Partner Login Search

HUNTRESS

Platform Solutions Why Huntress Resources About Free Trial

Home > Blog > A Catastrophe For Control: Understanding the ScreenConnect Authentication Bypass (CVE-2024-1709 & CVE-2024-1708)

February 21, 2024

A Catastrophe For Control: Understanding the ScreenConnect Authentication Bypass (CVE-2024-1709 & CVE-2024-1708)

By: Team Huntress

On February 19, 2024, ConnectWise published a security advisory for ScreenConnect version 23.9.8, referencing two vulnerabilities and software weaknesses. The same day, Huntress researchers worked to understand this threat and successfully recreated a proof-of-concept exploit demonstrating its impact.

This write-up will discuss our analysis efforts and the technical details behind this attack, which we're coining as "SlashAndGrab."

The ConnectWise advisory indicated that in all versions of ScreenConnect below 23.9.8 there were two vulnerabilities:

1. **CVE-2024-1709**: Authentication bypass using an alternate path or channel (CWE-288)
2. **CVE-2024-1708**: Improper limitation of a pathname to a restricted directory ("path traversal") (CWE-22)

The first vulnerability was disclosed with a critical base CVSS scoring of 10, the highest possible severity. The authentication bypass would ultimately open the door for the second vulnerability.

ConnectWise made a patch available and expressed that all **on-premise** versions of ScreenConnect 23.9.7 and below must be updated immediately—cloud instances were already automatically patched.

At the time of release, the ConnectWise advisory was very sparse on technical details. There was not much information available as to what these vulnerabilities really consisted of, how they might be taken advantage of, or any other threat intelligence or indicators of compromise to hunt for.

Once we recreated the exploit and attack chain, we came to the same conclusion: there should not be public details about the vulnerability until there had been adequate time for the industry to patch. It would be too dangerous for this information to be readily available to threat actors.

But, with other vendors now publicly sharing the proof-of-concept exploit, the cat is out of the bag. We now feel that sharing our analysis shares no more threat than what is already available. So, we're ready to spill the beans.

