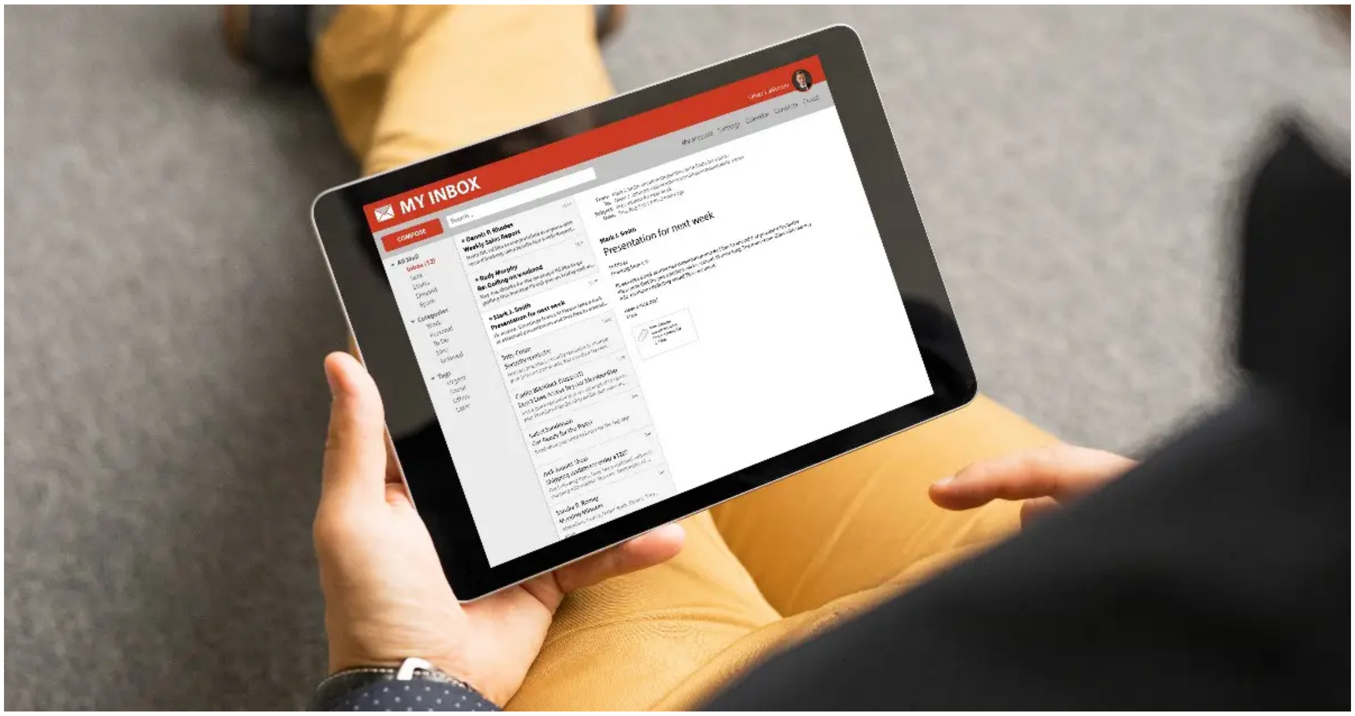
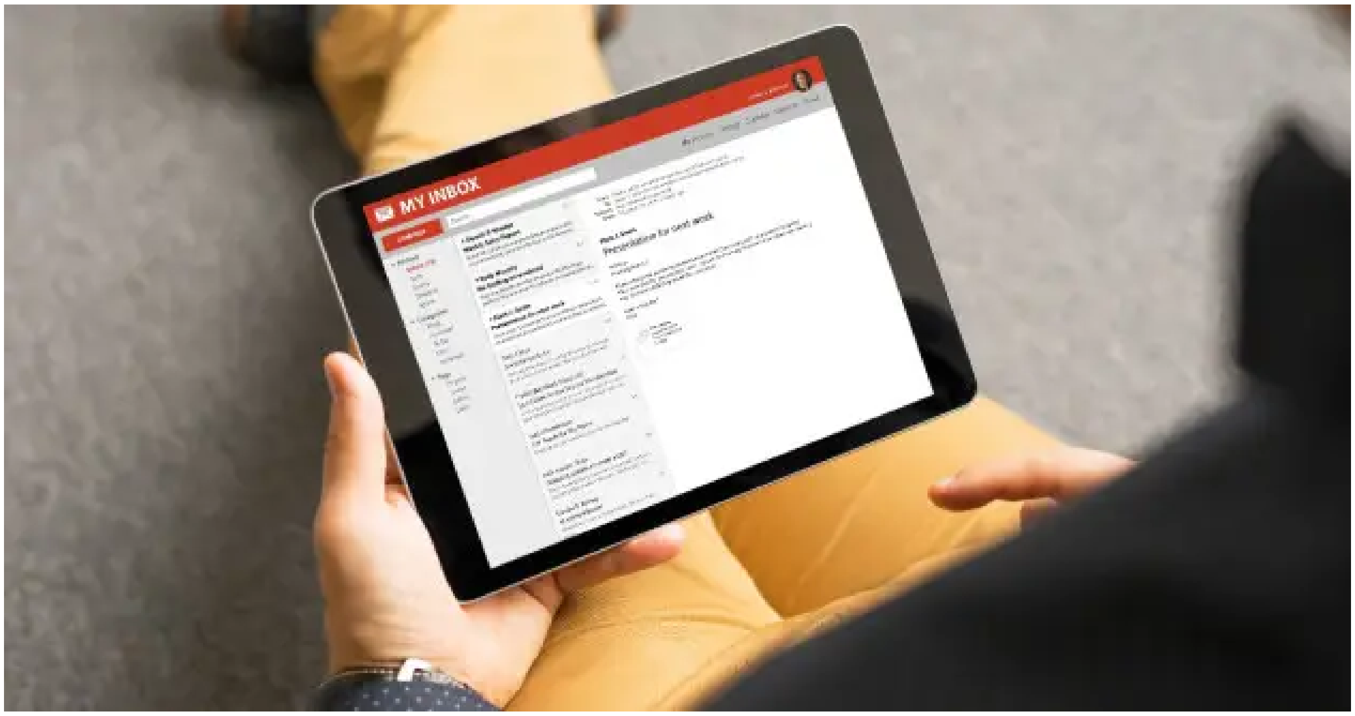
Trickbot Group's AnchorDNS Backdoor Upgrades to AnchorMail

