CRIMEWARE

Deep Dive Into TrickBot Executor Module “mexec”: Hidden “Anchor” Bot Nexus Operations

JASON REAVES / APRIL 8, 2020

New “mexec” module delivers tertiary malware and allows TrickBot to pivot within a network, deploy a variety of payloads and evade common detection methods.

By Jason Reaves, Joshua Platt & Vitali Kremez

Executive Summary

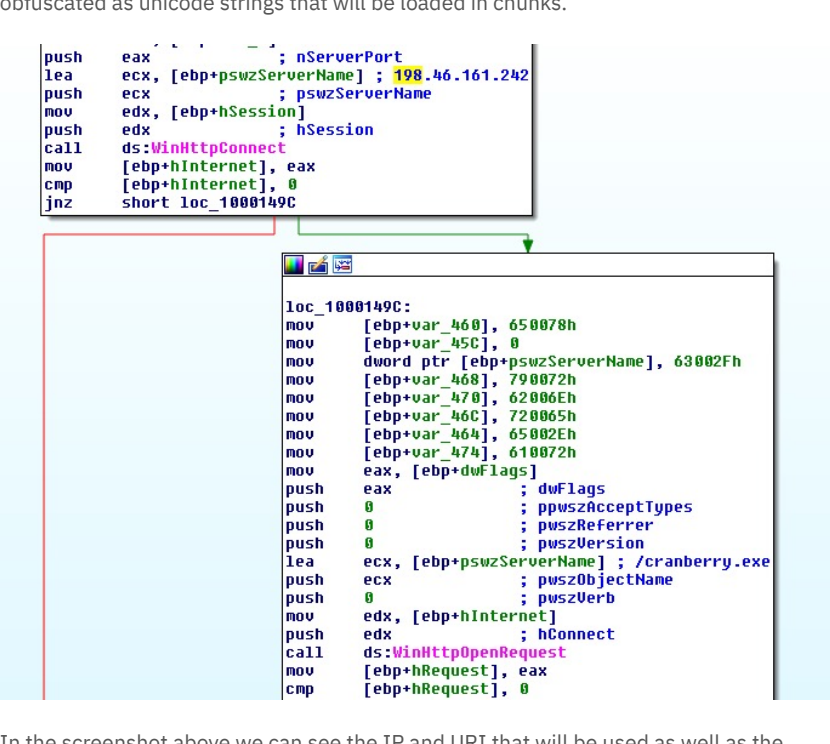
- TrickBot continues to be one of the most potent and actively developed malware frameworks in use on the crimeware landscape.
- TrickBot loads many modules leveraged for various tasks such as secondary tasks that normally revolve around credential loss, system and network profiling, data harvesting and propagation.
- The new executor `mexec` module is special in that it is primarily deployed for delivering tertiary malware which allows the TrickBot group threat actors to pivot within the compromised network environment while deploying different-purpose malware payloads and evading detection.
- The `mexec` module is primarily considered to be a loading module as its job is to load another malware payload onto the system.
- The module comes in two flavors: one as a “downloader” that will download and execute an arbitrary file and another as a “dropper” that embeds another malware within the `mexec` body to be dropped on the system.

Background

TrickBot is the successor of Dyre [1,2], and at first was primarily focused on banking fraud and utilized injection systems in the same manner. Over the years, TrickBot has shifted focus to enterprise environments to incorporate everything from network profiling and mass data collection to lateral traversal exploits. This focus shift is also prevalent in their incorporation of malware and techniques in their tertiary deliveries that are targeting enterprise environments. Such behavior is similar to a company where the focus will shift depending on what generates the best revenue.

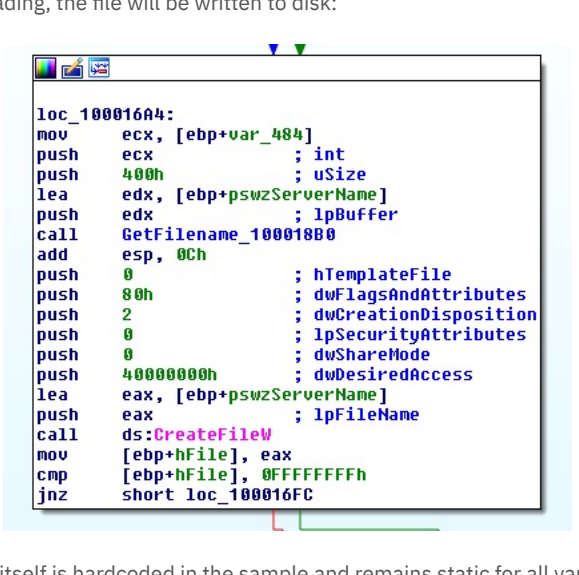
Research Insights

The `mexec` module, a possible initial internal naming for “memory executor”, acts as a downloader and can be described as a tool that can be detonated in memory designed to download and execute another executable. Most of the important strings are obfuscated as unicode strings that will be loaded in chunks.