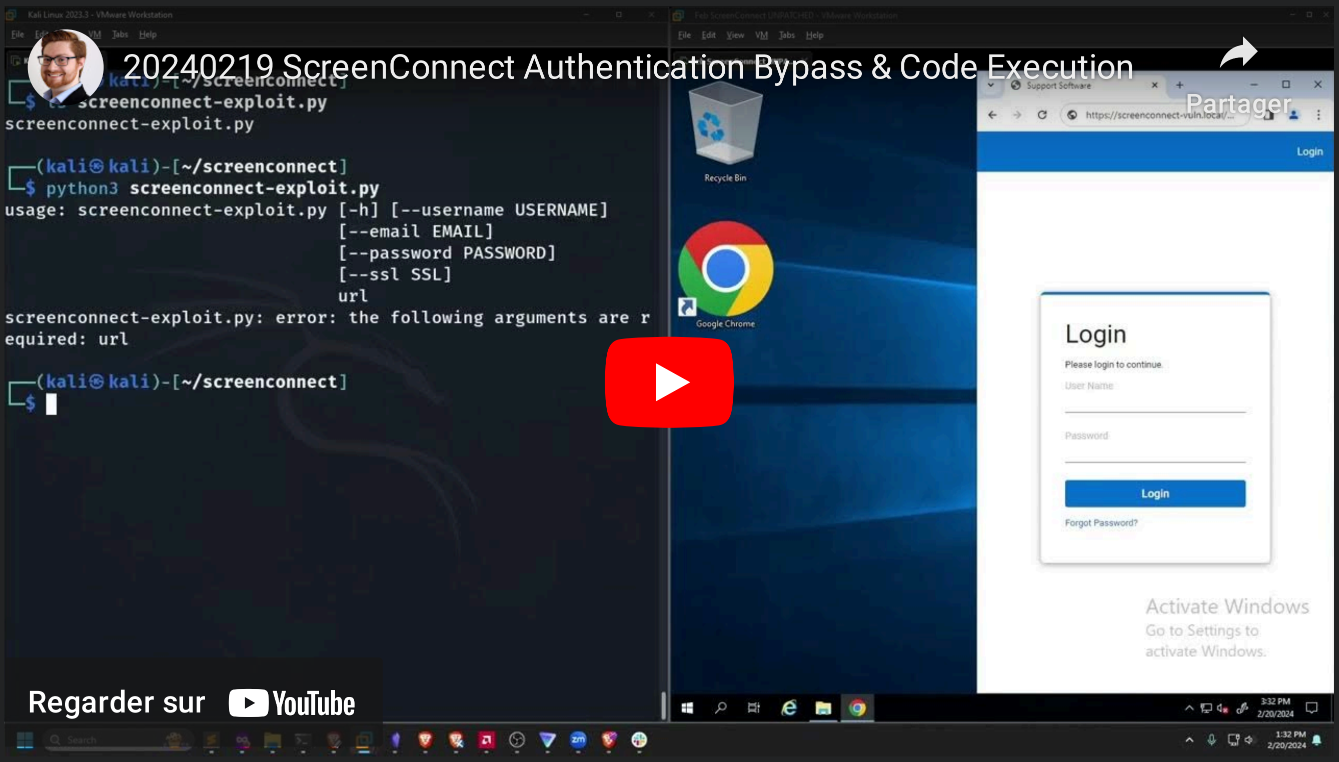
Categories

Response to Incidents

Share

The "exploit" is trivial and embarrassingly easy.

Below is a video demonstration of our recreated proof-of-concept exploit, which performs the simple authentication bypass but takes it a step further to showcase remote code execution.



```
┌──(kali㉿kali)-[~/screenconnect]
└─$ ls screenconnect-exploit.py
screenconnect-exploit.py

┌──(kali㉿kali)-[~/screenconnect]
└─$ python3 screenconnect-exploit.py
usage: screenconnect-exploit.py [-h] [--username USERNAME]
                                [--email EMAIL]
                                [--password PASSWORD]
                                [--ssl SSL]
                                url
screenconnect-exploit.py: error: the following arguments are r
equired: url

┌──(kali㉿kali)-[~/screenconnect]
└─$ █
```

Our Analysis

When the Huntress team was made aware of the ConnectWise advisory, our team began to dig into what changes were made in the patch that could help explain what these vulnerabilities were.

We could simply look for the differences between code present in the new, patched version, and the previous, unpatched version. Creating a local testing environment for both of these states of the ScreenConnect software, we could easily see the delta that might clue us into the potential exploit.

We observed that these files differed:

```
ScreenConnect/App_Data/User.xml
ScreenConnect/Bin/ScreenConnect.WindowsFileManager.exe
ScreenConnect/Bin/ScreenConnect.ClientService.exe
ScreenConnect/Bin/ScreenConnect.Client.Agent.zip
ScreenConnect/Bin/ServedArtifacts/ScreenConnect.Windows.dll
ScreenConnect/Bin/ServedArtifacts/ScreenConnect.Core.dll
ScreenConnect/Bin/ScreenConnect.Core.jar
ScreenConnect/Bin/ScreenConnect.Service.exe
ScreenConnect/Bin/ScreenConnect.WindowsClient.exe
ScreenConnect/Bin/ScreenConnect.WindowsServer.dll
ScreenConnect/Bin/ScreenConnect.Server.dll
ScreenConnect/Bin/ScreenConnect.ClientSetup.exe
ScreenConnect/Bin/ScreenConnect.Client.exe.jar
ScreenConnect/Bin/ScreenConnect.Web.dll
ScreenConnect/Bin/ScreenConnect.Client.Bundle.zip
ScreenConnect/Bin/ScreenConnect.Windows.dll
ScreenConnect/Bin/ScreenConnect.ClientSetup.msi
ScreenConnect/Bin/ScreenConnect.WindowsInstaller.dll
ScreenConnect/Bin/ScreenConnect.WindowsBackstageShell.exe
ScreenConnect/Bin/ScreenConnect.Client.apk
ScreenConnect/Bin/ScreenConnect.Client.dll
ScreenConnect/Bin/ScreenConnect.Client.application
ScreenConnect/Bin/ScreenConnect.Client.exe
ScreenConnect/Bin/ScreenConnect.Core.dll
ScreenConnect/Bin/ScreenConnect.Relay.dll
ScreenConnect/Bin/ScreenConnect.Client.manifest
ScreenConnect/Bin/ScreenConnect.Client.jar
ScreenConnect/Bin/ScreenConnect.ClientBootstrap.jar
ScreenConnect/Bin/ScreenConnect.ClientService.dll
ScreenConnect/SetupWizard.aspx
ScreenConnect/web.config
```

