]
HrefIndex, int[][] DataIndex, int[] RedColorFlag)
....
2647.           while(DR.Read())
....
2651.             varDataValues += DR.GetValue(0).ToString() + ",";
```

Client Potential DOM Open Redirect

查詢路徑:

JavaScript\Cx\JavaScript Low Visibility\Client Potential DOM Open Redirect 版本:1

類別

OWASP Top 10 2013: A10-Unvalidated Redirects and Forwards
FISMA 2014: System And Information Integrity
NIST SP 800-53: SI-10 Information Input Validation (P1)
ASD STIG 4.10: APSC-DV-002560 - CAT I The application must not be subject to input handling vulnerabilities.
OWASP Top 10 2010: A10-Unvalidated Redirects and Forwards
OWASP Top 10 2021: A1-Broken Access Control

描述

Client Potential DOM Open Redirect\路徑 1:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=133
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:01 PM

	來源	目的地
檔案	ChangeDocGroupName.aspx	ChangeDocGroupName.aspx
行	51	51
物件	value	replace

代碼片斷	
檔案名稱	ChangeDocGroupName.aspx
方法	function GoExit() {

```
....
51.           location.replace(strURL.value);
```

Client Potential DOM Open Redirect\路徑 2:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=134
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:01 PM

	來源	目的地
檔案	ChangeDocName.aspx	ChangeDocName.aspx
行	52	52
物件	value	replace

代碼片斷

檔案名稱 ChangeDocName.aspx

方法 function GoExit() {

```
....
52.             location.replace(strURL.value);
```

Client Potential DOM Open Redirect\路徑 3:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=135
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:01 PM

	來源	目的地
檔案	ChangeDocShowName.aspx	ChangeDocShowName.aspx
行	52	52
物件	value	replace

代碼片斷

檔案名稱 ChangeDocShowName.aspx

方法 function GoExit() {

```
....
52.             location.replace(strURL.value);
```

Client Potential DOM Open Redirect\路徑 4:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=136
狀態	反覆出現的問題

Detection Date	7/8/2022 3:05:01 PM

	來源	目的地
檔案	ChangeMarquee.aspx	ChangeMarquee.aspx
行	52	52
物件	value	replace

代碼片斷
檔案名稱 ChangeMarquee.aspx
方法 function GoExit() {

```
....
52.             location.replace(strURL.value);
```

Client Potential DOM Open Redirect\路徑 5:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=137
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:01 PM

	來源	目的地
檔案	ChangeSystemName.aspx	ChangeSystemName.aspx
行	52	52
物件	value	replace

代碼片斷
檔案名稱 ChangeSystemName.aspx
方法 function GoExit() {

```
....
52.             location.replace(strURL.value);
```

Client Potential DOM Open Redirect\路徑 6:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=138
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:01 PM

	來源	目的地
檔案	WebChangeDocName.aspx	WebChangeDocName.aspx

行	55	55
物件	value	replace

代碼片斷
檔案名稱 WebChangeDocName.aspx
方法 function GoExit() {

```
....
55.             location.replace(strURL.value);
```

Client Potential DOM Open Redirect\路徑 7:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=139
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:01 PM

	來源	目的地
檔案	Controls/OUPickerButton.ascx	Controls/OUPickerButton.ascx
行	11	22
物件	value	BinaryExpr

代碼片斷
檔案名稱 Controls/OUPickerButton.ascx
方法 function selectOU(sender, event, left, top, selectionType, titleText, features, logKey, objCompanyValue, objCompanyName, objDepartmentValue, objDepartmentName, objMemberValue, objMemberName, checkMode){

```
....
11.                                     selectedId = "_" +
document.getElementById(objCompanyValue).value;
....
22.                       var win = window.open('OUPicker.aspx?LogKey=' +
logKey + '&TitleText=' + titleText + '&SelectionType=' + selectionType +
'&CN=' + objCompanyName + '&CV=' + objCompanyValue + '&DN=' +
objDepartmentName + '&DV=' + objDepartmentValue + '&MN=' + objMemberName
+ '&MV=' + objMemberValue + "&SelectedId=" + selectedId + "&CheckMode="
+ checkMode, 'OUPicker',
'toolbar=no,location=no,directories=no,status=no,menubar=no,scrollbars=y
es,resizable=yes,copyhistory=no' + features);
```

Client Potential DOM Open Redirect\路徑 8:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=140
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:01 PM

	來源	目的地
檔案	Controls/OUPickerButton.ascx	Controls/OUPickerButton.ascx
行	14	22
物件	value	BinaryExpr

代碼片斷

檔案名稱 Controls/OUPickerButton.ascx

方法 function selectOU(sender, event, left, top, selectionType, titleText, features, logKey, objCompanyValue, objCompanyName, objDepartmentValue, objDepartmentName, objMemberValue, objMemberName, checkMode){

```
....
14.                                    selectedId = "_" +
document.getElementById(objCompanyValue).value + "_" +
document.getElementById(objDepartmentValue).value;
....
22.                       var win = window.open('OUPicker.aspx?LogKey=' +
logKey + '&TitleText=' + titleText + '&SelectionType=' + selectionType +
'&CN=' + objCompanyName + '&CV=' + objCompanyValue + '&DN=' +
objDepartmentName + '&DV=' + objDepartmentValue + '&MN=' + objMemberName
+ '&MV=' + objMemberValue + "&SelectedId=" + selectedId + "&CheckMode="
+ checkMode, 'OUPicker',
'toolbar=no,location=no,directories=no,status=no,menubar=no,scrollbars=y
es,resizable=yes,copyhistory=no' + features);
```

Client Potential DOM Open Redirect\路徑 9:

嚴重程度： 低風險

結果狀態： 校驗

線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=141

狀態 反覆出現的問題

Detection Date 7/8/2022 3:05:01 PM

	來源	目的地
檔案	Controls/OUPickerButton.ascx	Controls/OUPickerButton.ascx
行	14	22
物件	value	BinaryExpr

代碼片斷

檔案名稱 Controls/OUPickerButton.ascx

方法 function selectOU(sender, event, left, top, selectionType, titleText, features, logKey, objCompanyValue, objCompanyName, objDepartmentValue, objDepartmentName, objMemberValue, objMemberName, checkMode){

```
....
14.                                    selectedId = "_" +
document.getElementById(objCompanyValue).value + ",_" +
document.getElementById(objDepartmentValue).value;
....
22.                       var win = window.open('OUPicker.aspx?LogKey=' +
logKey + '&TitleText=' + titleText + '&SelectionType=' + selectionType +
'&CN=' + objCompanyName + '&CV=' + objCompanyValue + '&DN=' +
objDepartmentName + '&DV=' + objDepartmentValue + '&MN=' + objMemberName
+ '&MV=' + objMemberValue + "&SelectedId=" + selectedId + "&CheckMode="
+ checkMode, 'OUPicker',
'toolbar=no,location=no,directories=no,status=no,menubar=no,scrollbars=y
es,resizable=yes,copyhistory=no' + features);
```

Client Potential DOM Open Redirect\路徑 10:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=142
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:01 PM

	來源	目的地
檔案	Controls/OUPickerButton.ascx	Controls/OUPickerButton.ascx
行	17	22
物件	value	BinaryExpr

代碼片斷

檔案名稱 Controls/OUPickerButton.ascx

方法 function selectOU(sender, event, left, top, selectionType, titleText, features, logKey, objCompanyValue, objCompanyName, objDepartmentValue, objDepartmentName, objMemberValue, objMemberName, checkMode){

```
....
17.                                    selectedId = "_" +
document.getElementById(objCompanyValue).value + ",_" +
document.getElementById(objDepartmentValue).value + ",_" +
document.getElementById(objDepartmentValue).value;
....
22.                       var win = window.open('OUPicker.aspx?LogKey=' +
logKey + '&TitleText=' + titleText + '&SelectionType=' + selectionType +
'&CN=' + objCompanyName + '&CV=' + objCompanyValue + '&DN=' +
objDepartmentName + '&DV=' + objDepartmentValue + '&MN=' + objMemberName
+ '&MV=' + objMemberValue + "&SelectedId=" + selectedId + "&CheckMode="
+ checkMode, 'OUPicker',
'toolbar=no,location=no,directories=no,status=no,menubar=no,scrollbars=y
es,resizable=yes,copyhistory=no' + features);
```

Client Potential DOM Open Redirect\路徑 11:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=1

	0300&pathid=143
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:01 PM

	來源	目的地
檔案	Controls/OUPickerButton.ascx	Controls/OUPickerButton.ascx
行	17	22
物件	value	BinaryExpr

代碼片斷

檔案名稱 Controls/OUPickerButton.ascx

方法 function selectOU(sender, event, left, top, selectionType, titleText, features, logKey, objCompanyValue, objCompanyName, objDepartmentValue, objDepartmentName, objMemberValue, objMemberName, checkMode){

```
....
17.                                    selectedId = "_" +
document.getElementById(objCompanyValue).value + "_" +
document.getElementById(objDepartmentValue).value + "_" +
document.getElementById(objDepartmentValue).value;
....
22.                        var win = window.open('OUPicker.aspx?LogKey=' +
logKey + '&TitleText=' + titleText + '&SelectionType=' + selectionType +
'&CN=' + objCompanyName + '&CV=' + objCompanyValue + '&DN=' +
objDepartmentName + '&DV=' + objDepartmentValue + '&MN=' + objMemberName
+ '&MV=' + objMemberValue + "&SelectedId=" + selectedId + "&CheckMode="
+ checkMode, 'OUPicker',
'toolbar=no,location=no,directories=no,status=no,menubar=no,scrollbars=y
es,resizable=yes,copyhistory=no' + features);
```

Client Potential DOM Open Redirect\路徑 12:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=144
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:01 PM

	來源	目的地
檔案	Controls/OUPickerButton.ascx	Controls/OUPickerButton.ascx
行	17	22
物件	value	BinaryExpr

代碼片斷

檔案名稱 Controls/OUPickerButton.ascx

方法 function selectOU(sender, event, left, top, selectionType, titleText, features, logKey, objCompanyValue, objCompanyName, objDepartmentValue, objDepartmentName, objMemberValue, objMemberName, checkMode){

```
....
17.                                    selectedId = "_" +
document.getElementById(objCompanyValue).value + "_" +
document.getElementById(objDepartmentValue).value + "_" +
document.getElementById(objDepartmentValue).value;
....
22.                       var win = window.open('OUPicker.aspx?LogKey=' +
logKey + '&TitleText=' + titleText + '&SelectionType=' + selectionType +
'&CN=' + objCompanyName + '&CV=' + objCompanyValue + '&DN=' +
objDepartmentName + '&DV=' + objDepartmentValue + '&MN=' + objMemberName
+ '&MV=' + objMemberValue + "&SelectedId=" + selectedId + "&CheckMode="
+ checkMode, 'OUPicker',
'toolbar=no,location=no,directories=no,status=no,menubar=no,scrollbars=y
es,resizable=yes,copyhistory=no' + features);
```

Client Potential DOM Open Redirect\路徑 13:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=145
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:01 PM

	來源	目的地
檔案	KM_js/PopUpClose.js	KM_js/PopUpClose.js
行	13	105
物件	value	varUrl

代碼片斷

檔案名稱 KM_js/PopUpClose.js

方法 function AspxGetData(AspxName, ParmName, ParmValue, SelFlag, SelPos) {

```
....
13.          datValue = document.getElementById(aryValue[i]).value;
....
34.      varUrl += deli + aryName[i] + "=" + escape(datValue);
....
39.    return Utf8(varUrl);
```

檔案名稱 KM_js/PopUpClose.js

方法 function OpenAspxProgram(AspxName, ParmName, ParmValue, SelFlag, SelPos) {

```
....
103.      var varUrl = AspxGetData(AspxName, ParmName, ParmValue,
SelFlag, SelPos);
....
105.      window.open(varUrl, '', sCommand);
```

Client Potential DOM Open Redirect\路徑 14:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=146
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:01 PM

	來源	目的地
檔案	KM_js/PopUpClose.js	KM_js/PopUpClose.js
行	18	105
物件	text	varUrl

代碼片斷

檔案名稱 KM_js/PopUpClose.js

方法 function AspxGetData(AspxName, ParmName, ParmValue, SelFlag, SelPos) {

```
....
18.             if (oItem.options[j].selected) { datValue +=
oItem.options[j].text + '|,'; }
....
34.      varUrl += deli + aryName[i] + "=" + escape(datValue);
....
39.    return Utf8(varUrl);
```

檔案名稱 KM_js/PopUpClose.js

方法 function OpenAspxProgram(AspxName, ParmName, ParmValue, SelFlag, SelPos) {

```
....
103.      var varUrl = AspxGetData(AspxName, ParmName, ParmValue,
SelFlag, SelPos);
....
105.      window.open(varUrl, '', sCommand);
```

Client Potential DOM Open Redirect\路徑 15:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=147
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:01 PM

	來源	目的地
檔案	KM_js/PopUpClose.js	KM_js/PopUpClose.js
行	24	105
物件	value	varUrl

代碼片斷

檔案名稱 KM_js/PopUpClose.js

方法 function AspxGetData(AspxName, ParmName, ParmValue, SelFlag, SelPos) {

```
....
24.            if (oItem.options[j].selected) { datValue +=
oItem.options[j].value + '|,'; }
....
34.      varUrl += deli + aryName[i] + "=" + escape(datValue);
....
39.    return Utf8(varUrl);
```

檔案名稱 KM_js/PopUpClose.js

方法 function OpenAspxProgram(AspxName, ParmName, ParmValue, SelFlag, SelPos) {

```
....
103.      var varUrl = AspxGetData(AspxName, ParmName, ParmValue,
SelFlag, SelPos);
....
105.      window.open(varUrl, '', sCommand);
```

Client Potential DOM Open Redirect\路徑 16:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=148
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:01 PM

	來源	目的地
檔案	KM_js/PopUpClose.js	KM_js/PopUpClose.js
行	28	105
物件	value	varUrl

代碼片斷

檔案名稱 KM_js/PopUpClose.js

方法 function AspxGetData(AspxName, ParmName, ParmValue, SelFlag, SelPos) {

```
....
28.          datValue =
document.getElementById(aryValue[i]).value.split('/,/')[aryPos[i]];
....
34.      varUrl += deli + aryName[i] + "=" + escape(datValue);
....
39.    return Utf8(varUrl);
```

檔案名稱 KM_js/PopUpClose.js

方法	function OpenAspxProgram(AspxName, ParmName, ParmValue, SelFlag, SelPos) {

```
....
103.      var varUrl = AspxGetData(AspxName, ParmName, ParmValue,
SelFlag, SelPos);
....
105.      window.open(varUrl, '', sCommand);
```

Client Potential DOM Open Redirect\路徑 17:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=149
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:01 PM

	來源	目的地
檔案	KM_js/WebCatalogG.js	KM_js/WebCatalogG.js
行	871	871
物件	value	BinaryExpr

代碼片斷	
檔案名稱	KM_js/WebCatalogG.js
方法	function funBMK(Action, BookMarkID, BookMarkName, DocIDs, DocTitle) {

```
....
871.          window.open('WebMark.aspx?Type=1&GUID=' +
document.forms[0].GUID.value + '&DocID=' + DocIDs, '',
'titlebar=0,toolbar=0,location=0,directories=0,status=1,menubar=0,scroll
bars=0,resizable=1,fullscreen=0,top=' + newY + ',left=' + newX +
',width=' + winWidth + ',height=' + winHeight);
```

Client Potential DOM Open Redirect\路徑 18:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=150
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:01 PM

	來源	目的地
檔案	KM_js/WebCatalogG.js	KM_js/WebCatalogG.js
行	968	968
物件	value	BinaryExpr

代碼片斷	
檔案名稱	KM_js/WebCatalogG.js

方法	function funGroup(Action, GroupID, GroupName, DocIDs, DocTitle, DeleteFlag, RelationDoc) {

```
....
968.          window.open('WebMark.aspx?Type=2&GUID=' +
document.forms[0].GUID.value + '&DocID=' + DocIDs, '',
'toolbar=0,location=0,directories=0,status=1,menubar=0,scrollbars=0,resi
zable=1,fullscreen=0,top=' + newY + ',left=' + newX + ',width=' +
winWidth + ',height=' + winHeight);
```

Client DOM Open Redirect

查詢路徑:

JavaScript\Cx\JavaScript Low Visibility\Client DOM Open Redirect 版本:3

類別

OWASP Top 10 2013: A10-Unvalidated Redirects and Forwards
FISMA 2014: System And Information Integrity
NIST SP 800-53: SI-10 Information Input Validation (P1)
OWASP Top 10 2010: A10-Unvalidated Redirects and Forwards
OWASP Top 10 2021: A1-Broken Access Control

描述

Client DOM Open Redirect\路徑 1:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=84
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:51 PM

	來源	目的地
檔案	Controls/OUPickerButton.ascx	Controls/OUPickerButton.ascx
行	11	22
物件	value	open

代碼片斷	
檔案名稱	Controls/OUPickerButton.ascx
方法	function selectOU(sender, event, left, top, selectionType, titleText, features, logKey, objCompanyValue, objCompanyName, objDepartmentValue, objDepartmentName, objMemberValue, objMemberName, checkMode){

```
....
11.                                     selectedId = "_" +
document.getElementById(objCompanyValue).value;
....
22.                         var win = window.open('OUPicker.aspx?LogKey=' +
logKey + '&TitleText=' + titleText + '&SelectionType=' + selectionType +
'&CN=' + objCompanyName + '&CV=' + objCompanyValue + '&DN=' +
objDepartmentName + '&DV=' + objDepartmentValue + '&MN=' + objMemberName
+ '&MV=' + objMemberValue + "&SelectedId=" + selectedId + "&CheckMode="
+ checkMode, 'OUPicker',
'toolbar=no,location=no,directories=no,status=no,menubar=no,scrollbars=y
es,resizable=yes,copyhistory=no' + features);
```

Client DOM Open Redirect\路徑 2:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=85
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:51 PM

	來源	目的地
檔案	Controls/OUPickerButton.ascx	Controls/OUPickerButton.ascx
行	14	22
物件	value	open

代碼片斷

檔案名稱 Controls/OUPickerButton.ascx

方法 function selectOU(sender, event, left, top, selectionType, titleText, features, logKey, objCompanyValue, objCompanyName, objDepartmentValue, objDepartmentName, objMemberValue, objMemberName, checkMode){

```
....
14.                                     selectedId = "_" +
document.getElementById(objCompanyValue).value + "_" +
document.getElementById(objDepartmentValue).value;
....
22.                         var win = window.open('OUPicker.aspx?LogKey=' +
logKey + '&TitleText=' + titleText + '&SelectionType=' + selectionType +
'&CN=' + objCompanyName + '&CV=' + objCompanyValue + '&DN=' +
objDepartmentName + '&DV=' + objDepartmentValue + '&MN=' + objMemberName
+ '&MV=' + objMemberValue + "&SelectedId=" + selectedId + "&CheckMode="
+ checkMode, 'OUPicker',
'toolbar=no,location=no,directories=no,status=no,menubar=no,scrollbars=y
es,resizable=yes,copyhistory=no' + features);
```

Client DOM Open Redirect\路徑 3:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=86
狀態	反覆出現的問題

Detection Date	7/8/2022 3:04:51 PM

	來源	目的地
檔案	Controls/OUPickerButton.ascx	Controls/OUPickerButton.ascx
行	14	22
物件	value	open

代碼片斷

檔案名稱 Controls/OUPickerButton.ascx

方法 function selectOU(sender, event, left, top, selectionType, titleText, features, logKey, objCompanyValue, objCompanyName, objDepartmentValue, objDepartmentName, objMemberValue, objMemberName, checkMode){

```
....
14.                                    selectedId = "_" +
document.getElementById(objCompanyValue).value + "_" +
document.getElementById(objDepartmentValue).value;
....
22.                        var win = window.open('OUPicker.aspx?LogKey=' +
logKey + '&TitleText=' + titleText + '&SelectionType=' + selectionType +
'&CN=' + objCompanyName + '&CV=' + objCompanyValue + '&DN=' +
objDepartmentName + '&DV=' + objDepartmentValue + '&MN=' + objMemberName
+ '&MV=' + objMemberValue + "&SelectedId=" + selectedId + "&CheckMode="
+ checkMode, 'OUPicker',
'toolbar=no,location=no,directories=no,status=no,menubar=no,scrollbars=y
es,resizable=yes,copyhistory=no' + features);
```

Client DOM Open Redirect\路徑 4:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=87
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:51 PM

	來源	目的地
檔案	Controls/OUPickerButton.ascx	Controls/OUPickerButton.ascx
行	17	22
物件	value	open

代碼片斷

檔案名稱 Controls/OUPickerButton.ascx

方法 function selectOU(sender, event, left, top, selectionType, titleText, features, logKey, objCompanyValue, objCompanyName, objDepartmentValue, objDepartmentName, objMemberValue, objMemberName, checkMode){

```
....
17.                                     selectedId = "_" +
document.getElementById(objCompanyValue).value + ",_" +
document.getElementById(objDepartmentValue).value + ",_" +
document.getElementById(objDepartmentValue).value;
....
22.                         var win = window.open('OUPicker.aspx?LogKey=' +
logKey + '&TitleText=' + titleText + '&SelectionType=' + selectionType +
'&CN=' + objCompanyName + '&CV=' + objCompanyValue + '&DN=' +
objDepartmentName + '&DV=' + objDepartmentValue + '&MN=' + objMemberName
+ '&MV=' + objMemberValue + "&SelectedId=" + selectedId + "&CheckMode="
+ checkMode, 'OUPicker',
'toolbar=no,location=no,directories=no,status=no,menubar=no,scrollbars=y
es,resizable=yes,copyhistory=no' + features);
```

Client DOM Open Redirect\路徑 5:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=88
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:51 PM

	來源	目的地
檔案	Controls/OUPickerButton.ascx	Controls/OUPickerButton.ascx
行	17	22
物件	value	open

代碼片斷

檔案名稱 Controls/OUPickerButton.ascx

方法 function selectOU(sender, event, left, top, selectionType, titleText, features, logKey, objCompanyValue, objCompanyName, objDepartmentValue, objDepartmentName, objMemberValue, objMemberName, checkMode){

```
....
17.                                     selectedId = "_" +
document.getElementById(objCompanyValue).value + ",_" +
document.getElementById(objDepartmentValue).value + ",_" +
document.getElementById(objDepartmentValue).value;
....
22.                         var win = window.open('OUPicker.aspx?LogKey=' +
logKey + '&TitleText=' + titleText + '&SelectionType=' + selectionType +
'&CN=' + objCompanyName + '&CV=' + objCompanyValue + '&DN=' +
objDepartmentName + '&DV=' + objDepartmentValue + '&MN=' + objMemberName
+ '&MV=' + objMemberValue + "&SelectedId=" + selectedId + "&CheckMode="
+ checkMode, 'OUPicker',
'toolbar=no,location=no,directories=no,status=no,menubar=no,scrollbars=y
es,resizable=yes,copyhistory=no' + features);
```

Client DOM Open Redirect\路徑 6:

嚴重程度：	低風險
結果狀態：	校驗

線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=89
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:51 PM

	來源	目的地
檔案	Controls/OUPickerButton.ascx	Controls/OUPickerButton.ascx
行	17	22
物件	value	open

代碼片斷

檔案名稱 Controls/OUPickerButton.ascx

方法 function selectOU(sender, event, left, top, selectionType, titleText, features, logKey, objCompanyValue, objCompanyName, objDepartmentValue, objDepartmentName, objMemberValue, objMemberName, checkMode){

```
....
17.                                    selectedId = "_" +
document.getElementById(objCompanyValue).value + "_" +
document.getElementById(objDepartmentValue).value + "_" +
document.getElementById(objDepartmentValue).value;
....
22.                        var win = window.open('OUPicker.aspx?LogKey=' +
logKey + '&TitleText=' + titleText + '&SelectionType=' + selectionType +
'&CN=' + objCompanyName + '&CV=' + objCompanyValue + '&DN=' +
objDepartmentName + '&DV=' + objDepartmentValue + '&MN=' + objMemberName
+ '&MV=' + objMemberValue + "&SelectedId=" + selectedId + "&CheckMode="
+ checkMode, 'OUPicker',
'toolbar=no,location=no,directories=no,status=no,menubar=no,scrollbars=y
es,resizable=yes,copyhistory=no' + features);
```

Client DOM Open Redirect\路徑 7:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=90
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:51 PM

	來源	目的地
檔案	KM_js/PopUpClose.js	KM_js/PopUpClose.js
行	13	105
物件	value	open

代碼片斷

檔案名稱 KM_js/PopUpClose.js

方法 function AspxGetData(AspxName, ParmName, ParmValue, SelFlag, SelPos) {

```
....
13.           datValue = document.getElementById(aryValue[i]).value;
....
34.       varUrl += deli + aryName[i] + "=" + escape(datValue);
....
39.     return Utf8(varUrl);
```

檔案名稱 KM_js/PopUpClose.js

方法 function OpenAspxProgram(AspxName, ParmName, ParmValue, SelFlag, SelPos) {

```
....
103.       var varUrl = AspxGetData(AspxName, ParmName, ParmValue,
SelFlag, SelPos);
....
105.       window.open(varUrl, '', sCommand);
```

Client DOM Open Redirect\路徑 8:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=91
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:51 PM

	來源	目的地
檔案	KM_js/PopUpClose.js	KM_js/PopUpClose.js
行	28	105
物件	value	open

代碼片斷

檔案名稱 KM_js/PopUpClose.js

方法 function AspxGetData(AspxName, ParmName, ParmValue, SelFlag, SelPos) {

```
....
28.           datValue =
document.getElementById(aryValue[i]).value.split('/,/')[aryPos[i]];
....
34.       varUrl += deli + aryName[i] + "=" + escape(datValue);
....
39.     return Utf8(varUrl);
```

檔案名稱 KM_js/PopUpClose.js

方法 function OpenAspxProgram(AspxName, ParmName, ParmValue, SelFlag, SelPos) {

```
....
103.      var varUrl = AspxGetData(AspxName, ParmName, ParmValue,
SelFlag, SelPos);
....
105.      window.open(varUrl, '', sCommand);
```

Client DOM Open Redirect\路徑 9:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=92
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:51 PM

	來源	目的地
檔案	ChangeDocGroupName.aspx	ChangeDocGroupName.aspx
行	51	51
物件	value	replace

代碼片斷

檔案名稱 ChangeDocGroupName.aspx

方法 function GoExit() {

```
....
51.            location.replace(strURL.value);
```

Client DOM Open Redirect\路徑 10:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=93
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:51 PM

	來源	目的地
檔案	ChangeDocName.aspx	ChangeDocName.aspx
行	52	52
物件	value	replace

代碼片斷

檔案名稱 ChangeDocName.aspx

方法 function GoExit() {

```
....
52.             location.replace(strURL.value);
```

Client DOM Open Redirect\路徑 11:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=94
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:51 PM

	來源	目的地
檔案	ChangeDocShowName.aspx	ChangeDocShowName.aspx
行	52	52
物件	value	replace

代碼片斷

檔案名稱 ChangeDocShowName.aspx

方法 function GoExit() {

```
....
52.             location.replace(strURL.value);
```

Client DOM Open Redirect\路徑 12:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=95
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:51 PM

	來源	目的地
檔案	ChangeMarquee.aspx	ChangeMarquee.aspx
行	52	52
物件	value	replace

代碼片斷

檔案名稱 ChangeMarquee.aspx

方法 function GoExit() {

```
....
52.             location.replace(strURL.value);
```

Client DOM Open Redirect\路徑 13:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=96
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:51 PM

	來源	目的地
檔案	ChangeSystemName.aspx	ChangeSystemName.aspx
行	52	52
物件	value	replace

代碼片斷

檔案名稱 ChangeSystemName.aspx

方法 function GoExit() {

