New WhiteShadow downloader uses Microsoft SQL to retrieve malware | Proofpoint US

By Bryan Campbell and Jeremy Hedges with the Proofpoint Threat Insight Team

Published: 2019-09-26 · Archived: 2026-04-05 22:38:37 UTC

Overview

While not a new development, the use of Microsoft SQL queries to retrieve next-stage payloads has been relatively rare as a form of malware distribution. In August 2019, however, Proofpoint researchers encountered new Microsoft Office macros, which collectively act as a staged downloader that we dubbed "WhiteShadow." Since the first observed occurrence of WhiteShadow in a small campaign leading to infection with an instance of Crimson RAT, we have observed the introduction of detection evasion techniques. These changes include ordering of various lines of code as well as certain basic obfuscation attempts.

In August 2019, the macros that make up WhiteShadow appeared in English-language cleartext. The only observed obfuscation technique was in the simple case altering of strings such as **"Full_fILE"** or **"rUN_pATH."** In early September, we observed slight misspellings of certain variables such as **"ShellAppzz.Namespace(Unzz).**" Mid-September brought another change in macro code using reversed strings such as **"StrReverse("piz.Updates\stnemucoD\")"**. The most recently observed versions of the WhiteShadow macros contain long randomized text strings such as **"skjfhskfhksfhksfhksjfh1223sfsdf.eDrAerTerAererer".** Overall, the macro code still remains mostly human readable, with the same functionality as previously observed.

When recipients open malicious document attachments in these campaigns and activate macros, WhiteShadow operates by executing SQL queries against attacker-controlled Microsoft SQL Server databases. The malware is stored as long strings that are ASCII-encoded within the database. Once retrieved, the macro decodes the string and writes it to disk as a PKZip archive of a Windows executable. Once extracted by the macro, the executable is run on the system to start installing malware, which is determined by the actor based on the script configuration stored in the malicious Microsoft Office attachments.

Early campaigns delivered Crimson malware, which has historically been linked to specific actor activity. However, Proofpoint researchers currently have no evidence tying this current round of malware distribution with previous Crimson campaigns [1].

Campaigns

In August 2019, Proofpoint researchers began observing a series of malicious email campaigns distributing Microsoft Word and Microsoft Excel attachments containing the WhiteShadow downloader Visual Basic macro (Figure 1).