gistfile1.txt hosted with ❤ by GitHub view raw

CVE-2024-1709 - Authentication Bypass

Most of these changes observed in the new version were binary files, which require decompilation to compare; however, `SetupWizard.aspx` is a text file. When looking at the differences in this file, it immediately looked promising:

```
1a2
> <%@ Implements Interface="ScreenConnect.ISetupHandler" %>
5a7,14
>
>     protected override void OnInit(EventArgs e)
>     {
>        base.OnInit(e);
>
>        if (SetupModule.IsSetup)
>                throw new InvalidOperationException("Already setup"
>     }
```

SetupWizard.aspx.diff hosted with ❤ by GitHub view raw

In the newest update, we can see there is a new check to ensure that the instance is set up before allowing the user to use the setup wizard. This piqued our interest as, at face value, version 23.9.7 already blocked access to the setup wizard after authentication. Looking closer into `ScreenConnect.SetupModule` inside of `ScreenConnect.Web.dll` sheds more light on the exploitation path.

In previous versions of ScreenConnect, this handler was registered for the `OnBeginRequest` event, while in 28.9.8 it is registered for the `OnPostMapRequestHandler` event. This was seemingly done so that the `context.Handler` field will be populated. A new check is added to validate if the resolved HTTP handler implements the new `ISetupHandler` interface whereas the old code simply checked that `Request.Path` string matched "`/SetupWizard.aspx`". This change implies that there was some way to bypass the string comparison of `Request.Path` in previous versions. **If the request path does not match "`/SetupWizard.aspx`," then the setup wizard will be allowed regardless of the setup state of the instance.**

This would normally not be exploitable, but .Net has weird functionality that allows URL path components after a mapped legitimate URL to be passed along to the application. This information is exposed in `Request.FilePath`, `Request.PathInfo` and `Request.Path`. For example, if there is a legitimate path "`/something.aspx`", then .Net will allow the user to request "`/something.aspx/anythingelse`" and will pass along the trailing bit after the file path via the `Request.PathInfo`. However, `Request.Path` always contains *both* `Request.FilePath` *and* `Request.PathInfo` per MSDN documentation:

Putting this together, it means we can simply request "`/SetupWizard.aspx/literallyanything`" and we should be allowed to access the setup wizard on already-configured ScreenConnect instances.

The setup wizard is responsible for setting up the initial administrative user and installing a license on the system during initial setup. The user creation portion of this setup happens immediately after clicking the "Next" button on the setup page, so there is no need to complete the setup wizard fully to exploit the system (i.e. you do not need to replace or have access to a valid license key). **Completing this step will completely overwrite the internal user database. All other local users will be deleted aside from the user you specify in the setup wizard.**

Once you have administrative access to a compromised instance, it is trivial to create and upload a malicious ScreenConnect extension to gain Remote Code Execution (RCE). This is not a vulnerability, but a feature of ScreenConnect, which allows an administrator to create extensions that execute .Net code as `SYSTEM` on the ScreenConnect server.

