**Vulnerability Assessment Report**

**For**



**KTC\_burp**

**April 18, 2022**

**Document Security Level:** Confidential

**Document Version:** 1.0

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Edit Report** | **Editor** |
| 1.0 | April 18, 2022 | Creation | INET Managed Services |

|  |  |  |  |
| --- | --- | --- | --- |
| KTC\_burp | | INET Managed Services CO., LTD. | |
| Name |  | **Name** |  |
| Position |  | **Position** |  |
| Tel |  | **Tel** |  |
| Signature |  | **Signature** |  |

**Table of Contents**

[1. Restrictions on disclosure and use of information 4](#_Toc98504828)

[2. Operation Method 5](#_Toc98504829)

[3. Project Scope 6](#_Toc98504830)

[3.1 Web Application Vulnerability Assessment 6](#_Toc98504831)

[4. Testing Tools 7](#_Toc98504832)

[5. Web Application Vulnerability Assessment 7](#_Toc98504833)

[6.1 Target Information 8](#_Toc98504834)

[6.2 Executive summary 9](#_Toc98504835)

[6.2.1 Summary Vulnerability by Severity 9](#_Toc98504836)

[6.2.2 Vulnerability by Target 9](#_Toc98504837)

[6.3 Web Application Vulnerability Detail 10](#_Toc98504838)

[6. Port Discovery 11](#_Toc98504839)

[7. Appendix 12](#_Toc98504840)

[8.1 About Nessus 12](#_Toc98504841)

[8.1.1 Nessus vulnerabilities 12](#_Toc98504842)

[8.1.2 Nessus risk score 12](#_Toc98504843)

[8.2 About Burp Suite's web vulnerability scanner 14](#_Toc98504844)

[8.2.1 Burp Suite's web vulnerability scanner risk score 14](#_Toc98504845)

# Restrictions on disclosure and use of information

Restriction on Disclosure and Use of Confidential Information. The Executive understands and agrees that the Confidential Information constitutes an asset of the Company and its affiliated entities and may not be converted to the Executive's own use. Accordingly, the Executive hereby agrees that the Executive shall not, directly, or indirectly, at any time, reveal, divulge, or disclose to any Person not expressly authorized by the Company any Confidential Information, and the Executive shall not, directly, or indirectly, use or make use of any Confidential Information in connection with any business activity other than that of the Company. The parties acknowledge and agree that this Agreement is not intended to, and does not, alter either the Company's rights or the Executive's obligations under any state or federal statutory or common law regarding trade secrets and unfair trade.

# Operation Method

* 1. Posture Review
  2. Information Gathering
  3. Enumeration
  4. Vulnerability Assessment
  5. Analyze & Evaluate Risk Value
  6. Report



Figure 1: Operation Method

# Project Scope

## **3.1 Web Application Vulnerability Assessment**

**Target / IP Address:**

| **No.** | **Domain / Server Name** | **Public IP Address** | **Private IP Address** | **OS/Model** | **Functions** | **Public Assessment** | **Private Assessment** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | - | - | - | - | - | ✓ | - |

# Testing Tools

|  |  |
| --- | --- |
| **Tool Name** | **Testing Type** |
| Nmap | Host and Service Discovery |
| Nessus Professional | Infrastructure Vulnerability Assessment |
| Burp Suite's web vulnerability scanner | Web Application Vulnerability Assessment |