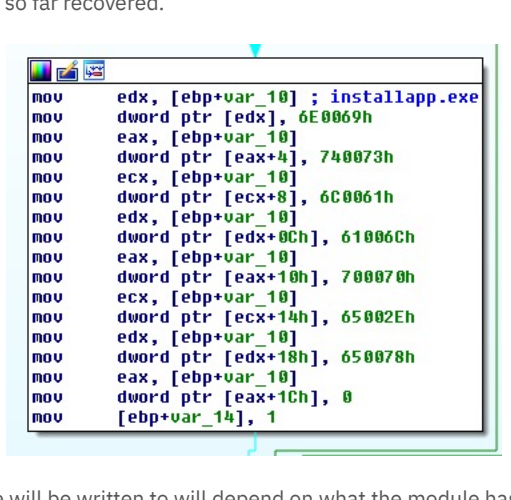


In the screenshot above we can see the IP and URI that will be used as well as the obfuscation of dynamically rebuilding the strings on the fly that was previously mentioned.

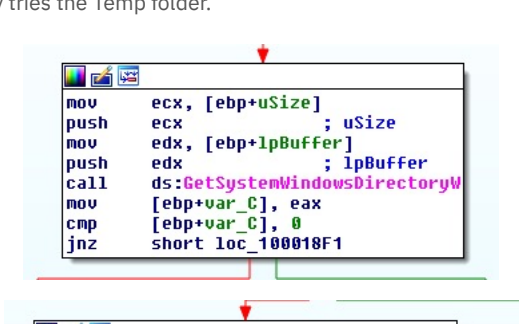
After downloading, the file will be written to disk:

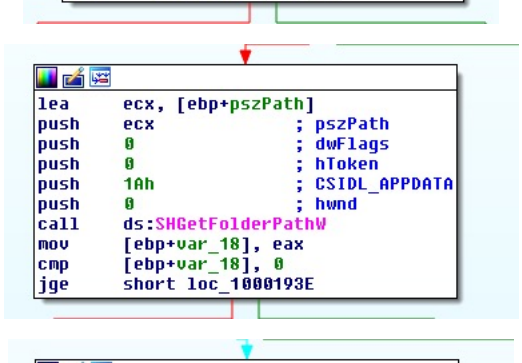


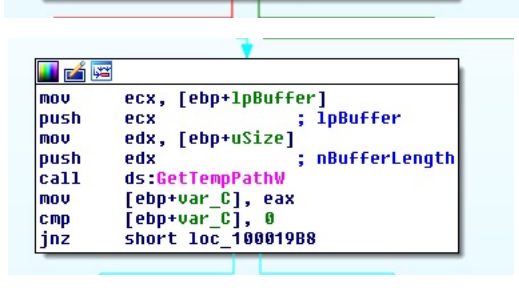
The filename itself is hardcoded in the sample and remains static for all variants and samples we have so far recovered.



The folder the file will be written to will depend on what the module has access to. First, it checks if it can write to the Windows system folder; if not it tries the AppData folder and finally tries the Temp folder.







Notably, the downloader also sets up process security information to adjust downloader permission leveraging via a sequence of Windows API GetNamedSecurityInfoW, SetEntriesInAclW, SetNamedSecurityInfoW. The possible security control list implementation is aimed to bypass file execution prevention as downloaded from a remote location.

Two other samples of `mexec` were recovered during our ongoing research:

SHA256:
5b729cd36cf3f0fdcfa0020b1f0f3bb98f9b456005814e61349bfdc50f390a7e
SHA1: f82753b1d526da357e4cbcfa24e80e79422b8bce
URL: 172.82[.]152[.]15/blueberry.exe

SHA256:
cd2e0341119cfbf734917f83d91a14d5855906a83066649bd49689e504181330
SHA1: d0a1bcc0df0ff70b5fb90704adab7fee734fc21d
URL: 172.82[.]152[.]15/aspen.exe

Pivoting on this IP in VirusTotal shows a number of URLs that look like TrickBot deliveries but also an EXE file that has the same naming structure as previously seen.

The sample downloaded as cloudberry.exe turns out to be the DNS variant of Anchor TrickBot[3], which is referenced as the gtag ‘anchor_dns’.

