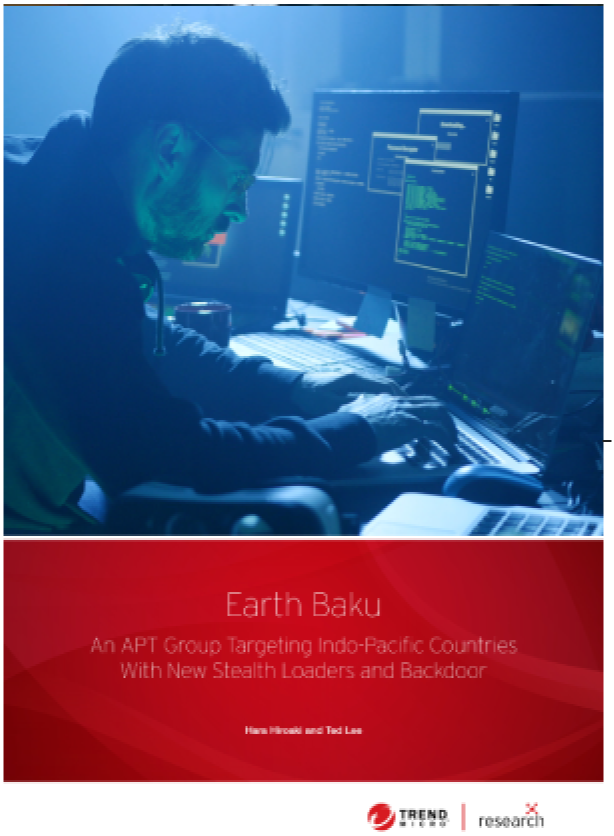
Earth Baku Returns: Uncovering the Upgraded Toolset Behind the APT Group’s New Cyberespionage Campaign

trendmicro.com/vinfo/us/security/news/cybercrime-and-digital-threats/earth-baku-returns



Earth Baku

An APT Group Targeting Indo-Pacific Countries With New Stealth Loaders and Backdoor

Hara Hiroaki and Ted Lee

TREND MICRO | research

Download Earth Baku: An APT Group Targeting Indo-Pacific Countries With New Stealth Loaders and Backdoor

Last year, we began studying new malware tools that surfaced as part of a cyberespionage campaign, which Earth Baku — a notorious advanced persistent threat (APT) group, better known as APT41 — had carried out against organizations in the Indo-Pacific region. While we have yet to determine the exact motives behind Earth Baku’s operations, we share our key findings from our analysis with a view to encouraging further research into this active campaign.

Victim profile

For this campaign, Earth Baku has leveled its attacks against entities in the airline, computer hardware, automotive, infrastructure, publishing, media, and IT industries. According to our detections, these organizations are located in the Indo-Pacific region. So far, we have registered hits in India, Indonesia, Malaysia, the Philippines, Taiwan, and Vietnam.

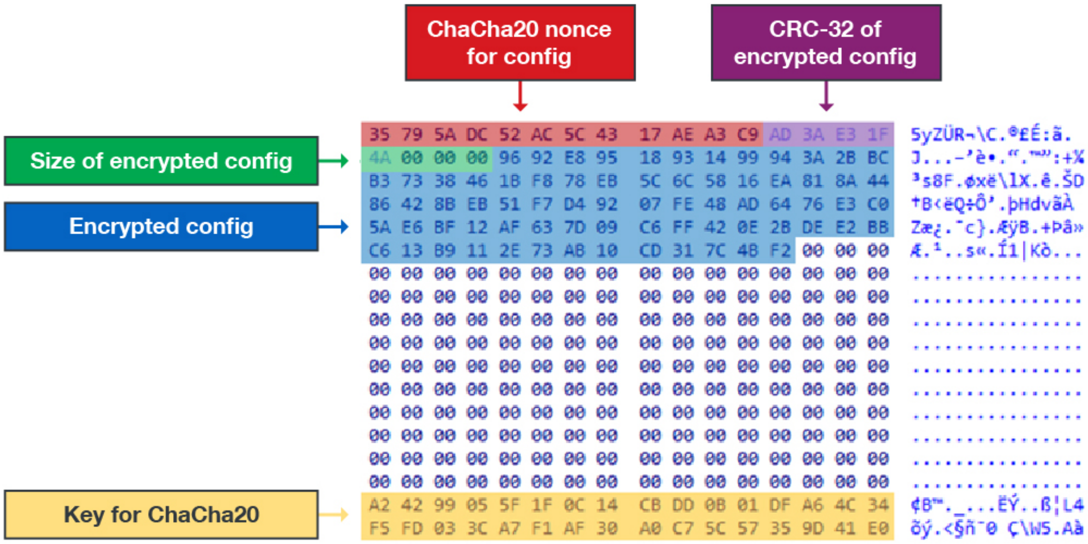
Figure 1. Countries affected by Earth Baku's campaign, all in the Indo-Pacific region
Source: Trend Micro™ Smart Protection Network™ infrastructure

New tools

Our analysis indicates that Earth Baku employs previously unidentified pieces of malware in this campaign: two shellcode loaders, which we have named StealthVector and StealthMutant, and a backdoor, which we have dubbed ScrambleCross.

