Internal Network Penetration Test Report

XXXXX

19 January 2022

Numen Cyber Labs - Security Services

Numen Cyber Technology Pte. Ltd.
11 North Buona Vista Drive, #04-09,
The Metropolis, Singapore 138589

Tel:	65-63555555
Fax:	65-63666666
Email:	sales@numencyber.com
Web:	https://numencyber.com

Table of Content

Document Control

Client Confidentiality

This document contains Client Confidential information and may not be copied without written permission.

Proprietary Information

The content of this document is considered proprietary information and should not be disclosed outside of the recipient organization's network.

Numen Cyber Technology gives permission to copy this report for the purposes of disseminating information within your organization or any regulatory agency.

Document Version Control

Issue No.	Issue Date	Issued By	Change Description
0.1	19/01/2022	Xxx	Draft for internal review
0.2	24/01/2022	xxx	Internal review

Document Distribution List

xxx	Security Consultant, Numen Cyber Labs
xxx	Security Consultant, Numen Cyber Labs

Executive Summary

XXXXX engaged Numen Cyber Labs to conduct a penetration test against their internal network from 12 January 2022 to 17 January 2022. The purpose of the engagement was to utilize active exploitation techniques by simulating real network attack behaviors to evaluate the security of the application. The assessment provides insight into the resilience of the internal defense to withstand attacks from unauthorized users.

This current report details the scope of testing conducted and all significant findings along with detailed remedial advice. The summary below provides non-technical audience with a summary of the key findings and section two of this report contains technical details of each vulnerability that was discovered during the assessment.

Assessment Summary

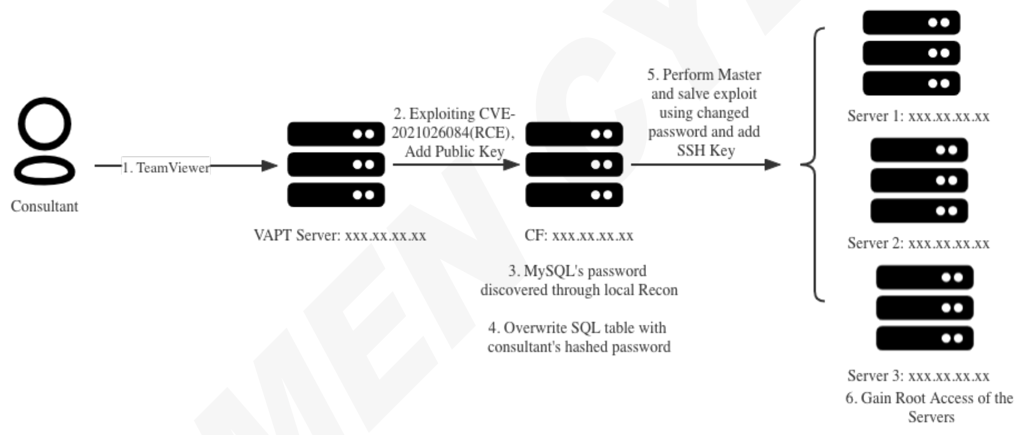

Consultant
1. TeamViewer
VAPT Server: xxx.xx.xx.xx
2. Exploiting CVE-2021026084(RCE), Add Public Key
CF: xxx.xx.xx.xx
3. MySQL's password discovered through local Recon
4. Overwrite SQL table with consultant's hashed password
5. Perform Master and salve exploit using changed password and add SSH Key
Server 1: xxx.xx.xx.xx
Server 2: xxx.xx.xx.xx
Server 3: xxx.xx.xx.xx
6. Gain Root Access of the Servers

Figure 1: Attack Roadmap

1. During the assessment, the only information provided was the IP address of the VAPT server (172.xx.xx.xx). It was accessed via TeamViewer from our testing machine. The consultant begins the assessment by conducting Information gathering and reconnaissance.

2. Multiple internal hosts were discovered, of which, one of them (xxx.xx.xx.xx) was found to be running Confluence and is affected by CVE-2021-26084. This vulnerability allows anyone to perform remote code execution. The consultant exploited this vulnerability by adding his own public key for SSH connection. Thereafter, SSH connection was initiated by the consultant and terminal access was obtained.

3. Having obtained terminal access in xxx.xx.xx.xx, the consultant began exploring local files and folders. Confluence's configuration file was found to contain root's password for mysql access. After connecting to mysql tables with root access, the consultant changed the passwords for Confluence and Jira admin accounts via SQL UPDATE function. The consultant successfully gained access to Confluence and Jira

with the respective admin accounts. Confluence was found to be displaying usernames and passwords in one of its web pages.

4. Three Redis servers were found to be vulnerable to password brute force and Master-Slave replication vulnerability. This resulted in root terminal access in all three Redis servers. After performing local file exploration, it was discovered that one of the Redis server contained sensitive information such as AWS secret key and access key.

Based on the security assessment for XXXX infrastructure, the status of the identified vulnerabilities set the risk at a **CRITICAL** level, which if not addressed in time, these vulnerabilities could be a trigger for a cybersecurity breach. These vulnerabilities can be easily fixed by following the best practices and recommendation given throughout the report.

1 Technical Summary

1.1 Scope

The security assessment was carried out in the internal network environment, and it included the following scope:

- xxx.xx.xx.xx

We discovered more hosts through information gathering, so we listed them in the scope of security assessment.

- xxx.xx.xx.xx
- xxx.xx.xx.xx
- xxx.xx.xx.xx
- xxx.xx.xx.xx
- xxx.xx.xx.xx

1.2 Post Assessment Clean-up

Any test accounts, which were created for the purpose of this assessment should be disabled or removed together with any associated content.

1.3 Risk Ratings

The risk level formulation is taken from Common Vulnerability Scoring System (CVSS) version 3.1 based on Base metric group.

CVSS is used for risk assessment of technical vulnerabilities due to its wide industry acceptance, and its ability to provide a composite metric based on multiple elements: the vulnerability's inherent technical risk, time-dependent risk, and organization unique perception of its business impact. CVSS scores can be considered as technically objective, up-to-date, and contextual - and is therefore will be used in this project as a way to propose prioritization of remediation efforts.

The likelihood that a potential vulnerability could be exercised by a given threat-source can be described as high, moderate, low. Assessing the likelihood consists of three steps; first determine the likelihood of event initiation or occurrence, next determine the likelihood threat event resulting in impact, and last determine overall likelihood level.

Definition of the value of the likelihood and the likelihood matrix are shown below. Definition table below describes 4 levels of severity ratings vulnerability mapped from the numeric CVSS score that will be assigned to each finding.

CVSS Score	Severity	Risk Levels Definition
9.0-10.0	CRITICAL	Exploitation is usually straightforward and likely result in compromise of data or root-level compromise of servers or infrastructure devices.
7.0-8.0	HIGH	Impact and likelihood of successful attack is high.
4.0-6.9	MEDIUM	Information from the system may be collected or modified, or which are not harmful on their own but potentially harmful when combined with other vulnerabilities.
0.0-3.9	LOW	No near-term risk exposure but may escalate in the future when the suspected vulnerability is validated and can be easily exploited.
0.0	INFO	A finding or observation which does not necessary warrant a risk in the assessment context.

