Open in app

Get started

N0ur5 Follow

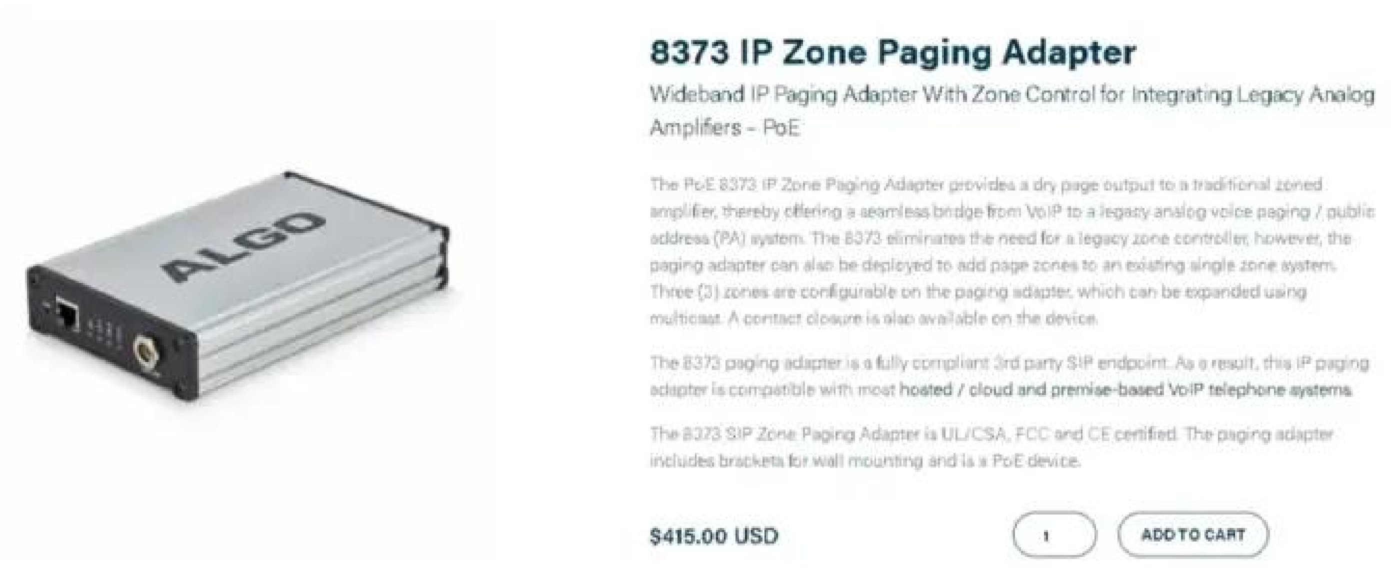
Jun 21 · 7 min read · Listen

Save

Achievement Unlocked: CVE-2022–31395

Hey All,

Been awhile, like usual :)



8373 IP Zone Paging Adapter

Wideband IP Paging Adapter With Zone Control for Integrating Legacy Analog Amplifers - PoE

The PoE 8373 IP Zone Paging Adapter provides a dry page output to a traditional zoned amplifier, thereby offering a seamless bridge from VoIP to a legacy analog voice paging / public address (PA) system. The 8373 eliminates the need for a legacy zone controller, however, the paging adapter can also be deployed to add page zones to an existing single zone system. Three (3) zones are configurable on the paging adapter, which can be expanded using multicast. A contact closure is also available on the device.

The 8373 paging adapter is a fully compliant 3rd party SIP endpoint. As a result, this IP paging adapter is compatible with most **hosted / cloud and premise-based VoIP telephone systems.**

The 8373 SIP Zone Paging Adapter is UL/CSA, FCC and CE certified. The paging adapter includes brackets for wall mounting and is a PoE device.

$415.00 USD 1 ADD TO CART

We will be talking about a CVE related to firmware running on one of these devices in this blog. Other devices by Algo running identical firmware seem impacted as well.