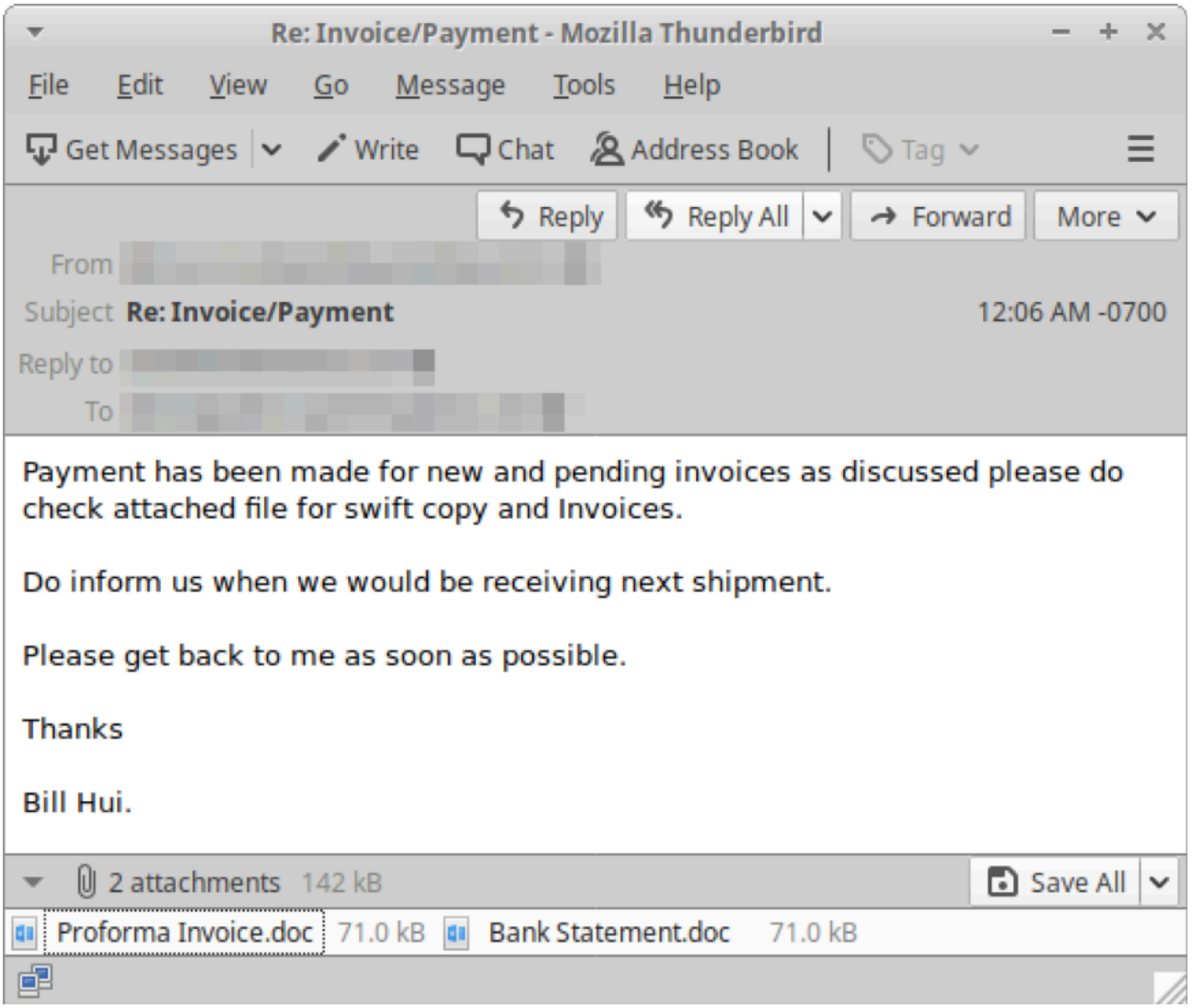

Figure 1: A malicious email from a threat actor containing Microsoft Word attachments that, when opened, will execute Visual Basic macros that collectively comprise the WhiteShadow downloader. WhiteShadow then installs additional malware.

It appears that WhiteShadow is one component of a malware delivery service, which includes a rented instance of Microsoft SQL Server to host payloads retrieved by the downloader.

The following chart is a chronology of campaigns, relative message volume, and payload summary since we first observed WhiteShadow in the wild.

Date	Relative Volume	Geo/Language	Attachment Type	Payload
8/26/2019	Low	AU/English	Excel Document	Crimson

8/29/2019	Low	AU/English	Excel Document	Crimson
9/2-9/4/2019	Low	AU, UK, China/English	Word and Excel Documents	Nanocore, njRAT, AgentTesla, Crimson
9/9/2019	Low	Chinese	Word Document	Crimson
9/12/2019	Low	UK/English	Word Document	Crimson
9/16-9/18/2019	Medium	US/English	Excel Document	AZORrult
9/16-9/17/2019	Low	English	Excel Document	Formbook
9/17-9/18/2019	Low	US/English	Excel Document	Nanocore
9/20/2019	Low	English	Excel Document	Nanocore
9/20/2019	Low	English	Excel Document	Crimson
9/24/2019	Low	US/English	Word Document	Crimson

Table 1: Chronology and relative volume of August and September 2019 WhiteShadow campaigns.

Downloader Analysis

WhiteShadow uses a SQLOLEDB connector to connect to a remote Microsoft SQL Server instance, execute a query, and save the results to a file in the form of a zipped executable.

