Exploring Bergard: Old Malware with New Tricks | Proofpoint US

By January 28, 2016 Darien Huss

Published: 2016-01-28 · Archived: 2026-04-05 12:49:02 UTC

Updated 06/24/2016

The Bergard Trojan and the C0d0so group that made it famous with the November 2014 watering hole attack [1] via Forbes.com have received renewed attention recently, with other researchers [2] potentially linking emerging tools and recent attacks to the group. Proofpoint researchers conducted a historical analysis of samples related to this research and uncovered new malware variants and likely origins and methods of infection.

Many of these samples have not been discussed publicly and several have very little or no anti-virus coverage. The analysis that follows is of completed, historical attacks as well as an extremely recent and ongoing attack, providing insight into the volume and timeline of infections, as well as a timeline for attacker-initiated actions using a novel malware family.

Common link

In the malware used in the Forbes watering hole attack (the Bergard Trojan [3]), a simple single-byte XOR encoding technique (Fig. 1) was used to encode potentially suspicious strings, along with 5-byte padding prior to each string (Fig. 2).

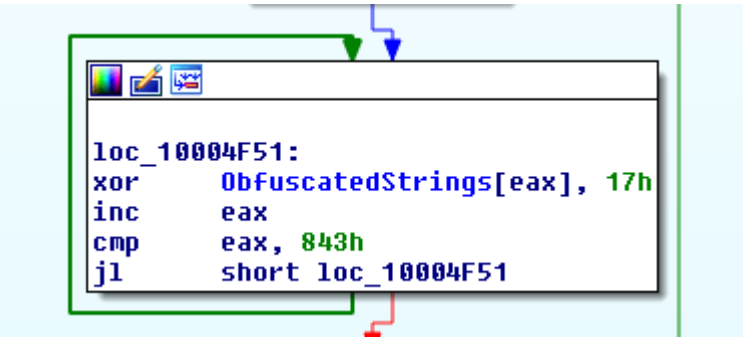

Figure 1: String deobfuscation in sample: 3e92802ba89f3f2f66ce04311e0f3882

```
.data:100150F8 ; char ObfuscatedStrings[]
.data:100150F8 ObfuscatedStrings db 4Bh                 ; DATA XREF: DllMain
.data:100150F9                   db  39h ; 9
.data:100150FA                   db  74h ; t
.data:100150FB                   db  53h ; S
.data:100150FC                   db  65h ; e
.data:100150FD aOpen             db 'open',0
.data:10015102                   db  77h ; w
.data:10015103                   db  77h ; w
.data:10015104                   db  4Bh ; K
.data:10015105                   db  4Ah ; J
.data:10015106                   db  6Dh ; m
.data:10015107 a@echoOffPing127_1NulDel1I db '@echo off',0Ah
.data:10015107                   db 'ping 127.1 > nul',0Ah
.data:10015107                   db 'del %%1',0Ah
.data:10015107                   db 'if exist %%1 del %%1 else goto Bye',0Ah
.data:10015107                   db 'ping 127.1 > nul',0Ah
.data:10015107                   db 'if exist %%1 del %%1 else goto Bye',0Ah
.data:10015107                   db 'ping 127.1 > nul',0Ah
.data:10015107                   db 'if exist %%1 del %%1 else goto Bye',0Ah
.data:10015107                   db ':Bye',0Ah
.data:10015107                   db 'del %%1',0
```

Figure 2: Deobfuscated strings in sample: 3e92802ba89f3f2f66ce04311e0f3882

Additionally, some of the variants contain a common PRNG algorithm that is used along with GetTickCount as a seed (Fig. 3) to pseudo-randomly generate lowercase letters (Fig. 4). By leveraging these two techniques utilized by Bergard, we were able to uncover new malware families and several adversary campaigns.