CVE-2024-1708 - Path Traversal

The second vulnerability referenced in the ConnectWise ScreenConnect advisory presents a risk of ZipSlip attacks. We observed changes to `ScreenConnect.Core.dll`:

```
11057c11057
<               public void ExtractAllEntries(string basePath)
---
>               public void ExtractAllEntries(string basePath, strin
11062c11062
<                               FileSystemExtensions.DemandInParentPa
---
>                               FileSystemExtensions.DemandInParentPa
```

ScreenConnect.Core.dll.cs.diff hosted with ❤ by GitHub view raw

and `ScreenConnect.Server.dll`:

```
8392a8393
>                       DirectoryInfo extensionsDirectory = GetExten
8398c8399
<                       zipFile.ExtractAllEntries(GetExtensionsDirec
---
>                       zipFile.ExtractAllEntries(extensionsDirector
```

ScreenConnect.Server.dll.cs.diff hosted with ❤ by GitHub view raw

This adds new function arguments to strictly handle file paths when extracting content from a ZIP file. Looking through the code base, we can see these utility functions only seem to be used when handling ScreenConnect extensions. Prior to this patch, a malicious extension could potentially write files anywhere within `C:\Program Files (x86)\ScreenConnect\App_Extensions` instead of being properly restricted to their extension subdirectory.

While this is another vulnerability fixed in version `23.9.8` that could lead to remote code execution in certain conditions, it relies upon having administrative credentials and access to be able to use the Extensions functionality. This capability is "unlocked" by the authentication bypass, but is less severe on its own.

In our recreated proof-of-concept exploit, taking advantage of this potential ZipSlip attack vector was not necessary; after gaining administrator access with the authentication bypass, it was already possible to run arbitrary code with the typical Extensions feature set. The primary difference being that malicious code written using this vulnerability does not necessarily need an installed extension to execute. This means that it may go unnoticed more easily within your environment. Any `.aspx` or `.ashx` files within the root of `C:\Program Files (x86)\ScreenConnect\App_Extensions\` are likely malicious artifacts. ScreenConnect does not place any files within this directory normally.

Detection Guidance

While researching the above vulnerabilities, Huntress identified Indicators Of Compromise (IOCs) and developed a set of detections that can be used to notify ScreenConnect users of potential malicious activity.

As a reminder, per Huntress's previous post, the below listed SIGMA rules that leverage Windows Event ID 4663 requires that the Advanced Auditing policy be configured, and an appropriate SACL applied to the directory. Please refer to the previous blog post for how to generate this event.

CVE-2024-1709 - Authentication Bypass

While replicating the Authentication Bypass exploit, Huntress observed IIS logs for the malicious `SetupWizard.aspx` URL path, which can be used as an indicator of compromise:

The following Sigma rule will detect requests made to `SetupWizard.aspx` with a trailing path segment. This behavior is never legitimate, and should clearly indicate malicious intent if observed in the wild.

```
title: CVE-2024-1709 - ScreenConnect Authentication Bypass Exploitati
id: d27eabad-9068-401a-b0d6-9eac744d6e67
status: experimental
description: |
    Detects GET requests to '/SetupWizard.aspx/[anythinghere]' that
references:
  - https://www.connectwise.com/company/trust/security-bulletins/con
  - https://www.huntress.com/blog/a-catastrophe-for-control-understa
  - https://www.cve.org/CVERecord?id=CVE-2024-1709
author: Matt Anderson, Huntress
date: 2024/02/20
tags:
  - attack.initial_access
  - attack.persistence
  - cve.2024.1709
logsource:
  category: webserver
detection:
  selection:
    cs-uri-stem|contains: '/SetupWizard.aspx/'
  condition: selection
falsepositives:
  - Unknown
level: critical
```

SetupWizard_sigma_rule.yml hosted with ❤ by GitHub view raw

After accessing the Setup Wizard, an attacker will input new credentials to gain access to Screen Connect. Doing so will write a new user database to a temporary XML file. The following Sigma rule can be used to detect the ScreenConnect server writing to a temporary XML file on the server. While this may be an indicator of compromise, this activity is known to also be observed when legitimately modifying local users or their permissions in ScreenConnect, so there is a non-zero false positive rate.

```
title: ScreenConnect User Database Modification
id: 4109cb6a-a4af-438a-9f0c-056abba41c6f
status: experimental
description: This detects file modifications to the temporary xml use
references:
  - https://www.connectwise.com/company/trust/security-bulletins/con
  - https://www.huntress.com/blog/a-catastrophe-for-control-understa
  - https://www.cve.org/CVERecord?id=CVE-2024-1709
```

```
author: Huntress DE&TH Team
date: 2024/02/20
logsource:
  product: windows
  service: security
  definition: 'Requirements: SACLs must be enabled for the ScreenCon
detection:
  selection:
    EventID: 4663
    ObjectType: 'File'
    AccessMask: '0x6'
    ObjectName|endswith: '.xml'
    ObjectName|contains|all:
      - 'Temp'
      - 'ScreenConnect'
    ProcessName|contains: ScreenConnect.Service.exe
  condition: selection
falsepositives:
  - Unknown
level: medium
tags:
  - cve.2024.1709
```