securityintelligence.com/posts/new-malware-trickbot-anchordns-backdoor-upgrades-anchormail/





Malware February 25, 2022
By Charlotte Hammond co-authored by Ole Villadsen 6 min read

IBM Security X-Force researchers have discovered a revamped version of the Trickbot Group's AnchorDNS backdoor being used in recent attacks ending with the deployment of Conti ransomware. The Trickbot Group, which X-Force tracks as ITG23, is a cybercriminal gang known primarily for developing the Trickbot banking Trojan, which was first identified in 2016 and initially used to facilitate online banking fraud. The group has adapted in recent years to the ransomware economy by using its Trickbot and Bazarloader payloads to gain a foothold for ransomware attacks and through its close relationship with the Conti ransomware-as-a-service (RaaS).

ITG23 is also known for developing the Anchor malware framework, including the AnchorDNS variant, in 2018 for use during attacks on high-profile targets following initial infection by Trickbot or Bazarbackdoor, an additional backdoor developed by ITG23. AnchorDNS is notable for communicating with its Command and Control (C2) server using the DNS protocol. The upgraded backdoor, identified by IBM Security X-Force researchers as AnchorMail or Delegatz, now uses an email-based C2 server which it communicates with using SMTP and IMAP protocols over TLS. With the exception of the overhauled C2 communication mechanism, AnchorMail's behavior aligns very closely to that of its AnchorDNS predecessor.