```
....
52.             location.replace(strURL.value);
```

Client DOM Open Redirect\路徑 14:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=97
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:51 PM

	來源	目的地
檔案	MenuControl_A.Html	MenuControl_A.Html
行	39	48
物件	substr	replace

代碼片斷

檔案名稱 MenuControl_A.Html

方法 function ReplaceParam(sHref, obj) {

```
....
39.         var sParam = location.search.substr(1);
40.         var aryParam = sParam.split("&");
....
42.           var aryNameValue = aryParam[i].split("=");
....
44.           sHref = sHref.replace(pat, aryNameValue[1]);
....
48.         location.replace(sHref);
```

Client DOM Open Redirect\路徑 15:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=98
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:51 PM

	來源	目的地
檔案	WebChangeDocName.aspx	WebChangeDocName.aspx
行	55	55
物件	value	replace

代碼片斷

檔案名稱 WebChangeDocName.aspx

方法 function GoExit() {

```
....
55.             location.replace(strURL.value);
```

Improper Exception Handling

查詢路徑:

CSharp\Cx\CSharp Low Visibility\Improper Exception Handling 版本:5

類別

PCI DSS v3.2.1: PCI DSS (3.2.1) - 6.5.5 - Improper error handling
NIST SP 800-53: SC-5 Denial of Service Protection (P1)
ASD STIG 4.10: APSC-DV-002570 - CAT II The application must generate error messages that provide information necessary for corrective actions without revealing information that could be exploited by adversaries.
OWASP Top 10 API: API7-Security Misconfiguration
OWASP Top 10 2021: A4-Insecure Design

描述

Improper Exception Handling\路徑 1:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=99
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:56 PM

方法ExportFiles在WebDownloadFiles.aspx.cs第266 行執行可預期拋出異常的操作，但未正確包裹在try-catch區塊中。這構成不當異常處理(Improper Exception Handling)。

	來源	目的地
檔案	WebDownloadFiles.aspx.cs	WebDownloadFiles.aspx.cs
行	281	281

物件	Delete	Delete

代碼片斷
檔案名稱 WebDownloadFiles.aspx.cs
方法 private void ExportFiles()

```
....
281.         if(System.IO.Directory.Exists(varZipSourcePath))
System.IO.Directory.Delete(varZipSourcePath, true);
```

Improper Exception Handling\路徑 2:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=100
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:56 PM

方法ExportFiles在WebDownloadFiles.aspx.cs第266 行執行可預期拋出異常的操作，但未正確包裹在try-catch區塊中。這構成不當異常處理(Improper Exception Handling)。

	來源	目的地
檔案	WebDownloadFiles.aspx.cs	WebDownloadFiles.aspx.cs
行	286	286
物件	CreateDirectory	CreateDirectory

代碼片斷
檔案名稱 WebDownloadFiles.aspx.cs
方法 private void ExportFiles()

```
....
286.         System.IO.Directory.CreateDirectory(varZipSourcePath);
```

Improper Exception Handling\路徑 3:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=101
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:56 PM

方法Page_Load在WebEditor.aspx.cs第115 行執行可預期拋出異常的操作，但未正確包裹在try-catch區塊中。這構成不當異常處理(Improper Exception Handling)。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	181	181

物件	CreateDirectory	CreateDirectory

代碼片斷
檔案名稱 WebEditor.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
181.          System.IO.Directory.CreateDirectory(_sFileUpPath);
```

Improper Exception Handling\路徑 4:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=102
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:56 PM

方法ExportXml在WebEditor.aspx.cs第724 行執行可預期拋出異常的操作，但未正確包裹在try-catch區塊中。這構成不當異常處理(Improper Exception Handling)。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	735	735
物件	CreateDirectory	CreateDirectory

代碼片斷
檔案名稱 WebEditor.aspx.cs
方法 private void ExportXml()

```
....
735.              System.IO.Directory.CreateDirectory(_sFileUpPath +
strSubDir);
```

Improper Exception Handling\路徑 5:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=103
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:56 PM

方法ExportXml在WebEditor.aspx.cs第724 行執行可預期拋出異常的操作，但未正確包裹在try-catch區塊中。這構成不當異常處理(Improper Exception Handling)。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	740	740

物件	GetFiles	GetFiles

代碼片斷
檔案名稱 WebEditor.aspx.cs
方法 private void ExportXml()

```
....
740.          string[] aryFilePath =
System.IO.Directory.GetFiles(_sUploadFilePath);
```

Improper Exception Handling\路徑 6:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=104
狀態 反覆出現的問題
Detection Date 7/8/2022 3:04:56 PM

方法ExportXml在WebEditor.aspx.cs第724 行執行可預期拋出異常的操作，但未正確包裹在try-catch區塊中。這構成不當異常處理(Improper Exception Handling)。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	752	752
物件	CreateDirectory	CreateDirectory

代碼片斷
檔案名稱 WebEditor.aspx.cs
方法 private void ExportXml()

```
....
752.                  System.IO.Directory.CreateDirectory(_sFileUpPath +
strSubDir);
```

Improper Exception Handling\路徑 7:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=105
狀態 反覆出現的問題
Detection Date 7/8/2022 3:04:56 PM

方法Sub_MSG在WebNotePad.aspx.cs第383 行執行可預期拋出異常的操作，但未正確包裹在try-catch區塊中。這構成不當異常處理(Improper Exception Handling)。

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs

行	439	439
物件	CreateDirectory	CreateDirectory

代碼片斷
檔案名稱 WebNotePad.aspx.cs
方法 private void Sub_MSG(string Action, string SysName, string ID, string Message, string URL, string StartDate, string EndDate)

```
....
439.                    System.IO.Directory.CreateDirectory(strPath);
```

Improper Exception Handling\路徑 8:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=106
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:56 PM

方法AddZipEntry在WebDownloadFiles.aspx.cs第330 行執行可預期拋出異常的操作，但未正確包裹在try-catch區塊中。這構成不當異常處理(Improper Exception Handling)。

	來源	目的地
檔案	WebDownloadFiles.aspx.cs	WebDownloadFiles.aspx.cs
行	341	341
物件	Read	Read

代碼片斷
檔案名稱 WebDownloadFiles.aspx.cs
方法 private void AddZipEntry(string strSourcePath)

```
....
341.            objFS.Read(aryBuffer, 0, aryBuffer.Length);
```

Improper Exception Handling\路徑 9:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=107
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:56 PM

方法AddZipEntry在WebDownloadFiles.aspx.cs第330 行執行可預期拋出異常的操作，但未正確包裹在try-catch區塊中。這構成不當異常處理(Improper Exception Handling)。

	來源	目的地
檔案	WebDownloadFiles.aspx.cs	WebDownloadFiles.aspx.cs

行	347	347
物件	Write	Write

代碼片斷
檔案名稱 WebDownloadFiles.aspx.cs
方法 private void AddZipEntry(string strSourcePath)

```
....
347.            objZipStream.Write(aryBuffer, 0, aryBuffer.Length);
```

Improper Exception Handling\路徑 10:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=108
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:56 PM

方法SaveDocHTML在WebPrint.aspx.cs第597 行執行可預期拋出異常的操作，但未正確包裹在try-catch區塊中。這構成不當異常處理(Improper Exception Handling)。

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	681	681
物件	WriteLine	WriteLine

代碼片斷
檔案名稱 WebPrint.aspx.cs
方法 private void SaveDocHTML()

```
....
681.            objSW.WriteLine(strSaveData);
```

Improper Exception Handling\路徑 11:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=109
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:56 PM

方法SaveDocXML在WebPrint.aspx.cs第688 行執行可預期拋出異常的操作，但未正確包裹在try-catch區塊中。這構成不當異常處理(Improper Exception Handling)。

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs

行	738	738
物件	WriteLine	WriteLine

代碼片斷
檔案名稱 WebPrint.aspx.cs
方法 private void SaveDocXML()

```
....
738.          objSW.WriteLine(strOutXML);
```

Improper Exception Handling\路徑 12:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=110
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:56 PM

方法SaveDocXMLa在WebPrint.aspx.cs第745 行執行可預期拋出異常的操作，但未正確包裹在try-catch區塊中。這構成不當異常處理(Improper Exception Handling)。

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	797	797
物件	WriteLine	WriteLine

代碼片斷
檔案名稱 WebPrint.aspx.cs
方法 private void SaveDocXMLa()

```
....
797.          objSW.WriteLine("<?xml version=\"1.0\" encoding=\"utf-8\"
?>\n<ALLXMLContents>\n" + objSB1.ToString() + "</ALLXMLContents>\n");
```

Information Exposure Through an Error Message

查詢路徑:
CSharp\Cx\CSharp Low Visibility\Information Exposure Through an Error Message 版本:3

類別

PCI DSS v3.2.1: PCI DSS (3.2.1) - 6.5.5 - Improper error handling
OWASP Top 10 2013: A6-Sensitive Data Exposure
FISMA 2014: Configuration Management
NIST SP 800-53: SI-11 Error Handling (P2)
OWASP Top 10 2017: A3-Sensitive Data Exposure
ASD STIG 4.10: APSC-DV-002570 - CAT II The application must generate error messages that provide information necessary for corrective actions without revealing information that could be exploited by adversaries.

OWASP Top 10 API: API3-Excessive Data Exposure
OWASP Top 10 2021: A4-Insecure Design

描述

Information Exposure Through an Error Message\路徑 1:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=111
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:58 PM

在 WebCatalog.aspx.cs 第 358 行的函式 Sub_Category，處理了 Message 例外(exception)或 Runtime Error。在例外處理的程式碼中，程式透露了例外的細項至 WebCatalog.aspx.cs 檔案中，第358行，Sub_Category 函式中的X02Y01。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	424	424
物件	Message	X02Y01

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Category(int Action)

```
....
424.                this.X02Y01.Text = string.Format("<font color=\"red\"
size=\"3\">分類：{0}刪除失敗！原因：{1}</font>", strCategoryName,
ex.Message);
```

Information Exposure Through an Error Message\路徑 2:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=112
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:58 PM

在 WebCatalog.aspx.cs 第 358 行的函式 Sub_Category，處理了 Message 例外(exception)或 Runtime Error。在例外處理的程式碼中，程式透露了例外的細項至 WebCatalog.aspx.cs 檔案中，第358行，Sub_Category 函式中的X02Y01。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	446	446
物件	Message	X02Y01

代碼片斷

檔案名稱	WebCatalog.aspx.cs
方法	private void Sub_Category(int Action)

```
....
446.                this.X02Y01.Text = string.Format("<font color=\"red\"
size=\"3\">分類：{0}新增失敗！原因：{1}</font>", strCategoryName,
ex.Message);
```

Information Exposure Through an Error Message\路徑 3:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=113
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:58 PM

在 WebCatalog.aspx.cs 第 632 行的函式 Sub_FolderMtn，處理了 Message 例外(exception)或 Runtime Error。在例外處理的程式碼中，程式透露了例外的細項至 WebCatalog.aspx.cs 檔案中，第632行，Sub_FolderMtn 函式中的X02Y01。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	714	714
物件	Message	X02Y01

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private void Sub_FolderMtn(int Action)

```
....
714.                this.X02Y01.Text = string.Format("<font
color=\"red\" size=\"3\">開放區域：{0}刪除失敗！<br />原因：{1}</font>",
strFolderName, ex.Message);
```

Information Exposure Through an Error Message\路徑 4:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=114
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:58 PM

在 WebCatalog.aspx.cs 第 632 行的函式 Sub_FolderMtn，處理了 Message 例外(exception)或 Runtime Error。在例外處理的程式碼中，程式透露了例外的細項至 WebCatalog.aspx.cs 檔案中，第632行，Sub_FolderMtn 函式中的X02Y01。

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs

行	801	801
物件	Message	X02Y01

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private void Sub_FolderMtn(int Action)

```
....
801.                this.X02Y01.Text = string.Format("<font
color=\"red\" size=\"3\">開放區域:{0}, 儲存失敗。<br />" + ex.Message +
"</font>", strFolderName);
```

Information Exposure Through an Error Message\路徑 5:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=115
狀態 反覆出現的問題
Detection Date 7/8/2022 3:04:58 PM

在 WebDocumentLog.aspx.cs 第 28 行的函式 Page_Load，處理了 Message 例外(exception)或 Runtime Error。 在例外處理的程式碼中，程式透露了例外的細項至 WebDocumentLog.aspx.cs 檔案中，第28行，Page_Load 函式中的Write。

	來源	目的地
檔案	WebDocumentLog.aspx.cs	WebDocumentLog.aspx.cs
行	48	48
物件	Message	Write

代碼片斷
檔案名稱 WebDocumentLog.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
48.           Response.Write(ex.Message);
```

Information Exposure Through an Error Message\路徑 6:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=116
狀態 反覆出現的問題
Detection Date 7/8/2022 3:04:58 PM

在 WebEditor.aspx.cs 第 724 行的函式 ExportXml，處理了 Source 例外(exception)或 Runtime Error。 在例外處理的程式碼中，程式透露了例外的細項至 WebEditor.aspx.cs 檔案中，第724行，ExportXml 函式中的Text。

	來源	目的地

檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	895	895
物件	Source	Text

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private void ExportXml()

```
....
895.          Literal_Msg.Text += KmXmlUI.GenerateError(ex.Source,
ex.Message, ex.StackTrace, varIsShowMsgTrace, varIsShowMsgResponse,
"WebEditor", objDoc.User.eMail);
```

檔案名稱 KmXmlUI.cs

方法 static public string GenerateError(string Source, string Description, string Trace, bool ShowTrace, bool Response, string EmailSubject, string MailFrom)

```
....
14.      static public string GenerateError(string Source, string
Description, string Trace, bool ShowTrace, bool Response, string
EmailSubject, string MailFrom)
....
20.          + "<tr valign='top'><td width='5%' nowrap><font
size='2'>ŗ<G</font></td><td style='text-align:left;'><font
size='2'>" + Source + "</font></td></tr>"
```

Information Exposure Through an Error Message\路徑 7:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=117
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:58 PM

在 WebEditor.aspx.cs 第 724 行的函式 ExportXml，處理了 Message 例外(exception)或 Runtime Error。 在例外處理的程式碼中，程式透露了例外的細項至 WebEditor.aspx.cs 檔案中，第724行，ExportXml 函式中的Text。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	895	895
物件	Message	Text

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private void ExportXml()

```
....
895.          Literal_Msg.Text += KmXmlUI.GenerateError(ex.Source,
ex.Message, ex.StackTrace, varIsShowMsgTrace, varIsShowMsgResponse,
"WebEditor", objDoc.User.eMail);
```

檔案名稱 KmXmlUI.cs

方法 static public string GenerateError(string Source, string Description, string Trace, bool ShowTrace, bool Response, string EmailSubject, string MailFrom)

```
....
14.      static public string GenerateError(string Source, string
Description, string Trace, bool ShowTrace, bool Response, string
EmailSubject, string MailFrom)
....
21.          + "<tr valign='top'><td><font
size='2'>�����G</font></td><td style='text-align:left;'><font
size='2'>" + Description + "</font></td></tr>"
```

Information Exposure Through an Error Message\路徑 8:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=118
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:58 PM

在 WebEditor.aspx.cs 第 724 行的函式 ExportXml，處理了 StackTrace 例外(exception)或 Runtime Error。 在例外處理的程式碼中，程式透露了例外的細項至 WebEditor.aspx.cs 檔案中，第724行，ExportXml 函式中的 Text。

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	895	895
物件	StackTrace	Text

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private void ExportXml()

```
....
895.          Literal_Msg.Text += KmXmlUI.GenerateError(ex.Source,
ex.Message, ex.StackTrace, varIsShowMsgTrace, varIsShowMsgResponse,
"WebEditor", objDoc.User.eMail);
```

檔案名稱 KmXmlUI.cs

方法	static public string GenerateError(string Source, string Description, string Trace, bool ShowTrace, bool Response, string EmailSubject, string MailFrom)

```
....
14.      static public string GenerateError(string Source, string
Description, string Trace, bool ShowTrace, bool Response, string
EmailSubject, string MailFrom)
....
22.          + (ShowTrace ? "<tr valign='top'><td><font
size='2'>l<font></td><td style='text-align:left;'>" +
Trace.Replace("\r\n", "<p>") + "</td></tr>" : "") + (Response ?
strResponse : "") + "</table></td></tr></table>";
```

Information Exposure Through an Error Message\路徑 9:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=119
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:58 PM

在 WebLogon.aspx.cs 第 30 行的函式 Page_Load，處理了 Source 例外(exception)或 Runtime Error。 在例外處理的程式碼中，程式透露了例外的細項至 WebLogon.aspx.cs 檔案中，第30行，Page_Load 函式中的 ErrorMessage。

	來源	目的地
檔案	WebLogon.aspx.cs	WebLogon.aspx.cs
行	196	196
物件	Source	ErrorMessage

代碼片斷	
檔案名稱	WebLogon.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
196.          this.ErrorMessage.Text = PG.GenerateError(ex.Source,
ex.Message, ex.StackTrace, true, true, varTitle, strCommand);
```

檔案名稱	PageSetting.cs
方法	public string GenerateError(string Source, string Description, string Trace, bool ShowTrace, bool Response, string EmailSubject, string Command)

```
....
764.      public string GenerateError(string Source, string Description,
string Trace, bool ShowTrace, bool Response, string EmailSubject, string
Command)
....
773.          strResult = "<table width='100%' cellpadding=2 cellspacing=2
style='border:1px solid #ff0000; border-collapse: collapse;'><tr
valign='top'><td style='border:1px solid #ff0000' width='100%'><table
width='100%' cellpadding=3 cellspacing=0 style='border:1px solid
#ffff00; border-collapse: collapse;'><tr valign='top'><td
style='border:1px solid #ffff00' width='5%' nowrap><font
size=2>��ץ��G</font></td><td style='border:1px solid #ffff00; text-
align:left;'><font size=2>" + Source + "</font></td></tr>"
....
779.          return strResult;
```

Information Exposure Through an Error Message\路徑 10:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=120
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:58 PM

在 WebLogon.aspx.cs 第 30 行的函式 Page_Load，處理了 Message 例外(exception)或 Runtime Error。 在例外處理的程式碼中，程式透露了例外的細項至 WebLogon.aspx.cs 檔案中，第30行，Page_Load 函式中的 ErrorMessage。

	來源	目的地
檔案	WebLogon.aspx.cs	WebLogon.aspx.cs
行	196	196
物件	Message	ErrorMessage

代碼片斷

檔案名稱 WebLogon.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
196.          this.ErrorMessage.Text = PG.GenerateError(ex.Source,
ex.Message, ex.StackTrace, true, true, varTitle, strCommand);
```

檔案名稱 PageSetting.cs

方法 public string GenerateError(string Source, string Description, string Trace, bool ShowTrace, bool Response, string EmailSubject, string Command)

```
....
764.      public string GenerateError(string Source, string Description,
string Trace, bool ShowTrace, bool Response, string EmailSubject, string
Command)
....
774.          + "<tr valign='top'><td style='border:1px solid
#ffff00'><font size=2>●●●●●G</font></td><td style='border:1px solid
#ffff00; text-align:left;'><font size=2>" + Description +
"</font></td></tr>"
....
773.        strResult = "<table width='100%' cellpadding=2 cellspacing=2
style='border:1px solid #ff0000; border-collapse: collapse;'><tr
valign='top'><td style='border:1px solid #ff0000' width='100%'><table
width='100%' cellpadding=3 cellspacing=0 style='border:1px solid
#ffff00; border-collapse: collapse;'><tr valign='top'><td
style='border:1px solid #ffff00' width='5%' nowrap><font
size=2>●╔●●G</font></td><td style='border:1px solid #ffff00; text-
align:left;'><font size=2>" + Source + "</font></td></tr>"
....
779.        return strResult;
```

Information Exposure Through an Error Message\路徑 11:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=121
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:58 PM

在 WebLogon.aspx.cs 第 30 行的函式 Page_Load，處理了 StackTrace 例外(exception)或 Runtime Error。 在例外處理的程式碼中，程式透露了例外的細項至 WebLogon.aspx.cs 檔案中，第30行，Page_Load 函式中的 ErrorMessage。

	來源	目的地
檔案	WebLogon.aspx.cs	WebLogon.aspx.cs
行	196	196
物件	StackTrace	ErrorMessage

代碼片斷

檔案名稱 WebLogon.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
196.          this.ErrorMessage.Text = PG.GenerateError(ex.Source,
ex.Message, ex.StackTrace, true, true, varTitle, strCommand);
```

檔案名稱 PageSetting.cs

方法 public string GenerateError(string Source, string Description, string Trace, bool ShowTrace, bool Response, string EmailSubject, string Command)

```
....
764.      public string GenerateError(string Source, string Description,
string Trace, bool ShowTrace, bool Response, string EmailSubject, string
Command)
....
776.          + (ShowTrace ? ("<tr valign='top'><td style='border:1px
solid #ffff00'><font size=2>l<G</font></td><td style='border:1px
solid #ffff00; text-align:left;'>" + Trace.Replace("\r\n", "<p>") +
"</td></tr>") : "")
....
773.        strResult = "<table width='100%' cellpadding=2 cellspacing=2
style='border:1px solid #ff0000; border-collapse: collapse;'><tr
valign='top'><td style='border:1px solid #ff0000' width='100%'><table
width='100%' cellpadding=3 cellspacing=0 style='border:1px solid
#ffff00; border-collapse: collapse;'><tr valign='top'><td
style='border:1px solid #ffff00' width='5%' nowrap><font
size=2><G</font></td><td style='border:1px solid #ffff00; text-
align:left;'><font size=2>" + Source + "</font></td></tr>"
....
779.        return strResult;
```

Information Exposure Through an Error Message\路徑 12:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=122
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:58 PM

在 WebPrint.aspx.cs 第 41 行的函式 Page_Load，處理了 Message 例外(exception)或 Runtime Error。 在例外處理的程式碼中，程式透露了例外的細項至 WebPrint.aspx.cs 檔案中，第41行，Page_Load 函式中的Write。

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	132	132
物件	Message	Write

代碼片斷

檔案名稱 WebPrint.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
132.          Response.Write(ex.Message);
```

Client JQuery Deprecated Symbols

查詢路徑:

JavaScript\Cx\JavaScript Low Visibility\Client JQuery Deprecated Symbols 版本:3

類別

OWASP Top 10 2013: A9-Using Components with Known Vulnerabilities
OWASP Top 10 2017: A9-Using Components with Known Vulnerabilities

描述

Client JQuery Deprecated Symbols\路徑 1:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=123
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:58 PM

在KM_js/WebAutoBatchUpd.js文件中的CheckData方法的第47行，引用了一個過時的API：unique。這已經被棄用，不應該在當前的版本中使用。

	來源	目的地
檔案	KM_js/WebAutoBatchUpd.js	KM_js/WebAutoBatchUpd.js
行	68	68
物件	unique	unique

代碼片斷

檔案名稱 KM_js/WebAutoBatchUpd.js

方法 function CheckData(tagID, type) {

```
....
68.           if ($.unique(aryItem.sort()).length != iLength) {
```

Client JQuery Deprecated Symbols\路徑 2:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=124
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:58 PM

在KM_js/WebTreeCodeMtn.js文件中的CheckCodeID方法的第18行，引用了一個過時的API：unique。這已經被棄用，不應該在當前的版本中使用。

	來源	目的地
檔案	KM_js/WebTreeCodeMtn.js	KM_js/WebTreeCodeMtn.js
行	30	30
物件	unique	unique

代碼片斷

檔案名稱 KM_js/WebTreeCodeMtn.js

方法 function CheckCodeID(obj, Col) {

```
....
30.     if ($.unique(aryItem.sort()).length != iLength) {
```

Client JQuery Deprecated Symbols\路徑 3:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=125
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:58 PM

在KM_js/WebAutoBatchUpd.js文件中的$方法的第185行，引用了一個過時的API：andSelf。這已經被棄用，不應該在當前的版本中使用。

	來源	目的地
檔案	KM_js/WebAutoBatchUpd.js	KM_js/WebAutoBatchUpd.js
行	186	186
物件	andSelf	andSelf

代碼片斷

檔案名稱 KM_js/WebAutoBatchUpd.js

方法 $('#FtableD tr.EditLine').each(function (index) {

```
....
186.        $(this).find('>td').andSelf().removeClass('EditLine');
```

Client JQuery Deprecated Symbols\路徑 4:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=126
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:58 PM

在KM_js/WebAutoBatchUpd.js文件中的TrClick方法的第182行，引用了一個過時的API：andSelf。這已經被棄用，不應該在當前的版本中使用。

	來源	目的地
檔案	KM_js/WebAutoBatchUpd.js	KM_js/WebAutoBatchUpd.js
行	189	189
物件	andSelf	andSelf

代碼片斷

檔案名稱 KM_js/WebAutoBatchUpd.js

方法 function TrClick(row) {

```
....
189.     $('#TRdata_' + row).find('>td').andSelf().addClass('EditLine');
```

Client JQuery Deprecated Symbols\路徑 5:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=127
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:58 PM

在KM_js/WebAutoClassMtn.js文件中的$方法的第322行，引用了一個過時的API：andSelf。這已經被棄用，不應該在當前的版本中使用。

	來源	目的地
檔案	KM_js/WebAutoClassMtn.js	KM_js/WebAutoClassMtn.js
行	323	323
物件	andSelf	andSelf

代碼片斷

檔案名稱 KM_js/WebAutoClassMtn.js

方法 $('#FtableD tr.EditLine').each(function (index) {

```
....
323.         $(this).find('>td').andSelf().removeClass('EditLine');
```

Client JQuery Deprecated Symbols\路徑 6:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=128
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:58 PM

在KM_js/WebAutoClassMtn.js文件中的TrClick方法的第319行，引用了一個過時的API：andSelf。這已經被棄用，不應該在當前的版本中使用。

	來源	目的地
檔案	KM_js/WebAutoClassMtn.js	KM_js/WebAutoClassMtn.js
行	326	326
物件	andSelf	andSelf

代碼片斷

檔案名稱 KM_js/WebAutoClassMtn.js

方法 function TrClick(row) {

```
....
326.     $('#TRdata_' + row).find('>td').andSelf().addClass('EditLine');
```

Client JQuery Deprecated Symbols\路徑 7:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=129
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:58 PM

在KM_js/WebTreeCodeMtn.js文件中的DeleteRow方法的第225行，引用了一個過時的API：unbind。這已經被棄用，不應該在當前的版本中使用。

	來源	目的地
檔案	KM_js/WebTreeCodeMtn.js	KM_js/WebTreeCodeMtn.js
行	264	264
物件	unbind	unbind

代碼片斷

檔案名稱 KM_js/WebTreeCodeMtn.js

方法 function DeleteRow(objTB, obj) {

```
....
264.
$($(objTB.rows[i])).find(":button,:submit").unbind("mouseover", function
() {
```

Client JQuery Deprecated Symbols\路徑 8:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=130
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:58 PM

在KM_js/WebTreeCodeMtn.js文件中的DeleteRow方法的第225行，引用了一個過時的API：unbind。這已經被棄用，不應該在當前的版本中使用。

	來源	目的地
檔案	KM_js/WebTreeCodeMtn.js	KM_js/WebTreeCodeMtn.js
行	267	267
物件	unbind	unbind

代碼片斷

檔案名稱 KM_js/WebTreeCodeMtn.js

方法 function DeleteRow(objTB, obj) {