The SQLOLEDB connector is an installable database connector from Microsoft but is included by default in many (if not all) installations of Microsoft Office. Once the connector is installed on the system, it can be used by

various parts of the Windows subsystem and by Visual Basic scripts including macros in Microsoft Office documents. We have observed several malware strains downloaded in this manner by WhiteShadow:

- Agent Tesla
- AZORult
- Crimson
- Nanocore
- njRat
- Orion Logger
- Remcos
- Formbook

The malware infection occurs in the following sequence:

- A user enables macros in a document or spreadsheet
- The macro reaches out to a Microsoft SQL server and pulls an ASCII string from the **'Byte_data'** column in the database table specified by a hardcoded **'Id_No'** in the macro
- The Macro 'decodes' the ASCII string and writes the data to a file in binary mode
    - Pseudo Format: **<byte><separator><byte><separator><byte>....**
    - See figures 3 and 4 for examples of the macro code that splits the data into an array before writing it to disk
- The file type of the decoded files have always been a ZIP to date, with a single executable inside
- The macro will then extract the executable from the ZIP and run it. The executable will be one of the malware payloads documented above

The sequence is illustrated in Figure 2:

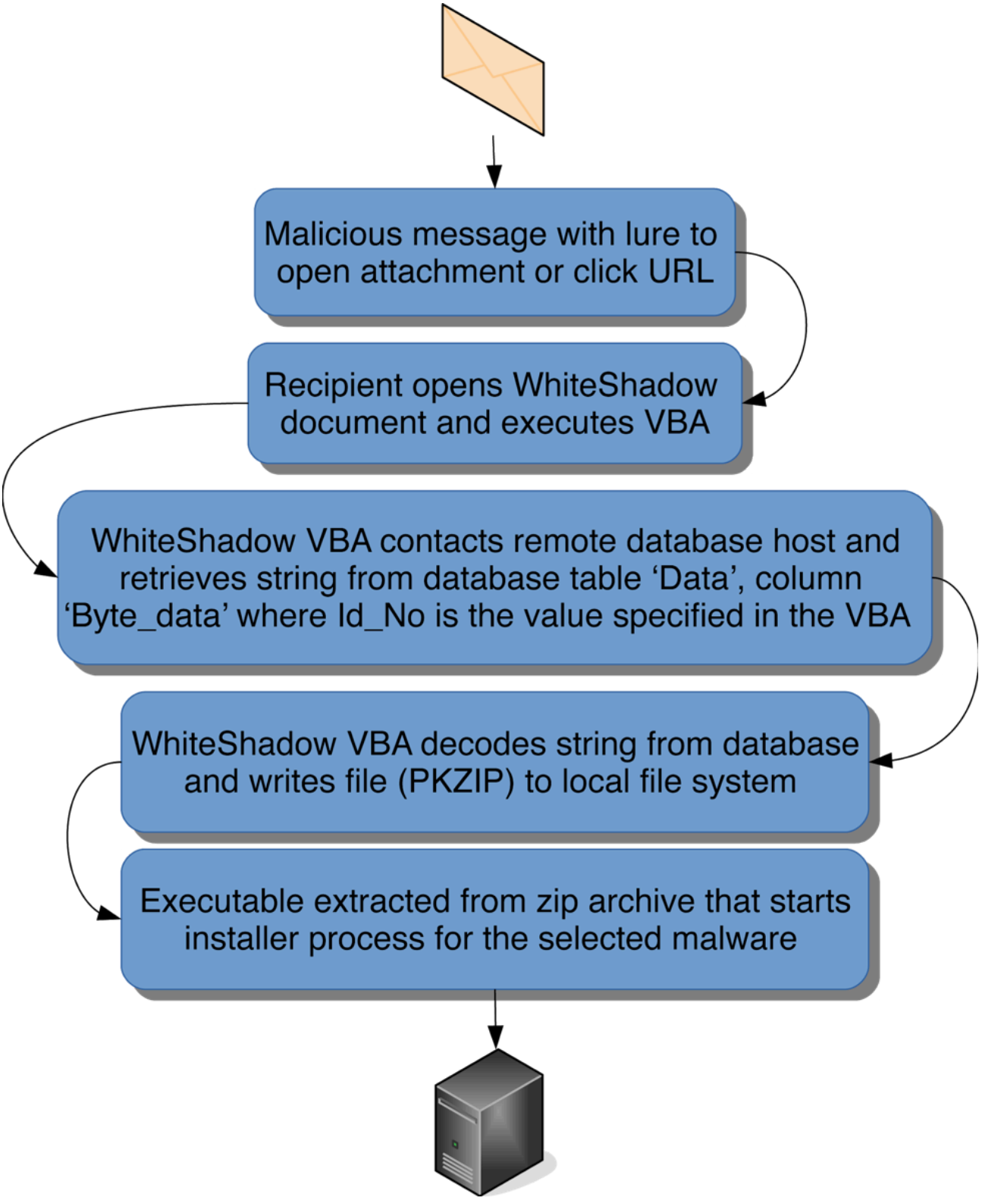

Figure 2. Illustration of WhiteShadow downloader and malware infection sequence.

```
Public Function Spliterzz()

Dim Array_bt() As String

Dim Bt As String

Bt = TextLoder_Frm.Bits_txt.Text

Array_bt = Split(Bt, "!")

Holder.DataStr = Array_bt

End Function
```