In terms of exploitability, the severity rating can be described as follows (this rating does not consider details of your installation and is to be used as guide only):

Severity Level: **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, it 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.

Severity Level: **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.

Severity Level: **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 be on the same local network as the victim.
- Vulnerabilities where exploitation provides only very limited access.
- Vulnerabilities that require user privileges for successful exploitation.

Severity Level: **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

1.4 Findings Overview

Ref	Description	Risk
Xxxx-n-1	Obtained Terminal Access	Critical
Xxxx-n-2	Obtained Root Privilege Access for Three Redis Servers	Critical
Xxxx-n-3	Gained Administrative Access for Jira and Confluence	HIGH
Xxxx-n-4	Obtained Multiple Xxxx Administrators' Password	HIGH
Xxxx-n-5	Exploiting Username Enumeration Vulnerability	LOW

2 Assessment Details

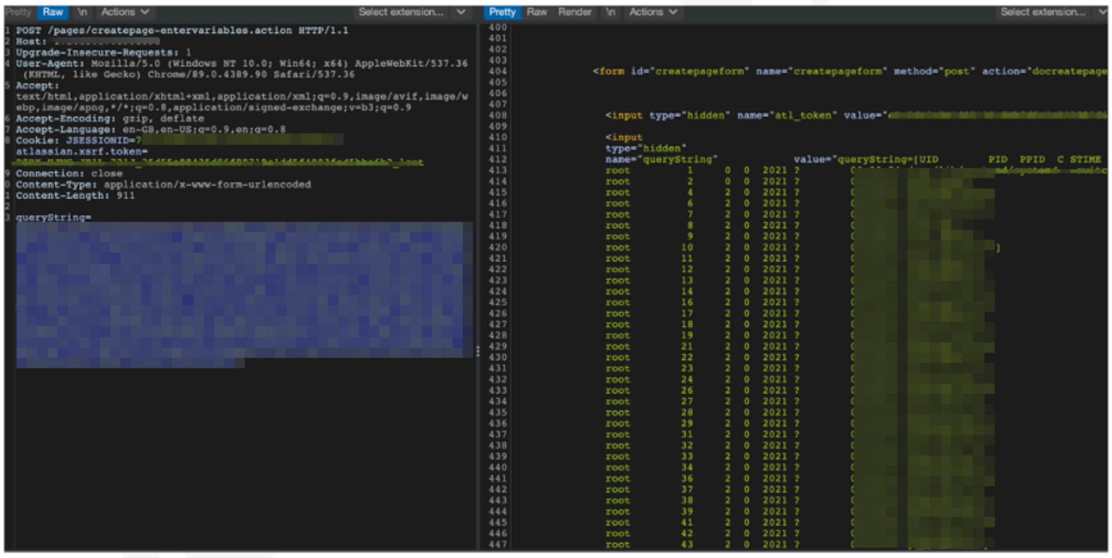
2.1 Obtained Terminal Access In xxx.xx.xx.xx

Ref-ID:	Xxxx-n-1	Risk Rating:	Critical

Attack Narrative

Initial access was granted via Team Viewer to the testing host machine (xxx.xx.xx.xx). After much reconnaissance an internal host machine (xxx.xx.xx.xx) was found to be running Confluence.

Upon further inspection, it was discovered that the Confluence version used was vulnerable to CVE-2021-26084. A remote, unauthenticated attacker could exploit this vulnerability by sending a malicious request to the target server. Successful exploitation can result in arbitrary code execution in the affected server.



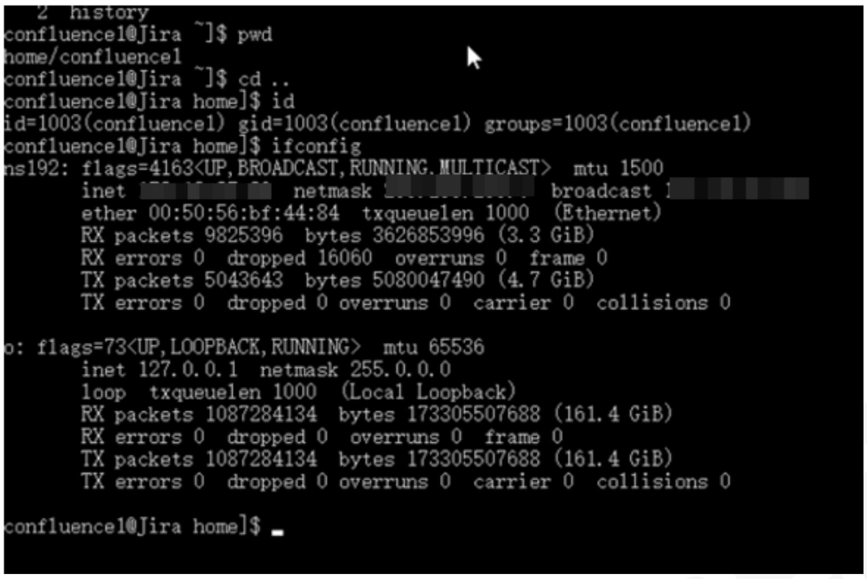
```
POST /pages/createpage-entervariables.action HTTP/1.1
Host: [redacted]
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
 (KHTML, like Gecko) Chrome/89.0.4389.90 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/w
ebp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Cookie: JSESSIONID=7[redacted]
atlassian.xsrf.token=
[redacted]
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 911

queryString=
[redacted]
```

```
<form id="createpageform" name="createpageform" method="post" action="docreatepage

  <input type="hidden" name="atl_token" value="[redacted]

  <input
  type="hidden"
  name="queryString"              value="queryString=[UID         PID  PPID  C STIME
  root            1       0   0  2021 ?          [redacted]
  root            2       0   0  2021 ?          [redacted]
  root            4       2   0  2021 ?          [redacted]
  root            6       2   0  2021 ?          [redacted]
  root            7       2   0  2021 ?          [redacted]
  root            8       2   0  2021 ?          [redacted]
  root            9       2   0  2021 ?          [redacted]
  root           10       2   0  2021 ?          [redacted]]
  root           11       2   0  2021 ?          [redacted]
  root           12       2   0  2021 ?          [redacted]
  root           13       2   0  2021 ?          [redacted]
  root           14       2   0  2021 ?          [redacted]
  root           16       2   0  2021 ?          [redacted]
  root           17       2   0  2021 ?          [redacted]
  root           18       2   0  2021 ?          [redacted]
  root           19       2   0  2021 ?          [redacted]
  root           21       2   0  2021 ?          [redacted]
  root           22       2   0  2021 ?          [redacted]
  root           23       2   0  2021 ?          [redacted]
  root           24       2   0  2021 ?          [redacted]
  root           26       2   0  2021 ?          [redacted]
  root           27       2   0  2021 ?          [redacted]
  root           28       2   0  2021 ?          [redacted]
  root           29       2   0  2021 ?          [redacted]
  root           31       2   0  2021 ?          [redacted]
  root           32       2   0  2021 ?          [redacted]
  root           33       2   0  2021 ?          [redacted]
  root           34       2   0  2021 ?          [redacted]
  root           36       2   0  2021 ?          [redacted]
  root           37       2   0  2021 ?          [redacted]
  root           38       2   0  2021 ?          [redacted]
  root           39       2   0  2021 ?          [redacted]
  root           41       2   0  2021 ?          [redacted]
  root           42       2   0  2021 ?          [redacted]
  root           43       2   0  2021 ?          [redacted]
```

Figure 2: HTTP request and response interaction with xxx.xx.xx.xx (host running confluence)

Figure 2 above shows a HTTP request being sent to xxx.xx.xx.xx. It contains a payload (***ps -ef***), which is a Linux command to get the list of processes currently running on the host. The HTTP response from the host machine successfully returned the process list.