```
....
267.
$($(objTB.rows[i])).find(":button,:submit").unbind("mouseout", function
() {
```

Password in Configuration File

查詢路徑:

CSharp\Cx\CSharp WebConfig\Password in Configuration File 版本:1

類別

PCI DSS v3.2.1: PCI DSS (3.2.1) - 6.5.8 - Improper access control
OWASP Top 10 2013: A6-Sensitive Data Exposure
FISMA 2014: Identification And Authentication
NIST SP 800-53: SC-28 Protection of Information at Rest (P1)
OWASP Top 10 2017: A3-Sensitive Data Exposure
ASD STIG 4.10: APSC-DV-003110 - CAT I The application must not contain embedded authentication data.
OWASP Top 10 2021: A5-Security Misconfiguration

描述

Password in Configuration File\路徑 1:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=341
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:08 PM

	來源	目的地
檔案	bin/Websys.KM.Authority.Control.CBCCASE.dll.config	bin/Websys.KM.Authority.Control.CBCCASE.dll.config
行	18	18
物件	"PasswordVarName"	"PasswordVarName"

代碼片斷

檔案名稱 bin/Websys.KM.Authority.Control.CBCCASE.dll.config

方法 <?xml version="1.0"?>

```
....
18.         <add key="PasswordVarName" />
```

Password in Configuration File\路徑 2:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=342
狀態	反覆出現的問題
Detection Date	11/14/2022 6:35:41 PM

	來源	目的地
檔案	Web.config	Web.config
行	51	51
物件	"PasswordChange"	"PasswordChange"

代碼片斷
檔案名稱 Web.config
方法 <?xml version="1.0"?>

```
....
51.      <add key="PasswordChange" value="true"/>
```

Password in Configuration File\路徑 3:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=343
狀態 反覆出現的問題
Detection Date 11/14/2022 6:35:41 PM

	來源	目的地
檔案	Web.config	Web.config
行	111	111
物件	"PasswordVarName"	"PasswordVarName"

代碼片斷
檔案名稱 Web.config
方法 <?xml version="1.0"?>

```
....
111.      <add key="PasswordVarName" value=""/>
```

Password in Configuration File\路徑 4:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=344
狀態 反覆出現的問題
Detection Date 11/14/2022 6:35:41 PM

	來源	目的地
檔案	Web.config	Web.config
行	117	117

物件	"SecureCnnStr.Pwd"	"SecureCnnStr.Pwd"

代碼片斷
檔案名稱 Web.config
方法 <?xml version="1.0"?>

```
....
117.      <add key="SecureCnnStr.Pwd" value="QST20190513" />
```

Password in Configuration File\路徑 5:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=345
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:08 PM

	來源	目的地
檔案	bin/Websys.KM.Authority.Control.CBCCASE.dll.config	bin/Websys.KM.Authority.Control.CBCCASE.dll.config
行	4	4
物件	"data source=km01;initial catalog=CAS;user id=sa;password=sa12247838!; Pooling=False;"	"data source=km01;initial catalog=CAS;user id=sa;password=sa12247838!; Pooling=False;"

代碼片斷
檔案名稱 bin/Websys.KM.Authority.Control.CBCCASE.dll.config
方法 <?xml version="1.0"?>

```
....
4.          <add name="CBCDataBase" connectionString="data
source=km01;initial catalog=CAS;user id=sa;password=sa12247838!;
Pooling=False;"/>
```

Password in Configuration File\路徑 6:

嚴重程度： 低風險
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=346
狀態 反覆出現的問題
Detection Date 11/14/2022 6:35:41 PM

	來源	目的地
檔案	Web.config	Web.config
行	4	4
物件	"data source=ap01;initial catalog=CAS;user	"data source=ap01;initial catalog=CAS;user

	id=chung;password=!QAZ2wsx#EDC; Pooling=False;"	id=chung;password=!QAZ2wsx#EDC; Pooling=False;"

代碼片斷
檔案名稱 Web.config
方法 <?xml version="1.0"?>

```
....
4.      <add name="CBCDataBase" connectionString="data
source=ap01;initial catalog=CAS;user id=chung;password=!QAZ2wsx#EDC;
Pooling=False;"/>
```

Client Side Only Validation

查詢路徑:
CSharp\Cx\CSharp Low Visibility\Client Side Only Validation 版本:2

類別

PCI DSS v3.2.1: PCI DSS (3.2.1) - 6.5.8 - Improper access control
OWASP Top 10 2013: A7-Missing Function Level Access Control
OWASP Top 10 2017: A5-Broken Access Control
OWASP Top 10 2021: A4-Insecure Design

描述

Client Side Only Validation\路徑 1:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=81
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:51 PM

在 Controls/CompanyList.ascx 檔中未找到伺服器端驗證，僅使用用戶端驗證是不夠的, 因為它很容易被略過。

	來源	目的地
檔案	Controls/CompanyList.ascx	Controls/CompanyList.ascx
行	1	1
物件	CompanyList	CompanyList

代碼片斷
檔案名稱 Controls/CompanyList.ascx
方法 <%@ Control Language="c#" AutoEventWireup="false" Codebehind="CompanyList.ascx.cs" Inherits="Websys.KM.Authority.Control.CompanyList" TargetSchema="http://schemas.microsoft.com/intellisense/ie5" %>

```
....
1.  <%@ Control Language="c#" AutoEventWireup="false"
Codebehind="CompanyList.ascx.cs"
Inherits="Websys.KM.Authority.Control.CompanyList"
TargetSchema="http://schemas.microsoft.com/intellisense/ie5" %>
```

Client Side Only Validation\路徑 2:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=82
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:51 PM

在 Controls/DepartmentList.ascx 檔中未找到伺服器端驗證，僅使用用戶端驗證是不夠的, 因為它很容易被略過。

	來源	目的地
檔案	Controls/DepartmentList.ascx	Controls/DepartmentList.ascx
行	1	1
物件	DepartmentList	DepartmentList

代碼片斷

檔案名稱 Controls/DepartmentList.ascx

方法 <%@ Control Language="c#" AutoEventWireup="false"
Codebehind="DepartmentList.ascx.cs"
Inherits="Websys.KM.Authority.Control.DepartmentList"
TargetSchema="http://schemas.microsoft.com/intellisense/ie5" %>

```
....
1.  <%@ Control Language="c#" AutoEventWireup="false"
Codebehind="DepartmentList.ascx.cs"
Inherits="Websys.KM.Authority.Control.DepartmentList"
TargetSchema="http://schemas.microsoft.com/intellisense/ie5" %>
```

Client Side Only Validation\路徑 3:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=83
狀態	反覆出現的問題
Detection Date	7/8/2022 3:04:51 PM

在 Controls/MemberEdit.ascx 檔中未找到伺服器端驗證，僅使用用戶端驗證是不夠的, 因為它很容易被略過。

	來源	目的地
檔案	Controls/MemberEdit.ascx	Controls/MemberEdit.ascx
行	1	1
物件	MemberEdit	MemberEdit

代碼片斷

檔案名稱 Controls/MemberEdit.ascx

方法 <%@ Control Language="c#" AutoEventWireup="false" CodeBehind="MemberEdit.ascx.cs"

```
....
1.  <%@ Control Language="c#" AutoEventWireup="false"
CodeBehind="MemberEdit.ascx.cs"
```

Information Exposure via Headers

查詢路徑:

CSharp\Cx\CSharp Low Visibility\Information Exposure via Headers 版本:1

類別

OWASP Top 10 API: API7-Security Misconfiguration
OWASP Top 10 2021: A1-Broken Access Control

描述

Information Exposure via Headers\路徑 1:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=131
狀態	反覆出現的問題
Detection Date	11/14/2022 6:35:30 PM

	來源	目的地
檔案	Web.config	Web.config
行	147	147
物件	HTTPRUNTIME	HTTPRUNTIME

代碼片斷

檔案名稱 Web.config

方法 <?xml version="1.0"?>

```
....
147.      <httpRuntime maxRequestLength="1024000"
executionTimeout="900"/>
```

Information Exposure via Headers\路徑 2:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=132
狀態	反覆出現的問題
Detection Date	11/14/2022 6:35:30 PM

	來源	目的地
檔案	Web.config	Web.config
行	151	151

物件	WEBSERVER	WEBSERVER

代碼片斷
檔案名稱 Web.config
方法 <?xml version="1.0"?>

```
....
151.    <system.webServer>
```

Unencrypted Web Config File

查詢路徑:
CSharp\公司\CSharp Low Visibility\Unencrypted Web Config File 版本:1
描述

Unencrypted Web Config File\路徑 1:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=336
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:07 PM

Web.config檔案:Web.config找不到加密區段，因此需檢視是否有敏感資訊需要加密。這些明文資訊任何人只要訪問檔案系統就可以查看。

	來源	目的地
檔案	Web.config	Web.config
行	1	1
物件	CxXmlConfigClass6f90b31a	CxXmlConfigClass6f90b31a

代碼片斷
檔案名稱 Web.config
方法 <?xml version="1.0"?>

```
....
1.  <?xml version="1.0"?>
```

Unencrypted Web Config File\路徑 2:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=337
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:07 PM

Web.config檔案:bin/Websys.KM.Authority.Control.CBCCASE.dll.config找不到加密區段，因此需檢視是否有敏感資訊需要加密。這些明文資訊任何人只要訪問檔案系統就可以查看。

	來源	目的地

檔案	bin/Websys.KM.Authority.Control.CBCCASE.dll.config	bin/Websys.KM.Authority.Control.CBCCASE.dll.config
行	1	1
物件	CxXmlConfigClassc7a3509a	CxXmlConfigClassc7a3509a

代碼片斷
檔案名稱 bin/Websys.KM.Authority.Control.CBCCASE.dll.config
方法 <?xml version="1.0"?>

```
....
1.  <?xml version="1.0"?>
```

DebugEnabled

查詢路徑:
CSharp\Cx\CSharp WebConfig\DebugEnabled 版本:2

類別

PCI DSS v3.2.1: PCI DSS (3.2.1) - 6.5.5 - Improper error handling
FISMA 2014: Configuration Management
NIST SP 800-53: SI-11 Error Handling (P2)
OWASP Top 10 2017: A3-Sensitive Data Exposure
ASD STIG 4.10: APSC-DV-002610 - CAT II The application must remove organization-defined software components after updated versions have been installed.
OWASP Top 10 API: API7-Security Misconfiguration
OWASP Top 10 2021: A5-Security Misconfiguration

描述

DebugEnabled\路徑 1:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=339
狀態	反覆出現的問題
Detection Date	11/14/2022 6:35:41 PM

應用程式原始碼包括 "true"，在Web.config 的行號 1中，這是由開發和除錯遺留下來的，並且不是預期的應用程式功能的一部分。

	來源	目的地
檔案	Web.config	Web.config
行	128	128
物件	"true"	"true"

代碼片斷
檔案名稱 Web.config
方法 <?xml version="1.0"?>

```
....
128.      <compilation debug="true" targetFramework="4.5"/>
```

DebugEnabled\路徑 2:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=340
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:08 PM

應用程式原始碼包括 "true"，在bin/Websys.KM.Authority.Control.CBCCASE.dll.config 的行號 1中，這是由開發和除錯遺留下來的，並且不是預期的應用程式功能的一部分。

	來源	目的地
檔案	bin/Websys.KM.Authority.Control.CBCCASE.dll.config	bin/Websys.KM.Authority.Control.CBCCASE.dll.config
行	40	40
物件	"true"	"true"

代碼片斷

檔案名稱 bin/Websys.KM.Authority.Control.CBCCASE.dll.config

方法 <?xml version="1.0"?>

```
....
40.          <compilation debug="true" targetFramework="4.5"/>
```

Client Regex Injection

查詢路徑:

JavaScript\Cx\JavaScript Low Visibility\Client Regex Injection 版本:3

類別

PCI DSS v3.2.1: PCI DSS (3.2.1) - 6.5.1 - Injection flaws - particularly SQL injection
OWASP Top 10 2013: A1-Injection
OWASP Top 10 2017: A1-Injection
OWASP Top 10 2021: A3-Injection

描述

Client Regex Injection\路徑 1:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=151
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:01 PM

來源	目的地

檔案	MenuControl_A.Html	MenuControl_A.Html
行	39	43
物件	substr	BinaryExpr

代碼片斷
檔案名稱 MenuControl_A.Html
方法 function ReplaceParam(sHref, obj) {

```
....
39.         var sParam = location.search.substr(1);
40.         var aryParam = sParam.split("&");
....
42.           var aryNameValue = aryParam[i].split("=");
43.           var pat = new RegExp("@" + aryNameValue[0] + "@", "gim");
```

Missing Content Security Policy

查詢路徑:
CSharp\Cx\CSharp Low Visibility\Missing Content Security Policy 版本:1

類別

OWASP Top 10 2010: A6-Security Misconfiguration
OWASP Top 10 2021: A7-Identification and Authentication Failures

描述

Missing Content Security Policy\路徑 1:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=152
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:02 PM

網站應用程序中未明確定義內容安全性政策 (CSP, Content Security Policy)。

	來源	目的地
檔案	Web.config	Web.config
行	1	1
物件	CxXmlConfigClass6f90b31a	CxXmlConfigClass6f90b31a

代碼片斷
檔案名稱 Web.config
方法 <?xml version="1.0"?>

```
....
1.  <?xml version="1.0"?>
```

Use Of Hardcoded Password

查詢路徑:

CSharp\公司\CSharp Low Visibility\Use Of Hardcoded Password 版本:1

描述

Use Of Hardcoded Password\路徑 1:

嚴重程度：	低風險
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=338
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:08 PM

此應用程式使用預先寫好的密碼"victory"進行單一驗證程序，無論是用它來驗證用戶的身份，或連接其他遠程系統。這個密碼以明文撰寫在檔案WebLogon.aspx.cs中的第30行，且不會因為重建專案而變動。

	來源	目的地
檔案	WebLogon.aspx.cs	WebLogon.aspx.cs
行	142	142
物件	"victory"	"victory"

代碼片斷

檔案名稱 WebLogon.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
142.                bVerify = (PWD == "victory");
```

Insufficient Logging of Exceptions

查詢路徑:

CSharp\公司\CSharp Best Coding Practice\Insufficient Logging of Exceptions 版本:1

描述

Insufficient Logging of Exceptions\路徑 1:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=514
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	KmXmlUI.cs	KmXmlUI.cs
行	176	176
物件	catch	catch

代碼片斷

檔案名稱 KmXmlUI.cs

方法 public void SetVaule(SqlDataReader dr, bool GetDefaultValue, bool bCopyDoc, string _sModule, Cdsys.KM.Struct.USER WhoAmI, string strKmAction)

```
....
176.         catch { }
```

Insufficient Logging of Exceptions\路徑 2:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=515
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	223	223
物件	catch	catch

代碼片斷

檔案名稱 PageSetting.cs

方法 private string Sub_QuickView(string MaxNumbers, string DocType, string ShowDocType, string Img, string ImageColor, string NavigateUrl, string Folders, string StartDate, string EndDate, DocOrder iOrderBy, string[] aryClass, bool[] aryClassAnd, string Companies, string Keywords, string SearchWords, string SearchMore, string DocIDs, bool FullSearchIsOR, bool KeyWordWithAccuracy, int Index, string[] AttachCMD, params string[] Limit)

```
....
223.           catch { }
```

Insufficient Logging of Exceptions\路徑 3:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=516
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	279	279
物件	catch	catch

代碼片斷

檔案名稱 PageSetting.cs

方法	private string Sub_QuickView(string MaxNumbers, string DocType, string ShowDocType, string Img, string ImageColor, string NavigateUrl, string Folders, string StartDate, string EndDate, DocOrder iOrderBy, string[] aryClass, bool[] aryClassAnd, string Companies, string Keywords, string SearchWords, string SearchMore, string DocIDs, bool FullSearchIsOR, bool KeyWordWithAccuracy, int Index, string[] AttachCMD, params string[] Limit)

```
....
279.         catch(Exception e)
```

Insufficient Logging of Exceptions\路徑 4:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=517
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	385	385
物件	catch	catch

代碼片斷	
檔案名稱	PageSetting.cs
方法	private string Sub_QuickViewG(string MaxNumbers, int GroupNo, string GroupName, string DocType, string Img, string ImageColor, string NavigateUrl, string Folders, string StartDate, string EndDate, DocOrder iOrderBy, string[] aryClass, bool[] aryClassAnd, string Companies, params string[] Limit)

```
....
385.         catch(Exception e)
```

Insufficient Logging of Exceptions\路徑 5:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=518
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	404	404
物件	catch	catch

代碼片斷	
檔案名稱	PageSetting.cs
方法	private string Sub_QuickViewP(string MaxNumbers, string DocType, string ShowDocType, string Folders, string StartDate, string EndDate, DocOrder iOrderBy, string[] aryClass, bool[] aryClassAnd, string Companies, params string[] Limit)

```
....
404.           catch { iMaxNo = 13; }
```

Insufficient Logging of Exceptions\路徑 6:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=519
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	444	444
物件	catch	catch

代碼片斷	
檔案名稱	PageSetting.cs
方法	private string Sub_QuickViewP(string MaxNumbers, string DocType, string ShowDocType, string Folders, string StartDate, string EndDate, DocOrder iOrderBy, string[] aryClass, bool[] aryClassAnd, string Companies, params string[] Limit)

```
....
444.         catch(Exception e)
```

Insufficient Logging of Exceptions\路徑 7:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=520
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	ShowDocument.aspx.cs	ShowDocument.aspx.cs
行	144	144
物件	catch	catch

代碼片斷

檔案名稱	ShowDocument.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
144.                catch { }
```

Insufficient Logging of Exceptions\路徑 8:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=521
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	ShowDocument.aspx.cs	ShowDocument.aspx.cs
行	154	154
物件	catch	catch

代碼片斷	
檔案名稱	ShowDocument.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
154.          catch { }
```

Insufficient Logging of Exceptions\路徑 9:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=522
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebAttachLink.aspx.cs	WebAttachLink.aspx.cs
行	82	82
物件	catch	catch

代碼片斷	
檔案名稱	WebAttachLink.aspx.cs
方法	protected void Page_Load(object sender, EventArgs e)

```
....
82.         catch { }
```

Insufficient Logging of Exceptions\路徑 10:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=523
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebAttachLink.aspx.cs	WebAttachLink.aspx.cs
行	165	165
物件	catch	catch

代碼片斷

檔案名稱 WebAttachLink.aspx.cs

方法 private void BuildFileTable(string strDocXML)

```
....
165.         catch { }
```

Insufficient Logging of Exceptions\路徑 11:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=524
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebBatchPrint.aspx.cs	WebBatchPrint.aspx.cs
行	411	411
物件	catch	catch

代碼片斷

檔案名稱 WebBatchPrint.aspx.cs

方法 private void Get_Data()

```
....
411.                 catch { }
```

Insufficient Logging of Exceptions\路徑 12:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=525
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebBatchPrint.aspx.cs	WebBatchPrint.aspx.cs
行	422	422
物件	catch	catch

代碼片斷

檔案名稱 WebBatchPrint.aspx.cs

方法 private void Get_Data()

```
....
422.                catch { }
```

Insufficient Logging of Exceptions\路徑 13:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=526
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebBatchPrint.aspx.cs	WebBatchPrint.aspx.cs
行	455	455
物件	catch	catch

代碼片斷

檔案名稱 WebBatchPrint.aspx.cs

方法 private void GetDefault()

```
....
455.            catch { }
```

Insufficient Logging of Exceptions\路徑 14:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=527
狀態	反覆出現的問題

Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebBatchPrint.aspx.cs	WebBatchPrint.aspx.cs
行	467	467
物件	catch	catch

代碼片斷
檔案名稱 WebBatchPrint.aspx.cs
方法 private void GetDefault()

```
....
467.            catch { }
```

Insufficient Logging of Exceptions\路徑 15:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=528
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebBatchPrint.aspx.cs	WebBatchPrint.aspx.cs
行	499	499
物件	catch	catch

代碼片斷
檔案名稱 WebBatchPrint.aspx.cs
方法 protected void btnRead_Click(object sender, System.EventArgs e)

```
....
499.            catch { }
```

Insufficient Logging of Exceptions\路徑 16:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=529
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebBatchPrint.aspx.cs	WebBatchPrint.aspx.cs

行	511	511
物件	catch	catch

代碼片斷
檔案名稱 WebBatchPrint.aspx.cs
方法 protected void btnRead_Click(object sender, System.EventArgs e)

```
....
511.            catch { }
```

Insufficient Logging of Exceptions\路徑 17:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=530
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebBatchPrint.aspx.cs	WebBatchPrint.aspx.cs
行	839	839
物件	catch	catch

代碼片斷
檔案名稱 WebBatchPrint.aspx.cs
方法 private string GenHtml(string Type)

```
....
839.            catch { iWidth = 152; }
```

Insufficient Logging of Exceptions\路徑 18:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=531
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebBatchPrint.aspx.cs	WebBatchPrint.aspx.cs
行	886	886
物件	catch	catch

代碼片斷

檔案名稱	WebBatchPrint.aspx.cs
方法	private string GenHtml(string Type)

```
....
886.                    catch { strData =
objDR1.GetValue(2).ToString().Split(' ')[0]; }
```

Insufficient Logging of Exceptions\路徑 19:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=532
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebBatchPrint.aspx.cs	WebBatchPrint.aspx.cs
行	924	924
物件	catch	catch

代碼片斷	
檔案名稱	WebBatchPrint.aspx.cs
方法	private string GenHtml(string Type)

```
....
924.            catch { }
```

Insufficient Logging of Exceptions\路徑 20:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=533
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebBatchPrint.aspx.cs	WebBatchPrint.aspx.cs
行	967	967
物件	catch	catch

代碼片斷	
檔案名稱	WebBatchPrint.aspx.cs
方法	private void Build_ExcelTable()

```
....
967.          catch { iWidth = 150; }
```

Insufficient Logging of Exceptions\路徑 21:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=534
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebBatchPrint.aspx.cs	WebBatchPrint.aspx.cs
行	1013	1013
物件	catch	catch

代碼片斷

檔案名稱 WebBatchPrint.aspx.cs

方法 private void Build_ExcelTable()

```
....
1013.                   catch { strData =
objDR1.GetValue(2).ToString().Split(' ')[0]; }
```

Insufficient Logging of Exceptions\路徑 22:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=535
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebBatchPrint.aspx.cs	WebBatchPrint.aspx.cs
行	1046	1046
物件	catch	catch

代碼片斷

檔案名稱 WebBatchPrint.aspx.cs

方法 private void Build_ExcelTable()

```
....
1046.          catch { }
```

Insufficient Logging of Exceptions\路徑 23:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=536
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	406	406
物件	catch	catch

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private void Sub_Category(int Action)

```
....
406.            catch { }
```

Insufficient Logging of Exceptions\路徑 24:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=537
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	422	422
物件	catch	catch

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private void Sub_Category(int Action)

```
....
422.            catch(Exception ex)
```

Insufficient Logging of Exceptions\路徑 25:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=538

狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	443	443
物件	catch	catch

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private void Sub_Category(int Action)

```
....
443.            catch(Exception ex)
```

Insufficient Logging of Exceptions\路徑 26:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=539
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	695	695
物件	catch	catch

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private void Sub_FolderMtn(int Action)

```
....
695.            catch(Exception ex)
```

Insufficient Logging of Exceptions\路徑 27:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=540
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地

檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	799	799
物件	catch	catch

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private void Sub_FolderMtn(int Action)

```
....
799.              catch(Exception ex)
```

Insufficient Logging of Exceptions\路徑 28:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=541
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	2487	2487
物件	catch	catch

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private DateTime Sub_CDate(string Date, int ChkFlag)

```
....
2487.         catch
```

Insufficient Logging of Exceptions\路徑 29:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=542
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	2511	2511
物件	catch	catch

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private string Sub_TodayNews(string StartDate, string EndDate, int Days)

```
....
2511.         catch { };
```

Insufficient Logging of Exceptions\路徑 30:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=543
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	2513	2513
物件	catch	catch

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private string Sub_TodayNews(string StartDate, string EndDate, int Days)

```
....
2513.         catch { };
```

Insufficient Logging of Exceptions\路徑 31:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=544
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	3161	3161
物件	catch	catch

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private DateTime Sub_ToDate(string Date)

```
....
3161.         catch
```

Insufficient Logging of Exceptions\路徑 32:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=545
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	3643	3643
物件	catch	catch

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_HotDocument(string DataView, string TopRecords, string DocShowNames, string StartDate, string EndDate)

```
....
3643.         catch(Exception e)
```

Insufficient Logging of Exceptions\路徑 33:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=546
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	3998	3998
物件	catch	catch

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 public void Sub_PersonalSetting(int Action)

```
....
3998.             catch(Exception err)
```

Insufficient Logging of Exceptions\路徑 34:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=547
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	4449	4449
物件	catch	catch

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 public void Sub_GetPersonalSetting()

```
....
4449.         catch { PSET.RecordsPerPage = "20"; }
```

Insufficient Logging of Exceptions\路徑 35:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=548
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	4457	4457
物件	catch	catch

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 public void Sub_GetPersonalSetting()

```
....
4457.         catch { DOC.intSummaryLength = 250; }
```

Insufficient Logging of Exceptions\路徑 36:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=549

狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	4632	4632
物件	catch	catch

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private string Sub_DocShowNamesOnly(string DocShowNames)

```
....
4632.                catch { }
```

Insufficient Logging of Exceptions\路徑 37:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=550
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	4795	4795
物件	catch	catch

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 public string base64Encode(string data)

```
....
4795.        catch(Exception e)
```

Insufficient Logging of Exceptions\路徑 38:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=551
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地

檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	4815	4815
物件	catch	catch

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 public string base64Decode(string data)

```
....
4815.         catch(Exception e)
```

Insufficient Logging of Exceptions\路徑 39:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=552
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	4876	4876
物件	catch	catch

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private string GetUrlLink(string url)

```
....
4876.         catch
```

Insufficient Logging of Exceptions\路徑 40:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=553
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	5296	5296
物件	catch	catch

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private int SmartWebConfig(string sKey, int iDefault)

```
....
5296.         catch { }
```

Insufficient Logging of Exceptions\路徑 41:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=554
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	5305	5305
物件	catch	catch

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private bool SmartWebConfig(string sKey, bool bDefault)

```
....
5305.         catch { }
```

Insufficient Logging of Exceptions\路徑 42:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=555
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	5318	5318
物件	catch	catch

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private string SmartWebConfig(string sKey, int iDefault, bool bLarge0, bool bPositive)

```
....
5318.         catch { }
```

Insufficient Logging of Exceptions\路徑 43:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=556
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	5400	5400
物件	catch	catch

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Initialize()

```
....
5400.           catch { }
```

Insufficient Logging of Exceptions\路徑 44:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=557
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	5409	5409
物件	catch	catch

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Initialize()

```
....
5409.         catch(Exception ex)
```

Insufficient Logging of Exceptions\路徑 45:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=558
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	5534	5534
物件	catch	catch

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
5534.          catch { }
```

Insufficient Logging of Exceptions\路徑 46:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=559
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	5874	5874
物件	catch	catch

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
5874.          catch { DOC.varIntDocReviewReject = int.MaxValue; }
```

Insufficient Logging of Exceptions\路徑 47:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=560
狀態	反覆出現的問題

Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	6121	6121
物件	catch	catch

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
6121.                    catch { }
```

Insufficient Logging of Exceptions\路徑 48:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=561
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	6332	6332
物件	catch	catch

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
6332.        catch(Exception ex)
```

Insufficient Logging of Exceptions\路徑 49:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=562
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebDocument.aspx.cs	WebDocument.aspx.cs

行	93	93
物件	catch	catch

代碼片斷
檔案名稱 WebDocument.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
93.         catch { }
```

Insufficient Logging of Exceptions\路徑 50:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=563
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebDocumentLog.aspx.cs	WebDocumentLog.aspx.cs
行	46	46
物件	catch	catch

代碼片斷
檔案名稱 WebDocumentLog.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
46.         catch(Exception ex)
```

Insufficient Logging of Exceptions\路徑 51:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=564
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebDownloadFiles.aspx.cs	WebDownloadFiles.aspx.cs
行	263	263
物件	catch	catch

代碼片斷

檔案名稱	WebDownloadFiles.aspx.cs
方法	private void BuildFileTable(string strDocXML)

```
....
263.         catch { }
```

Insufficient Logging of Exceptions\路徑 52:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=565
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	712	712
物件	catch	catch