# Web Application Vulnerability Assessment

**Vulnerability Assessment from Public Access (for public target)**

**Testing date:** <<Date SCAN>>

**Tester IP Address:** <<IP Target>>

Diagram

Description automatically generated

Figure 5: Vulnerability Assessment from Public Access

## **5.1 Target Information**

| **No.** | **Domain / Server Name** | **IP Address** | **OS/Model** | **Port** |
| --- | --- | --- | --- | --- |
| 1 | db.sac.or.th | 202.29.72.42 | - | TCP: 80, 443, 8008, 8010 |
| 2 | pmop.sac.or.th | 202.29.72.51 | - | TCP: 80, 443, 8008, 8010 |
| 3 | communityarchive.sac.or.th | 202.29.72.49 | - | TCP: 80, 443, 8008, 8010 |
| 4 | www.sac.or.th | 83.118.33.130 | - | TCP: 80, 443, 8008, 8010 |
| 5 | mis.sac.or.th | 83.118.33.135 | - | TCP: 80, 443, 8008, 8010 |
| 6 | tv.sac.or.th | 83.118.33.148 | - | TCP: 80, 443, 8008, 8010 |
| 7 | lib.sac.or.th | 83.118.33.131 | - | TCP: 80, 443, 8008, 8010 |

## **5.2 Executive summary**

The purpose of this activity is to find the vulnerability on the target web application.

### **5.2.1 Summary Vulnerability by Severity**

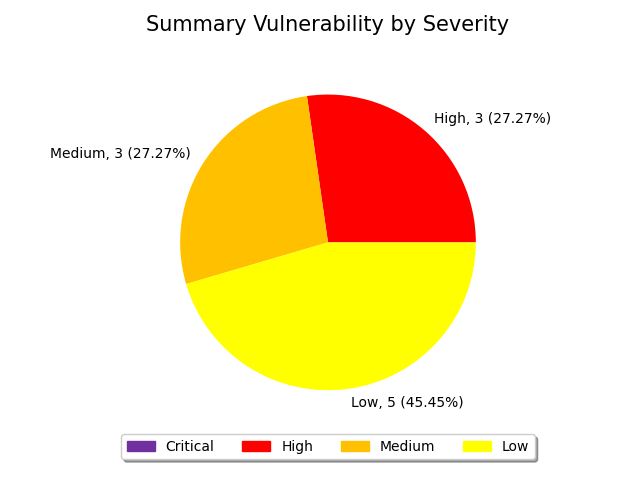


Figure 6: Summary by Severity of Web Application Vulnerability Assessment

### **5.2.2 Vulnerability by Target**

| **No.** | **Domain/Server Name** | **IP Address** | **Critical** | **High** | **Medium** | **Low** | **Total** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | http://10.150.1.21:30000 | 10.150.1.21 | 0 | 3 | 0 | 1 | 4 |
| 2 | http://34.96.83.158 | 34.96.83.158 | 0 | 0 | 0 | 1 | 1 |
| 3 | http://uatmobile.ktcgroup.dev | 34.96.83.158 | 0 | 0 | 0 | 1 | 1 |
| 4 | https://10.150.1.21:30000 | 10.150.1.21 | 0 | 0 | 1 | 2 | 3 |
| 5 | https://34.96.83.158 | 34.96.83.158 | 0 | 0 | 1 | 0 | 1 |
| 6 | https://uatmobile.ktcgroup.dev | 34.96.83.158 | 0 | 0 | 1 | 0 | 1 |
| **Total** | | | 0 | 3 | 3 | 5 | 11 |

