wireshark-1 Scan Report

Project Name	wireshark-1
Scan Start	Friday, June 21, 2024 6:11:06 PM
Preset	Checkmarx Default
Scan Time	01h:04m:25s
Lines Of Code Scanned	101145
Files Scanned	20
Report Creation Time	Friday, June 21, 2024 7:20:34 PM
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038
Team	CxServer
Checkmarx Version	8.7.0
Scan Type	Full
Source Origin	LocalPath
Density	6/10000 (Vulnerabilities/LOC)
Visibility	Public

Filter Settings

Severity

Included: High, Medium, Low, Information

Excluded: None

Result State

Included: Confirmed, Not Exploitable, To Verify, Urgent, Proposed Not Exploitable

Excluded: None

Assigned to

Included: All

Categories

Included:

Uncategorized	All
Custom	All
PCI DSS v3.2	All
OWASP Top 10 2013	All
FISMA 2014	All
NIST SP 800-53	All
OWASP Top 10 2017	All
OWASP Mobile Top 10 2016	All

Excluded:

Uncategorized	None
Custom	None
PCI DSS v3.2	None
OWASP Top 10 2013	None
FISMA 2014	None

NIST SP 800-53	None
OWASP Top 10 2017	None
OWASP Mobile Top 10 2016	None

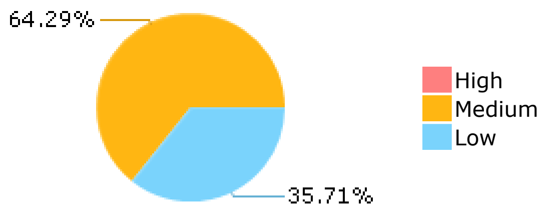
Results Limit

Results limit per query was set to 50

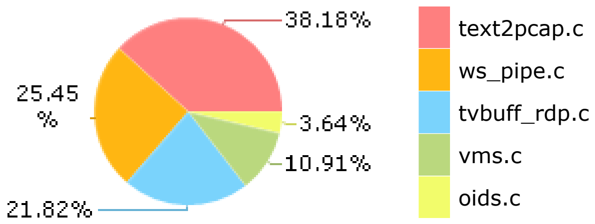
Selected Queries

Selected queries are listed in Result Summary

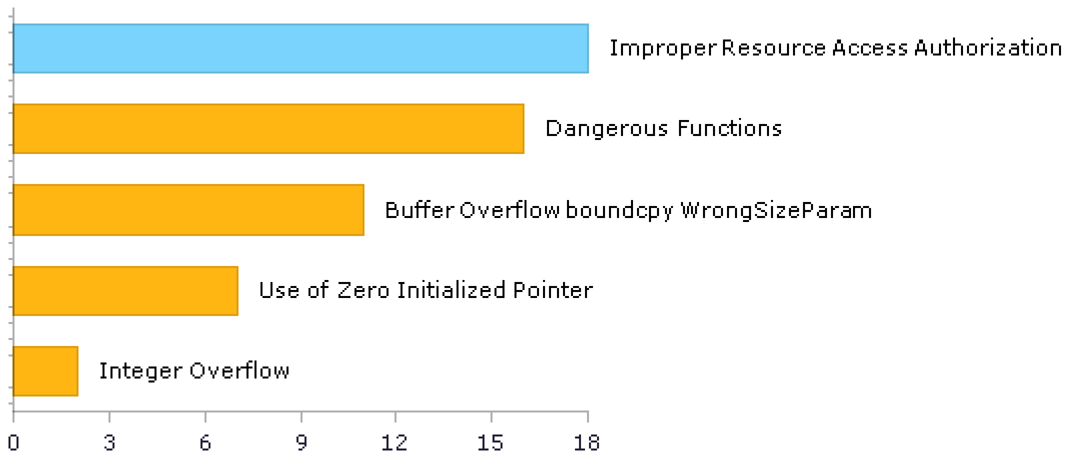
Result Summary



Most Vulnerable Files



Top 5 Vulnerabilities



Scan Summary - OWASP Top 10 2017

Further details and elaboration about vulnerabilities and risks can be found at: 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	11	11
A2-Broken Authentication	App. Specific	EASY	COMMON	AVERAGE	SEVERE	App. Specific	18	18
A3-Sensitive Data Exposure	App. Specific	AVERAGE	WIDESPREAD	AVERAGE	SEVERE	App. Specific	0	0
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	0	0
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	0	0
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	16	16
A10-Insufficient Logging & Monitoring	App. Specific	AVERAGE	WIDESPREAD	DIFFICULT	MODERATE	App. Specific	0	0

* Project scan results do not include all relevant queries. Presets and\or Filters should be changed to include all relevant standard queries.

Scan Summary - OWASP Top 10 2013

Further details and elaboration about vulnerabilities and risks can be found at: 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	0	0
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	0	0
A4-Insecure Direct Object References	SYSTEM USERS	EASY	COMMON	EASY	MODERATE	EXPOSED DATA	0	0
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	0	0
A7-Missing Function Level Access Control*	EXTERNAL, INTERNAL USERS	EASY	COMMON	AVERAGE	MODERATE	EXPOSED DATA AND FUNCTIONS	0	0
A8-Cross-Site Request Forgery (CSRF)	USERS BROWSERS	AVERAGE	COMMON	EASY	MODERATE	AFFECTED DATA AND FUNCTIONS	0	0
A9-Using Components with Known Vulnerabilities*	EXTERNAL USERS, AUTOMATED TOOLS	AVERAGE	WIDESPREAD	DIFFICULT	MODERATE	AFFECTED DATA AND FUNCTIONS	16	16
A10-Unvalidated Redirects and Forwards	USERS BROWSERS	AVERAGE	WIDESPREAD	DIFFICULT	MODERATE	AFFECTED DATA AND FUNCTIONS	0	0

***** Project scan results do not include all relevant queries. Presets and\or Filters should be changed to include all relevant standard queries.

Scan Summary - PCI DSS v3.2

Category	Issues Found	Best Fix Locations
PCI DSS (3.2) - 6.5.1 - Injection flaws - particularly SQL injection	0	0
PCI DSS (3.2) - 6.5.2 - Buffer overflows	13	13
PCI DSS (3.2) - 6.5.3 - Insecure cryptographic storage	0	0
PCI DSS (3.2) - 6.5.4 - Insecure communications	0	0
PCI DSS (3.2) - 6.5.5 - Improper error handling*	0	0
PCI DSS (3.2) - 6.5.7 - Cross-site scripting (XSS)	0	0
PCI DSS (3.2) - 6.5.8 - Improper access control	0	0
PCI DSS (3.2) - 6.5.9 - Cross-site request forgery	0	0
PCI DSS (3.2) - 6.5.10 - Broken authentication and session management	0	0

* Project scan results do not include all relevant queries. Presets and\or Filters should be changed to include all relevant standard queries.

Scan Summary - 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.	0	0
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.	0	0
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.	18	18
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.	2	2

***** Project scan results do not include all relevant queries. Presets and\or Filters should be changed to include all relevant standard queries.

Scan Summary - NIST SP 800-53

Category	Issues Found	Best Fix Locations
AC-12 Session Termination (P2)	0	0
AC-3 Access Enforcement (P1)	18	18
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)*	0	0
SC-28 Protection of Information at Rest (P1)	0	0
SC-4 Information in Shared Resources (P1)	0	0
SC-5 Denial of Service Protection (P1)*	7	7
SC-8 Transmission Confidentiality and Integrity (P1)	0	0
SI-10 Information Input Validation (P1)*	2	2
SI-11 Error Handling (P2)*	0	0
SI-15 Information Output Filtering (P0)	0	0
SI-16 Memory Protection (P1)	0	0

* Project scan results do not include all relevant queries. Presets and\or Filters should be changed to include all relevant standard queries.

Scan Summary - 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

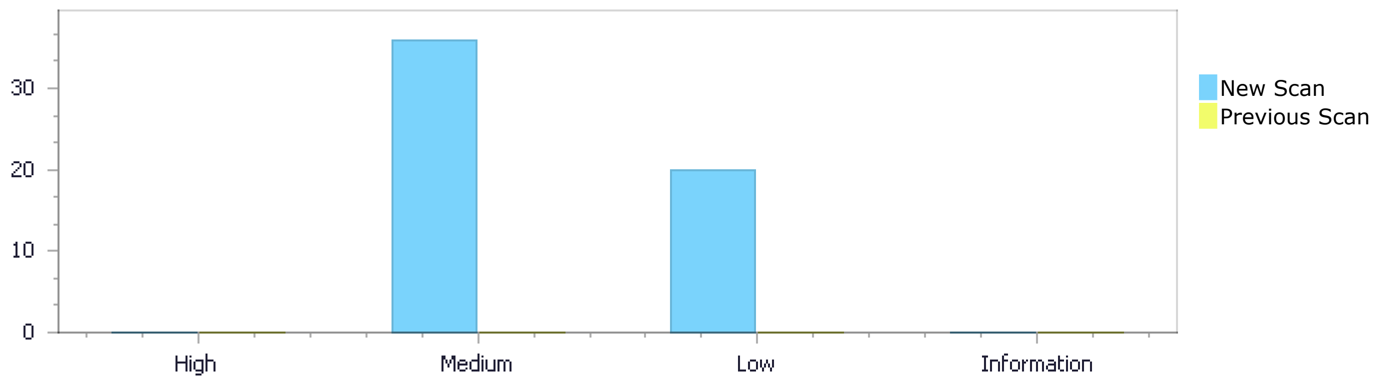
Scan Summary - Custom

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

Results Distribution By Status

First scan of the project

	High	Medium	Low	Information	Total
New Issues	0	36	20	0	56
Recurrent Issues	0	0	0	0	0
Total	0	36	20	0	56
Fixed Issues	0	0	0	0	0