代碼片斷	
檔案名稱	WebEditor.aspx.cs
方法	private string CopyAttachmentFiles(string OldDocID, string DocID, string DocXML)

```
....
712.                 catch { }
```

Insufficient Logging of Exceptions\路徑 53:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=566
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	716	716
物件	catch	catch

代碼片斷	
檔案名稱	WebEditor.aspx.cs
方法	private string CopyAttachmentFiles(string OldDocID, string DocID, string DocXML)

```
....
716.                catch { }
```

Insufficient Logging of Exceptions\路徑 54:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=567
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	762	762
物件	catch	catch

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private void ExportXml()

```
....
762.                catch { }
```

Insufficient Logging of Exceptions\路徑 55:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=568
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	888	888
物件	catch	catch

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private void ExportXml()

```
....
888.            catch(Exception ex)
```

Insufficient Logging of Exceptions\路徑 56:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=569
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	1395	1395
物件	catch	catch

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private void ListAutoGen0(XmlList list1, string rowStyle, DocFieldData fieldData)

```
....
1395.               catch { }
```

Insufficient Logging of Exceptions\路徑 57:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=570
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	1400	1400
物件	catch	catch

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private void ListAutoGen0(XmlList list1, string rowStyle, DocFieldData fieldData)

```
....
1400.               catch { }
```

Insufficient Logging of Exceptions\路徑 58:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=571
狀態	反覆出現的問題

Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	1516	1516
物件	catch	catch

代碼片斷
檔案名稱 WebEditor.aspx.cs
方法 private void ListAutoGen1(XmlList list1, string rowStyle, DocFieldData fieldData, int DefaultRow)

```
....
1516.           catch { }
```

Insufficient Logging of Exceptions\路徑 59:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=572
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	1694	1694
物件	catch	catch

代碼片斷
檔案名稱 WebEditor.aspx.cs
方法 private void ListAutoGen6(XmlList list1, string rowStyle, DocFieldData fieldData)

```
....
1694.         catch { }
```

Insufficient Logging of Exceptions\路徑 60:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=573
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地

檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	2418	2418
物件	catch	catch

代碼片斷
檔案名稱 WebEditor.aspx.cs
方法 private void ListAutoGen53(XmlList list1, string rowStyle, DocFieldData fieldData)

```
....
2418.         catch { }
```

Insufficient Logging of Exceptions\路徑 61:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=574
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	2589	2589
物件	catch	catch

代碼片斷
檔案名稱 WebEditor.aspx.cs
方法 private void ListAutoGen54(XmlList list1, string rowStyle, DocFieldData fieldData)

```
....
2589.         catch { }
```

Insufficient Logging of Exceptions\路徑 62:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=575
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebGoAnyPage.ashx.cs	WebGoAnyPage.ashx.cs
行	102	102
物件	catch	catch

代碼片斷
檔案名稱 WebGoAnyPage.ashx.cs
方法 public void ProcessRequest(HttpContext context)

```
....
102.         catch { }
```

Insufficient Logging of Exceptions\路徑 63:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=576
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebGoAnyPage.ashx.cs	WebGoAnyPage.ashx.cs
行	105	105
物件	catch	catch

代碼片斷
檔案名稱 WebGoAnyPage.ashx.cs
方法 public void ProcessRequest(HttpContext context)

```
....
105.         catch { }
```

Insufficient Logging of Exceptions\路徑 64:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=577
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebGoAnyPage.ashx.cs	WebGoAnyPage.ashx.cs
行	108	108
物件	catch	catch

代碼片斷
檔案名稱 WebGoAnyPage.ashx.cs
方法 public void ProcessRequest(HttpContext context)

```
....
108.        catch { }
```

Insufficient Logging of Exceptions\路徑 65:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=578
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebGoAnyPageLoad.ashx.cs	WebGoAnyPageLoad.ashx.cs
行	121	121
物件	catch	catch

代碼片斷

檔案名稱 WebGoAnyPageLoad.ashx.cs

方法 public void ProcessRequest(HttpContext context)

```
....
121.        catch { }
```

Insufficient Logging of Exceptions\路徑 66:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=579
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebGoAnyPageLoad.ashx.cs	WebGoAnyPageLoad.ashx.cs
行	124	124
物件	catch	catch

代碼片斷

檔案名稱 WebGoAnyPageLoad.ashx.cs

方法 public void ProcessRequest(HttpContext context)

```
....
124.        catch { }
```

Insufficient Logging of Exceptions\路徑 67:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=580
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebGoAnyPageLoad.ashx.cs	WebGoAnyPageLoad.ashx.cs
行	127	127
物件	catch	catch

代碼片斷

檔案名稱 WebGoAnyPageLoad.ashx.cs

方法 public void ProcessRequest(HttpContext context)

```
....
127.         catch { }
```

Insufficient Logging of Exceptions\路徑 68:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=581
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebLogon.aspx.cs	WebLogon.aspx.cs
行	96	96
物件	catch	catch

代碼片斷

檔案名稱 WebLogon.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
96.             catch { strNewsNo = "10"; }
```

Insufficient Logging of Exceptions\路徑 69:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=582
狀態	反覆出現的問題

Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebLogon.aspx.cs	WebLogon.aspx.cs
行	192	192
物件	catch	catch

代碼片斷
檔案名稱 WebLogon.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
192.         catch(Exception ex)
```

Insufficient Logging of Exceptions\路徑 70:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=583
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebLogout.aspx.cs	WebLogout.aspx.cs
行	42	42
物件	catch	catch

代碼片斷
檔案名稱 WebLogout.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
42.          catch { }
```

Insufficient Logging of Exceptions\路徑 71:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=584
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebMailA.aspx.cs	WebMailA.aspx.cs

行	48	48
物件	catch	catch

代碼片斷
檔案名稱 WebMailA.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
48.         catch { }
```

Insufficient Logging of Exceptions\路徑 72:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=585
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebMailA.aspx.cs	WebMailA.aspx.cs
行	140	140
物件	catch	catch

代碼片斷
檔案名稱 WebMailA.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
140.                    catch { }
```

Insufficient Logging of Exceptions\路徑 73:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=586
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebMailA.aspx.cs	WebMailA.aspx.cs
行	147	147
物件	catch	catch

代碼片斷

檔案名稱	WebMailA.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
147.            catch { }
```

Insufficient Logging of Exceptions\路徑 74:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=587
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebMailA.aspx.cs	WebMailA.aspx.cs
行	175	175
物件	catch	catch

代碼片斷	
檔案名稱	WebMailA.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
175.        catch(Exception err)
```

Insufficient Logging of Exceptions\路徑 75:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=588
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebMark.aspx.cs	WebMark.aspx.cs
行	181	181
物件	catch	catch

代碼片斷	
檔案名稱	WebMark.aspx.cs
方法	protected void fBookAdd_Click(object sender, System.EventArgs e)

```
....
181.          catch { }
```

Insufficient Logging of Exceptions\路徑 76:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=589
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebMark.aspx.cs	WebMark.aspx.cs
行	225	225
物件	catch	catch

代碼片斷

檔案名稱 WebMark.aspx.cs

方法 protected void fGroupAdd_Click(object sender, System.EventArgs e)

```
....
225.          catch { }
```

Insufficient Logging of Exceptions\路徑 77:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=590
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	90	90
物件	catch	catch

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
90.         catch
```

Insufficient Logging of Exceptions\路徑 78:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=591
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	130	130
物件	catch	catch

代碼片斷

檔案名稱 WebPrint.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
130.         catch(Exception ex)
```

Insufficient Logging of Exceptions\路徑 79:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=592
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	305	305
物件	catch	catch

代碼片斷

檔案名稱 WebPrint.aspx.cs

方法 private void BindData(bool bBindData)

```
....
305.         catch(Exception err)
```

Insufficient Logging of Exceptions\路徑 80:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=593
狀態	反覆出現的問題

Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	348	348
物件	catch	catch

代碼片斷
檔案名稱 WebPrint.aspx.cs
方法 private void BindData(bool bBindData)

```
....
348.                catch { }
```

Insufficient Logging of Exceptions\路徑 81:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=594
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	371	371
物件	catch	catch

代碼片斷
檔案名稱 WebPrint.aspx.cs
方法 private void BindData(bool bBindData)

```
....
371.            catch { }
```

Insufficient Logging of Exceptions\路徑 82:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=595
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs

行	541	541
物件	catch	catch

代碼片斷
檔案名稱 WebPrint.aspx.cs
方法 private void PrintDoc()

```
....
541.                catch { }
```

Insufficient Logging of Exceptions\路徑 83:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=596
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	647	647
物件	catch	catch

代碼片斷
檔案名稱 WebPrint.aspx.cs
方法 private void SaveDocHTML()

```
....
647.                catch
```

Insufficient Logging of Exceptions\路徑 84:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=597
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	664	664
物件	catch	catch

代碼片斷

檔案名稱	WebPrint.aspx.cs
方法	private void SaveDocHTML()

```
....
664.                catch { }
```

Insufficient Logging of Exceptions\路徑 85:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=598
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	727	727
物件	catch	catch

代碼片斷	
檔案名稱	WebPrint.aspx.cs
方法	private void SaveDocXML()

```
....
727.                catch
```

Insufficient Logging of Exceptions\路徑 86:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=599
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	781	781
物件	catch	catch

代碼片斷	
檔案名稱	WebPrint.aspx.cs
方法	private void SaveDocXMLa()

```
....
781.                catch
```

Insufficient Logging of Exceptions\路徑 87:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=600
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	815	815
物件	catch	catch

代碼片斷

檔案名稱 WebPrint.aspx.cs

方法 protected void DataGrid1_PreRender(object sender, System.EventArgs e)

```
....
815.          catch { }
```

Insufficient Logging of Exceptions\路徑 88:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=601
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	923	923
物件	catch	catch

代碼片斷

檔案名稱 WebPrint.aspx.cs

方法 private string GenHtml()

```
....
923.            catch { strC5 = ""; }
```

Insufficient Logging of Exceptions\路徑 89:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=602
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebPrintAllData.aspx.cs	WebPrintAllData.aspx.cs
行	67	67
物件	catch	catch

代碼片斷

檔案名稱 WebPrintAllData.aspx.cs

方法 private int SmartWebConfig(string key, int defualt)

```
....
67.         catch { iResult = defualt; }
```

Insufficient Logging of Exceptions\路徑 90:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=603
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebPrintAllData.aspx.cs	WebPrintAllData.aspx.cs
行	106	106
物件	catch	catch

代碼片斷

檔案名稱 WebPrintAllData.aspx.cs

方法 private void ProcessType_0()

```
....
106.         catch { }
```

Exposure of Resource to Wrong Sphere

查詢路徑:

CSharp\Cx\CSharp Best Coding Practice\Exposure of Resource to Wrong Sphere 版本:1

類別

PCI DSS v3.2.1: PCI DSS (3.2.1) - 6.5.8 - Improper access control

OWASP Top 10 2013: A7-Missing Function Level Access Control
OWASP Top 10 2017: A5-Broken Access Control
OWASP Top 10 2021: A4-Insecure Design

描述

Exposure of Resource to Wrong Sphere\路徑 1:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=435
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, varPageSettingSpeedMsg, in PageSetting.cs line 16.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	16	16
物件	varPageSettingSpeedMsg	varPageSettingSpeedMsg

代碼片斷	
檔案名稱	PageSetting.cs
方法	public string varPageSettingSpeedMsg = "";

```
....
16.      public string varPageSettingSpeedMsg = "";
```

Exposure of Resource to Wrong Sphere\路徑 2:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=436
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, varSQLcommand, in PageSetting.cs line 17.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	17	17
物件	varSQLcommand	varSQLcommand

代碼片斷	
檔案名稱	PageSetting.cs
方法	public string varSQLcommand = "";

```
....
17.      public string varSQLcommand = "";
```

Exposure of Resource to Wrong Sphere\路徑 3:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=437
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, oLightC, in PageSetting.cs line 23.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	23	23
物件	oLightC	oLightC

代碼片斷

檔案名稱 PageSetting.cs

方法 public string oLightC = "#f0f8ff";

```
....
23.      public string oLightC = "#f0f8ff";
```

Exposure of Resource to Wrong Sphere\路徑 4:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=438
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, LogKey, in PageSetting.cs line 787.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	787	787
物件	LogKey	LogKey

代碼片斷

檔案名稱 PageSetting.cs

方法 public string LogKey = "";

```
....
787.        public string LogKey = "";
```

Exposure of Resource to Wrong Sphere\路徑 5:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=439
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, SelectShowItem, in PageSetting.cs line 789.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	789	789
物件	SelectShowItem	SelectShowItem

代碼片斷

檔案名稱 PageSetting.cs

方法 public string SelectShowItem = "";

```
....
789.        public string SelectShowItem = "";
```

Exposure of Resource to Wrong Sphere\路徑 6:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=440
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, DocTypes, in PageSetting.cs line 790.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	790	790
物件	DocTypes	DocTypes

代碼片斷

檔案名稱 PageSetting.cs

方法 public string DocTypes = "";

```
....
790.        public string DocTypes = "";
```

Exposure of Resource to Wrong Sphere\路徑 7:

嚴重程度：	資訊

結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=441
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, DocShowNames, in PageSetting.cs line 791.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	791	791
物件	DocShowNames	DocShowNames

代碼片斷	
檔案名稱	PageSetting.cs
方法	public string DocShowNames = "";

```
....
791.        public string DocShowNames = "";
```

Exposure of Resource to Wrong Sphere\路徑 8:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=442
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, Categories, in PageSetting.cs line 792.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	792	792
物件	Categories	Categories

代碼片斷	
檔案名稱	PageSetting.cs
方法	public string Categories = "";

```
....
792.        public string Categories = "";
```

Exposure of Resource to Wrong Sphere\路徑 9:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=443

狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, Keywords, in PageSetting.cs line 793.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	793	793
物件	Keywords	Keywords

代碼片斷

檔案名稱 PageSetting.cs

方法 public string Keywords = "";

```
....
793.        public string Keywords = "";
```

Exposure of Resource to Wrong Sphere\路徑 10:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=444
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, SearchWords, in PageSetting.cs line 794.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	794	794
物件	SearchWords	SearchWords

代碼片斷

檔案名稱 PageSetting.cs

方法 public string SearchWords = "";

```
....
794.        public string SearchWords = "";
```

Exposure of Resource to Wrong Sphere\路徑 11:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=445
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, SearchMore, in PageSetting.cs line 795.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	795	795
物件	SearchMore	SearchMore

代碼片斷
檔案名稱 PageSetting.cs
方法 public string SearchMore = "";

```
....
795.        public string SearchMore = "";
```

Exposure of Resource to Wrong Sphere\路徑 12:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=446
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:09 PM

The application exposes a public field, HotWords, in PageSetting.cs line 796.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	796	796
物件	HotWords	HotWords

代碼片斷
檔案名稱 PageSetting.cs
方法 public string HotWords = "";

```
....
796.        public string HotWords = "";
```

Exposure of Resource to Wrong Sphere\路徑 13:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=447
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:09 PM

The application exposes a public field, Companies, in PageSetting.cs line 797.

	來源	目的地

檔案	PageSetting.cs	PageSetting.cs
行	797	797
物件	Companies	Companies

代碼片斷
檔案名稱 PageSetting.cs
方法 public string Companies = "";

```
....
797.        public string Companies = "";
```

Exposure of Resource to Wrong Sphere\路徑 14:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=448
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, DocIDs, in PageSetting.cs line 798.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	798	798
物件	DocIDs	DocIDs

代碼片斷
檔案名稱 PageSetting.cs
方法 public string DocIDs = "";

```
....
798.        public string DocIDs = "";
```

Exposure of Resource to Wrong Sphere\路徑 15:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=449
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, StartDate, in PageSetting.cs line 799.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	799	799

物件	StartDate	StartDate

代碼片斷
檔案名稱 PageSetting.cs
方法 public string StartDate = "";

```
....
799.        public string StartDate = "";
```

Exposure of Resource to Wrong Sphere\路徑 16:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=450
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, EndDate, in PageSetting.cs line 800.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	800	800
物件	EndDate	EndDate

代碼片斷
檔案名稱 PageSetting.cs
方法 public string EndDate = "";

```
....
800.        public string EndDate = "";
```

Exposure of Resource to Wrong Sphere\路徑 17:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=451
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, Folders, in PageSetting.cs line 801.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	801	801
物件	Folders	Folders

代碼片斷

檔案名稱	PageSetting.cs
方法	public string Folders = "";

```
....
801.        public string Folders = "";
```

Exposure of Resource to Wrong Sphere\路徑 18:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=452
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, ReplaceWords, in PageSetting.cs line 802.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	802	802
物件	ReplaceWords	ReplaceWords

代碼片斷	
檔案名稱	PageSetting.cs
方法	public string ReplaceWords = "";

```
....
802.        public string ReplaceWords = "";
```

Exposure of Resource to Wrong Sphere\路徑 19:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=453
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, OrderBy, in PageSetting.cs line 803.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	803	803
物件	OrderBy	OrderBy

代碼片斷	
檔案名稱	PageSetting.cs
方法	public int OrderBy = 0;

```
....
803.        public int OrderBy = 0;
```

Exposure of Resource to Wrong Sphere\路徑 20:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=454
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, PageNumber, in PageSetting.cs line 804.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	804	804
物件	PageNumber	PageNumber

代碼片斷

檔案名稱 PageSetting.cs

方法 public int PageNumber = 0;

```
....
804.        public int PageNumber = 0;
```

Exposure of Resource to Wrong Sphere\路徑 21:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=455
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, LastIndex, in PageSetting.cs line 805.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	805	805
物件	LastIndex	LastIndex

代碼片斷

檔案名稱 PageSetting.cs

方法 public int LastIndex = 0;

```
....
805.        public int LastIndex = 0;
```

Exposure of Resource to Wrong Sphere\路徑 22:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=456
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, PageCount, in PageSetting.cs line 806.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	806	806
物件	PageCount	PageCount

代碼片斷

檔案名稱 PageSetting.cs

方法 public int PageCount = 0;

```
....
806.         public int PageCount = 0;
```

Exposure of Resource to Wrong Sphere\路徑 23:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=457
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, RecordCount, in PageSetting.cs line 807.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	807	807
物件	RecordCount	RecordCount

代碼片斷

檔案名稱 PageSetting.cs

方法 public int RecordCount = 0;

```
....
807.         public int RecordCount = 0;
```

Exposure of Resource to Wrong Sphere\路徑 24:

嚴重程度：	資訊

結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=458
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, UsingAttachmentSearch, in PageSetting.cs line 808.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	808	808
物件	UsingAttachmentSearch	UsingAttachmentSearch

代碼片斷	
檔案名稱	PageSetting.cs
方法	public int UsingAttachmentSearch = 0; // 94/11/28 ����j�M

```
....
808.        public int UsingAttachmentSearch = 0;  // 94/11/28
����j�M
```

Exposure of Resource to Wrong Sphere\路徑 25:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=459
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, SearchTypes, in PageSetting.cs line 809.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	809	809
物件	SearchTypes	SearchTypes

代碼片斷	
檔案名稱	PageSetting.cs
方法	public string SearchTypes = ""; // 94/11/28 ����j�M

```
....
809.        public string SearchTypes = "";  // 94/11/28 ����j�M
```

Exposure of Resource to Wrong Sphere\路徑 26:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=1

	0300&pathid=460
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, DocReviewReject, in PageSetting.cs line 810.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	810	810
物件	DocReviewReject	DocReviewReject

代碼片斷

檔案名稱 PageSetting.cs

方法 public string DocReviewReject = "";

```
....
810.        public string DocReviewReject = "";
```

Exposure of Resource to Wrong Sphere\路徑 27:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=461
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, ClearRelation, in PageSetting.cs line 811.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	811	811
物件	ClearRelation	ClearRelation

代碼片斷

檔案名稱 PageSetting.cs

方法 public string ClearRelation = "";

```
....
811.        public string ClearRelation = "";
```

Exposure of Resource to Wrong Sphere\路徑 28:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=462
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, AppSearch, in PageSetting.cs line 812.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	812	812
物件	AppSearch	AppSearch

代碼片斷
檔案名稱 PageSetting.cs
方法 public string AppSearch = "";

```
....
812.        public string AppSearch = "";
```

Exposure of Resource to Wrong Sphere\路徑 29:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=463
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, RelationCategories, in PageSetting.cs line 815.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	815	815
物件	RelationCategories	RelationCategories

代碼片斷
檔案名稱 PageSetting.cs
方法 public string RelationCategories = "";

```
....
815.        public string RelationCategories = "";
```

Exposure of Resource to Wrong Sphere\路徑 30:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=464
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, RelationCompanies, in PageSetting.cs line 816.

	來源	目的地

檔案	PageSetting.cs	PageSetting.cs
行	816	816
物件	RelationCompanies	RelationCompanies

代碼片斷
檔案名稱 PageSetting.cs
方法 public string RelationCompanies = "";

```
....
816.        public string RelationCompanies = "";
```

Exposure of Resource to Wrong Sphere\路徑 31:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=465
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, Units, in PageSetting.cs line 817.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	817	817
物件	Units	Units

代碼片斷
檔案名稱 PageSetting.cs
方法 public string Units = "", RelationUnits = "";

```
....
817.        public string Units = "", RelationUnits = "";
```

Exposure of Resource to Wrong Sphere\路徑 32:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=466
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, RelationUnits, in PageSetting.cs line 817.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	817	817

物件	RelationUnits	RelationUnits

代碼片斷
檔案名稱 PageSetting.cs
方法 public string Units = "", RelationUnits = "";

```
....
817.        public string Units = "", RelationUnits = "";
```

Exposure of Resource to Wrong Sphere\路徑 33:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=467
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:09 PM

The application exposes a public field, TdAppAttribute, in PageSetting.cs line 1869.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	1869	1869
物件	TdAppAttribute	TdAppAttribute

代碼片斷
檔案名稱 PageSetting.cs
方法 public string[] TdAppAttribute = new string[0];

```
....
1869.        public string[] TdAppAttribute = new string[0];
```

Exposure of Resource to Wrong Sphere\路徑 34:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=468
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:09 PM

The application exposes a public field, TdWidth, in PageSetting.cs line 1870.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	1870	1870
物件	TdWidth	TdWidth

代碼片斷

檔案名稱	PageSetting.cs
方法	public string[] TdWidth = new string[0];

```
....
1870.         public string[] TdWidth = new string[0];
```

Exposure of Resource to Wrong Sphere\路徑 35:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=469
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, AppParm, in PageSetting.cs line 1871.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	1871	1871
物件	AppParm	AppParm

代碼片斷	
檔案名稱	PageSetting.cs
方法	public string AppParm = "";

```
....
1871.         public string AppParm = "";
```

Exposure of Resource to Wrong Sphere\路徑 36:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=470
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, varText, in PageSetting.cs line 1872.

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	1872	1872
物件	varText	varText

代碼片斷	
檔案名稱	PageSetting.cs
方法	public System.Text.StringBuilder varText = new System.Text.StringBuilder();

```
....
1872.        public System.Text.StringBuilder varText = new
System.Text.StringBuilder();
```

Exposure of Resource to Wrong Sphere\路徑 37:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=471
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, varObjMenuTips, in WebCatalog.aspx.cs line 22.

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	22	22
物件	varObjMenuTips	varObjMenuTips

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 public static System.Collections.Hashtable varObjMenuTips = new Hashtable();

```
....
22.      public static System.Collections.Hashtable varObjMenuTips = new
Hashtable();
```

Exposure of Resource to Wrong Sphere\路徑 38:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=472
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, varChild, in WebCatalog.aspx.cs line 37.

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	37	37
物件	varChild	varChild

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 public string varChild = ""; // 2007/01/19

```
....
37.      public string varChild = ""; // 2007/01/19
```

Exposure of Resource to Wrong Sphere\路徑 39:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=473
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, WhoAmI, in WebCatalog.aspx.cs line 72.

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	72	72
物件	WhoAmI	WhoAmI

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 public USER WhoAmI = new USER();

```
....
72.      public USER WhoAmI = new USER();
```

Exposure of Resource to Wrong Sphere\路徑 40:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=474
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, oHeadC1, in WebCatalog.aspx.cs line 115.

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	115	115
物件	oHeadC1	oHeadC1

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 public string oHeadC1 = "#c7dde4", oHeadC2 = "#e0ffff", oTopBottomC = "#afd0f1", oMiddleC = "#ffffff", oDarkC = "#3182c2", oLightC = "#eef3f5";

```
....
115.      public string oHeadC1 = "#c7dde4", oHeadC2 = "#e0ffff",
oTopBottomC = "#afd0f1", oMiddleC = "#ffffff", oDarkC = "#3182c2",
oLightC = "#eef3f5";
```

Exposure of Resource to Wrong Sphere\路徑 41:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=475
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, oHeadC2, in WebCatalog.aspx.cs line 115.

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	115	115
物件	oHeadC2	oHeadC2

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 public string oHeadC1 = "#c7dde4", oHeadC2 = "#e0ffff", oTopBottomC = "#afd0f1", oMiddleC = "#ffffff", oDarkC = "#3182c2", oLightC = "#eef3f5";

```
....
115.      public string oHeadC1 = "#c7dde4", oHeadC2 = "#e0ffff",
oTopBottomC = "#afd0f1", oMiddleC = "#ffffff", oDarkC = "#3182c2",
oLightC = "#eef3f5";
```

Exposure of Resource to Wrong Sphere\路徑 42:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=476
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, oTopBottomC, in WebCatalog.aspx.cs line 115.

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	115	115
物件	oTopBottomC	oTopBottomC

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法	public string oHeadC1 = "#c7dde4", oHeadC2 = "#e0ffff", oTopBottomC = "#afd0f1", oMiddleC = "#ffffff", oDarkC = "#3182c2", oLightC = "#eef3f5";

```
....
115.      public string oHeadC1 = "#c7dde4", oHeadC2 = "#e0ffff",
oTopBottomC = "#afd0f1", oMiddleC = "#ffffff", oDarkC = "#3182c2",
oLightC = "#eef3f5";
```

Exposure of Resource to Wrong Sphere\路徑 43:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=477
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, oMiddleC, in WebCatalog.aspx.cs line 115.

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	115	115
物件	oMiddleC	oMiddleC

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	public string oHeadC1 = "#c7dde4", oHeadC2 = "#e0ffff", oTopBottomC = "#afd0f1", oMiddleC = "#ffffff", oDarkC = "#3182c2", oLightC = "#eef3f5";

```
....
115.      public string oHeadC1 = "#c7dde4", oHeadC2 = "#e0ffff",
oTopBottomC = "#afd0f1", oMiddleC = "#ffffff", oDarkC = "#3182c2",
oLightC = "#eef3f5";
```

Exposure of Resource to Wrong Sphere\路徑 44:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=478
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, oDarkC, in WebCatalog.aspx.cs line 115.

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	115	115
物件	oDarkC	oDarkC

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	public string oHeadC1 = "#c7dde4", oHeadC2 = "#e0ffff", oTopBottomC = "#afd0f1", oMiddleC = "#ffffff", oDarkC = "#3182c2", oLightC = "#eef3f5";

```
....
115.      public string oHeadC1 = "#c7dde4", oHeadC2 = "#e0ffff",
oTopBottomC = "#afd0f1", oMiddleC = "#ffffff", oDarkC = "#3182c2",
oLightC = "#eef3f5";
```

Exposure of Resource to Wrong Sphere\路徑 45:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=479
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, oLightC, in WebCatalog.aspx.cs line 115.

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	115	115
物件	oLightC	oLightC

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	public string oHeadC1 = "#c7dde4", oHeadC2 = "#e0ffff", oTopBottomC = "#afd0f1", oMiddleC = "#ffffff", oDarkC = "#3182c2", oLightC = "#eef3f5";