The discovery of a `mexec` module used by TrickBot that is designed to be a loader is notable and is further evidence of the link between TrickBot and Anchor operations. In many aspects, the Anchor malware remains to be the adopted custom flexible version of the TrickBot fork codebase deployed on some of the most notable high-value government and corporate targets.

The new module also brings to light a feature within TrickBot that is commonly taken for granted: its ability to deliver other malware. This module adds another loading avenue to the existing arsenal present within TrickBot. In a follow up to this report, we will discuss a variant of `mexec` that delivers malware samples that are onboard instead of downloading them, which sheds more light on this connection between TrickBot and Anchor.

Delivery Names Discovered for `mexec` Downloader Variant

mexecDll(32|64)
mexectDll(32|64)
aexecDll(32|64)
onixDll(32|64)

TrickBot File Indicators

AppDataRoaming[^]+injectDll(32|64).dll
AppDataRoaming[^]+systeminfo(32|64).dll
AppDataRoaming[^]+pwgrab(32|64).dll
AppDataRoaming[^]+anubis(32|64).dll
AppDataRoaming[^]+shadnew(32|64).dll
AppDataRoaming[^]+onixDll(32|64).dll

Generic

AppDataRoaming[^]+[a-zA-Z]+(32|64).dll$
AppDataRoaming[^]+[a-zA-Z]+(32|64)_configs*

Observed `mexec` Filenames

Windowssystem32installapp.exe
Windowssyswow64installapp.exe
%AppData%installapp.exe
Tempinstallapp.exe

Indicators of Compromise

Download URLs

```
hxxp://172[.]82[.]152[.]15/cloudberry.exe
hxxp://64.91[.]251[.]250/UgaNda73n
hxxp://85.204[.]116[.]245/chishir.exe
hxxp://104[.]168[.]98[.]206/Nuclear.exe
hxxp://198[.]46[.]161[.]242/cranberry.exe
hxxp://185[.]98[.]87[.]185/Atomic.exe
hxxp://107.173.160[.]14/blueberry.exe
hxxp://107.173.160[.]14/cranberry.exe
```

OSINT mexec samples

```
SHA1: 3ef000cb90ab638ab0bae542c2d6e8e6ec146c53
SHA1: 0e29a1f93b003c31af46ab1ab7c8d3df150123e0
SHA1: dacd5b49ac628157fcb9cf8d6e537e851ef29a64
```

YARA

```
rule anchor_dns_32
{
meta:
author="Jason Reaves"
strings:
    $a1 = "/1001/" ascii wide
    $a2 = ":$GUID" ascii wide
    $a3 = ":$TASK" ascii wide
    $ua = "WinHTTP loader/1.0" ascii wide
    $hexlify = {0f be ?? ?? b8 f0 00 00 00 0f 45 ?? 8b ?? c1 e1 02 23 d0}
    $sdecode = {8a 04 0a 0f be c0 83 e8 ?? 88 04 0a 42 83}
    $xor_data = {80 b4 05 ?? ?? ff ff ?? 40 3b c6}

condition:
    3 of them
}

rule anchor_dns_64
{

meta:
author="Jason Reaves"
strings:
    $xor_data = {80 ?4 0? ?? ?? 48 ?? c? 48}
    $hexlify = {81 c1 f0 00 00 00 23 d1 41 8? ?? c1 e1 02}
    $a1 = "/1001/" ascii wide
    $a2 = ":$GUID" ascii wide
    $a3 = ":$TASK" ascii wide
    $ua = "WinHTTP loader/1.0" ascii wide

condition:
   3 of them
}
```

References

1: https://blog.malwarebytes.com/threat-analysis/2016/10/trick-bot-dyrezas-successor/
2: https://www.fidelissecurity.com/threatgeek/archive/trickbot-we-missed-you-dyre/
3: https://www.sentinelone.com/wp-content/uploads/the-deadly-planeswalker-how-the-trickbot-group-united-high-tech-crimeware-apt/

JASON REAVES

Jason Reaves is a Principal Threat Researcher at SentinelLabs who specializes in malware reverse-engineering. He has spent the majority of his career tracking threats in the Crimeware domain, including reverse-engineering data structures and algorithms found in malware in order to create automated frameworks for harvesting configuration and botnet data. Previously, he worked as a software developer and unix administrator in the financial industry and also spent six years in the U.S. Army. Jason holds multiple certifications related to reverse-engineering and application exploitation and has published numerous papers on topics such as writing malware scripts pretending to be a bot, unpackers, configuration data harvesters and covert channel utilities. He enjoys long walks in IDA and staring at RFCs for hours.