Figure 3. Sample code from WhiteShadow script with data decoding routine for array splitting using separator value of '!'.

```
Public Function Spliterzz()

Dim Array_bt() As String

Dim Bt As String

Bt = TextLoder_Frm.Bits_txt.Text

Array_bt = Split(Bt, ",")

Holder.DataStr = Array_bt

End Function
```

Figure 4. Sample code from WhiteShadow script with data decoding routine for array splitting using separator value of ','.

```
80#75#3#4#20#0#0#0#8#0#179#141#250#78#162#211#
228#47#4#159#0#0#0#30#1#0#16#0#0#0#84#111#
114#95#115#101#114#118#105#99#101#115#46#101#120#101#236#253#
7#92#84#73#243#48#10#247#153#204#12#12#12#97#134#12#
138#224#200#204#32#193#0#24#17#49#103#16#193#76#18#81#
```

Figure 5: ASCII representation of the 'encoded' PKZIP file before written to disk, line breaks added for comparison to Figure 6

```
0000:0000 080 075 003 004  020 000 000 000  008 000 179 141  250 078 162 211 PK........³.úN¢Ó
0000:0010 228 047 004 159  000 000 000 030  001 000 016 000  000 000 084 111 ä/.............To
0000:0020 114 095 115 101  114 118 105 099  101 115 046 101  120 101 236 253 r_services.exeìý
0000:0030 007 092 084 073  243 048 010 247  153 204 012 012  012 097 134 012 .\TIó0.÷.Ì...a..
0000:0040 138 224 200 204  032 193 000 024  017 049 103 016  193 076 018 081 .àÈÌ Á...1g.ÁL.Q
```

Figure 6: Hex editor representation of the 'decoded' PKZIP file as written to disk.

We have observed multiple databases all hosted on a subdomain of **mssql.somee[.]com:**

antinio.mssql.somee[.]com

bytesdata.mssql.somee[.]com

fabancho.mssql.somee[.]com

In each of the databases, the data that was being accessed by WhiteShadow was stored in a table called 'Data,' which always contained three columns:

1. **Id_No** ; A Primary Key 'int' identifier for the payload
2. **Byte_data** ; an encoded ASCII representation of the payload data
3. **Net_ver** ; This is likely a 'customer' identifier or versioning string for the payload

Proofpoint researchers have observed rows being added, removed, and in rare cases, updated in these databases.

Proofpoint researchers also noticed similarities in some cases with the data in the **'Net_Ver'** column and Affiliate/Group structures in some of the malware that was being dropped relative to a specific **'Id_No';**

Examples:

DB Subdomain	DB 'Id_No'	DB 'Net_ver'	Malware Config Affiliate	Malware Family
antinio	7	Del ec	DEL AEC V2	Crimson
bytesdata	4	jay2	JAY_V2	Crimson
bytesdata	9	oncode	oncode_Ver:1	Crimson
fabancho	2	oncode	Oncode-2	Crimson

Table 2: Illustration of how the Net_ver identifier column corresponds with the associated configured malware identifier

It should be noted that bytesdata's **Id_No '4'** was first identified as Nanocore, but was updated two days later to be Crimson, where the **Net_ver** was updated from **'jay'** to **'jay2'**

We also observed similarities between multiple MSSQL hosts, suggesting they are likely managed by the same actor:

bytesdata.mssql.somee[.]com -> Id_No: 9; Net_Ver: oncode

fabancho.mssql.somee[.]com -> Id_No: 2; Net_Ver: oncode

fabancho.mssql.somee[.]com -> Id_No: 2; Net_Ver: nano oncode

In addition to the **Net_ver** similarities (likely customer similarities, or perhaps repeat customers), it is clear that there is table schema reuse between the multiple databases which indicate the underlying architecture for these multiple databases are in fact related.