```
....
115.      public string oHeadC1 = "#c7dde4", oHeadC2 = "#e0ffff",
oTopBottomC = "#afd0f1", oMiddleC = "#ffffff", oDarkC = "#3182c2",
oLightC = "#eef3f5";
```

Exposure of Resource to Wrong Sphere\路徑 46:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=480
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, oRelTopBottomC, in WebCatalog.aspx.cs line 116.

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	116	116

物件	oRelTopBottomC	oRelTopBottomC

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 public string oRelTopBottomC = "#afd0f1", oRelMiddleC = "#eaf0f1";

```
....
116.      public string oRelTopBottomC = "#afd0f1", oRelMiddleC =
"#eaf0f1";
```

Exposure of Resource to Wrong Sphere\路徑 47:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=481
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:09 PM

The application exposes a public field, oRelMiddleC, in WebCatalog.aspx.cs line 116.

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	116	116
物件	oRelMiddleC	oRelMiddleC

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 public string oRelTopBottomC = "#afd0f1", oRelMiddleC = "#eaf0f1";

```
....
116.      public string oRelTopBottomC = "#afd0f1", oRelMiddleC =
"#eaf0f1";
```

Exposure of Resource to Wrong Sphere\路徑 48:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=482
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:09 PM

The application exposes a public field, strSourceData, in WebDownloadFiles.aspx.cs line 356.

	來源	目的地
檔案	WebDownloadFiles.aspx.cs	WebDownloadFiles.aspx.cs
行	356	356
物件	strSourceData	strSourceData

代碼片斷
檔案名稱 WebDownloadFiles.aspx.cs
方法 public string strSourceData;

```
....
356.      public string strSourceData;
```

Exposure of Resource to Wrong Sphere\路徑 49:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=483
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:09 PM

The application exposes a public field, strPath, in WebDownloadFiles.aspx.cs line 357.

	來源	目的地
檔案	WebDownloadFiles.aspx.cs	WebDownloadFiles.aspx.cs
行	357	357
物件	strPath	strPath

代碼片斷
檔案名稱 WebDownloadFiles.aspx.cs
方法 public string strPath;

```
....
357.      public string strPath;
```

Exposure of Resource to Wrong Sphere\路徑 50:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=484
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:09 PM

The application exposes a public field, strName, in WebDownloadFiles.aspx.cs line 358.

	來源	目的地
檔案	WebDownloadFiles.aspx.cs	WebDownloadFiles.aspx.cs
行	358	358
物件	strName	strName

代碼片斷
檔案名稱 WebDownloadFiles.aspx.cs

方法	public string strName;

```
....
358.      public string strName;
```

Exposure of Resource to Wrong Sphere\路徑 51:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=485
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, Status, in WebGetGroupNo.ashx.cs line 16.

	來源	目的地
檔案	WebGetGroupNo.ashx.cs	WebGetGroupNo.ashx.cs
行	16	16
物件	Status	Status

代碼片斷	
檔案名稱	WebGetGroupNo.ashx.cs
方法	public string Status = "";

```
....
16.        public string Status = "";
```

Exposure of Resource to Wrong Sphere\路徑 52:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=486
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, Message, in WebGetGroupNo.ashx.cs line 17.

	來源	目的地
檔案	WebGetGroupNo.ashx.cs	WebGetGroupNo.ashx.cs
行	17	17
物件	Message	Message

代碼片斷	
檔案名稱	WebGetGroupNo.ashx.cs
方法	public string Message = "";

```
....
17.         public string Message = "";
```

Exposure of Resource to Wrong Sphere\路徑 53:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=487
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, TextData, in WebGetGroupNo.ashx.cs line 18.

	來源	目的地
檔案	WebGetGroupNo.ashx.cs	WebGetGroupNo.ashx.cs
行	18	18
物件	TextData	TextData

代碼片斷

檔案名稱 WebGetGroupNo.ashx.cs

方法 public string TextData = "";

```
....
18.         public string TextData = "";
```

Exposure of Resource to Wrong Sphere\路徑 54:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=488
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, Status, in WebDeleteFiles.ashx.cs line 17.

	來源	目的地
檔案	WebDeleteFiles.ashx.cs	WebDeleteFiles.ashx.cs
行	17	17
物件	Status	Status

代碼片斷

檔案名稱 WebDeleteFiles.ashx.cs

方法 public string Status = "";

```
....
17.         public string Status = "";
```

Exposure of Resource to Wrong Sphere\路徑 55:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=489
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, Message, in WebDeleteFiles.ashx.cs line 18.

	來源	目的地
檔案	WebDeleteFiles.ashx.cs	WebDeleteFiles.ashx.cs
行	18	18
物件	Message	Message

代碼片斷

檔案名稱 WebDeleteFiles.ashx.cs

方法 public string Message = "";

```
....
18.         public string Message = "";
```

Exposure of Resource to Wrong Sphere\路徑 56:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=490
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, TextData, in WebDeleteFiles.ashx.cs line 19.

	來源	目的地
檔案	WebDeleteFiles.ashx.cs	WebDeleteFiles.ashx.cs
行	19	19
物件	TextData	TextData

代碼片斷

檔案名稱 WebDeleteFiles.ashx.cs

方法 public string TextData = "";

```
....
19.         public string TextData = "";
```

Exposure of Resource to Wrong Sphere\路徑 57:

嚴重程度：	資訊

結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=491
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, Status, in WebGoAnyPageLoad.ashx.cs line 16.

	來源	目的地
檔案	WebGoAnyPageLoad.ashx.cs	WebGoAnyPageLoad.ashx.cs
行	16	16
物件	Status	Status

代碼片斷

檔案名稱 WebGoAnyPageLoad.ashx.cs

方法 public string Status = "";

```
....
16.         public string Status = "";
```

Exposure of Resource to Wrong Sphere\路徑 58:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=492
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, Message, in WebGoAnyPageLoad.ashx.cs line 17.

	來源	目的地
檔案	WebGoAnyPageLoad.ashx.cs	WebGoAnyPageLoad.ashx.cs
行	17	17
物件	Message	Message

代碼片斷

檔案名稱 WebGoAnyPageLoad.ashx.cs

方法 public string Message = "";

```
....
17.         public string Message = "";
```

Exposure of Resource to Wrong Sphere\路徑 59:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=493

狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, Data1, in WebGoAnyPageLoad.ashx.cs line 18.

	來源	目的地
檔案	WebGoAnyPageLoad.ashx.cs	WebGoAnyPageLoad.ashx.cs
行	18	18
物件	Data1	Data1

代碼片斷

檔案名稱 WebGoAnyPageLoad.ashx.cs

方法 public string Data1 = "";

```
....
18.         public string Data1 = "";
```

Exposure of Resource to Wrong Sphere\路徑 60:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=494
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, Data2, in WebGoAnyPageLoad.ashx.cs line 19.

	來源	目的地
檔案	WebGoAnyPageLoad.ashx.cs	WebGoAnyPageLoad.ashx.cs
行	19	19
物件	Data2	Data2

代碼片斷

檔案名稱 WebGoAnyPageLoad.ashx.cs

方法 public string Data2 = "";

```
....
19.         public string Data2 = "";
```

Exposure of Resource to Wrong Sphere\路徑 61:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=495
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, Data3, in WebGoAnyPageLoad.ashx.cs line 20.

	來源	目的地
檔案	WebGoAnyPageLoad.ashx.cs	WebGoAnyPageLoad.ashx.cs
行	20	20
物件	Data3	Data3

代碼片斷
檔案名稱 WebGoAnyPageLoad.ashx.cs
方法 public string Data3 = "";

```
....
20.         public string Data3 = "";
```

Exposure of Resource to Wrong Sphere\路徑 62:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=496
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, Status, in WebGoAnyPage.ashx.cs line 11.

	來源	目的地
檔案	WebGoAnyPage.ashx.cs	WebGoAnyPage.ashx.cs
行	11	11
物件	Status	Status

代碼片斷
檔案名稱 WebGoAnyPage.ashx.cs
方法 public string Status = "";

```
....
11.       public string Status = "";
```

Exposure of Resource to Wrong Sphere\路徑 63:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=497
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, Message, in WebGoAnyPage.ashx.cs line 12.

	來源	目的地

檔案	WebGoAnyPage.ashx.cs	WebGoAnyPage.ashx.cs
行	12	12
物件	Message	Message

代碼片斷
檔案名稱 WebGoAnyPage.ashx.cs
方法 public string Message = "";

```
....
12.      public string Message = "";
```

Exposure of Resource to Wrong Sphere\路徑 64:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=498
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, Data1, in WebGoAnyPage.ashx.cs line 13.

	來源	目的地
檔案	WebGoAnyPage.ashx.cs	WebGoAnyPage.ashx.cs
行	13	13
物件	Data1	Data1

代碼片斷
檔案名稱 WebGoAnyPage.ashx.cs
方法 public string Data1 = "";

```
....
13.      public string Data1 = "";
```

Exposure of Resource to Wrong Sphere\路徑 65:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=499
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, Data2, in WebGoAnyPage.ashx.cs line 14.

	來源	目的地
檔案	WebGoAnyPage.ashx.cs	WebGoAnyPage.ashx.cs
行	14	14

物件	Data2	Data2

代碼片斷
檔案名稱 WebGoAnyPage.ashx.cs
方法 public string Data2 = "";

```
....
14.      public string Data2 = "";
```

Exposure of Resource to Wrong Sphere\路徑 66:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=500
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:09 PM

The application exposes a public field, Data3, in WebGoAnyPage.ashx.cs line 15.

	來源	目的地
檔案	WebGoAnyPage.ashx.cs	WebGoAnyPage.ashx.cs
行	15	15
物件	Data3	Data3

代碼片斷
檔案名稱 WebGoAnyPage.ashx.cs
方法 public string Data3 = "";

```
....
15.      public string Data3 = "";
```

Exposure of Resource to Wrong Sphere\路徑 67:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=501
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:09 PM

The application exposes a public field, Status, in WebGoAnyPage.ashx.cs line 25.

	來源	目的地
檔案	WebGoAnyPage.ashx.cs	WebGoAnyPage.ashx.cs
行	25	25
物件	Status	Status

代碼片斷

檔案名稱	WebGoAnyPage.ashx.cs
方法	public string Status = "";

```
....
25.         public string Status = "";
```

Exposure of Resource to Wrong Sphere\路徑 68:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=502
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, Message, in WebGoAnyPage.ashx.cs line 26.

	來源	目的地
檔案	WebGoAnyPage.ashx.cs	WebGoAnyPage.ashx.cs
行	26	26
物件	Message	Message

代碼片斷	
檔案名稱	WebGoAnyPage.ashx.cs
方法	public string Message = "";

```
....
26.         public string Message = "";
```

Exposure of Resource to Wrong Sphere\路徑 69:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=503
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, Data1, in WebGoAnyPage.ashx.cs line 27.

	來源	目的地
檔案	WebGoAnyPage.ashx.cs	WebGoAnyPage.ashx.cs
行	27	27
物件	Data1	Data1

代碼片斷	
檔案名稱	WebGoAnyPage.ashx.cs
方法	public string Data1 = "";

```
....
27.         public string Data1 = "";
```

Exposure of Resource to Wrong Sphere\路徑 70:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=504
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, Data2, in WebGoAnyPage.ashx.cs line 28.

	來源	目的地
檔案	WebGoAnyPage.ashx.cs	WebGoAnyPage.ashx.cs
行	28	28
物件	Data2	Data2

代碼片斷

檔案名稱 WebGoAnyPage.ashx.cs

方法 public string Data2 = "";

```
....
28.         public string Data2 = "";
```

Exposure of Resource to Wrong Sphere\路徑 71:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=505
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

The application exposes a public field, Data3, in WebGoAnyPage.ashx.cs line 29.

	來源	目的地
檔案	WebGoAnyPage.ashx.cs	WebGoAnyPage.ashx.cs
行	29	29
物件	Data3	Data3

代碼片斷

檔案名稱 WebGoAnyPage.ashx.cs

方法 public string Data3 = "";

```
....
29.         public string Data3 = "";
```

Exposure of Resource to Wrong Sphere\路徑 72:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=506
狀態	反覆出現的問題
Detection Date	11/14/2022 6:35:42 PM

The application exposes a public field, Status, in WebSaveParm1.ashx.cs line 20.

	來源	目的地
檔案	WebSaveParm1.ashx.cs	WebSaveParm1.ashx.cs
行	20	20
物件	Status	Status

代碼片斷

檔案名稱 WebSaveParm1.ashx.cs

方法 public string Status = "";

```
....
20.         public string Status = "";
```

Exposure of Resource to Wrong Sphere\路徑 73:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=507
狀態	反覆出現的問題
Detection Date	11/14/2022 6:35:42 PM

The application exposes a public field, Message, in WebSaveParm1.ashx.cs line 21.

	來源	目的地
檔案	WebSaveParm1.ashx.cs	WebSaveParm1.ashx.cs
行	21	21
物件	Message	Message

代碼片斷

檔案名稱 WebSaveParm1.ashx.cs

方法 public string Message = "";

```
....
21.         public string Message = "";
```

Insufficient Logging of Sensitive Operations

查詢路徑:

類別

OWASP Top 10 API: API10-Insufficient Logging and Monitoring
OWASP Top 10 2021: A9-Security Logging and Monitoring Failures

描述

Insufficient Logging of Sensitive Operations\路徑 1:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=604
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	6175	6175
物件	SysLogOut	SysLogOut

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
6175.               SysLogOut();
```

Insufficient Logging of Sensitive Operations\路徑 2:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=605
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebBatchPrint.aspx	WebBatchPrint.aspx
行	148	148
物件	btnDelete_Click	btnDelete_Click

代碼片斷
檔案名稱 WebBatchPrint.aspx
方法

```
....
148.
```

Insufficient Logging of Sensitive Operations\路徑 3:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=606
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebBatchPrint.aspx.cs	WebBatchPrint.aspx.cs
行	228	228
物件	Delete	Delete

代碼片斷

檔案名稱 WebBatchPrint.aspx.cs

方法 private void BuildGrid(string strDocDefID)

```
....
228.            aryRow0[0].Delete();
```

Insufficient Logging of Sensitive Operations\路徑 4:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=607
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebBatchPrint.aspx.cs	WebBatchPrint.aspx.cs
行	246	246
物件	Delete	Delete

代碼片斷

檔案名稱 WebBatchPrint.aspx.cs

方法 private void BuildGrid(string strDocDefID)

```
....
246.            aryRow0[0].Delete();
```

Insufficient Logging of Sensitive Operations\路徑 5:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=1

	0300&pathid=608
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	691	691
物件	DeleteFolder	DeleteFolder

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_FolderMtn(int Action)

```
....
691.                DOC.DeleteFolder(strFolderID, DeleteFlag);
```

Insufficient Logging of Sensitive Operations\路徑 6:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=609
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1217	1217
物件	DeleteDocument	DeleteDocument

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_Group(int Action)

```
....
1217.             if(DOC.DeleteDocument(strGroupID) > 0 && strDeleteFlag
!= "1") this.RequestQuery.DocIDs = "";
```

Insufficient Logging of Sensitive Operations\路徑 7:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=610
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	6055	6055
物件	sqlDeleteDocument	sqlDeleteDocument

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
6055.                this.Sub_Debug("Delete",
DOC.sqlDeleteDocument(strDelDocID, true, true, true, true).Replace(";",
";<br />"));
```

Insufficient Logging of Sensitive Operations\路徑 8:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=611
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	6060	6060
物件	DeleteDocument	DeleteDocument

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
6060.                if(DOC.DeleteDocument(strDelDocID, true, true, true,
true) > 0 && varAction != 1402) this.RequestQuery.DocIDs = "";
```

Insufficient Logging of Sensitive Operations\路徑 9:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=612
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebDeleteFiles.ashx.cs	WebDeleteFiles.ashx.cs

行	34	34
物件	DeleteAttachmentFilesByDocID	DeleteAttachmentFilesByDocID

代碼片斷
檔案名稱 WebDeleteFiles.ashx.cs
方法 public void ProcessRequest(HttpContext context)

```
....
34.          objDoc.DeleteAttachmentFilesByDocID(strDocID);
```

Insufficient Logging of Sensitive Operations\路徑 10:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=613
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebDeleteFiles.ashx.cs	WebDeleteFiles.ashx.cs
行	45	45
物件	DeleteDir	DeleteDir

代碼片斷
檔案名稱 WebDeleteFiles.ashx.cs
方法 public void ProcessRequest(HttpContext context)

```
....
45.              Cdsys.KM.Utility.FileUtil.DeleteDir(strUploadFilePath);
```

Insufficient Logging of Sensitive Operations\路徑 11:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=614
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebDownloadFiles.aspx.cs	WebDownloadFiles.aspx.cs
行	281	281
物件	Delete	Delete

代碼片斷

檔案名稱	WebDownloadFiles.aspx.cs
方法	private void ExportFiles()

```
....
281.         if(System.IO.Directory.Exists(varZipSourcePath))
System.IO.Directory.Delete(varZipSourcePath, true);
```

Insufficient Logging of Sensitive Operations\路徑 12:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=615
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebDownloadFiles.aspx.cs	WebDownloadFiles.aspx.cs
行	284	284
物件	Delete	Delete

代碼片斷	
檔案名稱	WebDownloadFiles.aspx.cs
方法	private void ExportFiles()

```
....
284.           if(System.IO.File.Exists(varZipSourcePath))
System.IO.File.Delete(varZipSourcePath);
```

Insufficient Logging of Sensitive Operations\路徑 13:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=616
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebDownloadFiles.aspx.cs	WebDownloadFiles.aspx.cs
行	318	318
物件	Delete	Delete

代碼片斷	
檔案名稱	WebDownloadFiles.aspx.cs
方法	private void ExportFiles()

```
....
318.            if(System.IO.Directory.Exists(varZipSourcePath))
System.IO.Directory.Delete(varZipSourcePath, true);
```

Insufficient Logging of Sensitive Operations\路徑 14:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=617
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebDownloadFiles.aspx.cs	WebDownloadFiles.aspx.cs
行	325	325
物件	Delete	Delete

代碼片斷

檔案名稱 WebDownloadFiles.aspx.cs

方法 private void ExportFiles()

```
....
325.            if(System.IO.Directory.Exists(varZipSourcePath))
System.IO.Directory.Delete(varZipSourcePath, true);
```

Insufficient Logging of Sensitive Operations\路徑 15:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=618
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	765	765
物件	DeleteDir	DeleteDir

代碼片斷

檔案名稱 WebEditor.aspx.cs

方法 private void ExportXml()

```
....
765.            Cdsys.KM.Utility.FileUtil.DeleteDir(_sUploadFilePath);
```

Insufficient Logging of Sensitive Operations\路徑 16:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=619
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	475	475
物件	DeleteBulletin	DeleteBulletin

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 private void Sub_MSG(string Action, string SysName, string ID, string Message, string URL, string StartDate, string EndDate)

```
....
475.            this.objDoc.DeleteBulletin(SysName, ID);
```

Insufficient Logging of Sensitive Operations\路徑 17:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=620
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCreateStdDoc.aspx	WebCreateStdDoc.aspx
行	120	120
物件	btnDelete_Click	btnDelete_Click

代碼片斷

檔案名稱 WebCreateStdDoc.aspx

方法

```
....
120.
```

Insufficient Logging of Sensitive Operations\路徑 18:

嚴重程度：	資訊
結果狀態：	校驗

線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=621
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebDocRecoverMtn.aspx	WebDocRecoverMtn.aspx
行	276	276
物件	btnDelete_Click	btnDelete_Click

代碼片斷

檔案名稱 WebDocRecoverMtn.aspx

方法

```
....
276.
```

Insufficient Logging of Sensitive Operations\路徑 19:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=622
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebUserIPEmail.aspx	WebUserIPEmail.aspx
行	181	181
物件	btnDelete_Click	btnDelete_Click

代碼片斷

檔案名稱 WebUserIPEmail.aspx

方法

```
....
181.
```

Insufficient Logging of Sensitive Operations\路徑 20:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=623
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	PageSetting.cs	PageSetting.cs
行	699	699
物件	Remove	Remove

代碼片斷
檔案名稱 PageSetting.cs
方法 public string GenerateRelatedTo(string ItemName, string[] ItemValue, string ItemCmd, bool IsWeight, string CheckItem, string BgColor, int HiddenNumber, string CheckCmd)

```
....
699.            strData.Remove(0, strData.Length);
```

Insufficient Logging of Sensitive Operations\路徑 21:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=624
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	417	417
物件	RemoveCategory	RemoveCategory

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private void Sub_Category(int Action)

```
....
417.              this.DOC.RemoveCategory(strCategoryID);
```

Insufficient Logging of Sensitive Operations\路徑 22:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=625
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	4965	4965

物件	Remove	Remove

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private string Sub_Sorting(string[] All, string[] Items)

```
....
4965.         if(strA.Length > 0) strA = strA.Remove(strA.Length - 1, 1);
```

Insufficient Logging of Sensitive Operations\路徑 23:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=626
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	5208	5208
物件	Remove	Remove

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private string ComparePaperItems(string OldPapers, string NewPapers)

```
....
5208.         if(strOldPapers.StartsWith(",")) strOldPapers =
strOldPapers.Remove(0, 1);
```

Insufficient Logging of Sensitive Operations\路徑 24:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=627
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	5209	5209
物件	Remove	Remove

代碼片斷
檔案名稱 WebCatalog.aspx.cs

方法	private string ComparePaperItems(string OldPapers, string NewPapers)

```
....
5209.         if(strOldPapers.EndsWith(",")) strOldPapers =
strOldPapers.Remove(strOldPapers.Length - 1, 1);
```

Insufficient Logging of Sensitive Operations\路徑 25:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=628
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	464	464
物件	RemoveChild	RemoveChild

代碼片斷	
檔案名稱	WebEditor.aspx.cs
方法	protected void Page_Load(object sender, System.EventArgs e)

```
....
464.                          if(xNodeX != null)
oXml.DocumentElement.RemoveChild(xNodeX);
```

Insufficient Logging of Sensitive Operations\路徑 26:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=629
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	804	804
物件	SetFolderAuthority	SetFolderAuthority

代碼片斷	
檔案名稱	WebCatalog.aspx.cs
方法	private void Sub_FolderMtn(int Action)

```
....
804.              DOC.SetFolderAuthority(strFolderID,
strPublicArea.Split(','), rtnPublicArea.Split(','));
```

Insufficient Logging of Sensitive Operations\路徑 27:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=630
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	892	892
物件	GetFolderAuthority	GetFolderAuthority

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_FolderMtn(int Action)

```
....
892.            objDR1 = DOC.GetFolderAuthority(strFolderID);
```

Insufficient Logging of Sensitive Operations\路徑 28:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=631
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1361	1361
物件	GetDocTypeAuthority	GetDocTypeAuthority

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_MIS(string DocShowNames)

```
....
1361.              strDocShowNames =
DOC.GetDocTypeAuthority(DocShowNames, true, AppFunction.Create);
```

Insufficient Logging of Sensitive Operations\路徑 29:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=632
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1386	1386
物件	GetDocTypeAuthority	GetDocTypeAuthority

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_MIS(string DocShowNames)

```
....
1386.                strDocShowNames =
DOC.GetDocTypeAuthority(DocShowNames, true, AppFunction.BatchPrint);
```

Insufficient Logging of Sensitive Operations\路徑 30:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=633
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1403	1403
物件	GetDocTypeAuthority	GetDocTypeAuthority

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_MIS(string DocShowNames)

```
....
1403.             if((_bShowPrint == false) || (DocShowNames ==
DOC.GetDocTypeAuthority(DocShowNames, true, AppFunction.HardCopy)) ||
(DOC.User.Class >= 3))
```

Insufficient Logging of Sensitive Operations\路徑 31:

嚴重程度：	資訊

結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=634
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1615	1615
物件	GetDocTypeAuthority	GetDocTypeAuthority

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_1301()

```
....
1615.                strTemp = DOC.GetDocTypeAuthority(strTemp, true,
AppFunction.Create);
```

Insufficient Logging of Sensitive Operations\路徑 32:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=635
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1621	1621
物件	GetDocTypeAuthority	GetDocTypeAuthority

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_1301()

```
....
1621.                  strTemp = DOC.GetDocTypeAuthority(strTemp, true,
AppFunction.Read);
```

Insufficient Logging of Sensitive Operations\路徑 33:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=636
狀態	反覆出現的問題

Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1628	1628
物件	GetDocTypeAuthority	GetDocTypeAuthority

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private void Sub_1301()

```
....
1628.                    strTemp = DOC.GetDocTypeAuthority(strTemp, true,
AppFunction.BatchPrint);
```

Insufficient Logging of Sensitive Operations\路徑 34:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=637
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1705	1705
物件	SysFunAuthorize	SysFunAuthorize

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private void Sub_2120()

```
....
1705.        if(DOC.SysFunAuthorize("9330")) // 2007/05/24 小品欣賞=9330
```

Insufficient Logging of Sensitive Operations\路徑 35:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=638
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地

檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1750	1750
物件	SysFunAuthorize	SysFunAuthorize

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private void Sub_2110()

```
....
1750.         if(DOC.SysFunAuthorize("9310")) // 2007/05/24 訊息公告=9310
```

Insufficient Logging of Sensitive Operations\路徑 36:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=639
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	1793	1793
物件	SysFunAuthorize	SysFunAuthorize

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private void Sub_2130()

```
....
1793.         if(DOC.SysFunAuthorize("9340")) // 2007/05/24 好書推薦=9340
```

Insufficient Logging of Sensitive Operations\路徑 37:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=640
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	2386	2386
物件	DocTypeAuthority	DocTypeAuthority

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private void Sub_100()

```
....
2386.          DOC.DocTypeAuthority(); // 2009/03/18
```

Insufficient Logging of Sensitive Operations\路徑 38:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=641
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	2938	2938
物件	DocumentAuthorize	DocumentAuthorize

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private void Sub_400()

```
....
2938.          DOC.DocumentAuthorize(RequestQuery.DocIDs); // 2008/01/07
```

Insufficient Logging of Sensitive Operations\路徑 39:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=642
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	4633	4633
物件	GetDocTypeAuthority1	GetDocTypeAuthority1

代碼片斷
檔案名稱 WebCatalog.aspx.cs
方法 private string Sub_DocShowNamesOnly(string DocShowNames)

```
....
4633.                 if(strProtect[0] !=
DOC.GetDocTypeAuthority1(strProtect[0], true, intFunction))
```

Insufficient Logging of Sensitive Operations\路徑 40:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=643
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	5244	5244
物件	GetDocTypeAuthority	GetDocTypeAuthority

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 private void Sub_BuildDocItems()

```
....
5244.           varDocShowNames =
this.Sub_DocShowNamesOnly(DOC.GetDocTypeAuthority(strDocAllNames, false,
AppFunction.Read, AppFunction.Create, AppFunction.Update,
AppFunction.Delete, AppFunction.Manage, AppFunction.BatchPrint,
AppFunction.HardCopy));
```

Insufficient Logging of Sensitive Operations\路徑 41:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=644
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	5614	5614
物件	GetDocTypeAuthority	GetDocTypeAuthority

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
5614.          bool bGroup = (this._bUseGroup &&
DOC.GetDocTypeAuthority("文件卷夾內的文件", true, AppFunction.Delete) !=
"") ? true : false;
```

Insufficient Logging of Sensitive Operations\路徑 42:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=645
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.cs	WebCatalog.aspx.cs
行	5616	5616
物件	GetDocTypeAuthority	GetDocTypeAuthority

代碼片斷

檔案名稱 WebCatalog.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
5616.          if(this._bUseGroup &&
DOC.GetDocTypeAuthority("文件卷夾內的文件", true, AppFunction.Create) !=
"")
```

Insufficient Logging of Sensitive Operations\路徑 43:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=646
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebGoAnyPage.ashx.cs	WebGoAnyPage.ashx.cs
行	67	67
物件	GetDocTypeAuthority	GetDocTypeAuthority