The discovery of this new Anchor variant adds a new stealthy backdoor for use during ransomware attacks and highlights the group's commitment to upgrading its malware.

Upon execution, AnchorMail creates a scheduled task for persistence which is set to run every 10 minutes. It then collects basic system information, registers with its C2 and enters a loop of checking for and executing received commands. The backdoor's command structure is also very similar to that of AnchorDNS and both versions appear to accept the same set of command codes, which provide a variety of different options for executing commands and payloads received from the C2.

The most notable feature of AnchorMail is its novel C2 communication mechanism. The C2 server appears to utilize mail server code, and the backdoor communicates with it through the sending and receiving of specially crafted email messages. AnchorMail uses the encrypted SMTPS protocol for sending data to the C2, and IMAPS is used for receiving it. In the analyzed sample the C2 server address was configured as 15906-28547.bacloud[.]info (213.252.247[.]230), with port 465 used for SMTPS communications, and 993 used for IMAPS.

AnchorMail is written in C++ and has so far only been observed targeting Windows systems. However, as AnchorDNS has been ported to Linux, it seems likely that a Linux-variant of AnchorMail may emerge too.

Persistence

AnchorMail starts by enumerating the scheduled tasks on the system and searching for one which executes itself. If it does not find one then it proceeds to create a new scheduled task for persistence.

It enumerates folders within the task scheduler library and creates a task within a randomly selected folder, using the folder name as a prefix for the task description followed by one of the following strings: 'Task', 'Updater', 'Backup', 'Service', 'Maintenance'. For example, it may select the Bluetooth folder and create its task within that folder with description 'BluetoothTask'. The task is configured to run every day, at 10-minute intervals, and the execution path is set as **<malware_commandline_path>,dllmain**. Once the scheduled task is created, the malware then exits.

Otherwise, if the malware finds the scheduled task has already been created, then it continues with its main functionality.

Registration

Once persistence has been achieved, AnchorMail proceeds to collect basic system information and register with its C2 server. The registration process is almost identical to that of AnchorDNS.
AnchorMail first generates a system id with the following format:

```
<hostname>_W<windows_version>.<random_guid>
```

For example:

```
DESKTOP-4LUGU5I_W10019041.D9B3AEAB44F8ED2F85EECD3ED81463CA
```

The malware also attempts to identify the external IP address of the system by making requests to the following URLs:

```
checkip.amazonaws.com
ipecho.net/plain
ipinfo.io/ip
api.ipify.org
icanhazip.com
myexternalip.com/raw
wtfismyip.com/text
ip.anysrc.net/plain/clientip
```

The collected information is then combined with a group id, in this case '**lackey**', and randomly generated tokens, and formatted as a type **/0/** request as per below:

```
/<group_id>/<system_id>/0/<os_version>/1001/<external_ip>/<token_1>/<token_2>/
```

For example:

```
/lackey/DESKTOP-
4LUGU5I_W10019041.D9B3AEAB44F8ED2F85EECD3ED81463CA/0/Windows_10_x64/1001/0.0.0.0/5E348391753F5C9D25C757E540685864939D952
```

This data is then sent to the C2 and the malware expects to receive a response containing the second token in order to validate that the data was received successfully.

Network Communications

AnchorMail is notable for communicating with its C2 server using SMTP and IMAP protocols over TLS, referred to as SMTPS and IMAPS respectively. The C2 server appears to utilize mail server code, and the backdoor communicates with it through the sending and receiving of specially crafted email messages. AnchorMail uses the encrypted SMTPS protocol for sending data to the C2, and IMAPS is used for receiving it.

