Connor Bressler

IAS311 – Ethical Hacking and Penetration Testing

2024 Vulnerability Testing Investigation Report Prepared For:

IAS311 Vulnerability Testing Investigation

Report Issued: 4/24/24

## Confidentiality Notice

*This report contains sensitive, privileged, and confidential information. Precautions should be taken to protect the confidentiality of the information in this document. Publication of this report may cause reputational damage to Penn College or facilitate attacks against Penn College.*

## Disclaimer

*Note that this assessment may not disclose all vulnerabilities that are present on the systems within the scope of the engagement. This report is a summary of the findings from a “point-in-time” assessment made on Penn College’s environment. Any changes made to the environment during the period of testing may affect the results of the assessment.*

# 

# EXECUTIVE SUMMARY

The penetration test of the intended target (192.168.2.4) was performed on the IAS311 internal network on 4/24/24. The penetration test simulated an attack from an external threat actor attempting to gain access to 192.168.2.4 within the IAS311 network. The purpose of this test was to discover and identify vulnerabilities in the target machine and to suggest methods to remediate the vulnerabilities. The test identified a total of five vulnerabilities within the scope of the engagement which are described below and broken down by severity in the table.

* The target system was a Linux machine and had an IP address of 192.168.2.4
* There were three active ports and services running on the target machine, this included:
  + **Port 22 – SSH**
  + **Port 80 – HTTP**
  + **Port 111 -Rpcbind**
* **Vulnerability 1 – Critical**
  + Insecure Configuration of web server on Port 80
* **Vulnerability 2 – High**
  + SQL Injection attack was able to be performed on the employee directory database that is stored on web server as **Vulnerability 1** allowed for sensitive directories and files to be viewed by unauthorized users.
* **Vulnerability 3 – High**
  + Employee information is able to be viewed in the clear with no hashing or encryption. Also resulted from **Vulnerability 1** as it allowed for sensitive directories and files to be viewed by unauthorized users.
* **Vulnerability 4 – Medium**
  + PHP info file about the web server is also able to be viewed in the clear. An outdated version of PHP is also being used, may allow for known vulnerabilities to be exploited. Also resulted from **Vulnerability 1.**
* **Vulnerability 5 – Medium**
  + An outdated version of SSH is being used, may allow for known vulnerabilities to be exploited.

|  |  |  |  |
| --- | --- | --- | --- |
| **CRITICAL** | **HIGH** | **MEDIUM** | **LOW** |
| **1** | **2** | **2** | **0** |

This penetration test is being performed with several expectations being prioritized. The primary expectation is to identify vulnerabilities on active ports and services within the target system that can potentially be exploited by malicious attackers. This includes assessing the effectiveness of security configurations and ensuring that they are serving their intended purposes. The activities that will be involved in this penetration test include reconnaissance and probing to gather information about the target system and its active services and ports, followed by scanning and discovering possible vulnerabilities on those active service and ports, and then trying methods to try and exploit those vulnerabilities. Post exploitation activities will include giving risk grades to all vulnerabilities found and giving recommendations on how to mitigate and resolve those vulnerabilities.

The penetration test is being done for multiple reasons. The main reason is to improve overall security posture by identifying weaknesses in applications, controls, and the system overall. It is important to find vulnerabilities and see how they are exploited in the test, then resolve them before they can be exploited by an actual attacker. This test also helps test response procedures and ensures that there can be an effective response to security incidents.

In order to ensure data confidentiality, integrity, and availability, security remediations should be implemented as described in the recommendations in the **Overview of All Vulnerabilities** section. Note that this assessment may not disclose all vulnerabilities that are present on the systems within the scope. Any changes made to the environment during the period of testing may affect the results of the assessment.

BRIEF OVERIVEW OF ALL TESTING METHODOLOGY PHASES

For the penetration test that will occur on this target machine, the methodology will be effective and thorough. The methodology will first begin when reconnaissance is performed during which basic information is gathered. Information including which domain the target system lives in, the status of the target machine, and identifying open ports and active services are all vital pieces of information that will be gathered during the reconnaissance phase. Tools such as Nmap whilst combining different parameters and Metasploit are used for reconnaissance, providing a basic but yet detailed view of the target. This information will be helpful when performing the two other phases later.