代碼片斷

檔案名稱 WebGoAnyPage.ashx.cs

方法 public void ProcessRequest(HttpContext context)

```
....
67.         if(strUseGroup == "true" &&
objDoc.GetDocTypeAuthority("虛擬卷夾內的文件", true,
Cdsys.KM.Struct.AppFunction.Create) != "")
```

Insufficient Logging of Sensitive Operations\路徑 44:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=647
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebGoAnyPageLoad.ashx.cs	WebGoAnyPageLoad.ashx.cs
行	86	86
物件	GetDocTypeAuthority	GetDocTypeAuthority

代碼片斷

檔案名稱 WebGoAnyPageLoad.ashx.cs

方法 public void ProcessRequest(HttpContext context)

```
....
86.         if(strUseGroup == "true" &&
objDoc.GetDocTypeAuthority("虛擬卷夾內的文件", true,
Cdsys.KM.Struct.AppFunction.Create) != "")
```

Insufficient Logging of Sensitive Operations\路徑 45:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=648
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	112	112
物件	SysFunAuthorize	SysFunAuthorize

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
112.                if(objDoc.SysFunAuthorize("9310")) // 2007/05/24
                    T     i=9310
```

Insufficient Logging of Sensitive Operations\路徑 46:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=649
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	138	138
物件	SysFunAuthorize	SysFunAuthorize

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
138.                if(objDoc.SysFunAuthorize("9340")) // 2007/05/24
                    n     =9340
```

Insufficient Logging of Sensitive Operations\路徑 47:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=650
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebNotePad.aspx.cs	WebNotePad.aspx.cs
行	164	164
物件	SysFunAuthorize	SysFunAuthorize

代碼片斷

檔案名稱 WebNotePad.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
164.              if(objDoc.SysFunAuthorize("9330")) // 2007/05/24
�p�~�Y��=9330
```

Insufficient Logging of Sensitive Operations\路徑 48:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=651
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebPrint.aspx.cs	WebPrint.aspx.cs
行	66	66
物件	DocTypeAuthority	DocTypeAuthority

代碼片斷

檔案名稱 WebPrint.aspx.cs

方法 protected void Page_Load(object sender, System.EventArgs e)

```
....
66.             objDoc.DocTypeAuthority(); // 2007/07/27
```

Insufficient Logging of Sensitive Operations\路徑 49:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=652
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebPrintAllData.aspx.cs	WebPrintAllData.aspx.cs
行	144	144
物件	GetDocTypeAuthority	GetDocTypeAuthority

代碼片斷

檔案名稱 WebPrintAllData.aspx.cs

方法 private void ProcessType_0()

```
....
144.          if(strUseGroup == "true" &&
objDOC.GetDocTypeAuthority("������������������", true,
Cdsys.KM.Struct.AppFunction.Create) != "")
```

Pages Without Global Error Handler

查詢路徑:

CSharp\Cx\CSharp Best Coding Practice\Pages Without Global Error Handler 版本:1

類別

PCI DSS v3.2.1: PCI DSS (3.2.1) - 6.5.5 - Improper error handling
OWASP Top 10 2013: A6-Sensitive Data Exposure
OWASP Top 10 2017: A3-Sensitive Data Exposure
OWASP Top 10 2021: A4-Insecure Design

描述

Pages Without Global Error Handler\路徑 1:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=653
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	ShowDocument.aspx.designer.cs	ShowDocument.aspx.designer.cs
行	13	13
物件	ShowDocument	ShowDocument

代碼片斷

檔案名稱 ShowDocument.aspx.designer.cs

方法 public partial class ShowDocument {

```
....
13.      public partial class ShowDocument {
```

Pages Without Global Error Handler\路徑 2:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=654
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	ShowLogDocument.aspx.designer.cs	ShowLogDocument.aspx.designer.cs

行	13	13
物件	ShowLogDocument	ShowLogDocument

代碼片斷
檔案名稱 ShowLogDocument.aspx.designer.cs
方法 public partial class ShowLogDocument {

```
....
13.      public partial class ShowLogDocument {
```

Pages Without Global Error Handler\路徑 3:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=655
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebAttachLink.aspx.designer.cs	WebAttachLink.aspx.designer.cs
行	13	13
物件	WebAttachLink	WebAttachLink

代碼片斷
檔案名稱 WebAttachLink.aspx.designer.cs
方法 public partial class WebAttachLink {

```
....
13.      public partial class WebAttachLink {
```

Pages Without Global Error Handler\路徑 4:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=656
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebBatchPrint.aspx.designer.cs	WebBatchPrint.aspx.designer.cs
行	13	13
物件	BatchPrint	BatchPrint

代碼片斷

檔案名稱	WebBatchPrint.aspx.designer.cs
方法	public partial class BatchPrint {

```
....
13.      public partial class BatchPrint {
```

Pages Without Global Error Handler\路徑 5:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=657
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebBulletin.aspx.designer.cs	WebBulletin.aspx.designer.cs
行	13	13
物件	WebBulletin	WebBulletin

代碼片斷	
檔案名稱	WebBulletin.aspx.designer.cs
方法	public partial class WebBulletin {

```
....
13.      public partial class WebBulletin {
```

Pages Without Global Error Handler\路徑 6:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=658
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebCatalog.aspx.designer.cs	WebCatalog.aspx.designer.cs
行	13	13
物件	WebCatalog	WebCatalog

代碼片斷	
檔案名稱	WebCatalog.aspx.designer.cs
方法	public partial class WebCatalog {

```
....
13.      public partial class WebCatalog {
```

Pages Without Global Error Handler\路徑 7:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=659
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebDocPrint.aspx.designer.cs	WebDocPrint.aspx.designer.cs
行	13	13
物件	WebDocPrint	WebDocPrint

代碼片斷

檔案名稱 WebDocPrint.aspx.designer.cs

方法 public partial class WebDocPrint {

```
....
13.      public partial class WebDocPrint {
```

Pages Without Global Error Handler\路徑 8:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=660
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebDocument.aspx.designer.cs	WebDocument.aspx.designer.cs
行	13	13
物件	WebDocument	WebDocument

代碼片斷

檔案名稱 WebDocument.aspx.designer.cs

方法 public partial class WebDocument {

```
....
13.      public partial class WebDocument {
```

Pages Without Global Error Handler\路徑 9:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=661
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebDocumentLog.aspx.designer.cs	WebDocumentLog.aspx.designer.cs
行	13	13
物件	WebDocumentLog	WebDocumentLog

代碼片斷

檔案名稱 WebDocumentLog.aspx.designer.cs

方法 public partial class WebDocumentLog {

```
....
13.      public partial class WebDocumentLog {
```

Pages Without Global Error Handler\路徑 10:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=662
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebDownloadFiles.aspx.designer.cs	WebDownloadFiles.aspx.designer.cs
行	13	13
物件	DownloadFiles	DownloadFiles

代碼片斷

檔案名稱 WebDownloadFiles.aspx.designer.cs

方法 public partial class DownloadFiles {

```
....
13.      public partial class DownloadFiles {
```

Pages Without Global Error Handler\路徑 11:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=663
狀態	反覆出現的問題

Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebEditor.aspx.designer.cs	WebEditor.aspx.designer.cs
行	13	13
物件	WebEditor	WebEditor

代碼片斷
檔案名稱 WebEditor.aspx.designer.cs
方法 public partial class WebEditor {

```
....
13.      public partial class WebEditor {
```

Pages Without Global Error Handler\路徑 12:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=664
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebError.aspx.designer.cs	WebError.aspx.designer.cs
行	13	13
物件	WebError	WebError

代碼片斷
檔案名稱 WebError.aspx.designer.cs
方法 public partial class WebError {

```
....
13.      public partial class WebError {
```

Pages Without Global Error Handler\路徑 13:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=665
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebExportXML.aspx.designer.cs	WebExportXML.aspx.designer.cs

行	13	13
物件	WebExportXML	WebExportXML

代碼片斷
檔案名稱 WebExportXML.aspx.designer.cs
方法 public partial class WebExportXML {

```
....
13.      public partial class WebExportXML {
```

Pages Without Global Error Handler\路徑 14:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=666
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebLogon.aspx.designer.cs	WebLogon.aspx.designer.cs
行	13	13
物件	WebLogon	WebLogon

代碼片斷
檔案名稱 WebLogon.aspx.designer.cs
方法 public partial class WebLogon {

```
....
13.      public partial class WebLogon {
```

Pages Without Global Error Handler\路徑 15:

嚴重程度： 資訊
結果狀態： 校驗
線上結果 http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=667
狀態 反覆出現的問題
Detection Date 7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebLogout.aspx.designer.cs	WebLogout.aspx.designer.cs
行	13	13
物件	WebLogout	WebLogout

代碼片斷

檔案名稱	WebLogout.aspx.designer.cs
方法	public partial class WebLogout {

```
....
13.      public partial class WebLogout {
```

Pages Without Global Error Handler\路徑 16:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=668
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebMailA.aspx.designer.cs	WebMailA.aspx.designer.cs
行	13	13
物件	WebMailA	WebMailA

代碼片斷	
檔案名稱	WebMailA.aspx.designer.cs
方法	public partial class WebMailA {

```
....
13.      public partial class WebMailA {
```

Pages Without Global Error Handler\路徑 17:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=669
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebMailTo.aspx.designer.cs	WebMailTo.aspx.designer.cs
行	13	13
物件	WebMailTo	WebMailTo

代碼片斷	
檔案名稱	WebMailTo.aspx.designer.cs
方法	public partial class WebMailTo {

```
....
13.      public partial class WebMailTo {
```

Pages Without Global Error Handler\路徑 18:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=670
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebMark.aspx.designer.cs	WebMark.aspx.designer.cs
行	13	13
物件	Mark	Mark

代碼片斷

檔案名稱 WebMark.aspx.designer.cs

方法 public partial class Mark {

```
....
13.      public partial class Mark {
```

Pages Without Global Error Handler\路徑 19:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=671
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebMIS.aspx.designer.cs	WebMIS.aspx.designer.cs
行	13	13
物件	WebMIS	WebMIS

代碼片斷

檔案名稱 WebMIS.aspx.designer.cs

方法 public partial class WebMIS {

```
....
13.      public partial class WebMIS {
```

Pages Without Global Error Handler\路徑 20:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=672
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebNotePad.aspx.designer.cs	WebNotePad.aspx.designer.cs
行	13	13
物件	WebNotePad	WebNotePad

代碼片斷

檔案名稱 WebNotePad.aspx.designer.cs

方法 public partial class WebNotePad {

```
....
13.      public partial class WebNotePad {
```

Pages Without Global Error Handler\路徑 21:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=673
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebPrint.aspx.designer.cs	WebPrint.aspx.designer.cs
行	13	13
物件	WebPrint	WebPrint

代碼片斷

檔案名稱 WebPrint.aspx.designer.cs

方法 public partial class WebPrint {

```
....
13.      public partial class WebPrint {
```

Pages Without Global Error Handler\路徑 22:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=674
狀態	反覆出現的問題

Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebPrintAllData.aspx.designer.cs	WebPrintAllData.aspx.designer.cs
行	13	13
物件	WebPrintAllData	WebPrintAllData

代碼片斷
檔案名稱 WebPrintAllData.aspx.designer.cs
方法 public partial class WebPrintAllData {

```
....
13.      public partial class WebPrintAllData {
```

Pages Without Global Error Handler\路徑 23:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=675
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:10 PM

	來源	目的地
檔案	WebRedirect.aspx.designer.cs	WebRedirect.aspx.designer.cs
行	13	13
物件	WebRedirect	WebRedirect

代碼片斷
檔案名稱 WebRedirect.aspx.designer.cs
方法 public partial class WebRedirect {

```
....
13.      public partial class WebRedirect {
```

Hardcoded Absolute Path

查詢路徑:

CSharp\Cx\CSharp Best Coding Practice\Hardcoded Absolute Path 版本:1

類別

OWASP Top 10 2021: A8-Software and Data Integrity Failures

描述

Hardcoded Absolute Path\路徑 1:

嚴重程度：	資訊
結果狀態：	校驗

線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=508
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

<?xml version="1.0"?> 方法使用 bin/Websys.KM.Authority.Control.CBCCASE.dll.config 第 1 行中寫死的絕對路徑 "c:\\temp" 來引用外部檔案

	來源	目的地
檔案	bin/Websys.KM.Authority.Control.CBCCASE.dll.config	bin/Websys.KM.Authority.Control.CBCCASE.dll.config
行	21	21
物件	"c:\\temp"	"c:\\temp"

代碼片斷	
檔案名稱	bin/Websys.KM.Authority.Control.CBCCASE.dll.config
方法	<?xml version="1.0"?>

```
....
21.          <add key="ExportPath" value="c:\temp"/>
```

Hardcoded Absolute Path\路徑 2:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=509
狀態	反覆出現的問題
Detection Date	11/14/2022 6:35:42 PM

<?xml version="1.0"?> 方法使用 Web.config 第 1 行中寫死的絕對路徑 "c:\\temp\\XmlExport\\" 來引用外部檔案

	來源	目的地
檔案	Web.config	Web.config
行	96	96
物件	"c:\\temp\\XmlExport\\"	"c:\\temp\\XmlExport\\"

代碼片斷	
檔案名稱	Web.config
方法	<?xml version="1.0"?>

```
....
96.       <add key="XmlExportPath#" value="c:\temp\XmlExport\"/>
```

Hardcoded Absolute Path\路徑 3:

嚴重程度：	資訊
結果狀態：	校驗

線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=510
狀態	反覆出現的問題
Detection Date	11/14/2022 6:35:42 PM

<?xml version="1.0"?> 方法使用 Web.config 第 1 行中寫死的絕對路徑 "c:\\temp\\SQLdata\\" 來引用外部檔案

	來源	目的地
檔案	Web.config	Web.config
行	97	97
物件	"c:\\temp\\SQLdata\\"	"c:\\temp\\SQLdata\\"

代碼片斷	
檔案名稱	Web.config
方法	<?xml version="1.0"?>

```
....
97.       <add key="SQLPath#" value="c:\temp\SQLdata\"/>
```

Hardcoded Absolute Path\路徑 4:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=511
狀態	反覆出現的問題
Detection Date	11/14/2022 6:35:42 PM

<?xml version="1.0"?> 方法使用 Web.config 第 1 行中寫死的絕對路徑 "c:\\temp\\" 來引用外部檔案

	來源	目的地
檔案	Web.config	Web.config
行	98	98
物件	"c:\\temp\\"	"c:\\temp\\"

代碼片斷	
檔案名稱	Web.config
方法	<?xml version="1.0"?>

```
....
98.       <add key="LogPath#" value="c:\temp\"/>
```

Hardcoded Absolute Path\路徑 5:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=512
狀態	反覆出現的問題

Detection Date	11/14/2022 6:35:42 PM

<?xml version="1.0"?> 方法使用 Web.config 第 1 行中寫死的絕對路徑 "c:\\temp\\" 來引用外部檔案

	來源	目的地
檔案	Web.config	Web.config
行	99	99
物件	"c:\\temp\\"	"c:\\temp\\"

代碼片斷
檔案名稱 Web.config
方法 <?xml version="1.0"?>

```
....
99.       <add key="ErrorPath#" value="c:\temp\"/>
```

Hardcoded Absolute Path\路徑 6:

嚴重程度：	資訊
結果狀態：	校驗
線上結果	http://checkmarx.tst.gov.tw/CxWebClient/ViewerMain.aspx?scanid=1132233&projectid=10300&pathid=513
狀態	反覆出現的問題
Detection Date	7/8/2022 3:05:09 PM

"C:\\temp\\CdsUploadTemp"; 方法使用 WebEditor.aspx.cs 第 54 行中寫死的絕對路徑 "C:\\temp\\CdsUploadTemp" 來引用外部檔案

	來源	目的地
檔案	WebEditor.aspx.cs	WebEditor.aspx.cs
行	54	54
物件	"C:\\temp\\CdsUploadTemp"	"C:\\temp\\CdsUploadTemp"

代碼片斷
檔案名稱 WebEditor.aspx.cs
方法 private string _sFileUpPath = "", _sUploadFilePath = "C:\\temp\\CdsUploadTemp";

```
....
54.       private string _sFileUpPath = "", _sUploadFilePath =
"C:\\temp\\CdsUploadTemp";
```

HttpOnlyCookies

風險

可能發生什麼問題

Cookies 中包含許多身分驗證所需的機敏資料，預設情況下可以透過客戶端腳本(例如JavaScript)輕易存取。除非 Web 應用程式使用 "httpOnly" 的 cookie 標記，明確阻止了這種情況發生，否則這些 cookie 可能被惡意的客戶端腳本存取，例如跨網站指令碼（XSS）。根據"深度防禦(Defense in Depth)"，若系統存在XSS漏洞，"httpOnly"標記可以減輕漏洞所造成的損害。

原因

如何發生

Web應用程式框架預設不會針對 Session cookie 與其他敏感性 cookie 去設定 "httpOnly" 標記。如上所述，若應用程式沒有使用 "httpOnly" cookie標記，則會允許客戶端腳本存取cookie。

一般建議

如何避免

- 對 server 端所有敏感的 cookie 設置 "httpOnly" 標記。
- 強烈建議實施 HTTP 強制安全傳輸技術（Strict Transport Security，HSTS），以確保cookie 可以透過安全通道發送。
- 在應用程式的 cookie 設定中明確的設置 "httpOnly" headers。
- 特別是，對於添加到response中的任何 cookie，將其`HttpCookie.HttpOnly`屬性設置為 `true` 。這包含透過 `Response.Cookies` collection屬性，被隱式添加到 response中的任何cookie。
- 更好的方法是將web 應用程式框架設定為自動幫全部的cookies設置 httpOnly 。方法是在應用程式的web.config檔中，於`<system.web>`元素下，將 `<httpCookies>` 元素的 `httpOnlyCookies`屬性設為 `"true"`。
- 如果直接將 cookie 寫入response headers，例如透過使用 `"Set-Cookie"` header name的 `Response.AppendHeader()`方法，建議在cookie 的尾端附加`";httpOnly;"`。

程式碼範例

CSharp

Creating Cookie in Code

```
private void setCookiesToResponse(string deptId)
{
      HttpCookie appCookie = new HttpCookie("department", deptId);
      Response.Cookies.add(appCookie);
}
```

Explicitly setting the HttpOnly flag on Cookies

```
private void setCookiesToResponse(string deptId)
{
      HttpCookie appCookie = new HttpCookie("department", deptId);
      appCookie.Expires = DateTime.Now.AddMinutes(30.0);
      appCookie.Secure = true;

      appCookie.HttpOnly = true;

      Response.Cookies.add(appCookie);
}
```

Heuristic DB Parameter Tampering

風險

可能發生什麼問題

惡意使用者可以單純透過修改傳送到伺服器的參照參數存取其他使用者的個人資訊，因此惡意使用者能夠繞過存取控制存取未授權的資料，像是其他使用者的帳號竊取機密或受限制的資訊。

原因

如何發生

應用程式在沒有過濾使用者 ID 的情況下存取使用者資訊，例如，僅透過傳送帳號 ID 就可以提供相關資訊，應用程式使用使用者輸入變數來從資料庫資料表中過濾特定筆數的資料，這些資料可能包含敏感性個人資訊 (如使用者帳號或是支付詳細資訊)，因為應用程式沒有根據任何使用者辨識元過濾資料，或是用預先產生的列表來限制可接受的變數值，惡意使用者可以輕易的修改被傳送的參照辨識元而取得未授權的資料。

一般建議

如何避免

一般建議：

- 在存取任何敏感性資料前強制進行授權驗證，包括特定的物件參考。
- 明確的阻擋任何未授權資料的存取，特別是其他使用者的資料。
- 如果可以，避免讓使用者可以單純透過傳送資料 ID 取得任意資料，例如與其讓使用者傳送帳號 ID 不如讓應用程式直接檢查目前授權使用者 session 的帳號 ID 。

針對性防禦措施：

- 根據使用者辨識元像是顧客編號等來過濾資料庫查詢。
- 對應使用者輸入變數到非直接參照變數中，例如透過預先準備好的可接受變數值名單來比對。

程式碼範例

Java

Unfiltered Direct Object Reference

```
public ResultSet getAccountInfo(request req){
    int accountId = Integer.parseInt(req.getParameter("accountId"));
    PreparedStatement stmt = connection.PreparedStatement("SELECT * from Accounts where
AccountId = ?");

    stmt.setInt(1, accountId);
    ResultSet accountRS = stmt.executeQuery();

    return accountRS;
}
```

Record References are Now Filtered and Indirect

```
    public ResultSet getAccountInfo(request req){
    int accountIndex = Integer.parseInt(req.getParameter("accountId"));
    int realAccountId = userAccountList.get(accountIndex);
    int userId = req.getSession().getAttribute("userId");
```

```
    PreparedStatement stmt = connection.PreparedStatement("SELECT * from Accounts where
AccountId = ? AND UserId = ?");

    stmt.setInt(1, realAccountId);
    stmt.setInt(2, userId);
    ResultSet accountRS = stmt.executeQuery();

    return accountRS;
}
```

Heuristic CSRF

風險

可能發生什麼問題

攻擊者可以代替受害者去執行所有被授權的行為，例如從受害者的帳戶將資金轉帳給攻擊者。但Log會記錄這個行為是由受害者執行。

原因

如何發生

應用程式執行一些修改資料庫內容的操作時，完全根據HTTP請求的內容，沒有重新進行用戶身份驗證（例如transaction驗證或加密形式的token），而是依賴瀏覽器或session驗證。 這代表攻擊者可以使用社交工程來讓受害者點擊包含交易請求的連結，應用程式會信任受害者的瀏覽器並執行操作。此種類型的攻擊稱為跨站請求偽造（XSRF或CSRF）。

一般建議

如何避免

使用標準或anti-CSRF library機制：最好是平台內建提供的機制或OWASP的CSRFGuard。 選擇性重新認證或交易認證（例如使用加密形式的token）也是可接受的。

程式碼範例

CSharp

The HttpRequest content is validated using AntiXsrfTokenKey

```
public class CSRFFixed
{
        public void foo(SqliteConnection connection, AntiXsrf AntiXsrfTokenKey, HttpRequest
Request)
        {
                string input = AntiXsrfTokenKey.Validate(Request.QueryString["user"]);
                string sql = "insert into Comments(comment) values (@user)";
                MySqlCommand cmd = new MySqlCommand(sql, connection);
            cmd.Parameters.AddwithValue(@user, input);
            connection.Open();
            SqlDataReader reader = cmd.ExecuteReader();
        }

}
```

HttpRequest content is used in a database query without any validation of that content

```
public class CSRF
{
        public void foo(SqliteConnection connection, HttpRequest Request)
```

```
        {
                string input = Request.QueryString["user"];
                string sql = "insert into Comments(comment) values (@user)";
                MySqlCommand cmd = new MySqlCommand(sql, connection);
            cmd.Parameters.AddwithValue(@user, input);
            connection.Open();
            SqlDataReader reader = cmd.ExecuteReader();
        }
}
```

Client Side Only Validation

風險

可能發生什麼問題

略過客戶端的驗證可能導致伺服器資料無法預期與竄改。

原因

如何發生

僅使用用戶端驗證 。

一般建議

如何避免

強烈建議在伺服器端驗證輸入，同時驗證用戶端。

程式碼範例

Client DOM Open Redirect

風險

可能發生什麼問題

攻擊者可能利用社交工程攻擊讓使用者點擊應用程式的連結，使用者將立即的被重新導向至任意的網站。 使用者可能認為他們仍然在原來的網站。第二個網站可能是具攻擊性的，包含惡意軟體，或者最常用於網絡釣魚。

原因

如何發生

應用程式重新導向使用者請求中提供的URL，且沒有警告使用者正重新導向至其他網站。 攻擊者可能利用社交工程攻擊讓受害者點擊連結到定義其他網站的應用程式將重新導向至使用者的瀏覽器參數，而使用者可能不知情的被重新導向。

一般建議

如何避免

1. 理想情況下，不允許重新導向至任意的URL。而應建立一個服務器端的對應從使用者提供的參數值，以合法的URL。 2. 如果有必要允許任意的URLs： ●對於應用程式內的網址，應先過濾和編碼使用者提供的參數，然後使用它作為一個相對URL通過與應用程式的網站域名前綴。 ● 對於應用程式(如果需要的話)之外的URL，使用中間免責聲明頁面，為使用者提供離開您的網站的明確警告。

程式碼範例

CSharp
Avoid redirecting to arbitrary URLs, instead map the parameter to a list of static URLs.

```
Response.Redirect(getUrlById(targetUrlId));
```

Java
Avoid redirecting to arbitrary URLs, instead map the parameter to a list of static URLs.

```
Response.Redirect(getUrlById(targetUrlId));
```

Apex

Open Redirection

```
String redirsite = ApexPages.currentPage().getParameters().get('redirlocation');
PageReference pageRef;
if(redirsite != null)
{
    pageRef = new PageReference(redirsite);
    pageRef.setRedirect(true);
    return pageRef;
}
pageRef = ApexPages.currentPage();
return pageRef;
```

Mitigating Open Redirection with Domain Name Prefix

```
String redirsite = ApexPages.currentPage().getParameters().get('redirlocation');
PageReference pageRef;
if(redirsite != null)
{
    pageRef = new PageReference('http://domain.com/page.jsp?' + redirsite);
    pageRef.setRedirect(true);
    return pageRef;
}
pageRef = ApexPages.currentPage();
return pageRef;
```

Improper Exception Handling

風險

可能發生什麼問題

●攻擊者可能會導致應用程式異常的崩潰，且造成拒絕服務(DoS)攻擊。 ●應用程式可能發生偶發性的崩潰。

原因

如何發生

'應用程式執行如資料庫或文件存取，這可能會引發一些異常狀況。若應用程式未妥善處理異常狀況，可能會當機。

一般建議

如何避免

可能導致異常的任何方法應包裝在一個try-catch區塊: ● 明確地處理預期的異常 ●包含一個預設的解決方案，以處理突發異常

程式碼範例

CSharp
Always catch exceptions explicitly.

```
try
{
// Database access or other potentially dangerous function
}
catch (SqlException ex)
{
// Handle exception
}
catch (Exception ex)
{
// Default handler for unexpected exceptions
}
```

Java
Always catch exceptions explicitly.