During startup, AnchorMail decrypts the server configuration which contains login credentials and server details. In the analyzed sample, these were as follows:

```
{[email protected]|ohvohNgaeT6Shoche8Ei|15906-28547.bacloud.info|15906-28547.bacloud.info}
{[email protected]|doo9ahxuuBug9cuuV8ga|15906-28547.bacloud.info|15906-28547.bacloud.info}{[email protected]}
```

However, a sample in the public domain was also identified with a different configuration:

```
{[email protected]|OaXah2shei6iL0Oohahj|15906-28547.bacloud.info|15906-28547.bacloud.info}
{[email protected]|xixo8eoCh3Aphae7phai|15906-28547.bacloud.info|15906-28547.bacloud.info}{[email protected]}
```

In the case of the analyzed sample, both the IMAP and SMTP servers are set to **15906-28547.bacloud[.]info**.

To initiate C2 communications, the malware performs a DNS request to retrieve the IP address of the server, which at the time of analysis is set to **213.252.247[.]230**. It then creates a TLS connection to SMTPS port 465 using the OpenSSL library, and then logs into the SMTP server using the configured credentials.

AnchorMail then crafts an email message containing the request string and any data to be sent to the C2. The recipient of the email is set as the final value from within the configuration, i.e. **z1[@]15906-28547.bacloud[.]info**.

AnchorMail generates a 16-byte GUID followed by a string such as '**1-1**', which it encodes using a custom encoding algorithm that encodes binary data into a series of lowercase letters, which it intersperses with spaces in order to make it appear like text. It then sets this encoded string as the email message subject.

The main request string and any accompanying data is also encoded using the encoding algorithm, then added to a zip file with the filename such as '**1-1.txt**'. Finally, it is encoded using base64 and added to the email body.

An additional set of data consisting of another 16 bytes value followed by the string '**GET**' is also encoded and added to the email body.

An example of a constructed email containing a type **/0/** request is follows:

From: <[email protected]>
To: <[email protected]>
Subject: Toduzu vypuwu asetadj tjwytu ycjsapo epeljhiru sohjsi jpjraze piwoge picuwe etetiwo tokoto wi.
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="4E7E98B61C1C3A56"

–4E7E98B61C1C3A56
Content-Type: application/octet-stream; name = "file.zip"
Content-Transfer-Encoding: base64

UEsDBBQAAAAIAMdjVFTnUv4FRgIAANUDAAAHAAAAMS0xLnR4dB1TQY4bMQw7t0D/oBfkD0XbQ08tFu1hj8JYzsgNYI0tzUT7+tI5BZ7IJEXSn9

–4E7E98B61C1C3A56
Content-Type: text/plain; charset="us-ascii"
Zdr vfl hdz krp srs rst rdl xts qvf rsz srp fps sp.

To receive responses from the C2, AnchorMail connects to the C2 server via IMAPS port 993 using TLS. It then logs into the IMAP server using the configured credentials and uses the IMAP LIST and STATUS commands to retrieve the status of the mailboxes. The malware then uses the SELECT and FETCH commands to retrieve messages which it parses until it finds one containing a unique identifier that matches one that the malware sent in its original request to the SMTP server.

The message contents are then decoded and the response and any commands and command parameters extracted.

Command Execution

If registration is successful, AnchorMail then sends a type **/1/** request to the C2 to check for any commands to be run.

```
/<group_id>/<system_id>/1/<token>/
```

The response from the C2 will be formatted as follows:

`<incode>/<group_id>/<system_id>/<token>/<cmdid>/<cmd_params>`

The **incode** field contains a number identifying the command which is to be run, and **cmd_params** contains the parameters for that command.

Overall, AnchorMail supports the following command codes.

Command Code: 0

Functionality:

Execute a given command using **%systemroot%\System32\cmd.exe**

Command Code: 1

Functionality:

- Retrieve executable file from C2 using type /5/ request.
- Save file to %temp% directory with a temp filename.
- Execute the dropped file via CreateProcess api.

Command Code: 2

Functionality:

- Retrieve executable file from C2 using request string specified in command params.
- Save file to %temp% directory with a temp filename.
- Execute the dropped file via CreateProcess api.