I am here today to write about a vulnerability I recently discovered during a security assessment and how it spawned into my first CVE. Getting a CVE to my name is something I've wanted to accomplish for several years. I got close a few times, and once I even found what I thought was a "Zero day" in a Fortinet Firewall just to find out it was reported and patched just wee[illegible]nd it. So close!

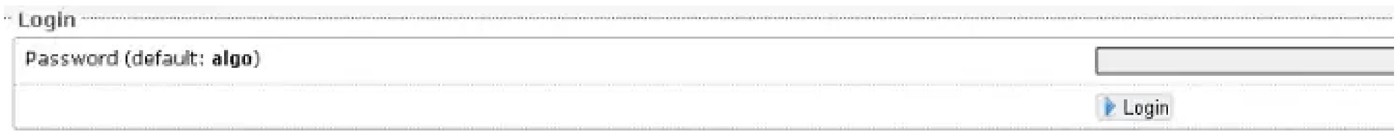
4

While I will be transparent in saying that this vulnerability is far from the most mind blowing or exciting vulnerability in recent times. I will also say I'm proud to have reported it through the CVE reporting process and was reserved CVE-2022–31395 by doing so. The final step in locking the CVE in is to point MITRE to a publication about the vulnerability. As you can maybe imagine, that is a large part of why I am writing this article. I did attempt to contact the manufacture to ask if they knew about this vulnerability. They simply told me the firmware in question was outdated and to update it. That however didn't tell me if this was a known vulnerability so I asked if they had any specific CVE's known to exist against the outdated firmware version I was working with. They said they did not have that information so I moved on to reporting to MITRE. As I sent the details to MITRE I was like:

I didn't hear anything for several days from MITRE so I pinged them for an updated and was told they are experiencing a high volume of reported CVE's and are working through them in a priority order. A few weeks passed and finally I saw a glorious email arrive. It stated that I should use CVE-2022–31395 to craft a public reference for the vulnerability in question, at which point the CVE should be official and no longer just "reserved".

Without further ado… I present to you the Directory Traversal vulnerability I found and reported. The vulnerability affects at least: **ALGO COMMUNICATION PRODUCTS LTD 8373 IP Zone Paging Adapter Control Panel, Firmware 1.7.6**. Other ALGO products containing the same firmware likely contain the same vulnerability as I have also confirmed the **1.7.6 Firmware on Algos' "8201 SIP PoE Intercom Control Panel"** contains the same vulnerability). I have not yet confirmed if other, newer firmwares are vulnerable at this time. In addition, the support team at Algo said the best way to know would be to update to their newest firmware and try the attack again. However, I do not have administrative privileges to this device as it belongs to a client of mine, so I am not able to update it and will have to wait if to see if my client decides to do so. I will report back if I find that newer versions are impacted.

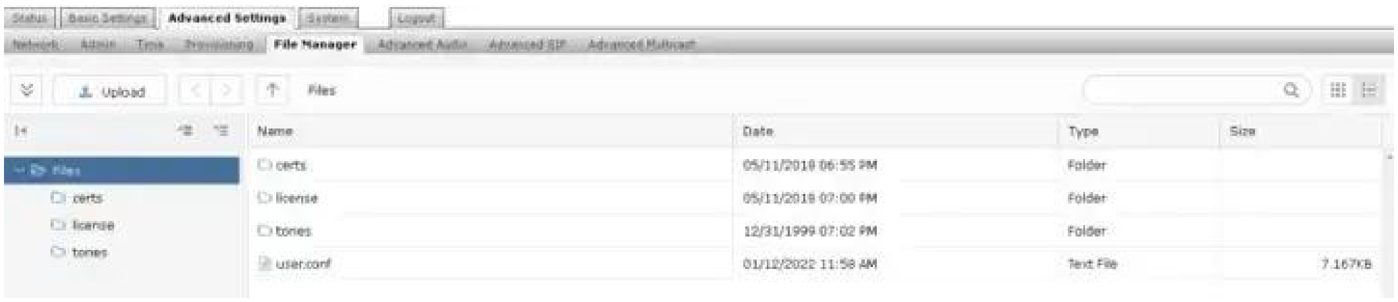
Lets' get into some of the details about this vulnerability! First, it is worth noting that this is an authenticated attack. The part of the application that is vulnerable is only accessible once authenticated. These devices appear to come pre-configured with a default password of "algo" with no username. In fact, I am saved a Google search for default password when initially analyzing the interface as it's clearly stated right in the web interface itself.