Payload Analysis

While analyzing the various payloads hosted in the database, we discovered one family of malware in particular - Crimson - that had several updates since our last analysis of the malware [2]. Updated commands are listed below:

cownar:

Adds an executable to Environment.SpecialFolder.CommonApplicationData\\%install_folder%\\updates\\ and executes it via Process.Start(exe_path);

cscreen:

Captures a JPEG format screenshot of the infected machine and sends it to the C&C using the C&C response command:

capScreen

getavs:

This is similar to the previously documented 'procl' command; it creates a concatenated string of processes with a format similar to:

>%process-id%>%process_module_name%><

for each process running on the system, enumerated via:

Process[] processes = Process.GetProcesses();

putsrt:

The input to this function is a string and it compares the string to the current running process executable path. If that path is different, it 'moves' the executable via:

File.WriteAllBytes(text, File.ReadAllBytes(executablePath));

It will then install the modified path into the common ‘CurrentVersion AutoRun’ registry key:

SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run

The remaining non-Crimson samples appeared to be largely the same as those currently documented and observed in the current threat landscape.

Conclusion

Although not a new development, the use of MSSQL queries to retrieve next-stage payloads is unusual in the wild. In late August 2019, Proofpoint researchers encountered a new staged downloader which uses this uncommon method and named it “WhiteShadow”. More importantly, this appears to be a new malware delivery service, allowing a range of threat actors to potentially incorporate the downloader and associated Microsoft SQL Server infrastructure into their attacks. We have observed actors using WhiteShadow to install RATs, downloaders, and keyloggers including Agent Tesla, AZORult, Crimson, and others. Organizations need to be cognizant of both the incoming malicious email and outbound traffic on TCP port 1433 which should be blocked or at least restricted on modern ACL configurations in firewalls today. Currently, these campaigns are relatively small, with message volumes in the hundreds and thousands, but we will continue to monitor associated trends.

References

[1]https://unit42.paloaltonetworks.com/tag/subaat/

[2]https://www.proofpoint.com/sites/default/files/proofpoint-operation-transparent-tribe-threat-insight-en.pdf

Indicators of Compromise (IOCs)

IOC	IOC Type	Description
antinio.mssql.somee[.]com	hostname	Dropper Next Stage
BytesData.mssql.somee[.]com	hostname	Dropper Next Stage

fabancho.mssql.somee[.]com	hostname	Dropper Next Stage
2cf21ddd9d369a2c88238606c5f84661cf0f62054e5cc44d65679834b166a931	SHA256	WhiteShadow Maldoc
ed8f4a7f09e428ceff8ede26102bb153b477b20775a0183be4ca2185999d20c8	SHA256	WhiteShadow Maldoc
539087ebb1d42c81c3be48705d153d75df550c047cf1056721f68724b78b73b7	SHA256	WhiteShadow Maldoc
dc90b12b71c4f8c08a789a5ec86ef9b05189d499c887f558f35eeb5e472551a0	SHA256	WhiteShadow Maldoc
fc068dda0efdaaa003c87bccd1880bc8953f18c2a8f1f0527a9a44e637e1fcce	SHA256	WhiteShadow Maldoc
a710a685bb4fbace08e26e32a8bb8a58665973cd802a3df2cb28581c287446e5	SHA256	WhiteShadow Maldoc
9cd62748e7be536f9bfb46493fc9704a93e4e4bcb38ef193b5d66e4a875756bc	SHA256	WhiteShadow Maldoc
95dbabe512ba4fc45e32786e87c292fb665e18bc0e2fea1cadb43ba1fe93f13b	SHA256	WhiteShadow Maldoc
a6a6b8c7cb72dd2670b6171576bc20c2f28198df12907b4d3ce010dcd97358e4	SHA256	WhiteShadow Maldoc
67d0347b8db05a7115d89507394760f41419e5e91ab88be50e27eded28ce503e	SHA256	WhiteShadow Maldoc