Command Code: 3

Functionality:

- Retrieve DLL file from C2 using type /5/ request.
- Save file to %temp% directory with a temp filename.
- Identify entrypoint by loading DLL and searching for the following exports: go, entry, run, start, exec, begin, StartW, ctrl, entryPoint, InitLib, dllmain
- Execute DLL at entrypoint using command **%systemroot%\System32\rundll32.exe <dll_path>,<entrypoint>**

Command Code: 4

Functionality:

- Retrieve DLL file from C2 using request string specified in command params.
- Save file to %temp% directory with a temp filename.
- Identify entrypoint by loading DLL and searching for the following exports: go, entry, run, start, exec, begin, StartW, ctrl, entryPoint, InitLib, dllmain
- Execute DLL at entrypoint using command **%systemroot%\System32\rundll32.exe <dll_path>,<entrypoint>**

Command Code: 5

Functionality:

- Retrieve file from C2 using type /5/ request.
- Inject file into new process using process hollowing technique and execute.
- Process injection target randomly selected from list: winhlp32.exe, write.exe, explorer.exe, svchost.exe, cmd.exe, notepad.exe, calc.exe, rundll32.exe

Command Code: 6

Functionality:

- Retrieve file from C2 using request string specified in command params.
- Inject file into new process using process hollowing technique and execute.
- Process injection target randomly selected from list: winhlp32.exe, write.exe, explorer.exe, svchost.exe, cmd.exe, notepad.exe, calc.exe, rundll32.exe

Command Code: 7

Functionality:

- Retrieve file from C2 using type /5/ request.
- Execute file via process doppelgänging technique.
- Process doppelgäng target randomly selected from list: winhlp32.exe, write.exe, explorer.exe, svchost.exe, cmd.exe, notepad.exe, calc.exe, rundll32.exe

Command Code: 8

Functionality:

- Retrieve file from C2 using request string specified in command params.
- Execute file via process doppelgänging technique.
- Process doppelgäng target randomly selected from list: winhlp32.exe, write.exe, explorer.exe, svchost.exe, cmd.exe, notepad.exe, calc.exe, rundll32.exe

Command Code: 9

Functionality:

- Create instance of cmd.exe with pipes attached.
- Execute specified command in cmd.exe by writing it to input pipe and then exit.

Command Code: 10

Functionality:

- Create instance of powershell.exe with pipes attached.
- Execute specified command in powershell.exe by writing it to input pipe and then exit.

Command Code: 11

Functionality:

- Retrieve shellcode from C2 using type /5/ request.
- Create a new virtual desktop with name **system_desktop** using CreateDesktopA.
- Create new process instances of mstsc.exe, explorer.exe, notepad.exe.
- Inject shellcode into these processes and execute code. It may also attempt to inject code into additional running processes

Command Code: 12

Functionality:

- Retrieve shellcode from C2 using request string specified in command params.
- Create a new virtual desktop with name **system_desktop** using CreateDesktopA.
- Create new process instances of mstsc.exe, explorer.exe, notepad.exe.
- Inject shellcode into these processes and execute code. It may also attempt to inject code into additional running processes

Command Code: 100

Functionality:

> Uninstall. Delete scheduled task and delete malware file.

The type /5/ request used by some of the commands to request payload files has the following format:

```
/<group_id>/<system_id>/5/<file_name>/
```

Once the specified command has been performed, results are sent back to the C2 using a type /10/ request.

```
/group_id>/<system_id>/10/<incode>/<cmdid>/<result>/
```

More cybersecurity threat resources are available here.

Charlotte Hammond
Malware Reverse Engineer, IBM Security

Charlotte is a malware reverse engineer for IBM Security's X-Force IRIS team. She has been working in the security industry for more than 7 years with a focu...

think 2022 IBM

IBM Think Broadcast
Let's think together.

Watch on demand →