After the reconnaissance phase, the focus of the penetration test shifts to finding and identifying possible vulnerabilities on the target system. Activities that are conducted during this phase mainly focus on scanning and probing for vulnerabilities in the ports, services, and applications running on the target machine. Metasploit and other tools such as Hydra, Nikto, Burp suite, and the Exploit Database may all be used during this phase. The goal of this phase is to find any possible vulnerabilities that can be exploited and that puts the machine at risk of compromise. Even using custom automated scripts may be an option during the phase.

Once vulnerabilities have been found in active ports, services, and applications on the target machine, they will try to be exploited but there is no guarantee of this being successful. Throughout this test, there will be a more detailed, documented methodology of each phase later in this document describing all actions taken in each phase. This detailed methodology will include the specific actions used in each phase, the vulnerabilities that were identified, and what resulted from those vulnerabilities. A later section in the document will go into detail describing each vulnerability, the risk level it brings, its exploitation level, and the remediation level describing how difficult it will be to remediate and resolve that vulnerability.

HIGH LEVEL ASSESSMENT OVERVIEW

Areas for Improvement

The major area of improvement that is recommended is enhancing the overall security of the webserver by updating its configurations to follow best practices, including adding on the use of transport layer security (TLS) to the webserver. All of the vulnerabilities except one resulted from the insecure configuration of the webserver. If the webserver already had better security protections before the test occurred, none of these vulnerabilities could’ve been exploited. Using just HTTP is not best practice as there is no encryption applied to the traffic being sent back and forth between a user and the webserver. Applying strict access controls to the webserver would also be an improvement, as users who would only have valid credentials would be allowed to access the directories that were able to be viewed in the penetration test.

Implementing stricter input validation and parameterized queries is another area that needs improvement. In the penetration test, a successful SQL injection attack was performed on the employee directory. Enforcing stricter input validation to ensure that the query a user types in conforms to desired formats could resolve this issue. Using parameterized queries and prepared statement would separate SQL code from user input as well. Instead of just having a search bar in the employee directory that allows anything to be typed into, parameterized queries such as a first name, last name, or email could’ve been used. These approaches ensure that user input is treated as data and not as code.

Protecting employee data is another area of improvement. Encrypting the data of the directoryresults.PHP file should be done. Using hashing algorithms such as SHA-256 and other methods should be done to safeguard employee data during storage. Using access controls to prevent unauthorized access to the directory and PHP info page should be done as well.

Upgrading current PHP and SSH versions is another area that needs improved. Updating PHP and SSH to their latest versions will address known vulnerabilities and ensure compatibility with security updates.

Short Term Recommendations

* Update PHP and SSH to their current versions as soon as possible to resolve known vulnerabilities and enhance overall security posture.
* Encrypt sensitive data and restrict access to directories and files using access control
* Conduct a security assessment to identify and resolve the insecure configurations that reside on the webserver.

Long Term Recommendations

* Perform continuous monitoring and scanning to regularly detect and address security misconfigurations and issues.
* Implement a patch management system to keep software and services up to date to maintain good security posture.

SCOPE

All testing was based on the scope of the network that is used by the internal network for IAS311. This information is detailed more in depth in the **Reconnaissance** section of the **Testing Methodology.**

## Network

|  |  |
| --- | --- |
| **Network** | **Note** |
| 192.168.2.0/24 | Internal Network for IAS311 |

## System Info

|  |  |  |
| --- | --- | --- |
| **Name** | **System Type** | **Note** |
| No host name specified. | Desktop or PC | Uses Linux Operating System |

EACH TESTING METHODOLOGY PHASE IN DEPTH

The testing methodology is split into three phases: *Reconnaissance*, *Identification of possible vulnerabilities*, and *The Risk Grading and Possible Exploitation of Vulnerabilities.*

**Reconnaissance**

During reconnaissance, a preliminary test for a ping sweep and port scan was performed to identify the target system, and the network it is on.