Password (default: **algo**)

I know it's not a secret… but it feels like a bad idea to have the default password printed next to the login feature.

Lucky for me, the device I was analyzing still had its' default password configured so I



Name	Date	Type	Size
certs	05/11/2018 06:55 PM	Folder	
license	05/11/2018 07:00 PM	Folder	
tones	12/31/1999 07:02 PM	Folder	
user.conf	01/12/2022 11:58 AM	Text File	7.167KB

My first move, is to attempt to download any of the files I am presented with here and to see what that HTTP request looks like using BurpSuite Professional… P.S. if you don't use BurpSuite Pro…. well I am just sad to hear that, it's an incredibly valuable tool and is priced super well in the lower hundreds per year ($300–400ish if I'm remembering correctly). Anyways… this is what I saw when reviewing the download request.

In this example, I'm downloading the "user.conf" file.

A natural change to attempt to make before resubmitting the request is, of course, to replace the file in the HTTP request that we want to fetch from the server with a directory traversal payload to see if we can "break out" of the file directory the application intends for us to browse and reach the "*passwd*" file in the "*/etc/*" directory of the underlying Linux OS.

Yehaw!

I was thrilled to see the output of the *passwd* file! Next, I thought to also grab the "*shadow*" file in the same directory (*/etc/*) which should contain hashed passwords for users! It is possible for us to maybe even crack the hash(es) if we can get them from the file.

Well, this is certainly awesome. I have the hashed password for the *root* account. This means I can possibly "jailbreak" or "root" the device if I can accomplish two things.

1. Crack the root password with Hashcat (my preferred hash cracking tool) or another password-hash cracking tool.
2. Find a way to execute code as root. My two theories on how to accomplish this was either to find a way to enable SSH (since it doesn't seem to be enabled on my target at least), or to leverage the directory traversal vulnerability while UPLOADING (instead of downloading as seen above). Perhaps I could drop a web shell or drop/overwrite a config file to get me further access.

I like Hashtcat for the GPU acceleration and the "Rules" engine that modifies a given wordlist on the fly! One or two of my first blog posts on https://n0ur5blog.wordpress.com/ (my old blog) talked a whole bunch about Hashcat!

if the web interface just uses whatever the root password is set to, and if changing the web interface password would change the root password. Regardless, I now have one last challenge and that is to turn this into some form of code execution via the methods I mentioned above!

As of the time of this writing I have not successfully executed code as the root user, but have confirmed I can upload files to outside the web root directory and even overwrite files already in place on the system! I am left with the following items to continue my research on this vulnerability.

- What other hardware and/or firmware versions are vulnerable to this? I have come across some other Algo devices containing default passwords for log-in that didn't even have the "File Manager" feature. The few I came across did have much newer firmware but also were different hardware from those that I've confirmed vulnerable so far. So I assume "File Manager" was either removed in newer firmware versions OR some hardware they produce simply doesn't have or need the File Manager.
- How can I use unrestricted file upload, combined with directory traversal to enable SSH or achieve some other form of remote code execution via ROOT. The web server seems serve "lua" files, so I assume if I can figure out the file structure/location of the web server root directory, I could possibly drop a web shell on it that would enable the remote code execution I'm looking for via web browser.
- Testing if changing the web interface password also changes the root password by grabbing the shadow file again after changing the password to see if the hash changed. If not, we can assume a large number of Algo devices ship with the root password of "*algo*".
- Create a python or bash script for this exploit once I discover the full extent of the items above!

Cheers!

N0ur5