The loaders: StealthVector and StealthMutant

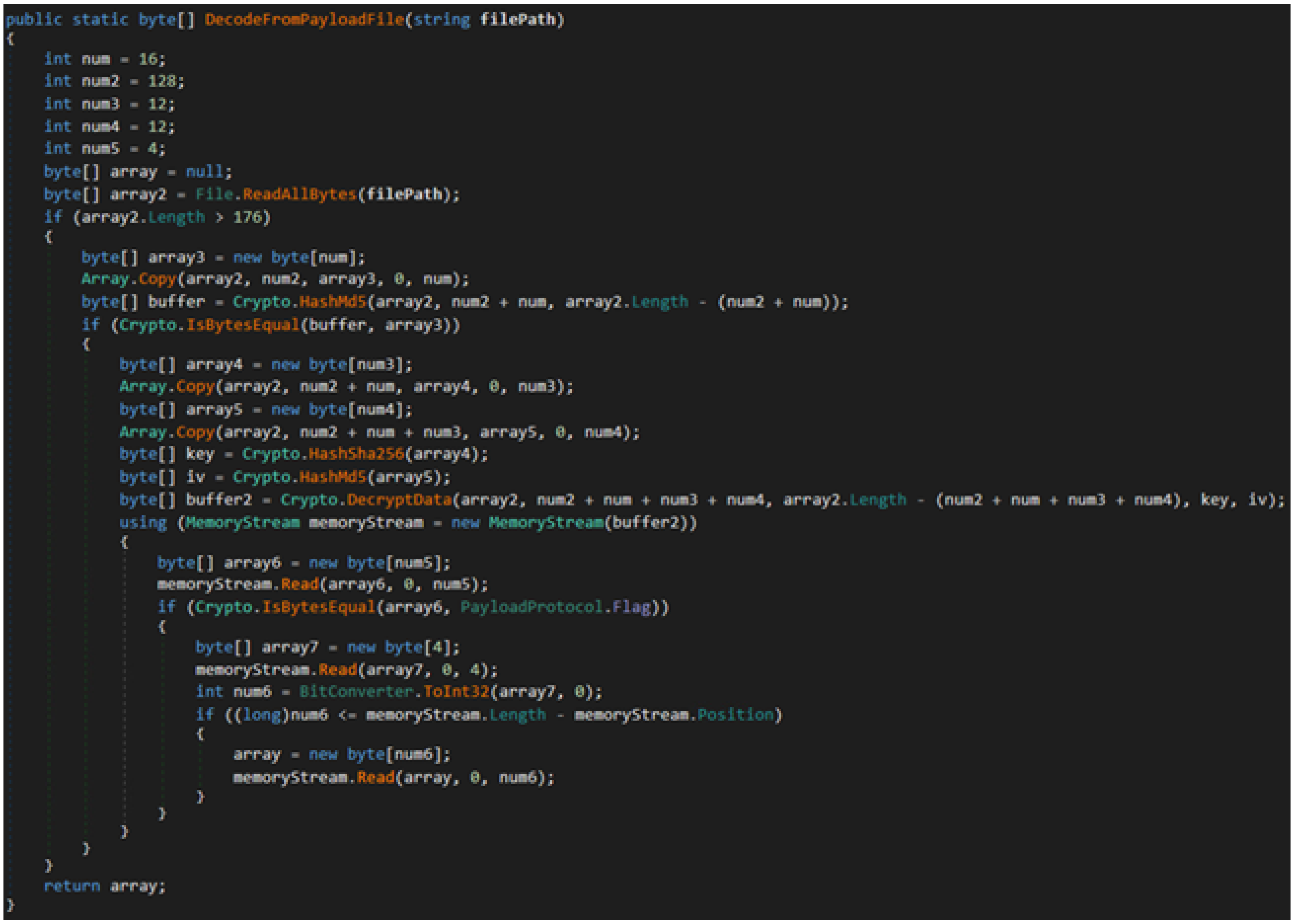
StealthVector, a shellcode loader written in C/C++, has various configurable features that malicious actors can easily implement without changing its source code. It can be configured to uninstall itself, run its payload in a specific location, avoid detection by disabling Event Tracing for Windows (ETW), and perform username checking for context awareness. StealthVector's configuration is difficult to decrypt because the loader is encrypted with the ChaCha20 routine and a fixed custom initial counter.



```
35 79 5A DC 52 AC 5C 43  17 AE A3 C9 AD 3A E3 1F  5yZÜR¬\C.®£É:ã.
4A 00 00 00 96 92 E8 95  18 93 14 99 94 3A 2B BC  J...–'è•.".™":+¼
B3 73 38 46 1B F8 78 EB  5C 6C 58 16 EA 81 8A 44  ³s8F.øxë\lX.ê.ŠD
86 42 8B EB 51 F7 D4 92  07 FE 48 AD 64 76 E3 C0  †B‹ëQ÷Ô'.þHdvãÀ
5A E6 BF 12 AF 63 7D 09  C6 FF 42 0E 2B DE E2 BB  Zæ¿.¯c}.ÆÿB.+Þâ»
C6 13 B9 11 2E 73 AB 10  CD 31 7C 4B F2 00 00 00  Æ.¹..s«.Í1|Kò...
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
A2 42 99 05 5F 1F 0C 14  CB DD 0B 01 DF A6 4C 34  ¢B™._...ËÝ..ß¦L4
F5 FD 03 3C A7 F1 AF 30  A0 C7 5C 57 35 9D 41 E0  õý.<§ñ¯0 Ç\W5.Aà
```