Results Distribution By State

	High	Medium	Low	Information	Total
Confirmed	0	0	0	0	0
Not Exploitable	0	0	0	0	0
To Verify	0	36	20	0	56
Urgent	0	0	0	0	0
Proposed Not Exploitable	0	0	0	0	0
Total	0	36	20	0	56

Result Summary

Vulnerability Type	Occurrences	Severity
Dangerous Functions	16	Medium
Buffer Overflow boundcpy WrongSizeParam	11	Medium
Use of Zero Initialized Pointer	7	Medium
Integer Overflow	2	Medium
Improper Resource Access Authorization	18	Low

Sizeof Pointer Argument	1	Low
Use of Sizeof On a Pointer Type	1	Low

10 Most Vulnerable Files

High and Medium Vulnerabilities

File Name	Issues Found
wireshark-2/ws_pipe.c	13
wireshark-2/tvbuff_rdp.c	12
wireshark-2/vms.c	6
wireshark-2/text2pcap.c	3
wireshark-2/oids.c	1
wireshark-2/packet-ubertooth.c	1

Scan Results Details

Dangerous Functions

Query Path:
CPP\Cx\CPP Medium Threat\Dangerous Functions Version:1

Categories

OWASP Top 10 2013: A9-Using Components with Known Vulnerabilities
OWASP Top 10 2017: A9-Using Components with Known Vulnerabilities

Description

Dangerous Functions\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=16
Status	New

The dangerous function, memcpy, was found in use at line 473 in wireshark-2/tvbuff_rdp.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	wireshark-2/tvbuff_rdp.c	wireshark-2/tvbuff_rdp.c
Line	488	488
Object	memcpy	memcpy

Code Snippet
File Name wireshark-2/tvbuff_rdp.c
Method rdp8_decompress(zgfx_context_t *zgfx, wmem_allocator_t *allocator, tvbuff_t *tvb, guint offset)

```
....
488.            memcpy(output, zgfx->outputSegment, zgfx->outputCount);
```

Dangerous Functions\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=17
Status	New

The dangerous function, memcpy, was found in use at line 473 in wireshark-2/tvbuff_rdp.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	wireshark-2/tvbuff_rdp.c	wireshark-2/tvbuff_rdp.c

Line	516	516
Object	memcpy	memcpy

Code Snippet
File Name wireshark-2/tvbuff_rdp.c
Method rdp8_decompress(zgfx_context_t *zgfx, wmem_allocator_t *allocator, tvbuff_t *tvb, guint offset)

```
....
516.                memcpy(output_ptr, zgfx->outputSegment, zgfx->outputCount);
```

Dangerous Functions\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=18
Status	New

The dangerous function, memcpy, was found in use at line 211 in wireshark-2/tvbuff_rdp.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	wireshark-2/tvbuff_rdp.c	wireshark-2/tvbuff_rdp.c
Line	223	223
Object	memcpy	memcpy

Code Snippet
File Name wireshark-2/tvbuff_rdp.c
Method zgfx_write_history_buffer(zgfx_context_t *zgfx, const guint8 *src, guint32 count)

```
....
223.            memcpy(&(zgfx->historyBuffer[zgfx->historyIndex]), src, count);
```

Dangerous Functions\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=19
Status	New

The dangerous function, memcpy, was found in use at line 211 in wireshark-2/tvbuff_rdp.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination

File	wireshark-2/tvbuff_rdp.c	wireshark-2/tvbuff_rdp.c
Line	228	228
Object	memcpy	memcpy

Code Snippet
File Name wireshark-2/tvbuff_rdp.c
Method zgfx_write_history_buffer(zgfx_context_t *zgfx, const guint8 *src, guint32 count)

```
....
228.            memcpy(&(zgfx->historyBuffer[zgfx->historyIndex]),
src, front);
```

Dangerous Functions\Path 5:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=20
Status New

The dangerous function, memcpy, was found in use at line 211 in wireshark-2/tvbuff_rdp.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	wireshark-2/tvbuff_rdp.c	wireshark-2/tvbuff_rdp.c
Line	229	229
Object	memcpy	memcpy

Code Snippet
File Name wireshark-2/tvbuff_rdp.c
Method zgfx_write_history_buffer(zgfx_context_t *zgfx, const guint8 *src, guint32 count)

```
....
229.            memcpy(&(zgfx->historyBuffer), src + front, count -
front);
```

Dangerous Functions\Path 6:

Severity Medium
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=21
Status New

The dangerous function, memcpy, was found in use at line 284 in wireshark-2/tvbuff_rdp.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	wireshark-2/tvbuff_rdp.c	wireshark-2/tvbuff_rdp.c
Line	301	301
Object	memcpy	memcpy

Code Snippet
File Name wireshark-2/tvbuff_rdp.c
Method zgfx_write_from_history(zgfx_context_t *zgfx, guint32 distance, guint32 count)

```
....
301.            memcpy(outputPtr, &(zgfx->historyBuffer[idx]),
toCopy);
```

Dangerous Functions\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=22
Status	New

The dangerous function, memcpy, was found in use at line 284 in wireshark-2/tvbuff_rdp.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	wireshark-2/tvbuff_rdp.c	wireshark-2/tvbuff_rdp.c
Line	304	304
Object	memcpy	memcpy

Code Snippet
File Name wireshark-2/tvbuff_rdp.c
Method zgfx_write_from_history(zgfx_context_t *zgfx, guint32 distance, guint32 count)

```
....
304.            memcpy(outputPtr, &(zgfx->historyBuffer[idx]),
partial);
```

Dangerous Functions\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=23
Status	New

The dangerous function, memcpy, was found in use at line 284 in wireshark-2/tvbuff_rdp.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination

File	wireshark-2/tvbuff_rdp.c	wireshark-2/tvbuff_rdp.c
Line	305	305
Object	memcpy	memcpy

Code Snippet
File Name wireshark-2/tvbuff_rdp.c
Method zgfx_write_from_history(zgfx_context_t *zgfx, guint32 distance, guint32 count)

```
....
305.                memcpy(outputPtr + partial, zgfx->historyBuffer,
toCopy - partial);
```

Dangerous Functions\Path 9:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=24
Status	New

The dangerous function, memcpy, was found in use at line 284 in wireshark-2/tvbuff_rdp.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	wireshark-2/tvbuff_rdp.c	wireshark-2/tvbuff_rdp.c
Line	314	314
Object	memcpy	memcpy

Code Snippet
File Name wireshark-2/tvbuff_rdp.c
Method zgfx_write_from_history(zgfx_context_t *zgfx, guint32 distance, guint32 count)

```
....
314.                memcpy(outputPtr, &(zgfx->outputSegment[zgfx-
>outputCount]), toCopy);
```

Dangerous Functions\Path 10:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=25
Status	New

The dangerous function, sscanf, was found in use at line 354 in wireshark-2/text2pcap.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	wireshark-2/text2pcap.c	wireshark-2/text2pcap.c

Line	437	437
Object	sscanf	sscanf

Code Snippet
File Name wireshark-2/text2pcap.c
Method parse_options(int argc, char *argv[], text_import_info_t * const info, wtap_dump_params * const params)

```
....
437.            if (sscanf(ws_optarg, "%x", &hdr_ethernet_proto) < 1)
{
```

Dangerous Functions\Path 11:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=26
Status	New

The dangerous function, sscanf, was found in use at line 321 in wireshark-2/vms.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	wireshark-2/vms.c	wireshark-2/vms.c
Line	361	361
Object	sscanf	sscanf

Code Snippet
File Name wireshark-2/vms.c
Method parse_vms_packet(FILE_T fh, wtap_rec *rec, Buffer *buf, int *err, gchar **err_info)

```
....
361.            num_items_scanned = sscanf(p,
```

Dangerous Functions\Path 12:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=27
Status	New

The dangerous function, sscanf, was found in use at line 321 in wireshark-2/vms.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	wireshark-2/vms.c	wireshark-2/vms.c

Line	368	368
Object	sscanf	sscanf

Code Snippet
File Name wireshark-2/vms.c
Method parse_vms_packet(FILE_T fh, wtap_rec *rec, Buffer *buf, int *err, gchar **err_info)

```
....
368.                    num_items_scanned = sscanf(p,
```

Dangerous Functions\Path 13:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=28
Status	New

The dangerous function, strlen, was found in use at line 184 in wireshark-2/vms.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	wireshark-2/vms.c	wireshark-2/vms.c
Line	205	205
Object	strlen	strlen

Code Snippet
File Name wireshark-2/vms.c
Method static gboolean vms_check_file_type(wtap *wth, int *err, gchar **err_info)

```
....
205.            reclen = (guint) strlen(buf);
```

Dangerous Functions\Path 14:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=29
Status	New

The dangerous function, strlen, was found in use at line 184 in wireshark-2/vms.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	wireshark-2/vms.c	wireshark-2/vms.c
Line	206	206

Object	strlen	strlen

Code Snippet
File Name wireshark-2/vms.c
Method static gboolean vms_check_file_type(wtap *wth, int *err, gchar **err_info)

```
....
206.            if (reclen < strlen(VMS_HDR_MAGIC_STR1) ||
```

Dangerous Functions\Path 15:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=30
Status	New

The dangerous function, strlen, was found in use at line 184 in wireshark-2/vms.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	wireshark-2/vms.c	wireshark-2/vms.c
Line	207	207
Object	strlen	strlen

Code Snippet
File Name wireshark-2/vms.c
Method static gboolean vms_check_file_type(wtap *wth, int *err, gchar **err_info)

```
....
207.                reclen < strlen(VMS_HDR_MAGIC_STR2) ||
```

Dangerous Functions\Path 16:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=31
Status	New

The dangerous function, strlen, was found in use at line 184 in wireshark-2/vms.c file. Such functions may expose information and allow an attacker to get full control over the host machine.