During reconnaissance, it was discovered that port 22 was open and running SSH service. The default configuration in most SSH implementations allows users to deploy new authorized keys for themselves. Hence, the next step was to attempt to add an entry in authorized_keys file, which specifies the SSH keys that can be used for logging into the user account. An entry was successfully added into ***/home/confluence1/.ssh/authorized_keys*** via arbitrary code execution vulnerability.



```
  2  history
confluence1@Jira ~]$ pwd
home/confluence1
confluence1@Jira ~]$ cd ..
confluence1@Jira home]$ id
id=1003(confluence1) gid=1003(confluence1) groups=1003(confluence1)
confluence1@Jira home]$ ifconfig
ns192: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet [redacted]  netmask [redacted]  broadcast [redacted]
        ether 00:50:56:bf:44:84  txqueuelen 1000  (Ethernet)
        RX packets 9825396  bytes 3626853996 (3.3 GiB)
        RX errors 0  dropped 16060  overruns 0  frame 0
        TX packets 5043643  bytes 5080047490 (4.7 GiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

o: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 1087284134  bytes 173305507688 (161.4 GiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1087284134  bytes 173305507688 (161.4 GiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

confluence1@Jira home]$ _
```

Figure 3: Execution of terminal commands and the respective output

Figure 3 above shows that SSH connection was established via authorized keys. This allows the attacker to explore the host machine via the Terminal display and commands. The current user session for the terminal is confluence1 and is a non-privileged account.

Remediation Guidance:

There are two ways to go about fixing CVE-2021-26084. The recommended solution would be to upgrade to the latest Long Term Support release. Only in the case where upgrading is not possible, there is a temporary workaround available in the official website of confluence (cve-2021-26084-update.sh).

Post exploitation of CVE-2021-26084 was writeable SSH public keys to gain SSH access without password. Hence, the direct fix will be to restrict write access to authorized_keys file to only the file owner or root.

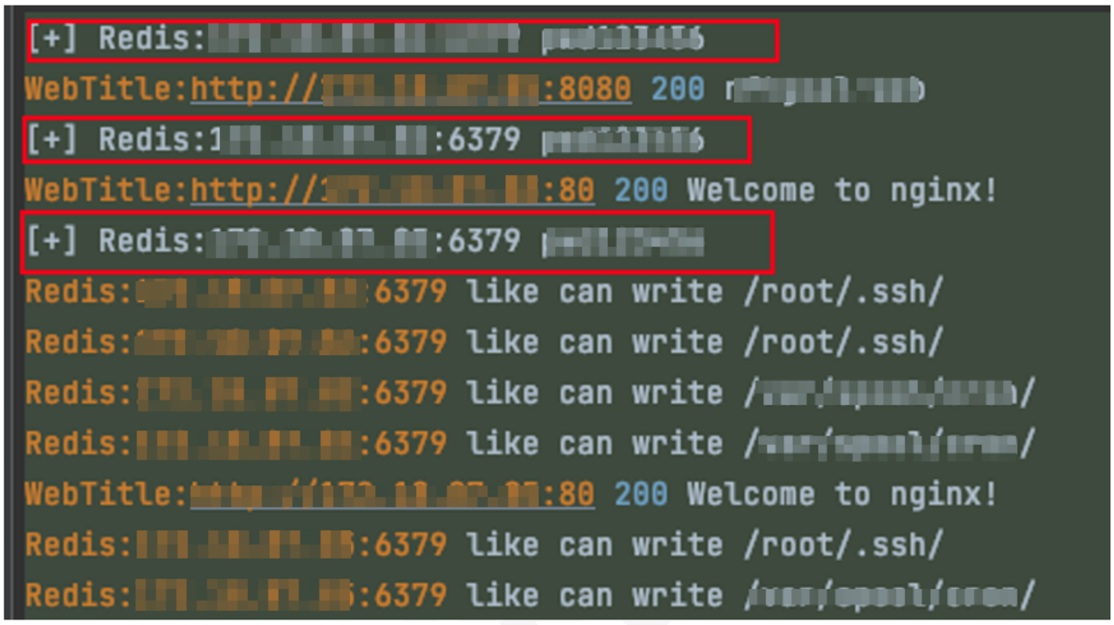
References:

- https://confluence.atlassian.com/doc/confluence-security-advisory-2021-08-25-1077906215.html
- https://steflan-security.com/linux-privilege-escalation-exploiting-misconfigured-ssh-keys/

2.2 Obtained Root Privilege Access for Three Servers

Ref-ID:	**Xxxx-n-2**	**Risk Rating:**	Critical

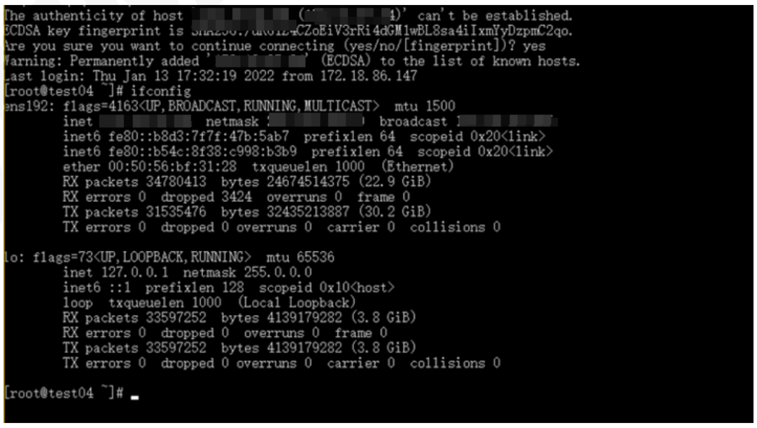
Attack Narrative



```
[+] Redis:[redacted] [redacted]
WebTitle:http://[redacted]:8080 200 [redacted]
[+] Redis:1[redacted]:6379 [redacted]
WebTitle:http://1[redacted]:80 200 Welcome to nginx!
[+] Redis:[redacted]:6379 [redacted]
Redis:[redacted]6379 like can write /root/.ssh/
Redis:[redacted]:6379 like can write /root/.ssh/
Redis:[redacted]:6379 like can write /[redacted]/
Redis:[redacted]:6379 like can write /[redacted]/
WebTitle:[redacted]:80 200 Welcome to nginx!
Redis:[redacted]:6379 like can write /root/.ssh/
Redis:[redacted]:6379 like can write /[redacted]/
```