1. **Introduction and Scope of Preliminary Test**
   1. This preliminary test is being conducted before the main parts of the penetration test occur. The procedures in this section involve reconnaissance and scanning of the network. This information will be used later in the main penetration test to perform more intensive procedures.
   2. The scope of the preliminary test is solely just testing the targeted Linux machine. The IP range that was tested during this scan was the 192.168.2.0-254 network. All that is being done in this test is a ping sweep of the chosen network and a port scan of the live systems.
   3. Restrictions were placed on the target machine. This test was for probing and determining where the target lived. It was placed on an internal network separate from other networks and the internet to limit spillage over on the internet.
2. **Commands Used**
   1. As mentioned, the scope of this test is a ping sweep of the network and a port scan. Two commands were used to complete these tasks.
   2. The command used to conduct the ping sweep was a simple Nmap command. The command was nmap -sn 192.168.2.0-254. The parameter -sn was used as the port scan was going to be conducted using another command. Using the parameter -sn leaves out the port scan to be conducted in a separate command. Combining both the ping sweep and port scan into one command was an option but most likely would take longer to do. As mentioned in section two, the scope of the preliminary test was the 192.168.2.0-254 network which is what was used in the ping sweep command.
   3. The command used to conduct the port scan was nmap -p 1-5000 192.168.2.2-4. The -p parameter was used so only a specific range of ports could be scanned. If the -p parameter wasn’t used, the command would scan all 65,000 thousand ports. It was determined that a scan between ports 1-5000 would be sufficient enough for this test. The IP range of 192.168.2.2-4 was used during this test although the data only from the target will be considered in this test. Mentioned below in **Ping Sweep Results**, the target system ends in .4, so the data from this machine will only be considered.
3. **Results of Ping Sweep and Port Scan**
   1. **Ping Sweep Results**
      1. The ping sweep occurred on April 17, 2024.
      2. The live systems that were identified in the 192.168.2.0-254 network were two machines. One machine was identified as the testing machine, and the other machine (192.168.2.4) was identified as the target machine.
      3. No host name was available for the target machine.
   2. **Port Scan Results**
      1. The port scan occurred on April 17,2024.
      2. The active ports and services that were identified were all on the 192.168.2.4 machine.
      3. Active Ports and Services on 192.168.2.4

* **Port 22- ssh**
* **Port 80 – http**
* **Port 111 – Rpcbind – Remote procedure call**

**Identification of Possible Vulnerabilities**

During the identification of possible vulnerabilities phase, actions will be taken, and tools will be used to actively try and exploit the three ports and services that were found active in the reconnaissance phase:

* **Port 22- ssh**
* **Port 80 – http**
* **Port 111 – Rpcbind – Remote procedure call**

1. **Searching for Vulnerabilities in Port 22-SSH**
   1. Port-22 that was running the service SSH was the first port to show up in the port scan in the reconnaissance phase. Since this is the case, this will be the first service that will be tested to see if there are any vulnerabilities to be exploited.
   2. For the first action of this phase, another Nmap scan was performed to determine the versions of all services running on the machine. The command used was nmap -sV -p 22,80,111. Based on the results, it seems that the current version of SSH being used (6.7p1), is outdated. This version of SSH is vulnerable considering it is very outdated.
   3. Next, a search was conducted to determine if there were any known vulnerabilities to SSH version 6.7. To find these results, searchsploit was used. The command “searchsploit OpenSSH 6.7p1” was used and five different vulnerabilities resulted. There were also scripts appended to each vulnerability that could be used as exploits. Overall, it seems there are five vulnerabilities that OpenSSH version 6.7p1 has. All five vulnerabilities also had a path assigned to them allowing for more information to be obtained. All five of these vulnerabilities will be explained more in depth later in the report.
   4. The command searchsploit -m was used to determine the specific CVE each vulnerability was assigned. An example can be found below.

**A screen shot of a computer

Description automatically generated**



* 1. So, each exploit for OpenSSH 6.7p1 has an assigned CVE number. Using this information later, more information can be learned about each exploit. The rest of the screenshotted data for all of the other vulnerabilities will be referenced in the Appendix B document.