	Source	Destination
File	wireshark-2/vms.c	wireshark-2/vms.c
Line	208	208
Object	strlen	strlen

Code Snippet	
File Name	wireshark-2/vms.c
Method	static gboolean vms_check_file_type(wtap *wth, int *err, gchar **err_info)

```
....
208.                reclen < strlen(VMS_HDR_MAGIC_STR3)) {
```

Buffer Overflow boundcpy WrongSizeParam

Query Path:
CPP\Cx\CPP Buffer Overflow\Buffer Overflow boundcpy WrongSizeParam Version:1

Categories

PCI DSS v3.2: PCI DSS (3.2) - 6.5.2 - Buffer overflows
OWASP Top 10 2017: A1-Injection

Description

Buffer Overflow boundcpy WrongSizeParam\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=2
Status	New

The size of the buffer used by ws_pipe_spawn_sync in OVERLAPPED, at line 229 of wireshark-2/ws_pipe.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that ws_pipe_spawn_sync passes to OVERLAPPED, at line 229 of wireshark-2/ws_pipe.c, to overwrite the target buffer.

	Source	Destination
File	wireshark-2/ws_pipe.c	wireshark-2/ws_pipe.c
Line	263	263
Object	OVERLAPPED	OVERLAPPED

Code Snippet	
File Name	wireshark-2/ws_pipe.c
Method	gboolean ws_pipe_spawn_sync(const gchar *working_directory, const gchar *command, gint argc, gchar **args, gchar **command_output)

```
....
263.        memset(&stdout_overlapped, 0, sizeof(OVERLAPPED));
```

Buffer Overflow boundcpy WrongSizeParam\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=3
Status	New

The size of the buffer used by ws_pipe_spawn_sync in OVERLAPPED, at line 229 of wireshark-2/ws_pipe.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that ws_pipe_spawn_sync passes to OVERLAPPED, at line 229 of wireshark-2/ws_pipe.c, to overwrite the target buffer.

	Source	Destination
File	wireshark-2/ws_pipe.c	wireshark-2/ws_pipe.c
Line	264	264
Object	OVERLAPPED	OVERLAPPED

Code Snippet
File Name wireshark-2/ws_pipe.c
Method gboolean ws_pipe_spawn_sync(const gchar *working_directory, const gchar *command, gint argc, gchar **args, gchar **command_output)

```
....
264.        memset(&stderr_overlapped, 0, sizeof(OVERLAPPED));
```

Buffer Overflow boundcpy WrongSizeParam\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=4
Status	New

The size of the buffer used by ws_pipe_spawn_sync in SECURITY_ATTRIBUTES, at line 229 of wireshark-2/ws_pipe.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that ws_pipe_spawn_sync passes to SECURITY_ATTRIBUTES, at line 229 of wireshark-2/ws_pipe.c, to overwrite the target buffer.

	Source	Destination
File	wireshark-2/ws_pipe.c	wireshark-2/ws_pipe.c
Line	283	283
Object	SECURITY_ATTRIBUTES	SECURITY_ATTRIBUTES

Code Snippet
File Name wireshark-2/ws_pipe.c
Method gboolean ws_pipe_spawn_sync(const gchar *working_directory, const gchar *command, gint argc, gchar **args, gchar **command_output)

```
....
283.        memset(&sa, 0, sizeof(SECURITY_ATTRIBUTES));
```

Buffer Overflow boundcpy WrongSizeParam\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=5
Status	New

The size of the buffer used by ws_pipe_spawn_sync in PROCESS_INFORMATION, at line 229 of wireshark-2/ws_pipe.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that ws_pipe_spawn_sync passes to PROCESS_INFORMATION, at line 229 of wireshark-2/ws_pipe.c, to overwrite the target buffer.

	Source	Destination
File	wireshark-2/ws_pipe.c	wireshark-2/ws_pipe.c
Line	313	313
Object	PROCESS_INFORMATION	PROCESS_INFORMATION

Code Snippet
File Name wireshark-2/ws_pipe.c
Method gboolean ws_pipe_spawn_sync(const gchar *working_directory, const gchar *command, gint argc, gchar **args, gchar **command_output)

```
....
313.      memset(&processInfo, 0, sizeof(PROCESS_INFORMATION));
```

Buffer Overflow boundcpy WrongSizeParam\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=6
Status	New

The size of the buffer used by ws_pipe_spawn_sync in STARTUPINFO, at line 229 of wireshark-2/ws_pipe.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that ws_pipe_spawn_sync passes to STARTUPINFO, at line 229 of wireshark-2/ws_pipe.c, to overwrite the target buffer.

	Source	Destination
File	wireshark-2/ws_pipe.c	wireshark-2/ws_pipe.c
Line	314	314
Object	STARTUPINFO	STARTUPINFO

Code Snippet
File Name wireshark-2/ws_pipe.c
Method gboolean ws_pipe_spawn_sync(const gchar *working_directory, const gchar *command, gint argc, gchar **args, gchar **command_output)

```
....
314.      memset(&info, 0, sizeof(STARTUPINFO));
```

Buffer Overflow boundcpy WrongSizeParam\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=7
Status	New

The size of the buffer used by ws_pipe_init in ws_pipe_t, at line 514 of wireshark-2/ws_pipe.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that ws_pipe_init passes to ws_pipe_t, at line 514 of wireshark-2/ws_pipe.c, to overwrite the target buffer.

	Source	Destination
File	wireshark-2/ws_pipe.c	wireshark-2/ws_pipe.c
Line	517	517
Object	ws_pipe_t	ws_pipe_t

Code Snippet
File Name wireshark-2/ws_pipe.c
Method void ws_pipe_init(ws_pipe_t *ws_pipe)

```
....
517.        memset(ws_pipe, 0, sizeof(ws_pipe_t));
```

Buffer Overflow boundcpy WrongSizeParam\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=8
Status	New

The size of the buffer used by ws_pipe_spawn_async in PROCESS_INFORMATION, at line 521 of wireshark-2/ws_pipe.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that ws_pipe_spawn_async passes to PROCESS_INFORMATION, at line 521 of wireshark-2/ws_pipe.c, to overwrite the target buffer.

	Source	Destination
File	wireshark-2/ws_pipe.c	wireshark-2/ws_pipe.c
Line	586	586
Object	PROCESS_INFORMATION	PROCESS_INFORMATION

Code Snippet
File Name wireshark-2/ws_pipe.c
Method GPid ws_pipe_spawn_async(ws_pipe_t *ws_pipe, GPtrArray *args)

```
....
586.        memset(&processInfo, 0, sizeof(PROCESS_INFORMATION));
```

Buffer Overflow boundcpy WrongSizeParam\Path 8:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=9
Status	New

The size of the buffer used by ws_pipe_spawn_async in STARTUPINFO, at line 521 of wireshark-2/ws_pipe.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that ws_pipe_spawn_async passes to STARTUPINFO, at line 521 of wireshark-2/ws_pipe.c, to overwrite the target buffer.

	Source	Destination
File	wireshark-2/ws_pipe.c	wireshark-2/ws_pipe.c
Line	587	587
Object	STARTUPINFO	STARTUPINFO

Code Snippet
File Name wireshark-2/ws_pipe.c
Method GPid ws_pipe_spawn_async(ws_pipe_t *ws_pipe, GPtrArray *args)

```
....
587.        memset(&info, 0, sizeof(STARTUPINFO));
```

Buffer Overflow boundcpy WrongSizeParam\Path 9:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=10
Status	New

The size of the buffer used by rdp8_decompress in zgfx, at line 473 of wireshark-2/tvbuff_rdp.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that rdp8_decompress passes to zgfx, at line 473 of wireshark-2/tvbuff_rdp.c, to overwrite the target buffer.

	Source	Destination
File	wireshark-2/tvbuff_rdp.c	wireshark-2/tvbuff_rdp.c
Line	488	488
Object	zgfx	zgfx

Code Snippet
File Name wireshark-2/tvbuff_rdp.c
Method rdp8_decompress(zgfx_context_t *zgfx, wmem_allocator_t *allocator, tvbuff_t *tvb, guint offset)

```
....
488.            memcpy(output, zgfx->outputSegment, zgfx->outputCount);
```

Buffer Overflow boundcpy WrongSizeParam\Path 10:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=11
Status	New

The size of the buffer used by rdp8_decompress in zgfx, at line 473 of wireshark-2/tvbuff_rdp.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that rdp8_decompress passes to zgfx, at line 473 of wireshark-2/tvbuff_rdp.c, to overwrite the target buffer.

	Source	Destination
File	wireshark-2/tvbuff_rdp.c	wireshark-2/tvbuff_rdp.c
Line	516	516
Object	zgfx	zgfx

Code Snippet
File Name wireshark-2/tvbuff_rdp.c
Method rdp8_decompress(zgfx_context_t *zgfx, wmem_allocator_t *allocator, tvbuff_t *tvb, guint offset)

```
....
516.                memcpy(output_ptr, zgfx->outputSegment, zgfx->outputCount);
```

Buffer Overflow boundcpy WrongSizeParam\Path 11:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=12
Status	New

The size of the buffer used by zgfx_write_from_history in partial, at line 284 of wireshark-2/tvbuff_rdp.c, is not properly verified before writing data to the buffer. This can enable a buffer overflow attack, using the source buffer that zgfx_write_from_history passes to partial, at line 284 of wireshark-2/tvbuff_rdp.c, to overwrite the target buffer.