```
try
{
// Database access or other potentially dangerous function
}
catch (SQLException ex)
{
// Handle exception
}
catch (Exception ex)
{
// Default handler for unexpected exceptions
}
```

Information Exposure Through an Error Message

風險

可能發生什麼問題

透露關於程式的環境、使用者或相關資訊 (例如：stack trace) 將會讓攻擊者找到其他的缺失也幫助攻擊者來發起攻擊。這也可能會使機密資料洩露，例如：密碼或資料庫欄位。

原因

如何發生

應用程式以不安全的方式處理例外(exception)，包括直接在 error message 中顯示完整的原始詳細訊息。這在幾個狀況下都可能會發生：不處理 exception；直接將 exception 輸出到頁面或檔案中；顯式return exception 物件；配置檔設定不嚴謹。這些 exception 細節可能包含機密資訊，並隨著 Runtime Error 而流出。

一般建議

如何避免

- 不將 excecption 資訊直接輸出或是透露給使用者，建議回傳一個制式化的錯誤訊息。Exception細節則應記錄於Log機制內。
- 任何會拋出 exception 的函式都應該要被包在處理 exception 的區塊內，而處理的方式有:
    - 明確的處理預期內的exception。
    - 包含了一個預設的解決方式來處理無預期的exceptions。
- 設定一個全域處理器來避免無處理的錯誤被送至使用者端。

程式碼範例

Client JQuery Deprecated Symbols

風險

可能發生什麼問題

參照使用已經棄用的模組會導致應用程式暴露在已知的漏洞底下，漏洞已經被公開回報而且已經被修復，普通的攻擊方式是掃描應用程式尋找這些已知漏洞，接著透過這些棄用版本的模組來濫用應用程式。即使棄用的程式碼以完全安全的方式使用，它存在函式庫中可能會影響開發人員將來重新利用被棄用的元件，導致應用程式暴露在風險中。這就是為什麼需要從函式庫中去除不推薦的程式碼。
請注意，真正的風險要看舊版本中的所有已知漏洞詳情來判斷。

原因

如何發生

應用程式參照已經被宣告為棄用的程式元素，元素包含函數、方法、屬性、模組或是過時的函式庫版本，有可能程式在開發後這些程式才被宣告成過時。

一般建議

如何避免

- 永遠使用最新版本的函式庫或套件以及其他相依程式。
- 不要用任何被宣告為棄用的方法、函數、屬性或其他元素。

程式碼範例

Java

Using Deprecated Methods for Security Checks

```
private void checkPermissions(InetAddress address) {

    SecurityManager secManager = System.getSecurityManager();

    if (secManager != null) {
        secManager.checkMulticast(address, 0)
    }

}
```

A Replacement Security Check

```
private void checkPermissions(InetAddress address) {

    SecurityManager secManager = System.getSecurityManager();

    if (secManager != null) {
        SocketPermission permission = new SocketPermission(address.getHostAddress(),
"accept,connect");

        secManager.checkPermission(permission)
    }
```

```
}
```

Information Exposure

Weakness ID: 200 *(Weakness Class)* **Status:** Incomplete

Description

Description Summary

An information exposure is the intentional or unintentional disclosure of information to an actor that is not explicitly authorized to have access to that information.

Extended Description

The information either

(1) is regarded as sensitive within the product's own functionality, such as a private message; or

(2) provides information about the product or its environment that could be useful in an attack but is normally not available to the attacker, such as the installation path of a product that is remotely accessible.

Many information exposures are resultant (e.g. path disclosure in PHP script error), but they can also be primary (e.g. timing discrepancies in crypto). There are many different types of problems that involve information exposures. Their severity can range widely depending on the type of information that is revealed.

Alternate Terms

Information Disclosure:	This term is frequently used in vulnerability databases and other sources, however "disclosure" does not always have security implications. The phrase "information disclosure" is also used frequently in policies and legal documents, but do not refer to disclosure of security-relevant information.
Information Leak:	This is a frequently used term, however the "leak" term has multiple uses within security. In some cases it deals with exposure of information, but in other cases (such as "memory leak") this deals with improper tracking of resources which can lead to exhaustion. As a result, CWE is actively avoiding usage of the "leak" term.

Time of Introduction

- Architecture and Design
- Implementation

Applicable Platforms

Languages

All

Likelihood of Exploit

High

Potential Mitigations

Compartmentalize your system to have "safe" areas where trust boundaries can be unambiguously drawn. Do not allow sensitive data to go outside of the trust boundary and always be careful when interfacing with a compartment outside of the safe area.

Weakness Ordinalities

Ordinality	Description
Resultant	*(where the weakness is typically related to the presence of some other weaknesses)*

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to

ChildOf	Category	199	Information Management Errors	**Development Concepts (primary)699**
ChildOf	Weakness Class	668	Exposure of Resource to Wrong Sphere	**Research Concepts (primary)1000**
ChildOf	Category	717	OWASP Top Ten 2007 Category A6 - Information Leakage and Improper Error Handling	**Weaknesses in OWASP Top Ten (2007) (primary)629**
ParentOf	Weakness Variant	201	Information Leak Through Sent Data	**Development Concepts (primary)699 Research Concepts (primary)1000**
ParentOf	Weakness Variant	202	Privacy Leak through Data Queries	**Development Concepts (primary)699**
ParentOf	Weakness Class	203	Information Exposure Through Discrepancy	**Development Concepts (primary)699 Research Concepts (primary)1000**
ParentOf	Weakness Base	209	Information Exposure Through an Error Message	**Development Concepts (primary)699 Research Concepts (primary)1000**
ParentOf	Weakness Base	212	Improper Cross-boundary Removal of Sensitive Data	**Development Concepts (primary)699 Research Concepts (primary)1000**
ParentOf	Weakness Base	213	Intended Information Leak	**Development Concepts (primary)699 Research Concepts (primary)1000**
ParentOf	Weakness Variant	214	Process Environment Information Leak	**Development Concepts (primary)699 Research Concepts (primary)1000**
ParentOf	Weakness Variant	215	Information Leak Through Debug Information	**Development Concepts (primary)699 Research Concepts (primary)1000**
ParentOf	Weakness Base	226	Sensitive Information Uncleared Before Release	**Development Concepts (primary)699** Research Concepts1000
ParentOf	Weakness Class	359	Privacy Violation	**Research Concepts (primary)1000**
ParentOf	Weakness Variant	497	Exposure of System Data to an Unauthorized Control Sphere	**Development Concepts (primary)699 Research Concepts (primary)1000**
ParentOf	Weakness Variant	524	Information Leak Through Caching	**Development Concepts (primary)699 Research Concepts (primary)1000**
ParentOf	Weakness Variant	526	Information Leak Through Environmental Variables	**Development Concepts (primary)699 Research Concepts (primary)1000**
ParentOf	Weakness Base	538	File and Directory Information Exposure	**Development Concepts (primary)699 Research Concepts (primary)1000**
ParentOf	Weakness Variant	598	Information Leak Through Query Strings in GET Request	**Development Concepts (primary)699**

				Research Concepts (primary)1000
ParentOf	Weakness Variant	612	Information Leak Through Indexing of Private Data	**Development Concepts (primary)699 Research Concepts (primary)1000**
MemberOf	View	635	Weaknesses Used by NVD	**Weaknesses Used by NVD (primary)635**
CanFollow	Weakness Variant	498	Information Leak through Class Cloning	Development Concepts699 Research Concepts1000
CanFollow	Weakness Variant	499	Serializable Class Containing Sensitive Data	Development Concepts699 Research Concepts1000

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
PLOVER			Information Leak (information disclosure)
OWASP Top Ten 2007	A6	CWE More Specific	Information Leakage and Improper Error Handling
WASC	13		Information Leakage

Related Attack Patterns

CAPEC-ID	Attack Pattern Name	*(CAPEC Version: 1.5)*
13	Subverting Environment Variable Values	
22	Exploiting Trust in Client (aka Make the Client Invisible)	
59	Session Credential Falsification through Prediction	
60	Reusing Session IDs (aka Session Replay)	
79	Using Slashes in Alternate Encoding	
281	Analytic Attacks	

Content History

Submissions

Submission Date	Submitter	Organization	Source
	PLOVER		Externally Mined

Modifications

Modification Date	Modifier	Organization	Source
2008-07-01	Eric Dalci	Cigital	External
	updated Time of Introduction		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Likelihood of Exploit, Relationships, Taxonomy Mappings, Weakness Ordinalities		
2008-10-14	CWE Content Team	MITRE	Internal
	updated Description		
2009-12-28	CWE Content Team	MITRE	Internal
	updated Alternate Terms, Description, Name		
2010-02-16	CWE Content Team	MITRE	Internal
	updated Taxonomy Mappings		
2010-04-05	CWE Content Team	MITRE	Internal
	updated Related Attack Patterns		

Previous Entry Names

Change Date	Previous Entry Name
2009-12-28	Information Leak (Information Disclosure)

BACK TO TOP

Client Potential DOM Open Redirect

風險

可能發生什麼問題

攻擊者可能利用社交工程攻擊讓使用者點擊應用程式的連結，使用者將立即的被重新導向至任意的網站。 使用者可能認為他們仍然在原來的網站。第二個網站可能是具攻擊性的，包含惡意軟體，或者最常用於網絡釣魚。

原因

如何發生

應用程式重新導向使用者請求中提供的URL，且沒有警告使用者正重新導向至其他網站。 攻擊者可能利用社交工程攻擊讓受害者點擊連結到定義其他網站的應用程式將重新導向至使用者的瀏覽器參數，而使用者可能不知情的被重新導向。

一般建議

如何避免

1. 理想情況下，不允許重新導向至任意的URL。而應建立一個服務器端的對應從使用者提供的參數值，以合法的URL。 2. 如果有必要允許任意的URLs： ●對於應用程式內的網址，應先過濾和編碼使用者提供的參數，然後使用它作為一個相對URL通過與應用程式的網站域名前綴。 ● 對於應用程式(如果需要的話)之外的URL，使用中間免責聲明頁面，為使用者提供離開您的網站的明確警告。

程式碼範例

Uncontrolled Resource Consumption ('Resource Exhaustion')

Weakness ID: 400 *(Weakness Base)* **Status:** Incomplete

Description

Description Summary

The software does not properly restrict the size or amount of resources that are requested or influenced by an actor, which can be used to consume more resources than intended.

Extended Description

Limited resources include memory, file system storage, database connection pool entries, or CPU. If an attacker can trigger the allocation of these limited resources, but the number or size of the resources is not controlled, then the attacker could cause a denial of service that consumes all available resources. This would prevent valid users from accessing the software, and it could potentially have an impact on the surrounding environment. For example, a memory exhaustion attack against an application could slow down the application as well as its host operating system.

Resource exhaustion problems have at least two common causes:

- (1) Error conditions and other exceptional circumstances
- (2) Confusion over which part of the program is responsible for releasing the resource

Time of Introduction

- Operation
- Architecture and Design
- Implementation

Applicable Platforms

Languages

All

Common Consequences

Scope	Effect
Availability	The most common result of resource exhaustion is denial of service. The software may slow down, crash due to unhandled errors, or lock out legitimate users.
Integrity	In some cases it may be possible to force the software to "fail open" in the event of resource exhaustion. The state of the software -- and possibly the security functionality - may then be compromised.

Likelihood of Exploit

Medium to High

Detection Methods

Automated Static Analysis

Automated static analysis typically has limited utility in recognizing resource exhaustion problems, except for program-independent system resources such as files, sockets, and processes. For system resources, automated static analysis may be able to detect circumstances in which resources are not released after they have expired. Automated analysis of configuration files may be able to detect settings that do not specify a maximum value.

Automated static analysis tools will not be appropriate for detecting exhaustion of custom resources, such as an intended security policy in which a bulletin board user is only allowed to make a limited number of posts per day.

Effectiveness: Limited

Automated Dynamic Analysis

Certain automated dynamic analysis techniques may be effective in spotting resource exhaustion problems, especially with

resources such as processes, memory, and connections. The technique may involve generating a large number of requests to the software within a short time frame.

Effectiveness: Moderate

Fuzzing

While fuzzing is typically geared toward finding low-level implementation bugs, it can inadvertently find resource exhaustion problems. This can occur when the fuzzer generates a large number of test cases but does not restart the targeted software in between test cases. If an individual test case produces a crash, but it does not do so reliably, then an inability to handle resource exhaustion may be the cause.

Effectiveness: Opportunistic

Demonstrative Examples

Example 1

(Bad Code)

Example Language: **Java**

```
class Worker implements Executor {
...
public void execute(Runnable r) {

try {
...
}
catch (InterruptedException ie) {

// postpone response
Thread.currentThread().interrupt();
}
}
public Worker(Channel ch, int nworkers) {
...
}
protected void activate() {

Runnable loop = new Runnable() {

public void run() {

try {
for (;;) {

Runnable r = ... r.run();
}
}
catch (InterruptedException ie) {
...
}
}
};
new Thread(loop).start();
}
```

There are no limits to runnables. Potentially an attacker could cause resource problems very quickly.

Example 2

This code allocates a socket and forks each time it receives a new connection.

(Bad Code)

Example Languages: **C and C++**

```
sock=socket(AF_INET, SOCK_STREAM, 0);
while (1) {

newsock=accept(sock, ...);
printf("A connection has been accepted\n");
pid = fork();
```

```
}
```

The program does not track how many connections have been made, and it does not limit the number of connections. Because forking is a relatively expensive operation, an attacker would be able to cause the system to run out of CPU, processes, or memory by making a large number of connections.

Observed Examples

Reference	Description
CVE-2009-2874	Product allows attackers to cause a crash via a large number of connections.
CVE-2009-1928	Malformed request triggers uncontrolled recursion, leading to stack exhaustion.
CVE-2009-2858	Chain: memory leak (CWE-404) leads to resource exhaustion.
CVE-2009-2726	Driver does not use a maximum width when invoking sscanf style functions, causing stack consumption.
CVE-2009-2540	Large integer value for a length property in an object causes a large amount of memory allocation.
CVE-2009-2299	Web application firewall consumes excessive memory when an HTTP request contains a large Content-Length value but no POST data.
CVE-2009-2054	Product allows exhaustion of file descriptors when processing a large number of TCP packets.
CVE-2008-5180	Communication product allows memory consumption with a large number of SIP requests, which cause many sessions to be created.
CVE-2008-2121	TCP implementation allows attackers to consume CPU and prevent new connections using a TCP SYN flood attack.
CVE-2008-2122	Port scan triggers CPU consumption with processes that attempt to read data from closed sockets.
CVE-2008-1700	Product allows attackers to cause a denial of service via a large number of directives, each of which opens a separate window.
CVE-2007-4103	Product allows resource exhaustion via a large number of calls that do not complete a 3-way handshake.
CVE-2006-1173	Mail server does not properly handle deeply nested multipart MIME messages, leading to stack exhaustion.
CVE-2007-0897	Chain: anti-virus product encounters a malformed file but returns from a function without closing a file descriptor (CWE-775) leading to file descriptor consumption (CWE-400) and failed scans.

Potential Mitigations

Phase: Architecture and Design

Design throttling mechanisms into the system architecture. The best protection is to limit the amount of resources that an unauthorized user can cause to be expended. A strong authentication and access control model will help prevent such attacks from occurring in the first place. The login application should be protected against DoS attacks as much as possible. Limiting the database access, perhaps by caching result sets, can help minimize the resources expended. To further limit the potential for a DoS attack, consider tracking the rate of requests received from users and blocking requests that exceed a defined rate threshold.

Phase: Architecture and Design

Mitigation of resource exhaustion attacks requires that the target system either:

- recognizes the attack and denies that user further access for a given amount of time, or
- uniformly throttles all requests in order to make it more difficult to consume resources more quickly than they can again be freed.

The first of these solutions is an issue in itself though, since it may allow attackers to prevent the use of the system by a particular valid user. If the attacker impersonates the valid user, he may be able to prevent the user from accessing the server in question.

The second solution is simply difficult to effectively institute -- and even when properly done, it does not provide a full solution. It simply makes the attack require more resources on the part of the attacker.

Phase: Architecture and Design

Ensure that protocols have specific limits of scale placed on them.

Phase: Implementation

Ensure that all failures in resource allocation place the system into a safe posture.

Other Notes

Database queries that take a long time to process are good DoS targets. An attacker would have to write a few lines of Perl code to generate enough traffic to exceed the site's ability to keep up. This would effectively prevent authorized users from using the site at all. Resources can be exploited simply by ensuring that the target machine must do much more work and consume more resources in order to service a request than the attacker must do to initiate a request.

A prime example of this can be found in old switches that were vulnerable to "macof" attacks (so named for a tool developed by Dugsong). These attacks flooded a switch with random IP and MAC address combinations, therefore exhausting the switch's cache, which held the information of which port corresponded to which MAC addresses. Once this cache was exhausted, the switch would fail in an insecure way and would begin to act simply as a hub, broadcasting all traffic on all ports and allowing for basic sniffing attacks.

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Category	399	Resource Management Errors	**Development Concepts (primary)699**
ChildOf	Weakness Class	664	Improper Control of a Resource Through its Lifetime	**Research Concepts (primary)1000**
ChildOf	Category	730	OWASP Top Ten 2004 Category A9 - Denial of Service	**Weaknesses in OWASP Top Ten (2004) (primary)711**
ParentOf	Category	769	File Descriptor Exhaustion	**Development Concepts (primary)699**
ParentOf	Weakness Base	770	Allocation of Resources Without Limits or Throttling	**Development Concepts (primary)699** Research Concepts1000
ParentOf	Weakness Base	771	Missing Reference to Active Allocated Resource	**Research Concepts (primary)1000**
ParentOf	Weakness Base	772	Missing Release of Resource after Effective Lifetime	Research Concepts1000
ParentOf	Weakness Base	779	Logging of Excessive Data	**Development Concepts (primary)699 Research Concepts (primary)1000**
CanFollow	Weakness Base	410	Insufficient Resource Pool	Development Concepts699 Research Concepts1000

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
CLASP			Resource exhaustion (file descriptor, disk space, sockets, ...)
OWASP Top Ten 2004	A9	CWE More Specific	Denial of Service
WASC	10		Denial of Service
WASC	41		XML Attribute Blowup

Related Attack Patterns

CAPEC-ID	Attack Pattern Name	*(CAPEC Version: 1.5)*
2	Inducing Account Lockout	
82	Violating Implicit Assumptions Regarding XML Content (aka XML Denial of Service (XDoS))	

147	XML Ping of Death
228	Resource Depletion through DTD Injection in a SOAP Message

References

Joao Antunes, Nuno Ferreira Neves and Paulo Verissimo. "Detection and Prediction of Resource-Exhaustion Vulnerabilities". Proceedings of the IEEE International Symposium on Software Reliability Engineering (ISSRE). November 2008. <http://homepages.di.fc.ul.pt/~nuno/PAPERS/ISSRE08.pdf>.

D.J. Bernstein. "Resource exhaustion". <http://cr.yp.to/docs/resources.html>.

Pascal Meunier. "Resource exhaustion". Secure Programming Educational Material. 2004. <http://homes.cerias.purdue.edu/~pmeunier/secprog/sanitized/class1/6.resource%20exhaustion.ppt>.

[REF-11] M. Howard and D. LeBlanc. "Writing Secure Code". Chapter 17, "Protecting Against Denial of Service Attacks" Page 517. 2nd Edition. Microsoft. 2002.

Content History

Submissions

Submission Date	Submitter	Organization	Source
	CLASP		Externally Mined

Modifications

Modification Date	Modifier	Organization	Source
2008-07-01	Eric Dalci	Cigital	External
	updated Time of Introduction		
2008-08-15		Veracode	External
	Suggested OWASP Top Ten 2004 mapping		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Common Consequences, Relationships, Other Notes, Taxonomy Mappings		
2008-10-14	CWE Content Team	MITRE	Internal
	updated Description, Name, Relationships		
2009-01-12	CWE Content Team	MITRE	Internal
	updated Description		
2009-05-27	CWE Content Team	MITRE	Internal
	updated Name, Relationships		
2009-07-27	CWE Content Team	MITRE	Internal
	updated Description, Relationships		
2009-10-29	CWE Content Team	MITRE	Internal
	updated Relationships		
2009-12-28	CWE Content Team	MITRE	Internal
	updated Common Consequences, Demonstrative Examples, Detection Factors, Likelihood of Exploit, Observed Examples, Other Notes, Potential Mitigations, References		
2010-02-16	CWE Content Team	MITRE	Internal
	updated Detection Factors, Potential Mitigations, References, Taxonomy Mappings		
2010-04-05	CWE Content Team	MITRE	Internal
	updated Related Attack Patterns		

Previous Entry Names

Change Date	Previous Entry Name
2008-10-14	Resource Exhaustion
2009-05-27	Uncontrolled Resource Consumption (aka 'Resource Exhaustion')

BACK TO TOP

Missing Content Security Policy

風險

可能發生什麼問題

CSP標頭強制要求當前的網頁該信任和允許那些內容(content)來源，例如腳本的來源、嵌入的 (子) 框架，嵌入的(父) 框架或圖像。如果在網頁中，某個content來源不符合CSP，那麼瀏覽器會立即拒絕它。未能正確的定義CSP可能會使應用程式的使用者面臨跨網站腳本（XSS）攻擊、點擊劫持、內容偽造等攻擊。

原因

如何發生

現代瀏覽器使用CSP標頭作為可信任的content來源指標(包括媒體、影像、腳本、框架等)。如果沒有明確定義這些策略，瀏覽器預設將允許不受信任的content，可能會使HTML頁面容易受到攻擊。

一般建議

如何避免

建議根據業務需求和外部檔案託管服務的部署，明確地設定合適的CSP標頭(框架、腳本、表單、腳本、媒體、圖片等...)，具體來說，不要使用萬用字元"*"來指定這些策略，因為這將允許來自外部的任何資源內容。
CSP可以在網頁應用程序代碼中明確定義，作為由web-server 配置所管理的標頭，或在HTML<head>下的 <meta>標籤中定義。

程式碼範例

PHP

Restricting Content-Security-Policy to Only Obtain Embedded Content from Current Web-Application

```
<?php
      header("Content-Security-Policy: default-src 'none'; script-src 'self'; connect-src
'self'; img-src 'self'; style-src 'self';");
?>
```

Heuristic SQL Injection

風險

可能發生什麼問題

攻擊者可以直接存取系統內的所有資料、竊取敏感資訊，包含私人使用者資訊，信用卡明細，商業機密，以及其他秘密資料。同樣的，攻擊者可能會修改或刪除已經存在的資料，或者甚至加入假資料。在有些情境下，甚至可能可以在資料庫上執行程式碼。

除了直接揭露及更改機密資訊以外，此漏洞還可以用來做其他應用，像是繞過身分驗證，越過安全驗證機制，或者偽造資料追蹤紀錄。

攻擊者輕易的找到程式的缺陷，進一步增加被漏洞利用的可能性。

請注意，由於注入行為發生在外部元件(external component)中，該元件可能有在內部進行過濾或消毒，視情況可以排除此風險。

原因

如何發生

應用程式利用SQL查詢語句來與資料庫引擎溝通，利用資料庫來儲存及管理資料。應用程式建構SQL查詢語句時，只是簡單的把字串相加，並插入不受信任的資料。而且資料與語句之間沒有分隔；此外，插入資料時，既沒檢查資料格式的正確性也沒有對其進行過濾。因此，不受信任的資料可能會包含在SQL查詢語句中，或者修改預期的SQL查詢語句。對資料庫而言，這是來自應用程式的SQL查詢，因此會如實的執行這些已被竄改過的查詢。

注意應用程式可能透過外部元件或API存取資料庫。因此，攻擊者一樣可以藉由改變使用者的輸入將惡意資料傳給API或其他元件(component)，隨著程式流，惡意語句最終會進入資料庫伺服器中。

一般建議

如何避免

- 無論來源為何，都要驗證所有不受信任的資料。驗證方式應採用白名單：僅接受符合指定結構的資料。不要建議採用黑名單的方式：僅拒絕非法字元。
- 尤其是要確認：
    - 資料格式(Data type)
    - 資料大小(Size)
    - 資料間距(Range)
    - 資料格式(Format)
    - 資料預期的值(Expected values)
- 根據最小權限原則，限制存取資料庫的物件及功能。
- 不要使用動態的字串串接來建立SQL查詢語句
- 最好使用DB Stored Procedures(預存程序)來存取資料，而非動態的組合查詢語句
- 建議使用安全的資料庫元件，像是參數化查詢及物件綁定(object bindings，例如：指令(commands)及參數(parameters))
- 或者，更好的解法是使用ORM的library，可以把應用程式上允許執行的指令預先定義及封裝起來，代替直接動態存取資料庫。如果使用這個方法就可以把指令部分與資料部分彼此隔離。
- 建議用標準資料存取的libraries，跟平台的API，而非使用不透明的第三方程式(drivers)。
- 建議使用資料庫特定的 `DbCommand` 子類別中的 `DbParameter` 物件與API。將指令的 `CommandType` 屬性設為 `CommandType.StoredProcedure`，接著添加參數到 `.Parameters` collection屬性中，而非使用字串串接。
- 考慮使用ORM的package，像是Entity Framework、LINQ-To-SQL、nHibernate等等。

程式碼範例

CSharp

Create Pseudo-Query Using String Concatenation

```
private int GetUserId_Unsafe(HttpRequest request)
{
	int userId = 0;

	string userName = request.Form["UserName"];
	string dataOperation = "SELECT [UserID] FROM [AppUsers] WHERE [UserName] = '" + userName
+ "' " ;

	using (CustomDbDriver db = new CustomDbDriver())
	{
		userId = db.runQuery(dataOperation);
	}

	return userId;
}
```

Use Standard DbCommand API to Call Regular Stored Procedures

```
private int GetUserId_SafeStandardAPI(HttpRequest request)
{
	int userId = 0;

	string userName = request.Form["UserName"];
	string spName = "GetUserId";

	using (SqlConnection conn = GetConnection())
	{
		using (SqlCommand command = new SqlCommand(spName, conn))
		{
			command.CommandType = CommandType.StoredProcedure;
			command.Parameters.AddWithValue("@userName", userName);

			using (SqlDataReader reader = command.ExecuteReader())
			{
				reader.Read();
				userId = reader.GetInt32(0);
			}
		}
	}

	return userId;
}
```

Wrap DB access with a stanard ORM such as EntityFramework

```
private int GetUserId_SafeEF(HttpRequest request)
{
	int userId = 0;

	string userName = request.Form["UserName"];

	using (var context = new DbContext(CONN_STRING))
	{
		var userSet = context.Set<AppUser>();
		var oneUser = userSet.Find(userName);

		if (oneUser != null)
			userId = oneUser.UserId;
```

```
        }

        return userId;
    }
```

Unencrypted Web Config File

風險

可能發生什麼問題

在.NET應用程式中，Web.config檔案基本上都含有敏感資訊，包含連線資訊、服務登入帳號密碼；所有潛在的敏感資訊都應該被加密，以避免惡意使用者訪問檔案檢索之。

原因

如何發生

網頁應用程式所參考的Web.config檔案，通常會儲存於該網站所架設Server的硬碟檔案系統上，由於檔案系統的特性，惡意使用者只要知道Web.config的所屬目錄和檔名即可訪問，若沒有對Web.config中的敏感資料進行加密，則這些明文資料會被竊取。

一般建議

如何避免

確保在Web.config中的敏感資料都經過加密。最佳實作是：利用.NET aspnet_regiis.exe's tool來進行敏感資料的加密，這些加密過的敏感資料，當Web-Server取用時工具會先進行解密，而透過目錄訪問時會顯示加密過的資料，以達到保護效果。

程式碼範例

ASP

web.config File with Plain-Text Sensitive Content

```
<configuration>

  <connectionStrings>
      <add name="ServiceName" connectionString="[connection strings]" />
   </connectionStrings>
   <system.web>
     <machineKey validationKey="[validation key]" decryptionKey="[decryption key]" />
   </system.web>
</configuration>
```

web.config File with Encrypted Sensitive Content

```
<configuration>

  <connectionStrings configProtectionProvider="RsaProtectedConfigurationProvider">
      <EncryptedData Type="http://www.w3.org/2001/04/xmlenc#Element"
         xmlns="http://www.w3.org/2001/04/xmlenc#">
         <EncryptionMethod Algorithm="[Encryption Algorithm]" />
         <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
            <EncryptedKey xmlns="http://www.w3.org/2001/04/xmlenc#">
               <EncryptionMethod Algorithm="[Encryption Algorithm]" />
               <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
                  <KeyName>RSA Key
                  </KeyName>
               </KeyInfo>
               <CipherData>[Cipher Value]</CipherValue>
               </CipherData>
            </EncryptedKey>
         </KeyInfo>
         <CipherData>[Cipher Value]</CipherValue>
         </CipherData>
```