## **5.3 Web Application Vulnerability Detail**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID.** | 1 | **Finding** | External service interaction (DNS) |
| **Severity** | **High** | **Port** | 30000 |
| **Target** | http://10.150.1.21:30000/robots.txt http://10.150.1.21:30000/ | | |
| **Detail** | External service interaction arises when it is possible to induce an application to interact with an arbitrary external service, such as a web or mail server. The ability to trigger arbitrary external service interactions does not constitute a vulnerability in its own right, and in some cases might even be the intended behavior of the application. However, in many cases, it can indicate a vulnerability with serious consequences. In cases where DNS-based interactions can be triggered, it is normally possible to trigger interactions using other service types, and these are reported as separate issues.   If a payload that specifies a particular service type (e.g. a URL) triggers only a DNS-based interaction,   then this strongly indicates that the application attempted to connect using that other service,   but was prevented from doing so by egress filters in place at the network layer. The ability to send requests to other systems can allow the vulnerable server to be used as an attack proxy.  By submitting suitable payloads, an attacker can cause the application server to attack other systems that it can interact with.   This may include public third-party systems, internal systems within the same organization, or services available on the local loopback adapter of the application server itself.   Depending on the network architecture, this may expose highly vulnerable internal services that are not otherwise accessible to external attackers. | | |
| **Solution** | You should review the purpose and intended use of the relevant application functionality,   and determine whether the ability to trigger arbitrary external service interactions is intended behavior.   If so, you should be aware of the types of attacks that can be performed via this behavior and take appropriate measures.   These measures might include blocking network access from the application server to other internal systems, and hardening the application server itself to remove any services available on the local loopback adapter. If the ability to trigger arbitrary external service interactions is not intended behavior, then you should implement a whitelist of permitted services and hosts, and block any interactions that do not appear on this whitelist.  Out-of-Band Application Security Testing (OAST) is highly effective at uncovering high-risk features, to the point where finding the root cause of an interaction can be quite challenging. To find the source of an external service interaction, try to identify whether it is triggered by specific application functionality, or occurs indiscriminately on all requests. If it occurs on all endpoints, a front-end CDN or application firewall may be responsible, or a back-end analytics system parsing server logs. In some cases, interactions may originate from third-party systems; for example, a HTTP request may trigger a poisoned email which passes through a link-scanner on its way to the recipient. | | |
| **Remark** | https://portswigger.net/blog/introducing-burp-collaborator https://portswigger.net/burp/application-security-testing/oast https://portswigger.net/research/cracking-the-lens-targeting-https-hidden-attack-surface | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **ID.** | 2 | **Finding** | TLS certificate |
| **Severity** | **Medium** | **Port** | 443 |
| **Target** | https://10.150.1.21:30000/ https://34.96.83.158/ https://uatmobile.ktcgroup.dev/ | | |
| **Detail** | TLS (or SSL) helps to protect the confidentiality and integrity of information in transit between the browser and server, and to provide authentication of the server's identity. To serve this purpose, the server must present an TLS certificate that is valid for the server's hostname, is issued by a trusted authority and is valid for the current date. If any one of these requirements is not met, TLS connections to the server will not provide the full protection for which TLS is designed. It should be noted that various attacks exist against TLS in general, and in the context of HTTPS web connections in particular. It may be possible for a determined and suitably-positioned attacker to compromise TLS connections without user detection even when a valid TLS certificate is used. | | |
| **Solution** |  | | |
| **Remark** | https://wiki.mozilla.org/Security/Server\_Side\_TLS | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **ID.** | 3 | **Finding** | Strict transport security not enforced |
| **Severity** | **Low** | **Port** | 30000 |
| **Target** | https://10.150.1.21:30000/robots.txt | | |