	Source	Destination
File	wireshark-2/tvbuff_rdp.c	wireshark-2/tvbuff_rdp.c
Line	304	304
Object	partial	partial

Code Snippet
File Name wireshark-2/tvbuff_rdp.c
Method zgfx_write_from_history(zgfx_context_t *zgfx, guint32 distance, guint32 count)

```
....
304.            memcpy(outputPtr, &(zgfx->historyBuffer[idx]), partial);
```

Use of Zero Initialized Pointer

Query Path:
CPP\Cx\CPP Medium Threat\Use of Zero Initialized Pointer Version:1

Categories

NIST SP 800-53: SC-5 Denial of Service Protection (P1)

Description

Use of Zero Initialized Pointer\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=32
Status	New

The variable declared in next at wireshark-2/oids.c in line 389 is not initialized when it is used by kl at wireshark-2/oids.c in line 389.

	Source	Destination
File	wireshark-2/oids.c	wireshark-2/oids.c
Line	438	494
Object	next	kl

Code Snippet
File Name wireshark-2/oids.c
Method static inline oid_kind_t smikind(SmiNode* sN, oid_key_t** key_p) {

```
....
438.                        k->next = NULL;
....
494.                        kl = k;
```

Use of Zero Initialized Pointer\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=33
Status	New

The variable declared in key at wireshark-2/packet-ubertooth.c in line 1359 is not initialized when it is used by command_data at wireshark-2/packet-ubertooth.c in line 1359.

	Source	Destination
File	wireshark-2/packet-ubertooth.c	wireshark-2/packet-ubertooth.c
Line	1717	1721
Object	key	command_data

Code Snippet
File Name wireshark-2/packet-ubertooth.c
Method dissect_ubertooth(tvbuff_t *tvb, packet_info *pinfo, proto_tree *tree, void *data)

```
....
1717.        key[2].key = NULL;
....
1721.            command_data = (command_data_t *)
wmem_tree_lookup32_le(wmem_tree, pinfo->num);
```

Use of Zero Initialized Pointer\Path 3:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=34
Status	New

The variable declared in list at wireshark-2/text2pcap.c in line 325 is not initialized when it is used by list at wireshark-2/text2pcap.c in line 325.

	Source	Destination
File	wireshark-2/text2pcap.c	wireshark-2/text2pcap.c
Line	328	341
Object	list	list

Code Snippet
File Name wireshark-2/text2pcap.c
Method list_encap_types(void) {

```
....
328.       GSList *list = NULL;
....
341.                   list = g_slist_insert_sorted(list, &encaps[i],
string_nat_compare);
```

Use of Zero Initialized Pointer\Path 4:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=35
Status	New

The variable declared in gerror at wireshark-2/text2pcap.c in line 354 is not initialized when it is used by regex at wireshark-2/text2pcap.c in line 354.

	Source	Destination
File	wireshark-2/text2pcap.c	wireshark-2/text2pcap.c
Line	371	485
Object	gerror	regex

Code Snippet

File Name	wireshark-2/text2pcap.c
Method	parse_options(int argc, char *argv[], text_import_info_t * const info, wtap_dump_params * const params)

```
....
371.       GError* gerror = NULL;
....
485.                regex = g_regex_new(ws_optarg, G_REGEX_DUPNAMES |
G_REGEX_OPTIMIZE | G_REGEX_MULTILINE, G_REGEX_MATCH_NOTEMPTY, &gerror);
```

Use of Zero Initialized Pointer\Path 5:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=36
Status	New

The variable declared in child_stdin_wr at wireshark-2/ws_pipe.c in line 521 is not initialized when it is used by child_stdin_wr at wireshark-2/ws_pipe.c in line 521.

	Source	Destination
File	wireshark-2/ws_pipe.c	wireshark-2/ws_pipe.c
Line	531	599
Object	child_stdin_wr	child_stdin_wr

Code Snippet

File Name	wireshark-2/ws_pipe.c
Method	GPid ws_pipe_spawn_async(ws_pipe_t *ws_pipe, GPtrArray *args)

```
....
531.       HANDLE child_stdin_wr = NULL;
....
599.           stdin_fd = _open_osfhandle((intptr_t)(child_stdin_wr),
_O_BINARY);
```

Use of Zero Initialized Pointer\Path 6:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=37
Status	New

The variable declared in child_stdout_rd at wireshark-2/ws_pipe.c in line 521 is not initialized when it is used by child_stdout_rd at wireshark-2/ws_pipe.c in line 521.

	Source	Destination
File	wireshark-2/ws_pipe.c	wireshark-2/ws_pipe.c
Line	532	600

Object	child_stdout_rd	child_stdout_rd

Code Snippet
File Name wireshark-2/ws_pipe.c
Method GPid ws_pipe_spawn_async(ws_pipe_t *ws_pipe, GPtrArray *args)

```
....
532.        HANDLE child_stdout_rd = NULL;
....
600.            stdout_fd = _open_osfhandle((intptr_t)(child_stdout_rd),
_O_BINARY);
```

Use of Zero Initialized Pointer\Path 7:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=38
Status	New

The variable declared in child_stderr_rd at wireshark-2/ws_pipe.c in line 521 is not initialized when it is used by child_stderr_rd at wireshark-2/ws_pipe.c in line 521.

	Source	Destination
File	wireshark-2/ws_pipe.c	wireshark-2/ws_pipe.c
Line	534	601
Object	child_stderr_rd	child_stderr_rd

Code Snippet
File Name wireshark-2/ws_pipe.c
Method GPid ws_pipe_spawn_async(ws_pipe_t *ws_pipe, GPtrArray *args)

```
....
534.        HANDLE child_stderr_rd = NULL;
....
601.            stderr_fd = _open_osfhandle((intptr_t)(child_stderr_rd),
_O_BINARY);
```

Integer Overflow

Query Path:
CPP\Cx\CPP Integer Overflow\Integer Overflow Version:0

Categories

PCI DSS v3.2: PCI DSS (3.2) - 6.5.2 - Buffer overflows
FISMA 2014: System And Information Integrity
NIST SP 800-53: SI-10 Information Input Validation (P1)

Description

Integer Overflow\Path 1:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-

	BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=13
Status	New

A variable of a larger data type, i, is being assigned to a smaller data type, in 229 of wireshark-2/ws_pipe.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	wireshark-2/ws_pipe.c	wireshark-2/ws_pipe.c
Line	379	379
Object	i	i

Code Snippet
File Name wireshark-2/ws_pipe.c
Method gboolean ws_pipe_spawn_sync(const gchar *working_directory, const gchar *command, gint argc, gchar **args, gchar **command_output)

```
....
379.                    int i = dw - WAIT_OBJECT_0;
```

Integer Overflow\Path 2:

Severity	Medium
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=14
Status	New

A variable of a larger data type, handle_idx, is being assigned to a smaller data type, in 677 of wireshark-2/ws_pipe.c. This will cause a loss of data, often the significant bits of a numerical value or the sign bit.

	Source	Destination
File	wireshark-2/ws_pipe.c	wireshark-2/ws_pipe.c
Line	752	752
Object	handle_idx	handle_idx

Code Snippet
File Name wireshark-2/ws_pipe.c
Method ws_pipe_wait_for_pipe(HANDLE * pipe_handles, int num_pipe_handles, HANDLE pid)

```
....
752.            int handle_idx = dw - WAIT_OBJECT_0;
```

Improper Resource Access Authorization

Query Path:
CPP\Cx\CPP Low Visibility\Improper Resource Access Authorization Version:1

Categories

FISMA 2014: Identification And Authentication
NIST SP 800-53: AC-3 Access Enforcement (P1)

OWASP Top 10 2017: A2-Broken Authentication

[Description](#)

Improper Resource Access Authorization\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=39
Status	New

	Source	Destination
File	wireshark-2/text2pcap.c	wireshark-2/text2pcap.c
Line	1043	1043
Object	fprintf	fprintf

Code Snippet
File Name wireshark-2/text2pcap.c
Method main(int argc, char *argv[])

```
....
1043.          fprintf(stderr, "\n-------------------------\n");
```

Improper Resource Access Authorization\Path 2:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=40
Status	New

	Source	Destination
File	wireshark-2/text2pcap.c	wireshark-2/text2pcap.c
Line	173	173
Object	fprintf	fprintf

Code Snippet
File Name wireshark-2/text2pcap.c
Method print_usage (FILE *output)

```
....
173.      fprintf(output,
```

Improper Resource Access Authorization\Path 3:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=41

Status	New

	Source	Destination
File	wireshark-2/text2pcap.c	wireshark-2/text2pcap.c
Line	269	269
Object	fprintf	fprintf

Code Snippet
File Name wireshark-2/text2pcap.c
Method print_usage (FILE *output)

```
....
269.        fprintf(output, "\n"
```

Improper Resource Access Authorization\Path 4:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=42
Status	New

	Source	Destination
File	wireshark-2/text2pcap.c	wireshark-2/text2pcap.c
Line	298	298
Object	fprintf	fprintf