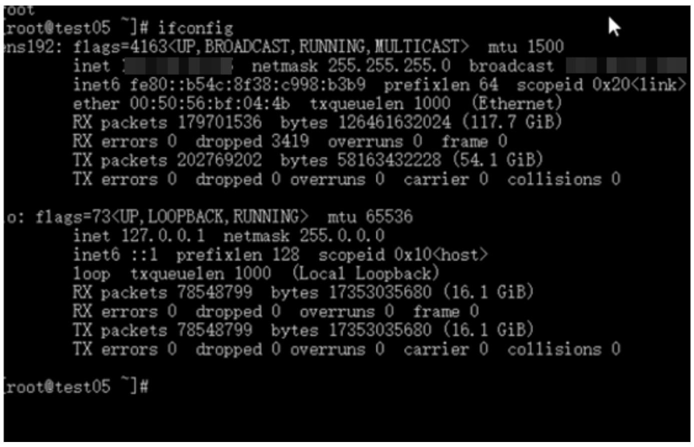
Figure 4: Password brute force

Figure 4 shows the results of the consultant's custom script to perform login brute force attempts against the 3 Redis servers. The weak Redis password (p*****6) was found to be the same for all 3 servers.



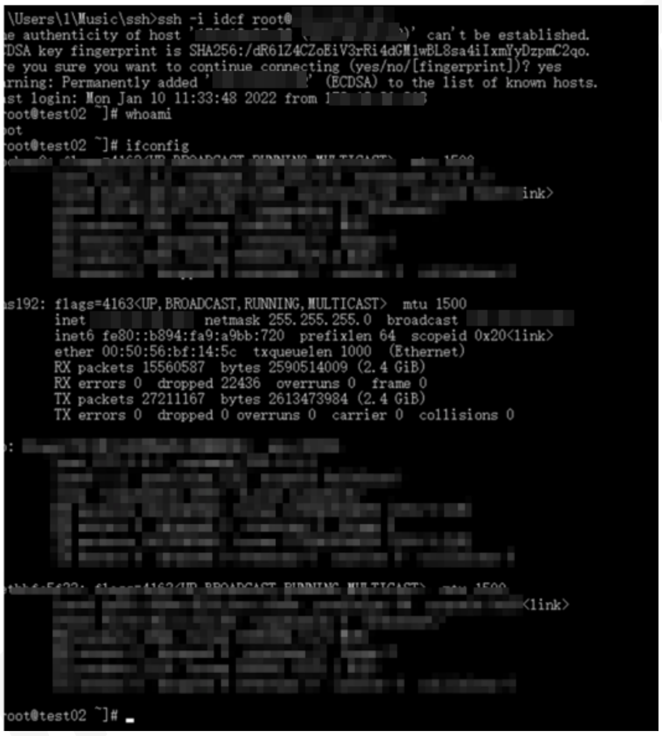
```
The authenticity of host [redacted] ([redacted]4)' can't be established.
ECDSA key fingerprint is [redacted]CZoEiV3rRi4dGM1wBL8sa4iIxmYyDzpmC2qo.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[redacted]' (ECDSA) to the list of known hosts.
Last login: Thu Jan 13 17:32:19 2022 from 172.18.86.147
[root@test04 ~]# ifconfig
ens192: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet [redacted]  netmask [redacted]  broadcast [redacted]
        inet6 fe80::b8d3:7f7f:47b:5ab7  prefixlen 64  scopeid 0x20<link>
        inet6 fe80::b54c:8f38:c998:b3b9  prefixlen 64  scopeid 0x20<link>
        ether 00:50:56:bf:31:28  txqueuelen 1000  (Ethernet)
        RX packets 34780413  bytes 24674514375 (22.9 GiB)
        RX errors 0  dropped 3424  overruns 0  frame 0
        TX packets 31535476  bytes 32435213887 (30.2 GiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 33597252  bytes 4139179282 (3.8 GiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 33597252  bytes 4139179282 (3.8 GiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[root@test04 ~]# _
```

Figure 5: xxx.xx.xx.xx proof of root access



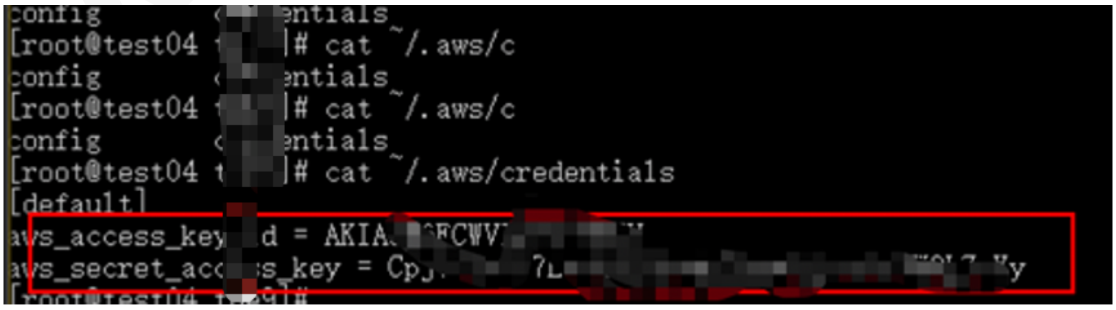
```
oot
root@test05 ~]# ifconfig
ns192: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet [redacted]  netmask 255.255.255.0  broadcast [redacted]
        inet6 fe80::b54c:8f38:c998:b3b9  prefixlen 64  scopeid 0x20<link>
        ether 00:50:56:bf:04:4b  txqueuelen 1000  (Ethernet)
        RX packets 179701536  bytes 126461632024 (117.7 GiB)
        RX errors 0  dropped 3419  overruns 0  frame 0
        TX packets 202769202  bytes 58163432228 (54.1 GiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

o: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 78548799  bytes 17353035680 (16.1 GiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 78548799  bytes 17353035680 (16.1 GiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@test05 ~]#
```

Figure 6: xxx.xx.xx.xx proof of root access



```
\Users\1\Music\ssh>ssh -i idcf root@[redacted]
e authenticity of host '[redacted] ([redacted])' can't be established.
DSA key fingerprint is SHA256:/dR61Z4CZoEiV3rRi4dGM1wBL8sa4iIxmYyDzpmC2qo.
e you sure you want to continue connecting (yes/no/[fingerprint])? yes
rning: Permanently added '[redacted]' (ECDSA) to the list of known hosts.
st login: Mon Jan 10 11:33:48 2022 from [redacted]
oot@test02 ~]# whoami
ot
oot@test02 ~]# ifconfig
[redacted]
s192: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet [redacted]  netmask 255.255.255.0  broadcast [redacted]
        inet6 fe80::b894:fa9:a9bb:720  prefixlen 64  scopeid 0x20<link>
        ether 00:50:56:bf:14:5c  txqueuelen 1000  (Ethernet)
        RX packets 15560587  bytes 2590514009 (2.4 GiB)
        RX errors 0  dropped 22436  overruns 0  frame 0
        TX packets 27211167  bytes 2613473984 (2.4 GiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[redacted]

[redacted]<link>
[redacted]
oot@test02 ~]#
```

Figure 7: xxx.xx.xx.xx proof of root access



```
config      [redacted]entials
[root@test04 [redacted]]# cat ~/.aws/c
config      [redacted]entials
[root@test04 [redacted]]# cat ~/.aws/c
config      [redacted]entials
[root@test04 [redacted]]# cat ~/.aws/credentials
[default]
aws_access_key[redacted]d = AKIA[redacted]FCWV[redacted]
aws_secret_acc[redacted]s_key = Cp[redacted]7L[redacted]y
[redacted]
```

Figure 8: xxx.xx.xx.xx AWS S3 keys

It was also observed that the Redis servers were vulnerable to Master-Slave replication exploit. The consultant used this exploit to insert entries /root/.ssh/authorized_keys, which specifies the SSH keys that can be used for logging into the root account.