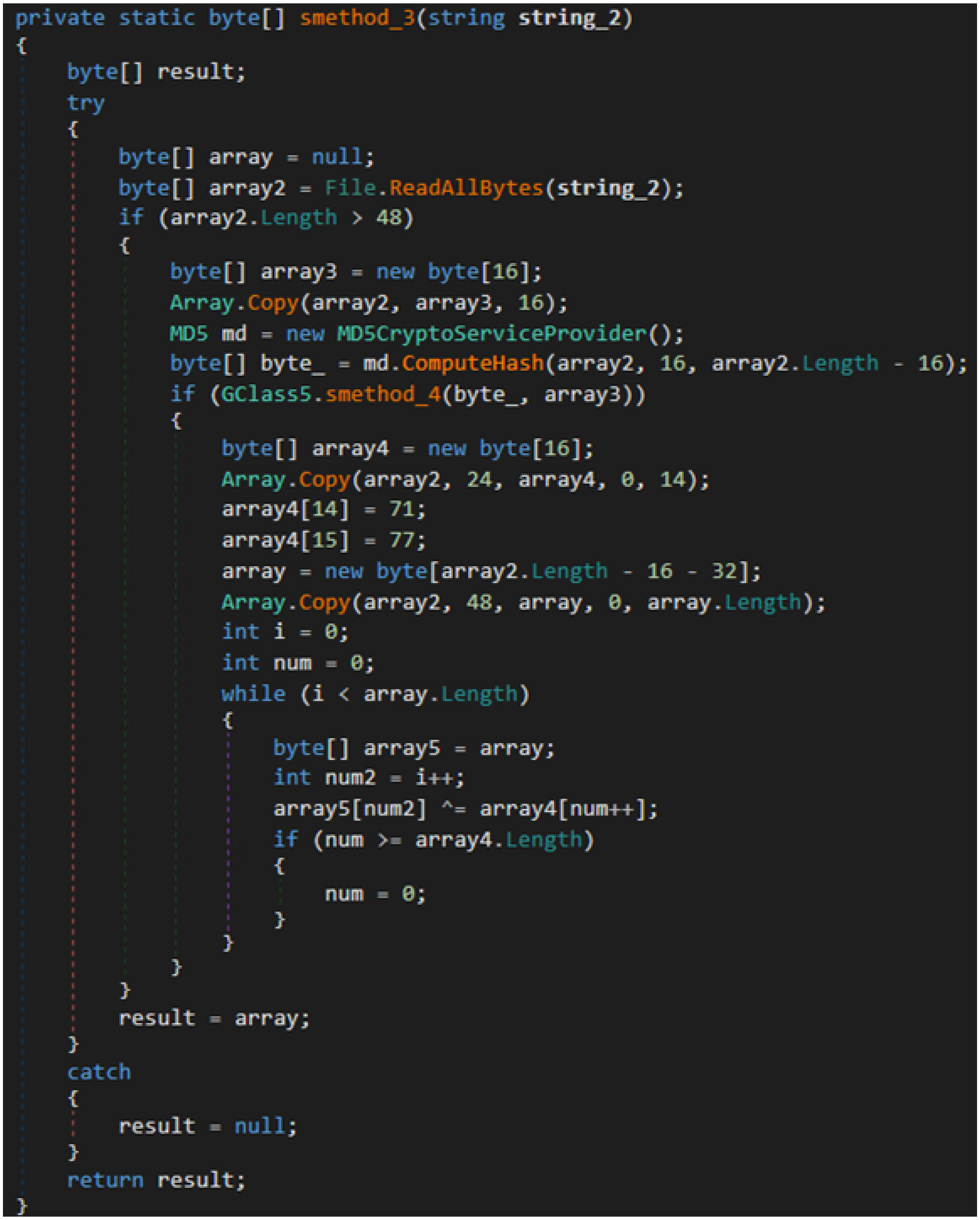
Figure 2. The locations of StealthVector's encrypted configuration and ChaCha20 key information