Code Snippet
File Name wireshark-2/text2pcap.c
Method list_capture_types(void) {

```
....
298.            fprintf(stderr, "    %s - %s\n",
wtap_file_type_subtype_name(ft),
```

Improper Resource Access Authorization\Path 5:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=43
Status	New

	Source	Destination
File	wireshark-2/text2pcap.c	wireshark-2/text2pcap.c
Line	319	319
Object	fprintf	fprintf

Code Snippet

File Name	wireshark-2/text2pcap.c
Method	string_elem_print(gpointer data, gpointer stream_ptr)

```
....
319.        fprintf((FILE *) stream_ptr, "    %s - %s\n",
```

Improper Resource Access Authorization\Path 6:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=44
Status	New

	Source	Destination
File	wireshark-2/text2pcap.c	wireshark-2/text2pcap.c
Line	950	950
Object	fprintf	fprintf

Code Snippet

File Name	wireshark-2/text2pcap.c
Method	parse_options(int argc, char *argv[], text_import_info_t * const info, wtap_dump_params * const params)

```
....
950.            fprintf(stderr, "Input from: %s\n", input_filename);
```

Improper Resource Access Authorization\Path 7:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=45
Status	New

	Source	Destination
File	wireshark-2/text2pcap.c	wireshark-2/text2pcap.c
Line	951	951
Object	fprintf	fprintf

Code Snippet

File Name	wireshark-2/text2pcap.c
Method	parse_options(int argc, char *argv[], text_import_info_t * const info, wtap_dump_params * const params)

```
....
951.            fprintf(stderr, "Output to: %s\n",  output_filename);
```

Improper Resource Access Authorization\Path 8:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=46
Status	New

	Source	Destination
File	wireshark-2/text2pcap.c	wireshark-2/text2pcap.c
Line	952	952
Object	fprintf	fprintf

Code Snippet
File Name wireshark-2/text2pcap.c
Method parse_options(int argc, char *argv[], text_import_info_t * const info, wtap_dump_params * const params)

```
....
952.            fprintf(stderr, "Output format: %s\n",
wtap_file_type_subtype_name(file_type_subtype));
```

Improper Resource Access Authorization\Path 9:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=47
Status	New

	Source	Destination
File	wireshark-2/text2pcap.c	wireshark-2/text2pcap.c
Line	953	953
Object	fprintf	fprintf

Code Snippet
File Name wireshark-2/text2pcap.c
Method parse_options(int argc, char *argv[], text_import_info_t * const info, wtap_dump_params * const params)

```
....
953.            if (hdr_ethernet) fprintf(stderr, "Generate dummy Ethernet
header: Protocol: 0x%0X\n",
```

Improper Resource Access Authorization\Path 10:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=48
Status	New

	Source	Destination
File	wireshark-2/text2pcap.c	wireshark-2/text2pcap.c
Line	955	955
Object	fprintf	fprintf

Code Snippet
File Name wireshark-2/text2pcap.c
Method parse_options(int argc, char *argv[], text_import_info_t * const info, wtap_dump_params * const params)

```
....
955.            if (hdr_ip) fprintf(stderr, "Generate dummy IP header:
Protocol: %u\n",
```

Improper Resource Access Authorization\Path 11:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=49
Status	New

	Source	Destination
File	wireshark-2/text2pcap.c	wireshark-2/text2pcap.c
Line	957	957
Object	fprintf	fprintf

Code Snippet
File Name wireshark-2/text2pcap.c
Method parse_options(int argc, char *argv[], text_import_info_t * const info, wtap_dump_params * const params)

```
....
957.            if (hdr_ipv6) fprintf(stderr, "Generate dummy IPv6 header:
Protocol: %u\n",
```

Improper Resource Access Authorization\Path 12:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=300

	[38&pathid=50](http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=50)
Status	New

	Source	Destination
File	wireshark-2/text2pcap.c	wireshark-2/text2pcap.c
Line	959	959
Object	fprintf	fprintf

Code Snippet
File Name wireshark-2/text2pcap.c
Method parse_options(int argc, char *argv[], text_import_info_t * const info, wtap_dump_params * const params)

```
....
959.            if (hdr_udp) fprintf(stderr, "Generate dummy UDP header:
Source port: %u. Dest port: %u\n",
```

Improper Resource Access Authorization\Path 13:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=51
Status	New

	Source	Destination
File	wireshark-2/text2pcap.c	wireshark-2/text2pcap.c
Line	961	961
Object	fprintf	fprintf

Code Snippet
File Name wireshark-2/text2pcap.c
Method parse_options(int argc, char *argv[], text_import_info_t * const info, wtap_dump_params * const params)

```
....
961.            if (hdr_tcp) fprintf(stderr, "Generate dummy TCP header:
Source port: %u. Dest port: %u\n",
```

Improper Resource Access Authorization\Path 14:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=52
Status	New

	Source	Destination

File	wireshark-2/text2pcap.c	wireshark-2/text2pcap.c
Line	963	963
Object	fprintf	fprintf

Code Snippet
File Name wireshark-2/text2pcap.c
Method parse_options(int argc, char *argv[], text_import_info_t * const info, wtap_dump_params * const params)

```
....
963.         if (hdr_sctp) fprintf(stderr, "Generate dummy SCTP header: Source port: %u. Dest port: %u. Tag: %u\n",
```

Improper Resource Access Authorization\Path 15:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=53
Status New

	Source	Destination
File	wireshark-2/text2pcap.c	wireshark-2/text2pcap.c
Line	965	965
Object	fprintf	fprintf

Code Snippet
File Name wireshark-2/text2pcap.c
Method parse_options(int argc, char *argv[], text_import_info_t * const info, wtap_dump_params * const params)

```
....
965.         if (hdr_data_chunk) fprintf(stderr, "Generate dummy DATA chunk header: TSN: %u. SID: %u. SSN: %u. PPID: %u\n",
```

Improper Resource Access Authorization\Path 16:

Severity Low
Result State To Verify
Online Results http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=54
Status New

	Source	Destination
File	wireshark-2/text2pcap.c	wireshark-2/text2pcap.c
Line	979	979
Object	fprintf	fprintf

Code Snippet
File Name wireshark-2/text2pcap.c
Method text2pcap_cmdarg_err(const char *msg_format, va_list ap)

```
....
979.        fprintf(stderr, "text2pcap: ");
```

Improper Resource Access Authorization\Path 17:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=55
Status	New

	Source	Destination
File	wireshark-2/text2pcap.c	wireshark-2/text2pcap.c
Line	981	981
Object	fprintf	fprintf

Code Snippet
File Name wireshark-2/text2pcap.c
Method text2pcap_cmdarg_err(const char *msg_format, va_list ap)

```
....
981.        fprintf(stderr, "\n");
```

Improper Resource Access Authorization\Path 18:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=56
Status	New

	Source	Destination
File	wireshark-2/text2pcap.c	wireshark-2/text2pcap.c
Line	991	991
Object	fprintf	fprintf

Code Snippet
File Name wireshark-2/text2pcap.c
Method text2pcap_cmdarg_err_cont(const char *msg_format, va_list ap)

```
....
991.        fprintf(stderr, "\n");
```

Use of Sizeof On a Pointer Type

Query Path:
CPP\Cx\CPP Low Visibility\Use of Sizeof On a Pointer Type Version:1
Description

Use of Sizeof On a Pointer Type\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=1
Status	New

	Source	Destination
File	wireshark-2/oids.c	wireshark-2/oids.c
Line	558	558
Object	sizeof	sizeof

Code Snippet
File Name wireshark-2/oids.c
Method static void register_mibs(void) {

```
....
558.        etta = g_array_new(FALSE,TRUE,sizeof(gint*));
```

Sizeof Pointer Argument

Query Path:
CPP\Cx\CPP Low Visibility\Sizeof Pointer Argument Version:0
Description

Sizeof Pointer Argument\Path 1:

Severity	Low
Result State	To Verify
Online Results	http://WIN-BA8RD5TJ8IG/CxWebClient/ViewerMain.aspx?scanid=1030045&projectid=30038&pathid=15
Status	New

	Source	Destination
File	wireshark-2/ws_pipe.c	wireshark-2/ws_pipe.c
Line	683	683
Object	pipeinsts	sizeof

Code Snippet
File Name wireshark-2/ws_pipe.c
Method ws_pipe_wait_for_pipe(HANDLE * pipe_handles, int num_pipe_handles, HANDLE pid)

```
....
683.      SecureZeroMemory(pipeinsts, sizeof(pipeinsts));
```

Buffer Overflow boundcpy WrongSizeParam

Risk

What might happen

Buffer overflow attacks, in their various forms, could allow an attacker to control certain areas of memory. Typically, this is used to overwrite data on the stack necessary for the program to function properly, such as code and memory addresses, though other forms of this attack exist. Exploiting this vulnerability can generally lead to system crashes, infinite loops, or even execution of arbitrary code.

Cause

How does it happen

Buffer Overflows can manifest in numerous different variations. In it's most basic form, the attack controls a buffer, which is then copied to a smaller buffer without size verification. Because the attacker's source buffer is larger than the program's target buffer, the attacker's data overwrites whatever is next on the stack, allowing the attacker to control program structures.

Alternatively, the vulnerability could be the result of improper bounds checking; exposing internal memory addresses outside of their valid scope; allowing the attacker to control the size of the target buffer; or various other forms.

General Recommendations

How to avoid it

- Always perform proper bounds checking before copying buffers or strings.
- Prefer to use safer functions and structures, e.g. safe string classes over `char*`, `strncpy` over `strcpy`, and so on.
- Consistently apply tests for the size of buffers.
- Do not return variable addresses outside the scope of their variables.