Figures 5, 6 and 7 above show successful SSH login via SSH public key method for the 3 Redis servers (xxx.xx.xx.xx, xxx.xx.xx.xx and xxx.xx.xx.xx).

Figure 8 shows that AWS S3 keys was found after exploring local files in xxx.xx.xx.xx.

Remediation Guidance:

Strong and complex password is always recommended for user accounts. As mentioned in Redis website, it is very fast at serving queries. This implies that a brute force login attempt will be faster than usual. Hence, it is important not to use dictionary type or common passwords for important accounts such as root.

Redis Master-Slave replication vulnerability should be mitigated with the following measures:

1. Run Redis using non-privileged user
2. Prohibit external network access to Redis
3. Modify the default port
4. Set firewall policy

References:

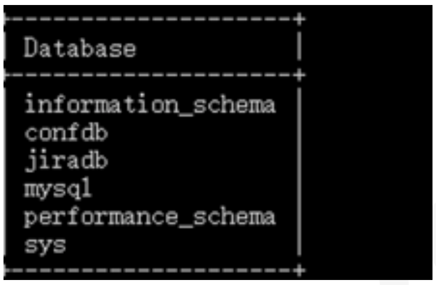
- https://nmap.org/nsedoc/scripts/redis-brute.html
- https://c0okb.github.io/2020/05/03/Redis%E6%BC%8F%E6%B4%9E%E5%88%A9%E7%94%A8/#%E5%89%8D%E8%A8%80
- https://redis.io/topics/security
- https://www.cnblogs.com/cute-puli/p/10870885.html

2.3 Gained Administrative Access for Jira and Confluence (xxx.xx.xx.xx)

Ref-ID:	**Xxxx-n-3**	**Risk Rating:**	**HIGH**

Attack Narrative

Using the terminal access to explore the local files, it was discovered that the xml file ***/data/atlassian/var/confluence/confluence.cfg.xml*** contains mysql root password (***root:asdf:********).



```
+--------------------+
| Database           |
+--------------------+
| information_schema |
| confdb             |
| jiradb             |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
```

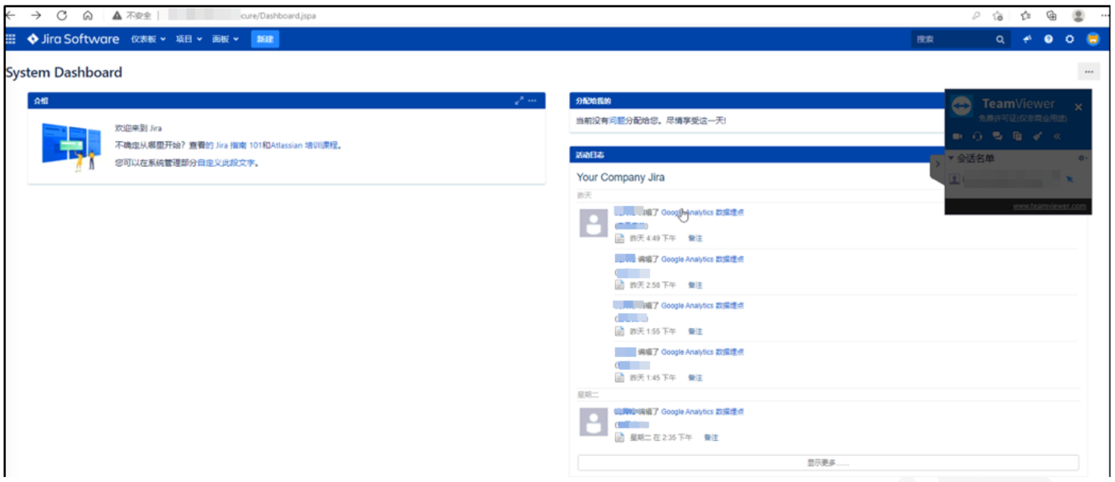
Figure 9: Output of ***mysql> show databases;***

After connecting to MySQL database as root and running the mysql command in Figure 9, a list of databases was displayed.

```
DIAGNOSTICS_ALERTS
EVENTS
FOLLOW_CONNECTIONS
IMAGEDETAILS
[redacted]
LABEL
LIKES
LINKS
[redacted]
------------------------------------------+
0 rows in set (0.00 sec)
```

Figure 10: Confluence database (confdb)

Figure 10 shows the contents within Confluence database.


不安全 | [redacted] cure/Dashboard.jspa

Jira Software 仪表板 项目 面板 新建 搜索

System Dashboard

介绍

欢迎来到 Jira

不确定从哪里开始？查看的 Jira 指南 101和Atlassian 培训课程。

您可以在系统管理部分自定义此段文字。

分配给我的

当前没有问题分配给您。尽情享受这一天!