StealthMutant, a C# implementation of StealthVector, executes its payload by performing process hollowing, a technique widely used by both malicious actors and red teams. Like StealthVector, StealthMutant can disable ETW and go undetected by Windows' built-in logging system. Most of the StealthMutant samples we have observed use AES-256-ECB to decrypt their payloads, but we have also found older versions of this loader that use XOR instead.



```
public static byte[] DecodeFromPayloadFile(string filePath)
{
    int num = 16;
    int num2 = 128;
    int num3 = 12;
    int num4 = 12;
    int num5 = 4;
    byte[] array = null;
    byte[] array2 = File.ReadAllBytes(filePath);
    if (array2.Length > 176)
    {
        byte[] array3 = new byte[num];
        Array.Copy(array2, num2, array3, 0, num);
        byte[] buffer = Crypto.HashMd5(array2, num2 + num, array2.Length - (num2 + num));
        if (Crypto.IsBytesEqual(buffer, array3))
        {
            byte[] array4 = new byte[num3];
            Array.Copy(array2, num2 + num, array4, 0, num3);
            byte[] array5 = new byte[num4];
            Array.Copy(array2, num2 + num + num3, array5, 0, num4);
            byte[] key = Crypto.HashSha256(array4);
            byte[] iv = Crypto.HashMd5(array5);
            byte[] buffer2 = Crypto.DecryptData(array2, num2 + num + num3 + num4, array2.Length - (num2 + num + num3 + num4), key, iv);
            using (MemoryStream memoryStream = new MemoryStream(buffer2))
            {
                byte[] array6 = new byte[num5];
                memoryStream.Read(array6, 0, num5);
                if (Crypto.IsBytesEqual(array6, PayloadProtocol.Flag))
                {
                    byte[] array7 = new byte[4];
                    memoryStream.Read(array7, 0, 4);
                    int num6 = BitConverter.ToInt32(array7, 0);
                    if ((long)num6 <= memoryStream.Length - memoryStream.Position)
                    {
                        array = new byte[num6];
                        memoryStream.Read(array, 0, num6);
                    }
                }
            }
        }
    }
    return array;
}
```

Figure 3. A StealthMutant sample that uses AES-256-ECB for decryption



```
private static byte[] smethod_3(string string_2)
{
    byte[] result;
    try
    {
        byte[] array = null;
        byte[] array2 = File.ReadAllBytes(string_2);
        if (array2.Length > 48)
        {
            byte[] array3 = new byte[16];
            Array.Copy(array2, array3, 16);
            MD5 md = new MD5CryptoServiceProvider();
            byte[] byte_ = md.ComputeHash(array2, 16, array2.Length - 16);
            if (GClass5.smethod_4(byte_, array3))
            {
                byte[] array4 = new byte[16];
                Array.Copy(array2, 24, array4, 0, 14);
                array4[14] = 71;
                array4[15] = 77;
                array = new byte[array2.Length - 16 - 32];
                Array.Copy(array2, 48, array, 0, array.Length);
                int i = 0;
                int num = 0;
                while (i < array.Length)
                {
                    byte[] array5 = array;
                    int num2 = i++;
                    array5[num2] ^= array4[num++];
                    if (num >= array4.Length)
                    {
                        num = 0;
                    }
                }
            }
        }
        result = array;
    }
    catch
    {
        result = null;
    }
    return result;
}
```

Figure 4. A sample of an older StealthMutant version that uses XOR for decryption

The payloads: ScrambleCross and Cobalt Strike beacon

A shellcode-based backdoor, ScrambleCross is one of the two kinds of payloads found in StealthMutant and StealthVector samples, the other being the Cobalt Strike beacon. ScrambleCross fields backdoor commands to and from its command-and-control (C&C)

server, enabling it to receive and then manipulate plug-ins. Because we have yet to retrieve any plug-ins from its C&C server, we have not ascertained the full extent of ScrambleCross' plug-in manipulation capabilities.