1. **Searching for Vulnerabilities In Port 80-HTTP**
   1. In this scenario, the tool Nikto was used to scan the active HTTP service on port 80. Nikto can take what it is given and discover vulnerabilities and misconfigurations on the active service on port 80. Based on the results from the command (Nikto -h http://192.168.2.4) , there is an active Apache webserver on port 80. The webserver is multiple versions behind of the most recent version of Apache. As mentioned, Nikto is able to discover vulnerabilities in web services, so it seems reasonable as to why there is a CVE number, 2003-1418, in the results of the command. Nikto found a vulnerability based on the target Apache server being outdated. The CVE that was displayed is a known vulnerability that has most likely resulted from the outdated Apache webserver.

A screenshot of a computer program

Description automatically generated

* 1. As mentioned, there seems to be some webserver that the target is running. To visit this website, the IP address of the target (192.168.2.4) will be typed into a web browser to see if we can get to it.
  2. Reaching the webserver on the target was a success. Based on the results, it seems that the webpage is for selling Marvel merchandise. The website shows items for sale and their prices and descriptions, and also shows where the company resides and the top officials within the company and their emails.

A close up of a website

Description automatically generated

A close-up of a text

Description automatically generated

* 1. Since the webserver is using HTTP, the webserver itself is not secured. By going into the page info of the website, it does show that it does not support encryption. This is a vulnerability as none of the traffic a user would be sending to this website would be encrypted over the internet, allowing the traffic to be viewed by anyone. This is the end of trying to find vulnerabilities for the webserver.
  2. This is the end of trying to find vulnerabilities within the webserver, other exploits are going to be used to try and gain access to the system through the webserver.

A screenshot of a computer error

Description automatically generated

* 1. Based on the results from Nikto, there seems to be some sort of vulnerability that is being missed. Using WMAP may possibly reveal the vulnerability that is being missed. First, WMAP has to be configured and setup to try and find a vulnerability on the server. All commands were found from the link: <https://null-byte.wonderhowto.com/how-to/use-metasploits-wmap-module-scan-web-applications-for-common-vulnerabilities-0187572/>
     1. The commands being used to setup WMAP are as follows:
        1. Msfdb init – creates a default database for Metasploit to interact with
        2. Service postgresql start – starts the PostgreSQL service
        3. Load wmap – loads the WMAP module
        4. Use the command wmap\_sites -a <http://192.168.2.4>. This will add the target IP as a site.
        5. Next, we need to specify the target of the scan. This will be done using the command wmap\_targets <http://192.168.2.4/index.html>.
        6. A site has been added and a target has been added, now all that has to be done is run the scanner.
        7. This will be done using wmap\_run -t. The -t parameter will show all enabled modules. The pictures in the **Appendix B** document will show all modules that are enabled when the scan is going to be ran.
        8. Now, time to run the scan. The command to perform this will be wmap\_run -e. The -e parameter will run all modules instead of just one.
        9. The scan has been completed, to interpret the results the command wmap\_vulns -l. The results are in the picture below.

A computer screen shot of a program

Description automatically generated

* + - 1. It seems that these are the different directories that are accessible in webserver. Access to these directories could all pose vulnerabilities if they have sensitive information in them. These options will be explored in the **Possible Exploitation** section below.

1. **Searching for vulnerabilities in Port 111- Rpcbind**
   1. There doesn’t seem to be a lot of tools that help with finding vulnerabilities relating to Rpcbind. There are some known exploits with Rpcbind that were found in searchsploit. At first, the command nmap -sV -p 111 192.168.2.4 was ran to determine the current version of Rpcbind running. The result came out to be 2-4 (RPC #100000). At this point, there doesn’t seem to be anything with Rpcbind that would present a vulnerability. More effort is going to be directed towards SSH and the webserver on port 80 to try and exploit one the vulnerabilities presented in those areas.