7b672eb80041a04f49198b1b51bcf0198321a1bdc1f7c56434c2320edb53fc43	SHA256	WhiteShadow Maldoc
acf9c1dda4a2076f0d503450db348ae2913345ebd134a3701baa2ff5ebaccd6e	SHA256	WhiteShadow Maldoc
fe88d40c56274a38ecd3a7178ac96970dd473c7ef3d0f54b5c8819f0b1fa41c3	SHA256	WhiteShadow Maldoc
hxxp[://]rebrand[.]ly/813ed538169eeeethczfz2346577777777788kfvmdkf	URL	Short URL Leading to WhiteShadow
hxxp[://]rebrand[.]ly/purchaseorder54326	URL	Short URL Leading to WhiteShadow
35e81258c4365fb97ae57f3989164ed4e8b8e62668af9d281a57c5e7a70c288c	SHA256	WhiteShadow Maldoc
c5193ba871414448c78cb516dfea622f2dbafa6bacb64e9d42c1769ebd4ffea3	SHA256	WhiteShadow Maldoc
b2c0b1535518321fbcde2c9d80f222e9477053e6ee505f2dd3b680277f80de1d	SHA256	WhiteShadow Maldoc
9a284b1ca8ac7fee1f8823d2457c935134ec61368ef42c8b2cbdfb338ad61ad7	SHA256	WhiteShadow Maldoc
29fe2bdf25d1739bb920c0776b1826661e8a459af44d1051faf08f3643d84d29	SHA256	WhiteShadow Maldoc

fcc8802b49bfb86d0cffb1cbc4f1b283887015b7da2263f9165a28f1b0f63f47	SHA256	Crimson
193.111.155[.]137	IPv4	Crimson C&C
c966830092abeb5ecb6747122b5c5d63dff064828b84e10a682770763e348713	SHA256	Nanocore
51.254.228[.]144	IPv4	Nanocore C&C
0df01a9e8ad097d6c2b48515497f12bfe9aaa29a3b1c509a0ae1e2a12a162f04	SHA256	Nanocore
jasoncarlosscot.dynu[.]net	hostname	Nanocore C&C (resolved to 79.134.225[.]77)
a7832e35fe571abed0a70b49c043e0fedb7fba28e6c212b6bbaa8fd4075c5f43	SHA256	Formbook
www.allixanes[.]com/ez3/	URL	Formbook C&C
0e54bf9380d40d34e6a3029b6e2357f4af1738968646fdaa0c369a6700e158f4	SHA256	AZORult
tslserv.duckdns[.]org	hostname	AZORult C&C over HTTP (resolved to 102.165.49[.]69)
bargainhoundblog[.]com	hostname	Expanded rebrandly URL hostname. Likely compromised.
17742a3ca746f7f13aff1342068b2b78df413f0c9cd6cdd02d6df7699874a13a	SHA256	Crimson

176.107.177[.]54	IPv4	Crimson C&C
globedigitalmedia[.]com	hostname	Expanded rebrandly URL hostname. Likely compromised.
ab962b7b932cb3478770c55a656baa657f9c58ea2409c89b68b5a7728ea721f8	SHA256	Crimson
d89772cfd63f7d5ce7c6740c6709ece4db624c85989e8d508c09f1baeef0a556	SHA256	Crimson
139.28.36[.]212	IPv4	Crimson C&C
64c5d3f729d9a1ec26d5686002ccb0111ee9ba6a6a8e7da6ad31251f5d5dde6a	SHA256	Agent Tesla
76e0104aa6c3a0cfc25c6f844edbbeed4e934e2ad21a56e8243f604f510cf723	SHA256	njRAT
139.28.36[.]212	IPv4	njRAT C&C
6a2acd6b97ce811ebf3d154c47b53cd16c500e075c3218e8628bf49f8d7cafe5	SHA256	Nanocore
45.92.156[.]76	IPv4	Nanocore C&C
96e274f1cb5f6918e6a24b714f7cbf2d3d007680050a16ba5232c67582ad0f3b	SHA256	Crimson
2ea787dfd65b0488b76b0a0a69ff2a632bb3bea3735ad007336b8dd1473f5768	SHA256	Crimson
192.3.157[.]104	IPv4	Crimson C&C
a9898d6d9054f301d0da9373b8cc38641d11c8409a1037112970aa47122561ff	SHA256	Crimson