Source Code Examples

CPP

Overflowing Buffers

```
const int BUFFER_SIZE = 10;
char buffer[BUFFER_SIZE];

void copyStringToBuffer(char* inputString)
{
    strcpy(buffer, inputString);
}
```

Checked Buffers

```
const int BUFFER_SIZE = 10;
const int MAX_INPUT_SIZE = 256;
```

CHECKMARX

```
char buffer[BUFFER_SIZE];

void copyStringToBuffer(char* inputString)
{
    if (strnlen(inputString, MAX_INPUT_SIZE) < sizeof(buffer))
    {
        strncpy(buffer, inputString, sizeof(buffer));
    }
}
```

Integer Overflow

Risk

What might happen

Assigning large data types into smaller data types, without proper checks and explicit casting, will lead to undefined behavior and unintentional effects, such as data corruption (e.g. value wraparound, wherein maximum values become minimum values); system crashes; infinite loops; logic errors, such as bypassing of security mechanisms; or even buffer overflows leading to arbitrary code execution.

Cause

How does it happen

This flaw can occur when implicitly casting numerical data types of a larger size, into a variable with a data type of a smaller size. This forces the program to discard some bits of information from the number. Depending on how the numerical data types are stored in memory, this is often the bits with the highest value, causing substantial corruption of the stored number. Alternatively, the sign bit of a signed integer could be lost, completely reversing the intention of the number.

General Recommendations

How to avoid it

- Avoid casting larger data types to smaller types.
- Prefer promoting the target variable to a large enough data type.
- If downcasting is necessary, always check that values are valid and in range of the target type, before casting

Source Code Examples

CPP

Unsafe Downsize Casting

```
int unsafe_addition(short op1, int op2) {

    // op2 gets forced from int into a short
    short total = op1 + op2;

    return total;
}
```

Safer Use of Proper Data Types

```
int safe_addition(short op1, int op2) {

    // total variable is of type int, the largest type that is needed
    int total = 0;

    // check if total will overflow available integer size
    if (INT_MAX - abs(op2) > op1)
```

```
    {
        total = op1 + op2;
    }
    else
    {
        // instead of overflow, saturate (but this is not always a good thing)
        total = INT_MAX
    }

    return total;
}
```

Dangerous Functions

Risk

What might happen

Use of dangerous functions may expose varying risks associated with each particular function, with potential impact of improper usage of these functions varying significantly. The presence of such functions indicates a flaw in code maintenance policies and adherence to secure coding practices, in a way that has allowed introducing known dangerous code into the application.

Cause

How does it happen

A dangerous function has been identified within the code. Functions are often deemed dangerous to use for numerous reasons, as there are different sets of vulnerabilities associated with usage of such functions. For example, some string copy and concatenation functions are vulnerable to Buffer Overflow, Memory Disclosure, Denial of Service and more. Use of these functions is not recommended.

General Recommendations

How to avoid it

- Deploy a secure and recommended alternative to any functions that were identified as dangerous.
    - If no secure alternative is found, conduct further researching and testing to identify whether current usage successfully sanitizes and verifies values, and thus successfully avoids the use-cases for whom the function is indeed dangerous
- Conduct a periodical review of methods that are in use, to ensure that all external libraries and built-in functions are up-to-date and whose use has not been excluded from best secure coding practices.

Source Code Examples

CPP

Buffer Overflow in gets()

```
int main()
{
	char buf[10];

	printf("Please enter your name: ");
	gets(buf); // veryveryverylongname
	if (buf == ACCEPTED_NAME)
	{
		// Do something
	}
	return 0;
}
```

Safe reading from user

```
int main()
{
	char buf[10];

	printf("Please enter your name: ");
	fgets(buf, sizeof(buf), stdin); //setting the amount of bytes to read
	if (buf == ACCEPTED_NAME)
	{
		//Do something
	}
	return 0;
}
```

Unsafe function for string copy

```
int main(int argc, char* argv[])
{
	char buf[10];
	strcpy(buf, argv[1]); // overflow occurs when len(argv[1]) > 10 bytes

	return 0;
}
```

Safe string copy

```
int main(int argc, char* argv[])
{
	char buf[10];
	strncpy(buf, argv[1], sizeof(buf));
	buf[9]= '\0'; //strncpy doesn't NULL terminates

	return 0;
}
```

Unsafe format string

```
int main(int argc, char* argv[])
{
	printf(argv[1]); // If argv[1] contains a format token, such as %s,%x or %d, will cause
an access violation
	return 0;
}
```

Safe format string

```
int main(int argc, char* argv[])
{
      printf("%s", argv[1]); // Second parameter is not a formattable string

      return 0;
}
```

Use of Zero Initialized Pointer

Risk

What might happen

A null pointer dereference is likely to cause a run-time exception, a crash, or other unexpected behavior.

Cause

How does it happen

Variables which are declared without being assigned will implicitly retain a null value until they are assigned. The null value can also be explicitly set to a variable, to ensure clear out its contents. Since null is not really a value, it may not have object variables and methods, and any attempt to access contents of a null object, instead of verifying it is set beforehand, will result in a null pointer dereference exception.

General Recommendations

How to avoid it

- For any variable that is created, ensure all logic flows between declaration and use assign a non-null value to the variable first.
- Enforce null checks on any received variable or object before it is dereferenced, to ensure it does not contain a null assigned to it elsewhere.
- Consider the need to assign null values in order to overwrite initialized variables. Consider reassigning or releasing these variables instead.

Source Code Examples

CPP

Explicit NULL Dereference

```
char * input = NULL;
printf("%s", input);
```

Implicit NULL Dereference

```
char * input;
printf("%s", input);
```

Java

Explicit Null Dereference

```
Object o = null;
out.println(o.getClass());
```

Use of sizeof() on a Pointer Type

Weakness ID: 467 *(Weakness Variant)* **Status:** Draft

Description

Description Summary

The code calls sizeof() on a malloced pointer type, which always returns the wordsize/8. This can produce an unexpected result if the programmer intended to determine how much memory has been allocated.

Time of Introduction

- Implementation

Applicable Platforms

Languages

C

C++

Common Consequences

Scope	Effect
Integrity	This error can often cause one to allocate a buffer that is much smaller than what is needed, leading to resultant weaknesses such as buffer overflows.

Likelihood of Exploit

High

Demonstrative Examples

Example 1

Care should be taken to ensure sizeof returns the size of the data structure itself, and not the size of the pointer to the data structure.

In this example, sizeof(foo) returns the size of the pointer.

(Bad Code)

Example Languages: **C and C++**

```
double *foo;
...
foo = (double *)malloc(sizeof(foo));
```

In this example, sizeof(*foo) returns the size of the data structure and not the size of the pointer.

(Good Code)

Example Languages: **C and C++**

```
double *foo;
...
foo = (double *)malloc(sizeof(*foo));
```

Example 2

This example defines a fixed username and password. The AuthenticateUser() function is intended to accept a username and a password from an untrusted user, and check to ensure that it matches the username and password. If the username and password match, AuthenticateUser() is intended to indicate that authentication succeeded.

(Bad Code)

```
/* Ignore CWE-259 (hard-coded password) and CWE-309 (use of password system for authentication) for this example. */

char *username = "admin";
char *pass = "password";

int AuthenticateUser(char *inUser, char *inPass) {
```

```
printf("Sizeof username = %d\n", sizeof(username));
printf("Sizeof pass = %d\n", sizeof(pass));

if (strncmp(username, inUser, sizeof(username))) {
printf("Auth failure of username using sizeof\n");
return(AUTH_FAIL);
}
/* Because of CWE-467, the sizeof returns 4 on many platforms and architectures. */
if (! strncmp(pass, inPass, sizeof(pass))) {
printf("Auth success of password using sizeof\n");
return(AUTH_SUCCESS);
}
else {
printf("Auth fail of password using sizeof\n");
return(AUTH_FAIL);
}
}

int main (int argc, char **argv)
{
int authResult;

if (argc < 3) {
ExitError("Usage: Provide a username and password");
}
authResult = AuthenticateUser(argv[1], argv[2]);
if (authResult != AUTH_SUCCESS) {
ExitError("Authentication failed");
}
else {
DoAuthenticatedTask(argv[1]);
}
}
```

In AuthenticateUser(), because sizeof() is applied to a parameter with an array type, the sizeof() call might return 4 on many modern architectures. As a result, the strncmp() call only checks the first four characters of the input password, resulting in a partial comparison (CWE-187), leading to improper authentication (CWE-287).

Because of the partial comparison, any of these passwords would still cause authentication to succeed for the "admin" user:

(Attack)

```
pass5
passABCDEFGH
passWORD
```

Because only 4 characters are checked, this significantly reduces the search space for an attacker, making brute force attacks more feasible.

The same problem also applies to the username, so values such as "adminXYZ" and "administrator" will succeed for the username.

Potential Mitigations

Phase: Implementation

Use expressions such as "sizeof(*pointer)" instead of "sizeof(pointer)", unless you intend to run sizeof() on a pointer type to gain some platform independence or if you are allocating a variable on the stack.

Other Notes

The use of sizeof() on a pointer can sometimes generate useful information. An obvious case is to find out the wordsize on a platform. More often than not, the appearance of sizeof(pointer) indicates a bug.