ScreenConnect_modification_sigma_rule.yml hosted with ❤ by GitHub view raw

The following YARA rule is similar to the IIS log Sigma rule given above. It will detect IIS log entries that reference `SetupWizard.aspx` with a trailing path component. As mentioned previously, a trailing path component on this file is always associated with malicious activity.

```
rule ScreenConnect_CVE_2024_1709_Exploitation {
  meta:
    description = "Detects a GET request to '/SetupWizard.aspx/' wit
    author = "Huntress DE&TH Team"
    reference = "https://www.huntress.com/blog/a-catastrophe-for-con
    date = "2024-02-20"
    id = "2886530b-e164-4c4b-b01e-950e3c40acb4"
  strings:
      $s1 = "/SetupWizard.aspx/" ascii

   condition:
      $s1
}
```

SetupWizard_IIS_log.yara hosted with ❤ by GitHub view raw

CVE-2024-1708 - Path Traversal

The path traversal exploit allows attackers to write files within the root of the `App_Extensions` directory. The following Sigma rule should alert on file modifications at the root of the `App_Extensions` directory while excluding legitimate modifications to extensions themselves.

```
title: CVE-2024-1708 - ScreenConnect Path Traversal Exploitation
id: 4c198a60-7d05-4daf-8bf7-4136fb6f5c62
status: experimental
description: This detects file modifications to ASPX and ASHX files w
references:
 - https://www.connectwise.com/company/trust/security-bulletins/conne
 - https://www.cve.org/CVERecord?id=CVE-2024-1708
 - https://www.huntress.com/blog/a-catastrophe-for-control-understand
author: Huntress DE&TH Team
date: 2024/02/20
tags:
 - attack.initial_access
 - attack.persistence
 - cve.2024.1709
logsource:
product: windows
service: security
definition: 'Requirements: SACLs must be enabled for the ScreenConne
detection:
selection:
    EventID: 4663
    ObjectType: 'File'
    ProcessName|contains: 'ScreenConnect.Service.exe'
    AccessMask: '0x6'
    ObjectName|endswith:
       - 'ScreenConnect\\App_Extensions\\*.ashx'
       - 'ScreenConnect\\App_Extensions\\*.aspx'
legitimate_path:
    ObjectName|contains: 'ScreenConnect\App_Extensions\\*\\'
condition: selection and not legitimate_path
falsepositives:
 - Unknown
level: critical
```

App_Extensions_sigma_rule.yml hosted with ❤ by GitHub view raw

We encourage customers and partners to reach out if they need assistance. If you are not currently using Huntress and need help monitoring for activity related to this vulnerability, you can use Huntress' free trial.

You Might Also Like

Incident Response: A Choose Your Own Adventure Exercise

Learn More

Cracks in the Foundation: Intrusions of FOUNDATION Accounting Software

Learn More

SlashAndGrab: The ConnectWise ScreenConnect Vulnerability Explained

Learn More

Platform

Huntress Managed Security Platform

Managed EDR

Managed EDR for macOS

MDR for Microsoft 365

Managed SIEM

Managed Security Awareness Training

Book A Demo

Solutions

Phishing

Compliance

Solutions by Topic

Business Email Compromise

Healthcare

Manufacturing

Education

Finance

Why Huntress?

Managed Service Providers

Value Added Resellers

Business & IT Teams

24/7 SOC

Case Studies

Resources

Resource Center

Blog

Upcoming Events

Support Documentation

About

Our Company

Leadership

News & Press

Careers

Contact Us

Privacy Policy | Cookie Policy | Terms of Use

Free Trial