活动日志

Your Company Jira

昨天

[redacted] 编辑了 Google Analytics 数据埋点 ([redacted])
昨天 4:49 下午 备注

[redacted] 编辑了 Google Analytics 数据埋点 ([redacted]
昨天 2:58 下午 备注

[redacted] 编辑了 Google Analytics 数据埋点 ([redacted])
昨天 1:55 下午 备注

[redacted] 编辑了 Google Analytics 数据埋点 ([redacted]
昨天 1:45 下午 备注

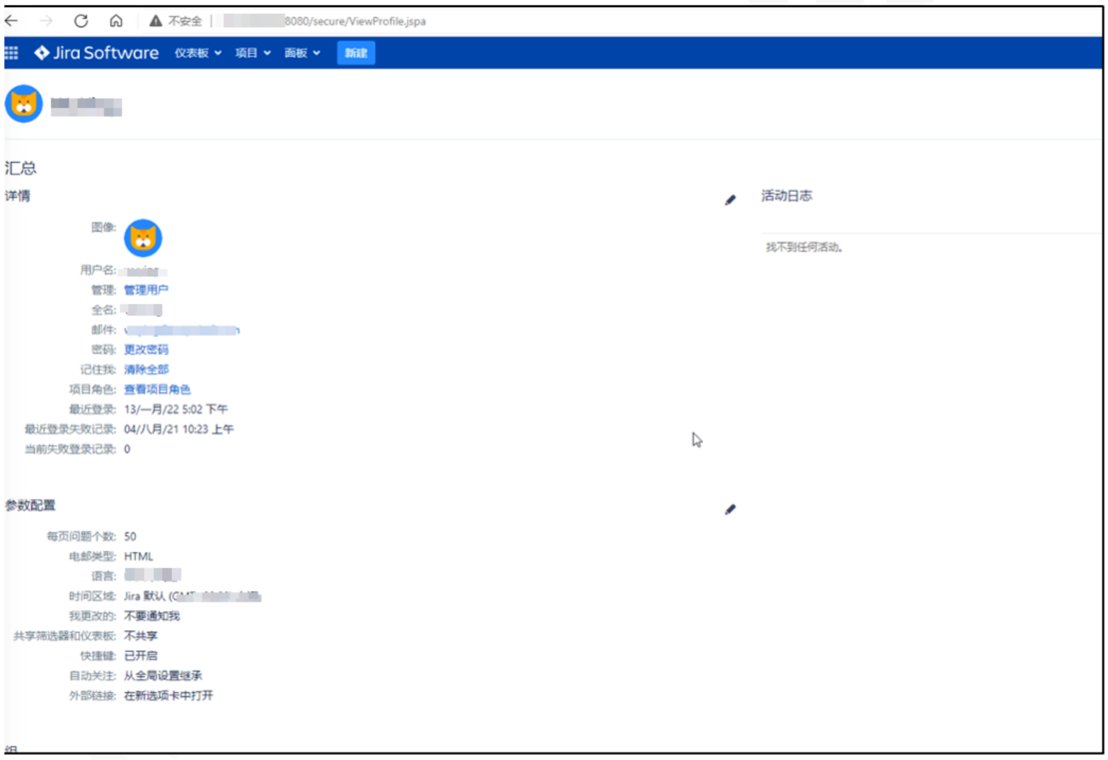
星期二

[redacted] 编辑了 Google Analytics 数据埋点 ([redacted]
星期二 在 2:35 下午 备注

显示更多......

TeamViewer
免费许可证(仅非商业用途)
会话名单
[redacted]
www.teamviewer.com

Figure 11: Jira management page with authenticated user session


不安全 | [redacted] 8080/secure/ViewProfile.jspa

Jira Software 仪表板 项目 面板 新建

[redacted]

汇总

详情

图像:	
用户名:	[redacted]
管理:	管理用户
全名:	[redacted]
邮件:	[redacted]
密码:	更改密码
记住我:	清除全部
项目角色:	查看项目角色
最近登录:	13/一月/22 5:02 下午
最近登录失败记录:	04/八月/21 10:23 上午
当前失败登录记录:	0

活动日志

找不到任何活动。

参数配置

每页问题个数:	50
电邮类型:	HTML
语言:	[redacted]
时间区域:	Jira 默认 (G[redacted]
我更改的:	不要通知我
共享筛选器和仪表板:	不共享
快捷键:	已开启
自动关注:	从全局设置继承
外部链接:	在新选项卡中打开

组

Figure 12: Jira management page with authenticated user session (XX)

Figure 11 & 12 shows an authenticated user session in Jira. There are many ways to obtain access to Jira. The easiest way was to execute SQL Update command to overwrite the current password for the user. After a successful overwrite, access was achieved by logging into Jira using the new password that was overwritten for that user.

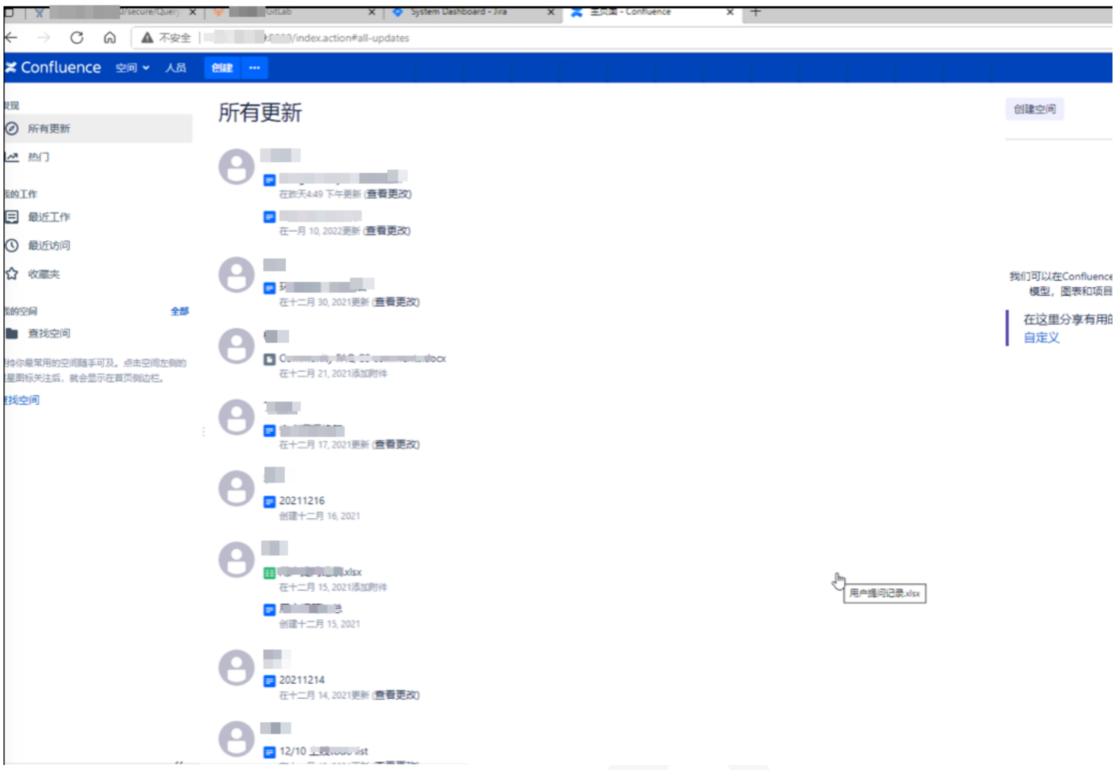


Figure 13: Confluence administrative page

Figure 13 shows an authenticated user session in Confluence. This successful login was also achieved by overwriting the current password for the admin via SQL Update statement.

Remediation Guidance:

Passwords should never be stored as plain text. However, the latest Confluence Long Term Support (LTS) is 7.13 and it does not support encryption. Therefore, it is recommended to implement a custom way of encryption to encrypt data at rest.