**Possible Exploitation of Found Vulnerabilities**

**Trying to Exploit SSH on Port 22**

1. Since the version of SSH being used is severely outdated, some known exploits are going to try to be used. To find a module for an exploit to use, Metasploit searched for viable exploits. This was done using the command “search ssh”. The command resulted in over 50 modules. At this point it was a guessing game as to which module could be used to try and exploit the outdated version of SSH. The first module to be used was 60, this dealt with exploiting the use of default credentials on SSH on 192.168.2.4.
2. A screenshot of a computer program

   Description automatically generated
3. In these commands, the target IP (192.168.2.4) was configured to be used along with the target port (22), but this exploit didn’t work. Most of the exploits for SSH require advanced knowledge to complete and learning that information may not be able to happen considering the time period in which this penetration test needs completed. At this point, the webserver on port 80 seems the most vulnerable, this is where all effort is going to be focused on.

**Trying to Exploit the Apache Server on Port 80-HTTP**

1. There was an attempt to run a PHP exploit based on the current version of the Apache web server. This was done using the auxiliary modules through Metasploit. The LHOST was set to the IP of the testing machine and the RHOST was set to the IP of the target. Unfortunately, the exploit did not work. None of these exploits that are being used to try and compromise SSH and HTTP may not work.

A screen shot of a computer

Description automatically generated

1. Based on the results that are referenced below from the WMAP scan that occurred in the **Identification of Possible Vulnerabilities** section, it seems that these are different directories that can be accessed through the webserver.

A computer screen shot of a program

Description automatically generated

1. It seems to be that all of these directories can be accessed through the web browser. The index.html file and products.html file won’t be necessary to inspect as these are just generic pages on the webserver that have already been found.

The /icons/ directory will try and be accessed first to see if it can be exploited.

A screenshot of a computer

Description automatically generated

1. Seems that this directory can’t be accessed, onto the /info/ directory. Here are the results of the next directory.

A screenshot of a computer

Description automatically generated



1. It seems that the files parent directory stores may pose major vulnerabilities. It shows three HTML files, and two PHP files. The about.html file was already found earlier, so there is no need to inspect this. The directory.html file will be inspected first.

A close up of a message

Description automatically generated

1. This file seems to server as an employee directory. Since the security of the entire web server is lacking, there could be a possibility that an SQL injection attack could be used in the search bar. We are going to try the ‘ OR 1=’1 attack. This attack is referenced in the “Go Hack Yourself” book.

A close-up of a computer

Description automatically generated



1. The SQL injection attack worked. There seems to be 10 total employees in the Superhero Merchandising Directory. This is definitely a huge vulnerability, not only are all of the employee’s names and email revealed, but the directory is also vulnerable to a SQL injection attack. Onto the directoryResults.php file in the picture below.

A close-up of a email

Description automatically generated

1. It seems that this PHP file is just another page that displays information about employees. Although, there is a slight difference between this file, and the directory.html file that got an SQL injection attack performed on it. The directoryResults.php file shows nine employees, the SQL injection attack that was successfully performed on the directory.html file showed 10 employees. There was an extra employee known as “Groot”. It seems that the SQL injection attack was able to pull more information from the employee database.
2. The next file that will be inspected will be the indx.html file. Based on the picture below, it seems that this file is the file to test if the webserver is running correctly. This also serves as the default configuration overview for the webserver. This doesn’t pose as a vulnerability; this is just a generic file for all Apache2 webservers. Onto the next file.

A screenshot of a computer

Description automatically generated

The next file poses as a major vulnerability as it shows details about the current PHP version of the target system. The info.PHP file shows an abundance of details such as basic information about the target system, configuration of the Apache webserver, HTTP header information, hashing information, and so much more.

A screenshot of a computer program

Description automatically generated

**Trying to Exploit Rpcbind on Port 111**

1. During the **Identification of Possible Vulnerabilities** phase of this test, there weren’t any real vulnerabilities found. Therefore, there isn’t going to be any real testing of vulnerabilities for this service and port. Most of the concerning vulnerabilities were found on the Apache webserver in port 80.