```
      </EncryptedData>
    </connectionStrings>
    <system.web>
      <machineKey configProtectionProvider="RsaProtectedConfigurationProvider">
        <EncryptedData Type="http://www.w3.org/2001/04/xmlenc#Element"
xmlns="http://www.w3.org/2001/04/xmlenc#">
          <EncryptionMethod Algorithm="[Encryption Algorithm]" />
          <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
            <EncryptedKey xmlns="http://www.w3.org/2001/04/xmlenc#">
              <EncryptionMethod Algorithm="[Encryption Algorithm]" />
              <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
                <KeyName>RSA Key
                </KeyName>
              </KeyInfo>
              <CipherData>
                <CipherValue>[Cipher Value]</CipherValue>
              </CipherData>
            </EncryptedKey>
          </KeyInfo>
          <CipherData>
            <CipherValue>[Cipher Value]</CipherValue>
          </CipherData>
        </EncryptedData>
      </machineKey>
    </system.web>
</configuration>
```

Use Of Hardcoded Password

風險

可能發生什麼問題

直接寫入的密碼會造成密碼的洩漏。如果攻擊者可以取得程式原始碼，他便可取得密碼，並利用它們來冒充合法使用者。攻擊者可以冒充自己是應用程式的末端使用者，或假裝應用程式登入遠端系統，例如資料庫或網路服務。 一旦攻擊者成功冒充使用者或應用程式，他便可取得完整的控制權，並做到任何能做的事。

原因

如何發生

應用程式程式庫含有嵌入在原始碼內的字串型態的密碼。這個直接寫入的值被直接使用或是用來和使用者輸入做驗證比對。或驗證末端程式連線到遠端系統（如資料庫或網路服務）。 攻擊者只需要取得原始碼即可揭露被直接寫入的密碼。同樣的，攻擊者也可以進行逆向工程反編譯應用程式的二進位程式碼，並簡單的取得寫入的密碼。一旦被發現，攻擊者可以很容易的使用這個密碼進行假冒攻擊，無論是對應用程式或遠端系統。 此外，一旦被偷取，將無法簡單的更改來預防更進一步的濫用，除非應用程式重新編譯過。此外，這個應用程式如果被分配到多個系統，從一個系統竊取的密碼可以自動允許在所有被部屬的系統上使用。

一般建議

如何避免

不要將機密資料直接寫入程式碼內。 特別是，用戶的密碼應該儲存在資料庫或是目錄服務，並使用夠強的雜湊演算法進行加密保護。(如 bcrypt, scrypt, PBKDF2, or Argon2)。不要用直接寫入的值進行比對。 系統密碼應該儲存在配置文件或資料庫，並以強大的加密方法保護（例如AES-256）。加密金鑰應該被安全的保護。

程式碼範例

Java

Hardcoded Admin Password

```
bool isAdmin(String username, String password) {
    bool isMatch = false;

    if (username.equals("admin")) {
        if (password.equals("P@ssw0rd"))
            return isMatch = true;
    }

    return isMatch;
}
```

No Hardcoded Credentials

```
bool isAdmin(String username, String password) {
    bool adminPrivs = false;

    if (authenticateUser(username, password)) {
        UserPrivileges privs = getUserPrivileges(username);

        if (privs.isAdmin)
            adminPrivs = true;
    }

    return adminPrivs;
```

```
}
```

DebugEnabled

風險

可能發生什麼問題

測試和除錯程式碼不應部署到正式環境中，可能創造非預期的進入點，從而增加應用程式的攻擊面。此外，此程式碼通常無正確測試或維護，可能留存其他已修復的歷史漏洞。通常，除錯程式碼將包含功能性的「後門」，使程式設計師能夠繞過操作安全機制，如身份驗證或存取控制。

原因

如何發生

在應用程式開發過程中，程式設計師通常會實作專門的程式碼，以方便除錯和測試。而程式設計師甚至會使除錯程式碼繞過安全機制，以便將測試集中在特定的功能上，並將其與安全架構隔離。
若此除錯或測試程式碼不會從程式中刪除，然後將其包含在軟體建置中並部署到正式環境中。

一般建議

如何避免

- 在部署或建置應用程式之前刪除所有除錯程式碼。確保未將配置設置定義為啟用除錯模式。
- 通過專用的測試框架實現所有測試碼，它可以將測試案例程式碼與應用程式的其餘部分隔離。
- 避免在應用程式程式碼本身中實作特殊的「測試程式碼」、「除錯時間(Debugging-time)」功能或「機密」介面或參數。
- 使用專用的 CI/CD 工具(可以自動設定已部署的應用程式)定義和實施標準和自動建置/部署過程，排除所有臨時程式碼，並只包含預期的應用程式程式碼。

程式碼範例

Java

Main in Servlet

```
public class AppServlet extends HttpServlet {
      protected void doGet(HttpServletRequest request, HttpServletResponse response)
                           throws ServletException, IOException {
            // handle request
    }

      private static String MODE = "";
      public static void main(String[] args) {
            // initialize app for debugging and testing
            MODE = "DEBUGGING";
    }
}
```

Ruby

Internal Test Method

```
class AppClass
      def run_app
       # Run the app
```

```
    end
      def test_app
       # Test and debug the app
    end
end
```

CSharp
Debug Configuration

```
<configuration>
      <system.web>
      <compilation debug="false" />
    </system.web>
</configuration>
```

Password in Configuration File

Weakness ID: 260 *(Weakness Variant)* **Status:** Incomplete

Description

Description Summary

The software stores a password in a configuration file that might be accessible to actors who do not know the password.

Extended Description

This can result in compromise of the system for which the password is used. An attacker could gain access to this file and learn the stored password or worse yet, change the password to one of their choosing.

Time of Introduction

- Architecture and Design
- Implementation

Applicable Platforms

Languages

All

Demonstrative Examples

Example 1

Below is a snippet from a Java properties file in which the LDAP server password is stored in plaintext.

(Bad Code)

Example Language: **Java**

```
webapp.ldap.username=secretUsername
webapp.ldap.password=secretPassword
```

Potential Mitigations

Avoid storing passwords in easily accessible locations.

Consider storing cryptographic hashes of passwords as an alternative to storing in plaintext.

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Category	254	Security Features	Development Concepts699 **Seven Pernicious Kingdoms (primary)700**
ChildOf	Weakness Base	522	Insufficiently Protected Credentials	**Development Concepts (primary)699 Research Concepts (primary)1000**
ChildOf	Category	632	Weaknesses that Affect Files or Directories	**Resource-specific Weaknesses (primary)631**
ParentOf	Weakness Variant	13	ASP.NET Misconfiguration: Password in Configuration File	**Research Concepts (primary)1000**
ParentOf	Weakness Variant	258	Empty Password in Configuration File	**Development Concepts (primary)699 Research Concepts (primary)1000**

Affected Resources

- File/Directory

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
7 Pernicious Kingdoms			Password Management: Password in Configuration File

References

J. Viega and G. McGraw. "Building Secure Software: How to Avoid Security Problems the Right Way". 2002.

Content History

Submissions			
Submission Date	**Submitter**	**Organization**	**Source**
	7 Pernicious Kingdoms		Externally Mined
Modifications			
Modification Date	**Modifier**	**Organization**	**Source**
2008-07-01	Sean Eidemiller	Cigital	External
	added/updated demonstrative examples		
2008-07-01	Eric Dalci	Cigital	External
	updated Time of Introduction		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Relationships, Taxonomy Mappings		
2008-10-14	CWE Content Team	MITRE	Internal
	updated Description		

BACK TO TOP

Heuristic Parameter Tampering

風險

可能發生什麼問題

惡意使用者可以透過直接請求來取得其他使用者的資訊，像是利用帳號編號攻擊者可以繞過授權程序竊取機密或存取受限的資訊 (例如銀行帳號餘額)，使用的僅僅只是直接物件參考。

原因

如何發生

應用程式在沒有透過使用者 ID 過濾資訊的情況下提供資料給使用者，舉例來說，可以單純透過傳送帳號 ID，應用程式就會連接這個使用者輸入變數到 SQL 查詢語法的字串中，不經過任何過濾程序，應用程式也沒有對輸入變數進行任何驗證，或是用一個預先計算好可接受數值的列表來進行限制比對。

一般建議

如何避免

一般建議：

- 在存取任何敏感性資料前強制進行授權驗證，包括特定的物件參考。
- 明確的阻擋任何未授權資料的存取，特別是其他使用者的資料。
- 如果可以，避免讓使用者可以單純透過傳送資料 ID 取得任意資料，例如與其讓使用者傳送帳號 ID 不如讓應用程式直接檢查目前授權使用者 session 的帳號 ID 。

針對性防禦措施：

- 不要將使用者輸入變數直接與 SQL 查詢進行字串連接。
- 將使用者專屬辨識元包含在 SQL 查詢的 WHERE 字句中進行過濾。
- 將使用者輸入變數對照非直接參照變數，像是透過一個準備好的可接受值列表。

程式碼範例

Java

Unfiltered Direct Object Reference

```
public ResultSet getAccountInfo(request req)  {
    int accountId = Integer.parseInt(req.getParameter("accountId"));

    string sql = "select * from [Accounts] where [AccountId] = "
                       + accountId.toString();

    Statement stmt = conn.createStatement();
    ResultSet accountRS = stmt.executeQuery(sql);

    return accountRS;
}
```

Record References are Now Filtered and Indirect

```
public ResultSet getAccountInfo(request req)  {
    int accountIndex = Integer.parseInt(req.getParameter("accountIndex"));
    int realAccountId = userAccountList.get(accountIndex);

    int userId = req.getSession().getAttribute("userId");
```

```
    string sql = "select * from [Accounts] where [AccountId] = "
                        + realAccountId.toString()
                        + " and [UserId] = " + userId.toString();

    Statement stmt = conn.createStatement();
    ResultSet accountRS = stmt.executeQuery(sql);

    return accountRS;
}
```

Heuristic Stored XSS

風險

可能發生什麼問題

攻擊者可以利用合法存取應用程式提交資料到資料庫。當另一個使用者隨後存取該資料，網頁可能會被改寫且惡意腳本可能會被啟動。

原因

如何發生

從資料庫中的資料建立網頁。資料直接嵌入至HTML的頁面，利用瀏覽器顯示。 資料可能源自於其他使用者的輸入。如果資料包含HTML片段或Javascript，使用者無法分辨是否為預期的頁面。 該漏洞主因為未先對嵌入資料庫中的資料進行編碼(Encode)來預防瀏覽器將其當為HTML的格式而非純文字。

一般建議

如何避免

'1. 驗證所有資料，無論其來源為何。驗證應基於白名單：僅接受預定結構的資訊，而不是拒絕不良的樣式(Patterns)。 應確認： ● 資料型態 ● 大小 ● 範圍 ● 格式 ● 期望值 2. 驗證無法取代編碼. 輸出嵌入之前， 不論其來源， 將所有動態資料進行編碼. 編碼方式應該是上下文相關的. 例如: ●HTML內容使用HTML的編碼方式 ●HTML編碼特性是將資料輸出到特性的值 ●JavaScript的編碼方式為伺服器產生的Javascript 3. 考慮使用ESAPI的編碼庫，或它的內置功能。對於舊版的ASP.NET，請考慮使用AntiXSS. 4. 在HTTP類型對應的表頭， 明確定義整個頁面的字元編碼. 5. 設置 httpOnly 標誌於會期資訊， 以防止利用XSS來竊取資訊.

程式碼範例

CSharp

Data obtained from the execution of an SQL command is output to a label

```
public class StoredXss
{
        public string foo(Label lblOutput, SqlConnection connection, int id)
        {
                string sql = "select email from CustomerLogin where customerNumber = @id";
                SqlCommand cmd = new SqlCommand(sql, connection);
                cmd.Prepare();
                cmd.Parameters.AddWithValue("@id", id);
                string output = (string)cmd.ExecuteScalar();
                lblOutput.Text = String.IsNullOrEmpty(output) ? "Customer Number does not
exist" : output;
        }
}
```

The outputed string is Html encoded before it is displayed in the label

```
public class StoredXssFixed
{
        public string foo(Label lblOutput, SqlConnection connection, HttpServerUtility Server,
int id)
        {
                string sql = "select email from CustomerLogin where customerNumber = @id";
                SqlCommand cmd = new SqlCommand(sql, connection);
                cmd.Prepare();
                cmd.Parameters.AddWithValue("@id", id);
                string output = (string)cmd.ExecuteScalar()
                lblOutput.Text = String.IsNullOrEmpty(output) ? "Customer Number does not
```

```
exist" : Server.HtmlEncode(output);
        }
}
```

Java

Data obtained from the excecution of an SQL command is output to a label

```
public class Stored_XSS {
      public static void XSSExample(Statement stmt) throws SQLException {
            Label label = new Label();
            ResultSet rs;
            rs = stmt.executeQuery("SELECT * FROM Customers WHERE UserName = Mickey");
            String lastNames = "";
            while (rs.next()) {
                  lastNames += rs.getString("Lname") + ", ";
            }
            label.setText("Mickey last names are: " + lastNames + " ");
      }
}
```

The outputed string is encoded to hard-coded string before it is displayed in the label

```
public class Stored_XSS_Fix {
      public static void XSSExample(Statement stmt) throws SQLException {
            Label label = new Label();
            ResultSet rs;
            HashMap<String, String> sanitize = new HashMap<String, String>();
            sanitize.put("A", "Cohen");
            sanitize.put("B", "Smith");
            sanitize.put("C", "Bond");
            rs = stmt.executeQuery("SELECT * FROM Customers WHERE UserName = Mickey");
            String lastNames = "";
            while (rs.next()) {
                  lastNames += sanitize.get(rs.getString("Lname")) + ", ";
            }
            label.setText("Mickey last names are: " + lastNames + " ");
      }
}
```

JavaScript

Data obtained from the execution of an SQL command is rendered to a web-page template

```
function renderUserProfileTable(res, connection, user_id) {
      connection.query('SELECT id,name,description from user WHERE id= ?',
[user_id],function(err, results) {
            var table = "<table>"
            table += "<table class='profile-html-table'>"
            table += "<tr><td>" + results[0].name + "</td></tr>"
            table += "<tr><td>" + results[0].description + "</td></tr>"
            table += "</table>"
            res.render("profile", table)
      });
}
```

Data obtained from the execution of an SQL command is encoded and then rendered to a web-page template

```
var htmlencoder = require('htmlencode');

function renderUserProfileTable(res, connection, user_id) {
      connection.query('SELECT id,name,description from user WHERE id= ?',
[user_id],function(err, results) {
            var table = "<table>"
            table += "<table class='profile-html-table'>"
            table += "<tr><td>" + htmlencoder.htmlEncode(results[0].name) + "</td></tr>"
            table += "<tr><td>" + htmlencoder.htmlEncode(results[0].description) + "</td></tr>"
            table += "</table>"
            res.render("profile", table)
      });
}
```

Exposure of Resource to Wrong Sphere

風險

可能發生什麼問題

如果沒有嚴謹的規劃變數存取權限，不小心將class的內部變數設為public，則該變數可能被預期之外的方式修改，並允許此class的外部使用者為該變數設定任意或不被允許的值。 若class或其他使用者嘗試修改該變數的值，可能使程式異常。根據此變數的使用方式，甚至能衍生出其他漏洞。

原因

如何發生

應用程式中的一個class將其屬性(property)宣告為public，而沒有對其限制存取權限。或者是忘了對public變數加上不可從外部修改的保護。

一般建議

如何避免

- 避免將內部變數和特定實現宣告為public。
- 若要公開，建議將變數設定為屬性(property)，並根據需求在程式碼中進行資料驗證和控制。
- 當需要將變數宣告為public，可以加上 `final` 修飾字將值限制為唯讀。

程式碼範例

Java

Exposing Public Field

```
public class MyProduct {
    // This value can be modified by any external code
    public float price;

    public MyProduct()  {
        this.price = ReadPriceFromDB("MyProduct");
    }
}
```

Exposing Read-Only Field

```
public class MyProduct {
    // This value can be read by external code,
    //   but can only be modified by the constructor
    public final float price;

    public MyProduct()  {
        this.price = ReadPriceFromDB("MyProduct");
    }
}
```

Wrapping with Properties

```
public class MyProduct {
    // This value can only be accessed by the class itself
```

```
    private float price;

    // External code can only read the value by calling the accessor property
    public float getPrice() {
        return price;
    }

    public MyProduct()  {
        this.price = ReadPriceFromDB("MyProduct");
    }
}
```

Hardcoded Absolute Path

風險

可能發生什麼問題

通常，寫死絕對路徑會使應用程式變得脆弱，並且會使程式在某些沒有相同檔案系統結構的環境中無法正常運行。如果應用程式未來版本的的設計或需求發生變化，這還會為軟體帶來維護問題。

此外，如果應用程式使用此路徑來讀取或寫入資料，則可能導致洩漏機密資料或程式被惡意輸入資料。在某些情況下，此漏洞甚至會使惡意使用者能夠覆寫預期的功能，使應用程式運行任意程式並執行攻擊者部署到伺服器的任意程式碼。

原因

如何發生

寫死的路徑不太靈活，使得應用程式難以適用環境變化。例如，程式可能被安裝在與預設目錄不同的目錄中。同樣，不同的系統語言和OS 架構會更改系統資料夾的名稱；例如，在西班牙語Windows 機器中會是"`C:\Archivos de programa (x86)\`" 而非"`C:\Program Files\`"。

此外，在Windows預設情況下，於系統資料夾和使用者設定檔之外創建的所有目錄和檔案，對任何經過身份驗證的使用者都給予完全的讀寫權限。因此，儘管應用程式假設它們是受到保護的，未經授權的惡意使用者還是可以訪問這些資料夾並讀取所有敏感資料。更糟糕的是，這些資料夾中未受保護的現有程式可能遭攻擊者覆寫並植入惡意程式碼，然後由應用程式啟動。

一般建議

如何避免

- 不要將絕對路徑寫死到應用程式中。
- 建議將絕對路徑儲存在外部設定檔中，以便根據每個環境的情況進行修改。
- 或者，如果目標檔案位於應用程式根目錄的一個子目錄中，也可使用相對當前應用程式的路徑。
- 不要在應用程式子目錄外假設特定的檔案系統結構。在 Windows 上，使用內建的可擴充變數，例如 `%WINDIR%`、`%PROGRAMFILES%`、和 `%TEMP%`。
- 在 Linux 和其他作業系統上，可以為應用程式設置系統監獄(system jail, 如 chroot)，並將所有程式和資料檔儲存到那裡。
- 最好將所有執行檔儲存在受保護的程式目錄下 (Windows 預設在 "`C:\Program Files\`")。
- 不要在任意資料夾中儲存敏感資料或設定檔。同樣，不要將資料檔儲存在程式目錄中。建議使用指定的資料夾，例如 Windows 上分別是 `%PROGRAMDATA%` 和 `%APPDATA%`。
- 根據最小權限原則，盡量嚴格地配置權限。請考慮在安裝和設置例程中自動實作此功能。

程式碼範例

Java

Hardcoded Path to Data File

```
public File getLogFile() {
      String filename = "C:\Logs\myapp.log";
      File logFile = new File(filename);

      return logFile;
}
```

Configured Path for Data File

```
public File getLogFile() {
```

```
        Properties props = this.Properties;
        String filename = (String)props.get("logDirectory") + (String)props.get("logFilename");

        File logFile = new File(filename);

        return logFile;
    }
```

Insufficient Logging

Weakness ID: 778 *(Weakness Base)* **Status:** Draft

Description

Description Summary

When a security-critical event occurs, the software either does not record the event or omits important details about the event when logging it.

Extended Description

When security-critical events are not logged properly, such as a failed login attempt, this can make malicious behavior more difficult to detect and may hinder forensic analysis after an attack succeeds.

Time of Introduction

- Operation

Applicable Platforms

Languages

Language-independent

Common Consequences

Scope	Effect
Accountability	If security critical information is not recorded, there will be no trail for forensic analysis and discovering the cause of problems or the source of attacks may become more difficult or impossible.

Likelihood of Exploit

Medium

Demonstrative Examples

Example 1

The example below shows a configuration for the service security audit feature in the Windows Communication Foundation (WCF).

(Bad Code)

Example Language: **XML**

```
<system.serviceModel>
<behaviors>
<serviceBehaviors>
<behavior name="NewBehavior">
<serviceSecurityAudit auditLogLocation="Default"
suppressAuditFailure="false"
serviceAuthorizationAuditLevel="None"
messageAuthenticationAuditLevel="None" />
...
</system.serviceModel>
```

The previous configuration file has effectively disabled the recording of security-critical events, which would force the administrator to look to other sources during debug or recovery efforts.

Logging failed authentication attempts can warn administrators of potential brute force attacks. Similarly, logging successful authentication events can provide a useful audit trail when a legitimate account is compromised. The following configuration shows appropriate settings, assuming that the site does not have excessive traffic, which could fill the logs if there are a large number of success or failure events (CWE-779).

(Good Code)

Example Language: **XML**

```
<system.serviceModel>
<behaviors>
<serviceBehaviors>
<behavior name="NewBehavior">
<serviceSecurityAudit auditLogLocation="Default"
suppressAuditFailure="false"
serviceAuthorizationAuditLevel="SuccessAndFailure"
messageAuthenticationAuditLevel="SuccessAndFailure" />
...
</system.serviceModel>
```

Observed Examples

Reference	Description
CVE-2008-4315	server does not log failed authentication attempts, making it easier for attackers to perform brute force password guessing without being detected
CVE-2008-1203	admin interface does not log failed authentication attempts, making it easier for attackers to perform brute force password guessing without being detected
CVE-2007-3730	default configuration for POP server does not log source IP or username for login attempts
CVE-2007-1225	proxy does not log requests without "http://" in the URL, allowing web surfers to access restricted web content without detection
CVE-2003-1566	web server does not log requests for a non-standard request type

Potential Mitigations

Phase: Architecture and Design

Use a centralized logging mechanism that supports multiple levels of detail. Ensure that all security-related successes and failures can be logged.

Phase: Operation

Be sure to set the level of logging appropriately in a production environment. Sufficient data should be logged to enable system administrators to detect attacks, diagnose errors, and recover from attacks. At the same time, logging too much data (CWE-779) can cause the same problems.

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Weakness Base	223	Omission of Security-relevant Information	**Development Concepts (primary)699 Research Concepts (primary)1000**
ChildOf	Category	254	Security Features	Development Concepts699
ChildOf	Weakness Class	693	Protection Mechanism Failure	Research Concepts1000

Content History

Submissions			
Submission Date	**Submitter**	**Organization**	**Source**
2009-07-02			Internal CWE Team
Contributions			
Contribution Date	**Contributor**	**Organization**	**Source**
2009-07-02		Fortify Software	Content
	Provided code example and additional information for description and consequences.		

BACK TO TOP

Insufficient Logging

Weakness ID: 778 *(Weakness Base)* **Status:** Draft

Description

Description Summary

When a security-critical event occurs, the software either does not record the event or omits important details about the event when logging it.

Extended Description

When security-critical events are not logged properly, such as a failed login attempt, this can make malicious behavior more difficult to detect and may hinder forensic analysis after an attack succeeds.

Time of Introduction

- Operation

Applicable Platforms

Languages

Language-independent

Common Consequences

Scope	Effect
Accountability	If security critical information is not recorded, there will be no trail for forensic analysis and discovering the cause of problems or the source of attacks may become more difficult or impossible.

Likelihood of Exploit

Medium

Demonstrative Examples

Example 1

The example below shows a configuration for the service security audit feature in the Windows Communication Foundation (WCF).

(Bad Code)

Example Language: **XML**

```
<system.serviceModel>
<behaviors>
<serviceBehaviors>
<behavior name="NewBehavior">
<serviceSecurityAudit auditLogLocation="Default"
suppressAuditFailure="false"
serviceAuthorizationAuditLevel="None"
messageAuthenticationAuditLevel="None" />
...
</system.serviceModel>
```

The previous configuration file has effectively disabled the recording of security-critical events, which would force the administrator to look to other sources during debug or recovery efforts.

Logging failed authentication attempts can warn administrators of potential brute force attacks. Similarly, logging successful authentication events can provide a useful audit trail when a legitimate account is compromised. The following configuration shows appropriate settings, assuming that the site does not have excessive traffic, which could fill the logs if there are a large number of success or failure events (CWE-779).

(Good Code)

Example Language: **XML**

```
<system.serviceModel>
<behaviors>
<serviceBehaviors>
<behavior name="NewBehavior">
<serviceSecurityAudit auditLogLocation="Default"
suppressAuditFailure="false"
serviceAuthorizationAuditLevel="SuccessAndFailure"
messageAuthenticationAuditLevel="SuccessAndFailure" />
...
</system.serviceModel>
```

Observed Examples

Reference	Description
CVE-2008-4315	server does not log failed authentication attempts, making it easier for attackers to perform brute force password guessing without being detected
CVE-2008-1203	admin interface does not log failed authentication attempts, making it easier for attackers to perform brute force password guessing without being detected
CVE-2007-3730	default configuration for POP server does not log source IP or username for login attempts
CVE-2007-1225	proxy does not log requests without "http://" in the URL, allowing web surfers to access restricted web content without detection
CVE-2003-1566	web server does not log requests for a non-standard request type

Potential Mitigations

Phase: Architecture and Design

Use a centralized logging mechanism that supports multiple levels of detail. Ensure that all security-related successes and failures can be logged.

Phase: Operation

Be sure to set the level of logging appropriately in a production environment. Sufficient data should be logged to enable system administrators to detect attacks, diagnose errors, and recover from attacks. At the same time, logging too much data (CWE-779) can cause the same problems.

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Weakness Base	223	Omission of Security-relevant Information	**Development Concepts (primary)699 Research Concepts (primary)1000**
ChildOf	Category	254	Security Features	Development Concepts699
ChildOf	Weakness Class	693	Protection Mechanism Failure	Research Concepts1000

Content History

Submissions			
Submission Date	**Submitter**	**Organization**	**Source**
2009-07-02			Internal CWE Team
Contributions			
Contribution Date	**Contributor**	**Organization**	**Source**
2009-07-02		Fortify Software	Content
	Provided code example and additional information for description and consequences.		

BACK TO TOP

Failure to Use a Standardized Error Handling Mechanism

Weakness ID: 544 *(Weakness Base)* **Status:** Draft

Description

Description Summary

The software does not use a standardized method for handling errors throughout the code, which might introduce inconsistent error handling and resultant weaknesses.

Extended Description

If the application handles error messages individually, on a one-by-one basis, this is likely to result in inconsistent error handling. The causes of errors may be lost. Also, detailed information about the causes of an error may be unintentionally returned to the user.

Time of Introduction

- Architecture and Design

Potential Mitigations

Phase: Architecture and Design

define a strategy for handling errors of different severities, such as fatal errors versus basic log events. Use or create built-in language features, or an external package, that provides an easy-to-use API and define coding standards for the detection and handling of errors.

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Category	388	Error Handling	**Development Concepts (primary)699**
ChildOf	Category	746	CERT C Secure Coding Section 12 - Error Handling (ERR)	**Weaknesses Addressed by the CERT C Secure Coding Standard (primary)734**
ChildOf	Weakness Class	755	Improper Handling of Exceptional Conditions	**Research Concepts (primary)1000**

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
Anonymous Tool Vendor (under NDA)			
CERT C Secure Coding	ERR00-C		Adopt and implement a consistent and comprehensive error-handling policy

Content History

Submissions

Submission Date	Submitter	Organization	Source
	Anonymous Tool Vendor (under NDA)		Externally Mined

Modifications

Modification Date	Modifier	Organization	Source
2008-07-01	Eric Dalci	Cigital	External
	updated Potential Mitigations, Time of Introduction		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Description, Relationships, Taxonomy Mappings		
2008-10-14	CWE Content Team	MITRE	Internal
	updated Relationships		
2008-11-24	CWE Content Team	MITRE	Internal
	updated Relationships, Taxonomy Mappings		
2009-03-10	CWE Content Team	MITRE	Internal
	updated Description, Name, Relationships		
2009-10-29	CWE Content Team	MITRE	Internal
	updated Potential Mitigations, Time of Introduction		

Previous Entry Names	
Change Date	**Previous Entry Name**
2009-03-10	Missing Error Handling Mechanism

BACK TO TOP

檢測的語言

語言	HASH值	變更的日期
CSharp	0763899333634536	2022/6/8
JavaScript	1012728348149017	2022/6/8
VbScript	0386000544005133	2022/6/8
Common	1867855616463800	2022/6/8