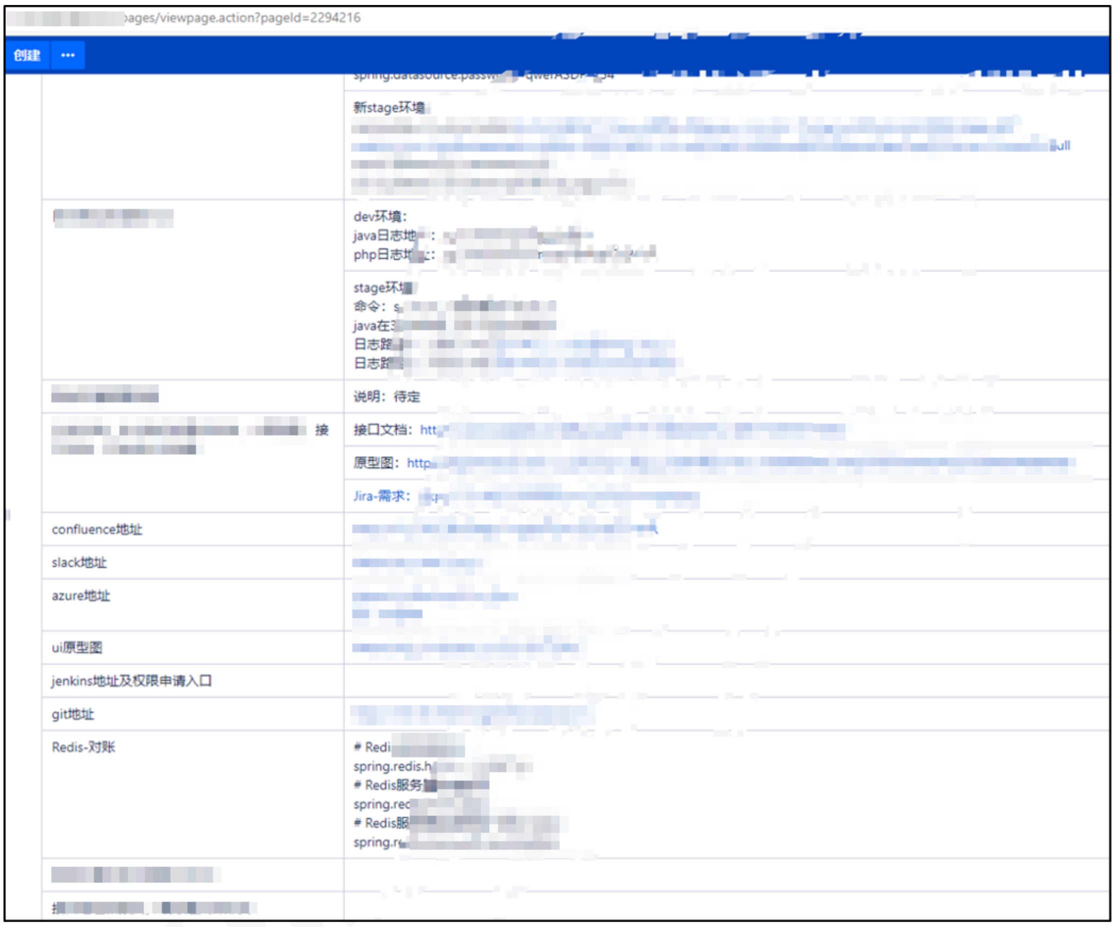
References:

- https://jira.atlassian.com/browse/CONFSERVER-60073

2.4 Obtained Multiple Xxxx Administrators' Password (xxx.xx.xx.xx)

Ref-ID:	Xxxx-n-4	Risk Rating:	HIGH

Attack Narrative


ages/viewpage.action?pageId=2294216

spring.datasource.passw qwerASDF

新stage环境

dev环境：
java日志地 ：
php日志地 ：

stage环境
命令：s
java在
日志路
日志路

说明：待定

接

接口文档：htt

原型图：http

Jira-需求：

confluence地址

slack地址

azure地址

ui原型图

jenkins地址及权限申请入口

git地址

Redis-对账

Redi
spring.redis.h
Redis服务
spring.red
Redis服
spring.r

Figure 14: XXXX administrator account username and passwords

After exploring Confluence web pages as an administrator, a web page containing user account passwords were found. As shown in Figure 14, x stage xx and Redis fields have some usernames and passwords displayed.

Remediation Guidance:

Plain text password should never be exposed even to administrators whereas other sensitive information should be stored elsewhere besides in Confluence.

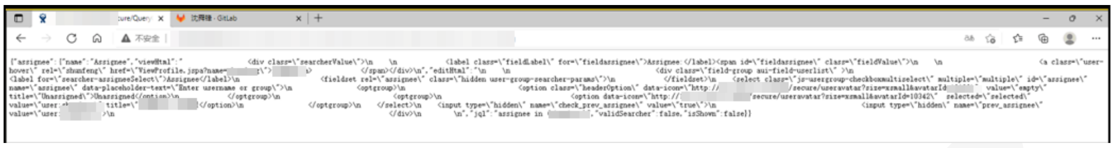
References:

- https://owasp.org/www-project-top-ten/2017/A3_2017-Sensitive_Data_Exposure

2.5 Exploiting Username Enumeration Vulnerability (xxx.xx.xx.xx)

Ref-ID:	**Xxxx-n-5**	**Risk Rating:**	LOW

Attack Narrative



```
{"assignee":{"name":"Assignee","viewHtml":"        <div class=\"searcherValue\">\n    \n        <label class=\"fieldLabel\" for=\"fieldassignee\">Assignee:</label><span id=\"fieldassignee\" class=\"fieldValue\">\n    \n        <a class=\"user-
hover\" rel=\"shunfeng\" href=\"ViewProfile.jspa?name=[redacted]\">[redacted]>        </span></div>\n","editHtml":"\n    \n        <div class=\"field-group aui-field-userlist\" >\n
<label for=\"searcher-assigneeSelect\">Assignee</label>\n        <fieldset rel=\"assignee\" class=\"hidden user-group-searcher-params\">\n        </fieldset>\n    <select class=\"js-usergroup-checkboxmultiselect\" multiple=\"multiple\" id=\"assignee\"
name=\"assignee\" data-placeholder-text=\"Enter username or group\">\n        <optgroup>\n                <option class=\"headerOption\" data-icon=\"http://[redacted]/secure/useravatar?size=xsmall&avatarId[redacted] value=\"empty\"
title=\"Unassigned\">Unassigned</option>\n        </optgroup>\n        <optgroup>\n        <option data-icon=\"http://[redacted]secure/useravatar?size=xsmall&avatarId=10342\"  selected=\"selected\"
value=\"user:[redacted] title=\"[redacted]</option>\n        </optgroup>\n    </select>\n    <input type=\"hidden\" name=\"check_prev_assignee\" value=\"true\">\n        <input type=\"hidden\" name=\"prev_assignee\"
value=\"user:[redacted]>\n                </div>\n        \n","jql":"assignee in [redacted],"validSearcher":false,"isShown":false}}
```

Figure 15: Output of a valid user “XX”

Atlassian Jira Server was observed to be affected by CVE-2020-36289. This allows an unauthenticated user to enumerate users via an Information Disclosure vulnerability in the QueryComponentRendererValue!Default.jspa endpoint.

As shown in Figure 15, the user “XX” was found to exists in the system using this vulnerability.

Remediation Guidance:

CVE-2020-36289 could be remediated by upgrading to versions 8.5.15, 8.13.7 or 8.17.0 and disabling anonymous user access.

References:

- https://jira.atlassian.com/browse/JRASERVER-71559

3 Conclusion

Xxxx's internal network contained multiple points of control failures, which led to a complete compromise of four host machines. These failures could have been disastrous to Xxxx if a malicious attacker had exploited them.

The purpose of the penetration test was to utilize active exploitation techniques by simulating real network attack behaviors to evaluate the security of the application. The goal was met when root access was obtained for several host machines.

It is important to note that compromise of the host machines was possible due to insufficient access controls at both the network boundary and host levels. Appropriate efforts should be undertaken to introduce effective network segmentation in addition to host hardening.