CLASSIFICATION DEFINITIONS

Risk Classifications

|  |  |  |
| --- | --- | --- |
| **Level** | **Score** | **Description** |
| **Critical** | **10** | The vulnerability poses an immediate threat to the organization. Successful exploitation may permanently affect the organization. Remediation should be immediately performed. |
| **High** | **7-9** | The vulnerability poses an urgent threat to the organization, and remediation should be prioritized. |
| **Medium** | **4-6** | Successful exploitation is possible and may result in notable disruption of business functionality. This vulnerability should be remediated when feasible. |
| **Low** | **1-3** | The vulnerability poses a negligible/minimal threat to the organization. The presence of this vulnerability should be noted and remediated if possible. |
| **Informational** | **0** | These findings have no clear threat to the organization but may cause business processes to function differently than desired or reveal sensitive information about the company. |

Exploitation Likelihood Classifications

|  |  |
| --- | --- |
| **Likelihood** | **Description** |
| **Likely** | Exploitation methods are well-known and can be performed using publicly available tools. Low-skilled attackers and automated tools could successfully exploit the vulnerability with minimal difficulty. |
| **Possible** | Exploitation methods are well-known, may be performed using public tools, but require configuration. Understanding of the underlying system is required for successful exploitation. |
| **Unlikely** | Exploitation requires deep understanding of the underlying systems or advanced technical skills. Precise conditions may be required for successful exploitation. |

Remediation Difficulty Classifications

|  |  |
| --- | --- |
| **Difficulty** | **Description** |
| **Hard** | Remediation may require extensive reconfiguration of underlying systems that is time consuming. Remediation may require disruption of normal business functions. |
| **Moderate** | Remediation may require minor reconfigurations or additions that may be time-intensive or expensive. |
| **Easy** | Remediation can be accomplished in a short amount of time, with little difficulty. |

## ASSESSMENT FINDINGS

|  |  |  |  |
| --- | --- | --- | --- |
| **Number** | **Finding** | **Risk Score** | **Risk** |
| 1 | Insecure Configuration of Webserver | **10** | **Critical** |
| 2 | Webserver Vulnerable to SQL Injection Attack | **8** | **High** |
| 3 | Employee Information in the Clear  (Resulting from vulnerability 1) | **6** | **Medium** |
| 4 | PHP Info File being in the Clear and the use of an outdated version of PHP  (Resulting from vulnerability 1) | **8** | **High** |
| 5 | Using Outdated Version of SSH6.7p1 | **4** | **Medium** |

TEMPLATE NOTE: (Sorting by descending risk score)

OVERVIEW OF ALL VULNERABILTIES FOUND

1. **Insecure Configuration of Webserver**
   1. Incorrect security configurations on the server leads for the ability for an attacker to access the /info/ directory where sensitive information is stored. These incorrect security configurations of the server lead to other vulnerabilities mentioned in the document and this therefore constitutes this vulnerability having a maximum risk score of 10/10.

|  |  |
| --- | --- |
| **CRTICAL RISK (10/10)** | |
| **Exploitation Likelihood** | **Likely** |
| **Remediation Difficulty** | **Moderate** |

**Recommendations**

* Review file and directory permissions on the webserver to ensure that directories with sensitive information are inaccessible to unauthorized users with invalid credentials.
* Disable directory listing on the web server. This prevents users from seeing the contents of directories when there is no default index file.
* Implement a web application firewall (WAF), for the web server. Using specific rules on the WAF will block access to sensitive directories and files and prevent directory transversal attacks.

1. **SQL Injection Attack**
   1. The discovery of an SQL Injection Attack upon the employee directory indicates a misconfiguration in the webserver’s input validation, allowing an attacker to execute malicious SQL queries. The execution of SQL queries upon the employee database can lead to the leakage of employee data and possibly unauthorized access to the employee database.

|  |  |
| --- | --- |
| **HIGH RISK (8/10)** | |
| **Exploitation Likelihood** | **Likely** |
| **Remediation Difficulty** | **Moderate** |

A close-up of a computer

Description automatically generated

**Recommendations**

* Use prepared statements, and parameterized queries to prevent user inputs from being treated as executable code. Implementing input validation and sanitizing user input is also recommended.