Attack vectors

This campaign uses different means to enter and infect a target system:

- Injection of an SQL script into the system's Microsoft SQL Server to upload a malicious file
- Exploitation of the Microsoft Exchange Server ProxyLogon vulnerability CVE-2021-26855 to upload a malicious web shell
- Possible distribution through emails containing malicious attachments
- Use of the installer application *InstallUtil.exe* in a scheduled task

Attribution

This campaign is tied to one of Earth Baku's earlier cyberespionage campaigns, which the group is perpetrating under the alias APT41. This older campaign has been ongoing since November 2018 and uses a different shellcode loader, which we have named LavagokLdr, but these two campaigns are alike in many ways.

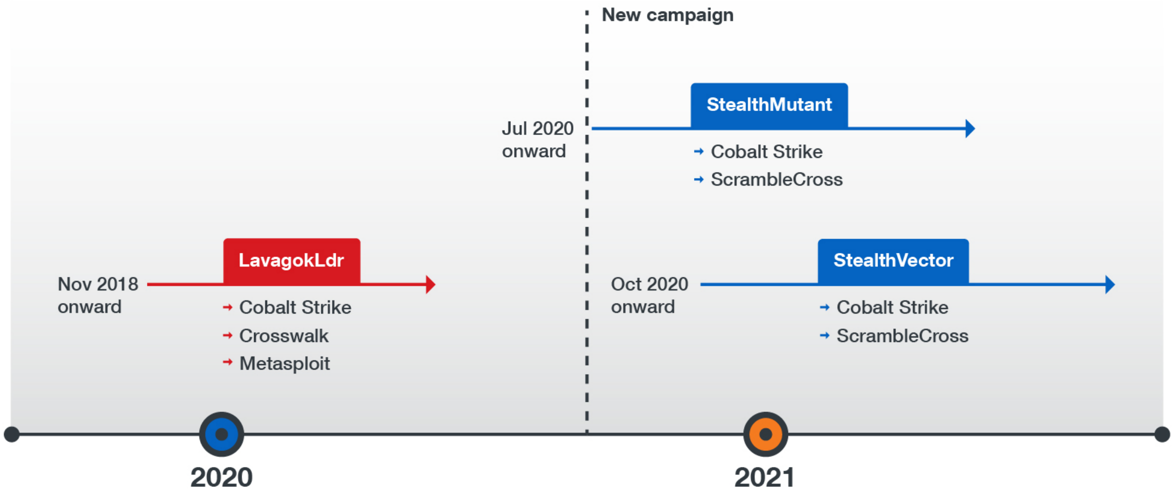


Figure 5. A timeline of Earth Baku's previous campaign and its new campaign

We have attributed this new campaign to Earth Baku on the basis of its code similarities to the other campaign:

- Both campaigns use the installer script called *install.bat*.

- Their shellcode loaders have the same kind of dynamic link library (DLL), *Storesyncsvc.dll*, and similar procedures for loading APIs.
- Their payloads perform similar processes for signature checking and decoding their main functions.

Skilled actors, upgraded tools

Our analysis of StealthMutant, StealthVector, and ScrambleCross demonstrates that Earth Baku has improved its malware tools since its last campaign. This suggests that the group's members specialize in different areas, including low-level programming, software development, and techniques used by red teams. While we have yet to ascertain Earth Baku's motives behind this campaign, the group has designed these sophisticated new tools to be easily modified and to avoid detection more efficiently when infiltrating a targeted network.

Our research paper "Earth Baku: An APT Group Targeting Indo-Pacific Countries With New Stealth Loaders and Backdoor" sheds more light on Earth Baku's operations in general and the capabilities of its new pieces of malware in particular. It also provides security recommendations that can help organizations protect their networks from campaigns like Earth Baku's.

HIDE

Like it? Add this infographic to your site:
1. Click on the box below. 2. Press Ctrl+A to select all. 3. Press Ctrl+C to copy. 4. Paste the code into your page (Ctrl+V).

Image will appear the same size as you see above.

Posted in Cybercrime & Digital Threats, Research, Targeted Attacks, Cybercrime