Weakness Ordinalities

Ordinality	Description
Primary	*(where the weakness exists independent of other weaknesses)*

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Category	465	Pointer Issues	**Development Concepts (primary)699**
ChildOf	Weakness Class	682	Incorrect Calculation	**Research Concepts (primary)1000**
ChildOf	Category	737	CERT C Secure Coding Section 03 - Expressions (EXP)	**Weaknesses Addressed by the CERT C Secure Coding Standard (primary)734**
ChildOf	Category	740	CERT C Secure Coding Section 06 - Arrays (ARR)	Weaknesses Addressed by the CERT C Secure Coding Standard734
CanPrecede	Weakness Base	131	Incorrect Calculation of Buffer Size	Research Concepts1000

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
CLASP			Use of sizeof() on a pointer type
CERT C Secure Coding	ARR01-C		Do not apply the sizeof operator to a pointer when taking the size of an array
CERT C Secure Coding	EXP01-C		Do not take the size of a pointer to determine the size of the pointed-to type

White Box Definitions

A weakness where code path has:

1. end statement that passes an identity of a dynamically allocated memory resource to a sizeof operator
2. start statement that allocates the dynamically allocated memory resource

References

Robert Seacord. "EXP01-A. Do not take the sizeof a pointer to determine the size of a type". <https://www.securecoding.cert.org/confluence/display/seccode/EXP01-A.+Do+not+take+the+sizeof+a+pointer+to+determine+the+size+of+a+type>.

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		
2008-08-01		KDM Analytics	External
	added/updated white box definitions		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Common Consequences, Relationships, Other Notes, Taxonomy Mappings, Weakness Ordinalities		
2008-11-24	CWE Content Team	MITRE	Internal
	updated Relationships, Taxonomy Mappings		
2009-03-10	CWE Content Team	MITRE	Internal
	updated Demonstrative Examples		
2009-12-28	CWE Content Team	MITRE	Internal
	updated Demonstrative Examples		
2010-02-16	CWE Content Team	MITRE	Internal
	updated Relationships		

BACK TO TOP

Use of sizeof() on a Pointer Type

Weakness ID: 467 *(Weakness Variant)* **Status:** Draft

Description

Description Summary

The code calls sizeof() on a malloced pointer type, which always returns the wordsize/8. This can produce an unexpected result if the programmer intended to determine how much memory has been allocated.

Time of Introduction

- Implementation

Applicable Platforms

Languages

C

C++

Common Consequences

Scope	Effect
Integrity	This error can often cause one to allocate a buffer that is much smaller than what is needed, leading to resultant weaknesses such as buffer overflows.

Likelihood of Exploit

High

Demonstrative Examples

Example 1

Care should be taken to ensure sizeof returns the size of the data structure itself, and not the size of the pointer to the data structure.

In this example, sizeof(foo) returns the size of the pointer.

(Bad Code)

Example Languages: **C and C++**

```
double *foo;
...
foo = (double *)malloc(sizeof(foo));
```

In this example, sizeof(*foo) returns the size of the data structure and not the size of the pointer.

(Good Code)

Example Languages: **C and C++**

```
double *foo;
...
foo = (double *)malloc(sizeof(*foo));
```

Example 2

This example defines a fixed username and password. The AuthenticateUser() function is intended to accept a username and a password from an untrusted user, and check to ensure that it matches the username and password. If the username and password match, AuthenticateUser() is intended to indicate that authentication succeeded.

(Bad Code)

```
/* Ignore CWE-259 (hard-coded password) and CWE-309 (use of password system for authentication) for this example. */

char *username = "admin";
char *pass = "password";

int AuthenticateUser(char *inUser, char *inPass) {
```

```
printf("Sizeof username = %d\n", sizeof(username));
printf("Sizeof pass = %d\n", sizeof(pass));

if (strncmp(username, inUser, sizeof(username))) {
printf("Auth failure of username using sizeof\n");
return(AUTH_FAIL);
}
/* Because of CWE-467, the sizeof returns 4 on many platforms and architectures. */
if (! strncmp(pass, inPass, sizeof(pass))) {
printf("Auth success of password using sizeof\n");
return(AUTH_SUCCESS);
}
else {
printf("Auth fail of password using sizeof\n");
return(AUTH_FAIL);
}
}

int main (int argc, char **argv)
{
int authResult;

if (argc < 3) {
ExitError("Usage: Provide a username and password");
}
authResult = AuthenticateUser(argv[1], argv[2]);
if (authResult != AUTH_SUCCESS) {
ExitError("Authentication failed");
}
else {
DoAuthenticatedTask(argv[1]);
}
}
```

In AuthenticateUser(), because sizeof() is applied to a parameter with an array type, the sizeof() call might return 4 on many modern architectures. As a result, the strncmp() call only checks the first four characters of the input password, resulting in a partial comparison (CWE-187), leading to improper authentication (CWE-287).

Because of the partial comparison, any of these passwords would still cause authentication to succeed for the "admin" user:

(Attack)

```
pass5
passABCDEFGH
passWORD
```

Because only 4 characters are checked, this significantly reduces the search space for an attacker, making brute force attacks more feasible.

The same problem also applies to the username, so values such as "adminXYZ" and "administrator" will succeed for the username.

Potential Mitigations

Phase: Implementation

Use expressions such as "sizeof(*pointer)" instead of "sizeof(pointer)", unless you intend to run sizeof() on a pointer type to gain some platform independence or if you are allocating a variable on the stack.

Other Notes

The use of sizeof() on a pointer can sometimes generate useful information. An obvious case is to find out the wordsize on a platform. More often than not, the appearance of sizeof(pointer) indicates a bug.

Weakness Ordinalities

Ordinality	Description
Primary	*(where the weakness exists independent of other weaknesses)*

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Category	465	Pointer Issues	**Development Concepts (primary)699**
ChildOf	Weakness Class	682	Incorrect Calculation	**Research Concepts (primary)1000**
ChildOf	Category	737	CERT C Secure Coding Section 03 - Expressions (EXP)	**Weaknesses Addressed by the CERT C Secure Coding Standard (primary)734**
ChildOf	Category	740	CERT C Secure Coding Section 06 - Arrays (ARR)	Weaknesses Addressed by the CERT C Secure Coding Standard734
CanPrecede	Weakness Base	131	Incorrect Calculation of Buffer Size	Research Concepts1000

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
CLASP			Use of sizeof() on a pointer type
CERT C Secure Coding	ARR01-C		Do not apply the sizeof operator to a pointer when taking the size of an array
CERT C Secure Coding	EXP01-C		Do not take the size of a pointer to determine the size of the pointed-to type

White Box Definitions

A weakness where code path has:

1. end statement that passes an identity of a dynamically allocated memory resource to a sizeof operator
2. start statement that allocates the dynamically allocated memory resource

References

Robert Seacord. "EXP01-A. Do not take the sizeof a pointer to determine the size of a type". <https://www.securecoding.cert.org/confluence/display/seccode/EXP01-A.+Do+not+take+the+sizeof+a+pointer+to+determine+the+size+of+a+type>.

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		
2008-08-01		KDM Analytics	External
	added/updated white box definitions		
2008-09-08	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Common Consequences, Relationships, Other Notes, Taxonomy Mappings, Weakness Ordinalities		
2008-11-24	CWE Content Team	MITRE	Internal
	updated Relationships, Taxonomy Mappings		
2009-03-10	CWE Content Team	MITRE	Internal
	updated Demonstrative Examples		
2009-12-28	CWE Content Team	MITRE	Internal
	updated Demonstrative Examples		
2010-02-16	CWE Content Team	MITRE	Internal
	updated Relationships		

BACK TO TOP

Improper Access Control (Authorization)

Weakness ID: 285 *(Weakness Class)* **Status:** Draft

Description

Description Summary

The software does not perform or incorrectly performs access control checks across all potential execution paths.

Extended Description

When access control checks are not applied consistently - or not at all - users are able to access data or perform actions that they should not be allowed to perform. This can lead to a wide range of problems, including information leaks, denial of service, and arbitrary code execution.

Alternate Terms

AuthZ:	"AuthZ" is typically used as an abbreviation of "authorization" within the web application security community. It is also distinct from "AuthC," which is an abbreviation of "authentication." The use of "Auth" as an abbreviation is discouraged, since it could be used for either authentication or authorization.

Time of Introduction

- Architecture and Design
- Implementation
- Operation

Applicable Platforms

Languages

Language-independent

Technology Classes

Web-Server: *(Often)*

Database-Server: *(Often)*

Modes of Introduction

A developer may introduce authorization weaknesses because of a lack of understanding about the underlying technologies. For example, a developer may assume that attackers cannot modify certain inputs such as headers or cookies.

Authorization weaknesses may arise when a single-user application is ported to a multi-user environment.

Common Consequences

Scope	Effect
Confidentiality	An attacker could read sensitive data, either by reading the data directly from a data store that is not properly restricted, or by accessing insufficiently-protected, privileged functionality to read the data.
Integrity	An attacker could modify sensitive data, either by writing the data directly to a data store that is not properly restricted, or by accessing insufficiently-protected, privileged functionality to write the data.
Integrity	An attacker could gain privileges by modifying or reading critical data directly, or by accessing insufficiently-protected, privileged functionality.

Likelihood of Exploit

High

Detection Methods

Automated Static Analysis

Automated static analysis is useful for detecting commonly-used idioms for authorization. A tool may be able to analyze related configuration files, such as .htaccess in Apache web servers, or detect the usage of commonly-used authorization libraries.

Generally, automated static analysis tools have difficulty detecting custom authorization schemes. In addition, the software's design may include some functionality that is accessible to any user and does not require an authorization check; an automated technique that detects the absence of authorization may report false positives.