| **Detail** | The application fails to prevent users from connecting to it over unencrypted connections. An attacker able to modify a legitimate user's network traffic could bypass the application's use of SSL/TLS encryption, and use the application as a platform for attacks against its users. This attack is performed by rewriting HTTPS links as HTTP, so that if a targeted user follows a link to the site from an HTTP page, their browser never attempts to use an encrypted connection. The sslstrip tool automates this process.   To exploit this vulnerability, an attacker must be suitably positioned to intercept and modify the victim's network traffic.This scenario typically occurs when a client communicates with the server over an insecure connection such as public Wi-Fi, or a corporate or home network that is shared with a compromised computer. Common defenses such as switched networks are not sufficient to prevent this. An attacker situated in the user's ISP or the application's hosting infrastructure could also perform this attack. Note that an advanced adversary could potentially target any connection made over the Internet's core infrastructure. | | |
| **Solution** | The application should instruct web browsers to only access the application using HTTPS. To do this, enable HTTP Strict Transport Security (HSTS) by adding a response header with the name 'Strict-Transport-Security' and the value 'max-age=expireTime', where expireTime is the time in seconds that browsers should remember that the site should only be accessed using HTTPS. Consider adding the 'includeSubDomains' flag if appropriate. Note that because HSTS is a "trust on first use" (TOFU) protocol, a user who has never accessed the application will never have seen the HSTS header, and will therefore still be vulnerable to SSL stripping attacks. To mitigate this risk, you can optionally add the 'preload' flag to the HSTS header, and submit the domain for review by browser vendors. | | |
| **Remark** | https://developer.mozilla.org/en-US/docs/Web/Security/HTTP\_strict\_transport\_security https://github.com/moxie0/sslstrip https://hstspreload.appspot.com/ | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **ID.** | 4 | **Finding** | Vulnerable JavaScript dependency |
| **Severity** | **Low** | **Port** | 30000 |
| **Target** | https://10.150.1.21:30000/static/js/30.d8e9e774d799fb50c50b.js | | |
| **Detail** | The use of third-party JavaScript libraries can introduce a range of DOM-based vulnerabilities, including some that can be used to hijack user accounts like DOM-XSS.     Common JavaScript libraries typically enjoy the benefit of being heavily audited. This may mean that bugs are quickly identified and patched upstream, resulting in a steady stream of security updates that need to be applied. Although it may be tempting to ignore updates, using a library with missing security patches can make your website exceptionally easy to exploit. Therefore, it's important to ensure that any available security updates are applied promptly.    Some library vulnerabilities expose every application that imports the library, but others only affect applications that use certain library features. Accurately identifying which library vulnerabilities apply to your website can be difficult, so we recommend applying all available security updates regardless. | | |
| **Solution** | Develop a patch-management strategy to ensure that security updates are promptly applied to all third-party libraries in your application. Also, consider reducing your attack surface by removing any libraries that are no longer in use. | | |
| **Remark** |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **ID.** | 5 | **Finding** | Unencrypted communications |
| **Severity** | **Low** | **Port** | 80 |
| **Target** | http://10.150.1.21:30000/ http://34.96.83.158/ http://uatmobile.ktcgroup.dev/ | | |
| **Detail** | The application allows users to connect to it over unencrypted connections. An attacker suitably positioned to view a legitimate user's network traffic could record and monitor their interactions with the application and obtain any information the user supplies. Furthermore, an attacker able to modify traffic could use the application as a platform for attacks against its users and third-party websites. Unencrypted connections have been exploited by ISPs and governments to track users, and to inject adverts and malicious JavaScript. Due to these concerns, web browser vendors are planning to visually flag unencrypted connections as hazardous.  To exploit this vulnerability, an attacker must be suitably positioned to eavesdrop on the victim's network traffic. This scenario typically occurs when a client communicates with the server over an insecure connection such as public Wi-Fi, or a corporate or home network that is shared with a compromised computer. Common defenses such as switched networks are not sufficient to prevent this. An attacker situated in the user's ISP or the application's hosting infrastructure could also perform this attack. Note that an advanced adversary could potentially target any connection made over the Internet's core infrastructure.  Please note that using a mixture of encrypted and unencrypted communications is an ineffective defense against active attackers, because they can easily remove references to encrypted resources when these references are transmitted over an unencrypted connection. | | |
| **Solution** | Applications should use transport-level encryption (SSL/TLS) to protect all communications passing between the client and the server. The Strict-Transport-Security HTTP header should be used to ensure that clients refuse to access the server over an insecure connection. | | |
| **Remark** | https://www.chromium.org/Home/chromium-security/marking-http-as-non-secure https://wiki.mozilla.org/Security/Server\_Side\_TLS https://developer.mozilla.org/en-US/docs/Web/Security/HTTP\_strict\_transport\_security | | |