4 Appendices - Penetration Testing Methodologies

4.1 Web/API Application Assessment

Web application assessments can be performed either remotely or on site, depending on the exposure of the application. The purpose of the assessment is to identify any vulnerabilities that can be exploited in order to attack the system or other users, bypass controls, escalate privileges, or extract sensitive data.

During the assessment, the consultants will use proven non-invasive testing techniques to quickly identify any weaknesses. The application is viewed and manipulated from several perspectives, including with no credentials, user credentials, and privileged user credentials.

The primary areas of concern in web application security are authentication bypass, injection, account traversal, privilege escalation, and data extraction.

Our methodology covers all of the OWASP Top 10 web application security risks and more.

Ref	Risk	Description
A1	Injection	Injection flaws, such as SQL, NoSQL, OS, and LDAP injection, occur when untrusted data is sent to an interpreter as part of a command or query. The attacker's hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.
A2	Broken Authentication	Application functions related to authentication and session management are often implemented incorrectly, allowing attackers to compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users' identities temporarily or permanently.
A3	Sensitive Data Exposure	Many web applications and APIs do not properly protect sensitive data, such as financial, healthcare, and PII. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data may be compromised without extra protection, such as encryption at rest or in transit.
A4	XML External Entities (XXE)	Many older or poorly configured XML processors evaluate external entity references within XML documents. External entities can be used to disclose internal files using the file URI handler, internal file shares, internal port scanning, remote code execution, and denial of service attacks.
A5	Broken Access Control	Restrictions on what authenticated users are allowed to do are often not properly enforced. Attackers can exploit these flaws to access unauthorized functionality and/or data, such

		as access other users' accounts, view sensitive files, modify other users' data, change access rights, etc.
A6	Security Misconfiguration	Security misconfiguration is the most commonly seen issue. This is commonly a result of insecure default configurations, incomplete or ad hoc configurations, open cloud storage, misconfigured HTTP headers, and verbose error messages containing sensitive information. Not only must all operating systems, frameworks, libraries, and applications be securely configured, but they must be patched/upgraded in a timely fashion.
A7	Cross-Site Scripting (XSS)	XSS flaws occur whenever an application includes untrusted data in a web page without proper validation or escaping, or updates an existing web page with user-supplied data using a browser API that can create HTML or JavaScript. XSS allows attackers to execute scripts in the victim's browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites.
A8	Insecure Deserialization	Insecure deserialization often leads to remote code execution. Even if deserialization flaws do not result in remote code execution, they can be used to perform attacks, including replay attacks, injection attacks, and privilege escalation attacks.
A9	Using Known Vulnerable Components	Components, such as libraries, frameworks, and other software modules, run with the same privileges as the application. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover.
A10	Insufficient Logging and Monitoring	Insufficient logging and monitoring, coupled with missing or ineffective integration with incident response, allows attackers to further attack systems, maintain persistence, pivot to other systems, and tamper, extract, or destroy data. Most breach studies show time to detect a breach is over 200 days, typically detected by external parties rather than internal processes or monitoring.

4.2 Internal Infrastructure Assessment

Internal penetration testing will assessment identify how far an attacker can laterally move through a network once an external breach has occurred. This style of Assessment has a similar goal to external penetration testing (find sensitive data, take administrative control of the network, etc.), but provides a completely different attack surface for the assessment team to analyze.

The assessment is divided into Four distinct phases: Reconnaissance, Threat Modeling, Vulnerability Analysis and Exploitation:

1. Reconnaissance

Once the test has officially begun, a start notification will be sent to the client. The first phase will involve passive intelligence gathering, which will include sniffing the network and analyzing the traffic we can see. Additionally, some open source intelligence gathering will be performed to gather information such as user accounts, software information, etc. The goal of this phase is to identify any sensitive information that may help during the following phases of testing.

Tools may include: Wireshark, TCPDump, GHDB, custom scripts

2. Threat Modeling

For this assessment, the threat modeling phase serves to evaluate the types of threats that may affect the targets that are in scope. The types of attacks and likelihood of these threats materializing will serve to inform risk rankings/priorities that are assigned to vulnerabilities throughout the assessment. This phase of the assessment should also include host and service discovery, combining the passive methods of detection from step 1 with active enumeration. The ultimate goal of an internal penetration test is to emulate an attacker who has already gained access to the internal network or a malicious insider. As such, during this stage we will evaluate the targets in scope and determine an attack path that follows real-world attacks.

Tools may include: nmap, responder, Wireshark

3. Vulnerability Analysis

The vulnerability analysis phase will start with the active enumeration of all in-scope targets/applications. For each service discovered, automated and manual techniques will be used to attempt to find vulnerabilities on those hosts. The engineer will attempt to identify the version of the service and identify any previously published vulnerabilities. The engineer will also test the unauthenticated portion of any web applications discovered. Each service will be tested for default credentials, misconfigurations will be identified, and the attack surface will be prioritized ahead of the exploit phase.

Tools may include: Nessus, nmap, Burp Suite Pro, Metasploit Framework, netcat, dirb, SSLscan

4. Exploitation

This phase will involve taking all potential vulnerabilities identified in the previous phases of the assessment and attempting to exploit them as an attacker would. This helps to evaluate the realistic risk level associated with the successful exploitation of the vulnerability, analyze the possibility of exploit/attack chains, and account for any mitigating controls that may be in place. Additionally, we'll identify any false positives during this activity. That attack team focuses on using the identified vulnerabilities to reach the compromise goals laid out for the assessment, which usually involve accessing/exfiltrating sensitive data and/or taking administrative control of the corporate network.

Additionally, for an internal penetration test, there are attacks that take advantage of being on the same local network as in-scope targets. These include NBNS Spoofing, LLMNR Spoofing Attacks, ARP Cache Poisoning, etc. During the exploitation stage, the test team will try to gather credentials throughout the environment in various ways. For each login prompt discovered, password attacks will be attempted to try to gain access to in-scope systems.

Tools may include: Metasploit Framework, Responder, Bettercap, Hashcat, BurpSuite Pro, custom scripts

5. Post Exploitation

After successful exploitation analysis will continue, including infrastructure analysis, pivoting, sensitive data identification, data exfiltration, and identification of high-value targets/data. Our engineer will attempt to elevate their permissions and eventually achieve domain administrator, or top level permissions, on the domain. Once permissions are elevated, the test team will demonstrate the risk by identifying sensitive information such as credit cards, protected health information, internal communications, etc. We'll use the information collected here in the prioritization and criticality ranking of identified vulnerabilities.

Tools may include: Metasploit Framework, Burp Suite Pro, ExploitDB, custom scripts

5. References

- https://www.zerodayinitiative.com/blog/2021/9/21/cve-2021-26084-details-on-the-recently-exploited-atlassian-confluence-ognl-injection-bug
- https://steflan-security.com/linux-privilege-escalation-exploiting-misconfigured-ssh-keys/
- https://medium.com/@knownsec404team/rce-exploits-of-redis-based-on-master-slave-replication-ef7a664ce1d0
- https://github.com/projectdiscovery/nuclei-templates/blob/master/cves/2020/CVE-2020-36289.yaml
- https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-30r1.pdf