Effectiveness: Limited

Automated Dynamic Analysis

Automated dynamic analysis may find many or all possible interfaces that do not require authorization, but manual analysis is required to determine if the lack of authorization violates business logic

Manual Analysis

This weakness can be detected using tools and techniques that require manual (human) analysis, such as penetration testing, threat modeling, and interactive tools that allow the tester to record and modify an active session.

Specifically, manual static analysis is useful for evaluating the correctness of custom authorization mechanisms.

Effectiveness: Moderate

These may be more effective than strictly automated techniques. This is especially the case with weaknesses that are related to design and business rules. However, manual efforts might not achieve desired code coverage within limited time constraints.

Demonstrative Examples

Example 1

The following program could be part of a bulletin board system that allows users to send private messages to each other. This program intends to authenticate the user before deciding whether a private message should be displayed. Assume that LookupMessageObject() ensures that the $id argument is numeric, constructs a filename based on that id, and reads the message details from that file. Also assume that the program stores all private messages for all users in the same directory.

(Bad Code)

Example Language: **Perl**

```perl
sub DisplayPrivateMessage {
my($id) = @_;
my $Message = LookupMessageObject($id);
print "From: " . encodeHTML($Message->{from}) . "<br>\n";
print "Subject: " . encodeHTML($Message->{subject}) . "\n";
print "<hr>\n";
print "Body: " . encodeHTML($Message->{body}) . "\n";
}

my $q = new CGI;
# For purposes of this example, assume that CWE-309 and
# CWE-523 do not apply.
if (! AuthenticateUser($q->param('username'), $q->param('password'))) {
ExitError("invalid username or password");
}

my $id = $q->param('id');
DisplayPrivateMessage($id);
```

While the program properly exits if authentication fails, it does not ensure that the message is addressed to the user. As a result, an authenticated attacker could provide any arbitrary identifier and read private messages that were intended for other users.

One way to avoid this problem would be to ensure that the "to" field in the message object matches the username of the authenticated user.

Observed Examples

Reference	Description
CVE-2009-3168	Web application does not restrict access to admin scripts, allowing authenticated users to reset administrative passwords.

CVE-2009-2960	Web application does not restrict access to admin scripts, allowing authenticated users to modify passwords of other users.
CVE-2009-3597	Web application stores database file under the web root with insufficient access control (CWE-219), allowing direct request.
CVE-2009-2282	Terminal server does not check authorization for guest access.
CVE-2009-3230	Database server does not use appropriate privileges for certain sensitive operations.
CVE-2009-2213	Gateway uses default "Allow" configuration for its authorization settings.
CVE-2009-0034	Chain: product does not properly interpret a configuration option for a system group, allowing users to gain privileges.
CVE-2008-6123	Chain: SNMP product does not properly parse a configuration option for which hosts are allowed to connect, allowing unauthorized IP addresses to connect.
CVE-2008-5027	System monitoring software allows users to bypass authorization by creating custom forms.
CVE-2008-7109	Chain: reliance on client-side security (CWE-602) allows attackers to bypass authorization using a custom client.
CVE-2008-3424	Chain: product does not properly handle wildcards in an authorization policy list, allowing unintended access.
CVE-2009-3781	Content management system does not check access permissions for private files, allowing others to view those files.
CVE-2008-4577	ACL-based protection mechanism treats negative access rights as if they are positive, allowing bypass of intended restrictions.
CVE-2008-6548	Product does not check the ACL of a page accessed using an "include" directive, allowing attackers to read unauthorized files.
CVE-2007-2925	Default ACL list for a DNS server does not set certain ACLs, allowing unauthorized DNS queries.
CVE-2006-6679	Product relies on the X-Forwarded-For HTTP header for authorization, allowing unintended access by spoofing the header.
CVE-2005-3623	OS kernel does not check for a certain privilege before setting ACLs for files.
CVE-2005-2801	Chain: file-system code performs an incorrect comparison (CWE-697), preventing defauls ACLs from being properly applied.
CVE-2001-1155	Chain: product does not properly check the result of a reverse DNS lookup because of operator precedence (CWE-783), allowing bypass of DNS-based access restrictions.

Potential Mitigations

Phase: Architecture and Design

Divide your application into anonymous, normal, privileged, and administrative areas. Reduce the attack surface by carefully mapping roles with data and functionality. Use role-based access control (RBAC) to enforce the roles at the appropriate boundaries.

Note that this approach may not protect against horizontal authorization, i.e., it will not protect a user from attacking others with the same role.

Phase: Architecture and Design

Ensure that you perform access control checks related to your business logic. These checks may be different than the access control checks that you apply to more generic resources such as files, connections, processes, memory, and database records. For example, a database may restrict access for medical records to a specific database user, but each record might only be intended to be accessible to the patient and the patient's doctor.

Phase: Architecture and Design

Strategy: Libraries or Frameworks

Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid.

For example, consider using authorization frameworks such as the JAAS Authorization Framework and the OWASP ESAPI Access Control feature.

Phase: Architecture and Design

For web applications, make sure that the access control mechanism is enforced correctly at the server side on every page. Users should not be able to access any unauthorized functionality or information by simply requesting direct access to that page.

One way to do this is to ensure that all pages containing sensitive information are not cached, and that all such pages restrict access to requests that are accompanied by an active and authenticated session token associated with a user who has the required permissions to access that page.

Phases: System Configuration; Installation

Use the access control capabilities of your operating system and server environment and define your access control lists accordingly. Use a "default deny" policy when defining these ACLs.

Relationships

Nature	Type	ID	Name	View(s) this relationship pertains to
ChildOf	Category	254	Security Features	**Seven Pernicious Kingdoms (primary)700**
ChildOf	Weakness Class	284	Access Control (Authorization) Issues	**Development Concepts (primary)699** **Research Concepts (primary)1000**
ChildOf	Category	721	OWASP Top Ten 2007 Category A10 - Failure to Restrict URL Access	**Weaknesses in OWASP Top Ten (2007) (primary)629**
ChildOf	Category	723	OWASP Top Ten 2004 Category A2 - Broken Access Control	**Weaknesses in OWASP Top Ten (2004) (primary)711**
ChildOf	Category	753	2009 Top 25 - Porous Defenses	**Weaknesses in the 2009 CWE/SANS Top 25 Most Dangerous Programming Errors (primary)750**
ChildOf	Category	803	2010 Top 25 - Porous Defenses	**Weaknesses in the 2010 CWE/SANS Top 25 Most Dangerous Programming Errors (primary)800**
ParentOf	Weakness Variant	219	Sensitive Data Under Web Root	**Research Concepts (primary)1000**
ParentOf	Weakness Base	551	Incorrect Behavior Order: Authorization Before Parsing and Canonicalization	**Development Concepts (primary)699** Research Concepts1000
ParentOf	Weakness Class	638	Failure to Use Complete Mediation	Research Concepts1000
ParentOf	Weakness Base	804	Guessable CAPTCHA	**Development Concepts (primary)699** **Research Concepts (primary)1000**

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
7 Pernicious Kingdoms			Missing Access Control
OWASP Top Ten 2007	A10	CWE More Specific	Failure to Restrict URL Access
OWASP Top Ten 2004	A2	CWE More Specific	Broken Access Control

Related Attack Patterns

CAPEC-ID	Attack Pattern Name	*(CAPEC Version: 1.5)*
1	Accessing Functionality Not Properly Constrained by ACLs	
13	Subverting Environment Variable Values	

17	Accessing, Modifying or Executing Executable Files
87	Forceful Browsing
39	Manipulating Opaque Client-based Data Tokens
45	Buffer Overflow via Symbolic Links
51	Poison Web Service Registry
59	Session Credential Falsification through Prediction
60	Reusing Session IDs (aka Session Replay)
77	Manipulating User-Controlled Variables
76	Manipulating Input to File System Calls
104	Cross Zone Scripting

References

NIST. "Role Based Access Control and Role Based Security". <http://csrc.nist.gov/groups/SNS/rbac/>.

[REF-11] M. Howard and D. LeBlanc. "Writing Secure Code". Chapter 4, "Authorization" Page 114; Chapter 6, "Determining Appropriate Access Control" Page 171. 2nd Edition. Microsoft. 2002.

Content History

Submissions

Submission Date	Submitter	Organization	Source
	7 Pernicious Kingdoms		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 Relationships, Other Notes, Taxonomy Mappings		
2009-01-12	CWE Content Team	MITRE	Internal
	updated Common Consequences, Description, Likelihood of Exploit, Name, Other Notes, Potential Mitigations, References, Relationships		
2009-03-10	CWE Content Team	MITRE	Internal
	updated Potential Mitigations		
2009-05-27	CWE Content Team	MITRE	Internal
	updated Description, Related Attack Patterns		
2009-07-27	CWE Content Team	MITRE	Internal
	updated Relationships		
2009-10-29	CWE Content Team	MITRE	Internal
	updated Type		
2009-12-28	CWE Content Team	MITRE	Internal
	updated Applicable Platforms, Common Consequences, Demonstrative Examples, Detection Factors, Modes of Introduction, Observed Examples, Relationships		
2010-02-16	CWE Content Team	MITRE	Internal
	updated Alternate Terms, Detection Factors, Potential Mitigations, References, Relationships		
2010-04-05	CWE Content Team	MITRE	Internal
	updated Potential Mitigations		

Previous Entry Names

Change Date	Previous Entry Name
2009-01-12	Missing or Inconsistent Access Control

BACK TO TOP

Scanned Languages

Language	Hash Number	Change Date
CPP	4541647240435660	6/19/2024
Common	0105849645654507	6/19/2024