5ff5817e325c78a1a706035811bfb976421222c208a7fce694a25bb609a9a2fb	SHA256	Crimson
176.107.177[.]77	IPv4	Crimson C&C
0c1623662a7ad222ed753b9ffc0d85912e3a075c348b752b671a0b1c755fd1e7	SHA256	Crimson
193.228.53[.]0	IPv4	Crimson C&C
4c487ba8dfded5d050d01ab656ef3916c5269551e51ed60f9cfa5995f55e3264	SHA256	njRAT
mundial2018.duckdns[.]org	hostname	njRAT C&C (resolved to 46.246.85[.]129)
d2158cfd1bb9116a04dcb919fa35402d64b9e9b39a9c6cd57460ca113cde488e	SHA256	Orion Logger
0943a968cc9e00f83c0bb44685c67890c59ad7785db7fc12e9a0de8df309cbfa	SHA256	Crimson
185.157.79[.]115	IPv4	Crimson C&C
a40987639b464c2d7864faa0cc84da7f996feecc7a7f0225a474e282d2d81c37	SHA256	Crimson
542c6ed8e77987ca01152784a38ab4d288a959d7e2144989ae7f1eb89866d65b	SHA256	Agent Tesla
a8e0c6387dd77500a0593c0c26ba3b1e72b9bce200c232d7dcc1f2e75a449512	SHA256	Crimson
98d5fc49a5153cd8035ca0e83cf46a81dd573c175884821473aa4d07f719031f	SHA256	Crimson
dfea73a64bcb2aeab104dd3d78b83c859c4319be08a8da4edf77cc631bbdd623	SHA256	Crimson

07aa897c146f9443876930f1b69807ec7034a73a93dec0dc36157f17dedd3069	SHA256	Crimson
f6ffbd8762b893aa9d7907917ee0b11457fbdbf37f4aacdf8d1d4a4f7f3badca	SHA256	Crimson
87.247.155[.]111	IPv4	Crimson C&C
a2b5168fb4b6a18d66571c6debc54f9f462f5b05a82313123feecc96dab0e595	SHA256	Netwire
halwachi50.mymediapc[.]net	hostname	Netwire C&C (resolved to 45.138.172[.]161)
robinmmadi.servehumour[.]com	hostname	Netwire C&C (resolved to 45.138.172[.]161)
bde269bf69582312c1ec76090991e7369e11dbee47a153af53e49528c8bd1b27	SHA256	Crimson
185.161.209[.]183	IPv4	Crimson C&C
bd7abfaa0d3b1d315c2565c83c1003c229c700176c894752df11e6ecae7ad7e6	SHA256	Crimson
185.161.210[.]111	IPv4	Crimson C&C
4738c2849f2c81dec71427adaed489a84299563da31b62ce5489b84c95426ada	SHA256	Crimson
ee0f3eb8a4d7c87a4c33a1f8b08e78bb95fa7ee41ddf0b07d9b6eabe87a33b2e	SHA256	Remcos
naddyto.warzonedns[.]com	hostname	Remcos C&C

4b554367f8069f64201418cddcec82d7857dcc2573be7f0fb387c1b4802040b6	SHA256	Formbook
www.scaker[.]com	hostname	Formbook C&C over HTTP

ET and ETPRO Suricata/Snort Signatures

2013410 ET POLICY Outbound MSSQL Connection to Standard port (1433)

2814263 ETPRO TROJAN MSIL/Crimson C&C Server Command (info)

2832108 ETPRO TROJAN MSIL/Crimson Client Command (info)

2832107 ETPRO TROJAN MSIL/Crimson Receiving Command (getavs)

2816280 ETPRO TROJAN MSIL/Crimson Receiving Command (ping)

2815463 ETPRO TROJAN Win32/Megalodon/AgentTesla Conn Check

2837782 ETPRO TROJAN Win32/Origin Logger SMTP Exfil

2816766 ETPRO TROJAN NanoCore RAT C&C 7

2810290 ETPRO TROJAN NanoCore RAT Keepalive Response 1

2816718 ETPRO TROJAN NanoCore RAT Keep-Alive Beacon

2829000 ETPRO TROJAN FormBook C&C Checkin (GET)

2829004 ETPRO TROJAN FormBook C&C Checkin (POST)