# Port Discovery

| **Port** | **Protocol** | **Service** |
| --- | --- | --- |
| 53 | tcp | domain |
| 80 | tcp | http |
| 443 | tcp | https |
| 8008 | tcp | http |
| 8010 | tcp | xmpp |

# Appendix

## **7.1 About Nessus**

Nessus is a proprietary vulnerability scanner developed by Tenable, Inc. Nessus is trusted by more than 30,000 organizations worldwide as one of the most widely deployed security technologies on the planet - and the gold standard for vulnerability assessment.

Reference: https://www.tenable.com/products/nessus

### **7.1.1 Nessus vulnerabilities**

As information about new vulnerabilities are discovered and released into the public domain, Tenable, Inc. research staff designs programs to enable Nessus to detect them. These programs are named plugins, and are written in the Nessus proprietary scripting language, called Nessus Attack Scripting Language (NASL). Plugins contain vulnerability information, a generic set of remediation actions, and the algorithm to test for the presence of the security issue.

Reference: https://www.tenable.com/plugins

### **7.1.2 Nessus risk score**

There are four risk levels in this document: Critical, High, Medium, and Low. There are methods for determining the risk level. Based on the Common Vulnerability Scoring System (CVSS), a standard for assessing the severity of vulnerabilities in computer systems. Regarded by the NIAC (National Infrastructure Advisory Council), expert assessments are measured in a range of 0 – 10

| **Severity** | **Description** | **Score** |
| --- | --- | --- |
| Critical | Vulnerabilities that score in the critical range usually have most of the following characteristics:   * Exploitation of the vulnerability likely results in root-level compromise of servers or infrastructure devices. * Exploitation is usually straightforward, in the sense that the attacker does not need any special authentication credentials or knowledge about individual victims, and does not need to persuade a target user, for example via social engineering, into performing any special functions.   For critical vulnerabilities, is advised that you patch or upgrade as soon as possible, unless you have other mitigating measures in place. For example, a mitigating factor could be if your installation is not accessible from the Internet. | 9.0 – 10.0 |
| High | Vulnerabilities that score in the high range usually have some of the following characteristics:   * The vulnerability is difficult to exploit. * Exploitation could result in elevated privileges. * Exploitation could result in a significant data loss or downtime. | 7.0 – 8.9 |
| Medium | Vulnerabilities that score in the medium range usually have some of the following characteristics:   * Vulnerabilities that require the attacker to manipulate individual victims via social engineering tactics. * Denial of service vulnerabilities that are difficult to set up. * Exploits that require an attacker to reside on the same local network as the victim. * Vulnerabilities where exploitation provides only very limited access. * Vulnerabilities that require user privileges for successful exploitation. | 4.0 – 6.9 |
| Low | Vulnerabilities in the low range typically have very little impact on an organization's business. Exploitation of such vulnerabilities usually requires local or physical system access. | 0.1 – 3.9 |

## **7.2 About Burp Suite's web vulnerability scanner**

The web vulnerability scanner behind Burp Suite's popularity has more to it than most. Burp Scanner uses PortSwigger's world-leading research to help its users find a wide range of vulnerabilities in web applications, automatically. Sitting at the core of both Burp Suite Enterprise Edition and Burp Suite Professional, Burp Scanner is the weapon of choice for over 60,000 users across more than 15,000 organizations.

Reference: https://portswigger.net/burp/vulnerability-scanner

### **7.2.1 Burp Suite's web vulnerability scanner risk score**

The level of severity for an issue that was found by a scan. The higher the severity level, the larger the impact is likely to be if an attacker is able to exploit this vulnerability. Note that the severity level is only a rough approximation based on a typical website. You should use your knowledge of the purpose and context of the associated functionality to determine how serious each issue is in your individual case.

Reference: https://portswigger.net/burp/extensibility/enterprise/graphql-api/severity.html

| **Severity** | **Description** |
| --- | --- |
| High | An attacker can **fully** compromise the confidentiality, integrity, or availability, of a target system without specialized access, user interaction or circumstances that are beyond the attacker’s control. Very likely to allow lateral movement and escalation of attack to other systems on the internal network of the vulnerable application. |
| Medium | An attacker can **partially** compromise the confidentiality, integrity, or availability, of a target system. Specialized access, user interaction, or circumstances that are beyond the attacker’s control may be required for an attack to succeed. Very likely to be used in conjunction with other vulnerabilities to escalate an attack. |
| Low | An attacker can **limitedly** compromise the confidentiality, integrity, or availability, of a target system. Specialized access, user interaction, or circumstances that are beyond the attacker’s control is required for an attack to succeed. Needs to be used in conjunction with other vulnerabilities to escalate an attack. |