1. **Employee information in the clear**
   1. The discovery of employee information being in the clear resulting from insecure configuration of the webserver can be severe. Employee information being in the clear can lead to data breaches, compromised privacy for employees, and possibly even unauthorized access to an account depending on the employee information found.

|  |  |
| --- | --- |
| **Medium RISK (6/10)** | |
| **Exploitation Likelihood** | **Likely** |
| **Remediation Difficulty** | **Moderate** |

A close-up of a email

Description automatically generated

**Recommendations**

* Implement user authentication when trying to gain access to the employee directory. Ensure that if a user wants to gain access to the employee directory, they have to have a valid set of credentials to do so. This will make the attempt to view the employee directory more challenging for an intruder if they do not have a valid set of credentials.
* Implementing hashing mechanisms such as SHA-256 on the data directory would be a good recommendation as well, this prevents employee data from being easily readable in the case of unauthorized access.

1. **PHP info data in the clear and the use of an outdated version of PHP**
   1. The discovery of PHP info data being in the clear resulting from insecure configuration on the webserver poses a risk. The PHP info data from the file shows the version of PHP being used, basic but yet important information about the webserver such as HTTP header information, hashing information, Apache environment variables, and PHP variables being used. Allowing an attacker to see this information enables them to learn about information about the webserver and helps them in searching for vulnerabilities based on the PHP configuration.
   2. The info.php file showing the version of PHP being used poses a large risk as well. Currently, on the webserver, the version of PHP being used is Version 5.6.30-0 +deb8u1. After a little bit of research, it was discovered that this version is severely outdated. By looking at cve.mitre.org and typing in the current version of PHP that is being used on the webserver, hundreds of vulnerabilities came up. This was most likely due to the fact that this version of PHP being used is severely outdated. So, not only does the info.php file show information like HTTP headers, hashing information, and Apache environment variables being used, the entire version of PHP itself that is being used has hundreds of known vulnerabilities.

|  |  |
| --- | --- |
| **HIGH RISK (8/10)** | |
| **Exploitation Likelihood** | **Likely** |
| **Remediation Difficulty** | **Easy** |

**Recommendations**

* Just like the employee directory information being in the clear, the PHP info data is also in the clear. To remediate this vulnerability, access controls should be put on the info.php file. This will ensure that if an appropriate user wants to look at this file, they have to have a valid set of credentials. Enabling user authentication to look at this file ensures that if an attacker wants to view it, they have to go obtain good credentials which makes it more challenging.
* Implementing hashing mechanisms such as SHA-256 on the info.php file is a good recommendation as well. This prevents this information from being read in the clear in the case of unauthorized access.
* Using the most recent version of PHP is also recommended. Using the most recent version will remediate the hundreds of vulnerabilities that the older version is currently at risk for.

1. **Use of outdated version of SSH**
   1. The discovery of the use of an outdated version of SSH on the target system poses an intermediate risk. Using older versions of SSH such as 6.7p1 contains vulnerabilities that attackers can exploit to gain unauthorized access, execute arbitrary code, and perform other malicious activities. Older versions of SSH are more susceptible to brute-force attacks as well. Overall, just using older versions of SSH makes the system susceptible to more attacks.

|  |  |
| --- | --- |
| **Medium RISK (4/10)** | |
| **Exploitation Likelihood** | **Likely** |
| **Remediation Difficulty** | **Easy** |

**Recommendations**

* The easiest recommendation to resolve this vulnerability is to install and use the most recent version of SSH. Installing the most recent version will address vulnerabilities that version 6.7 is susceptible to, it will improve overall security of the system and of SSH, and new features and performance enhancements can be put to good use.

APPENDIX A - TOOLS USED

|  |  |
| --- | --- |
| **TOOL** | **DESCRIPTION** |
| **Searchsploit** | Used for finding known exploits within ports and services. |
| **Metasploit** | Used for exploitation of vulnerable services and vulnerability scanning. |
| **Nmap** | Used for scanning ports on hosts. |
| **Nikto** | Used for finding vulnerabilities in web servers and applications |
| **WMAP** | Used for automating web application penetration test tasks and integrating it with Metasploit |
| **GoHackYourself** | Book that was used to reference SQL injection attack. |

***Table A.1:*** *Tools used during assessment*

## Version Information

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Description** |
| 1.0 | 4/24/24 | Initial report to client |

## Contact Information

|  |  |
| --- | --- |
| **Name** | Connor Bressler |
| **Email** | Cjb86@pct.edu |