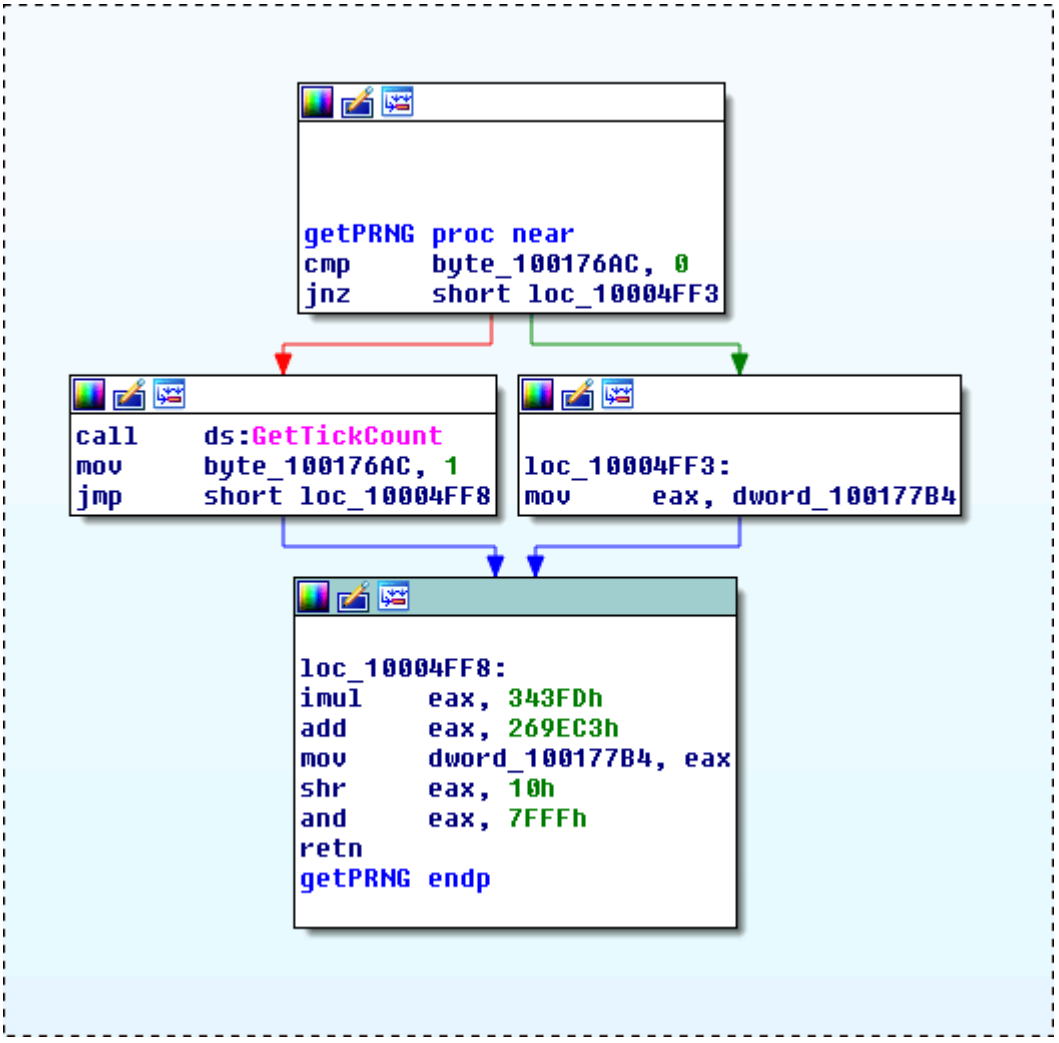

Figure 3: PRNG algorithm in sample: 3e92802ba89f3f2f66ce04311e0f3882

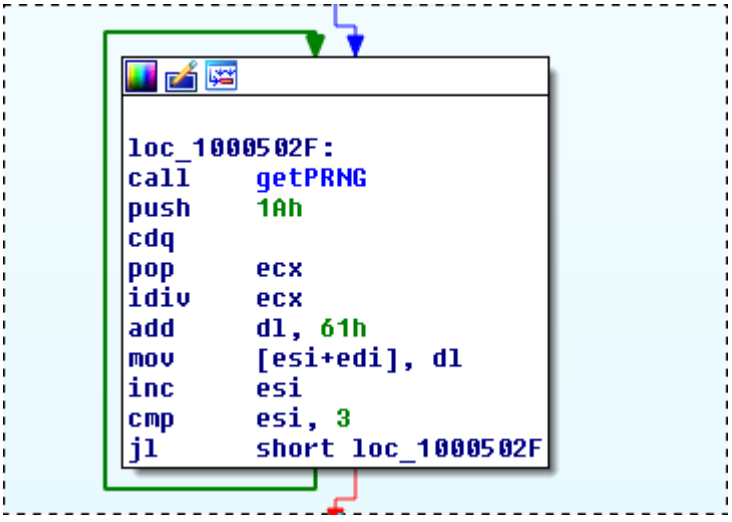

Figure 4: PRNG algorithm usage in sample: 3e92802ba89f3f2f66ce04311e0f3882

Related variants

There are several unique strings embedded in the first samples we analyzed which were also found in completely different malware families. In most of the samples we analyzed, they were found encoded using the same encoding mentioned in the previous section, however some older variants contained no XOR encoding but still used a 5-byte padding layout for interesting/suspicious strings. To further research possibly related samples as well as find completely new malware families, we developed a “shotgun yara” [4] approach that, when coupled with unique strings found commonly in related samples, allowed us to discover completely new malware families. In the next sub-sections we discuss several of the already known malware families and clusters we have found, as well as several previously unreported malware families that may exist in currently active attack campaigns.

PGV_PVID Variant

We refer to this family as the PGV_PVID variant based on the cookie variable utilized in the network beacons generated by these samples (Fig. 5). The samples that we were able to find with similar string encoding, PRNG algorithm, and similarly structured C2 beacon may be found in the IOCs section as well as a graph of the samples and their associated C2 in Figure 6.

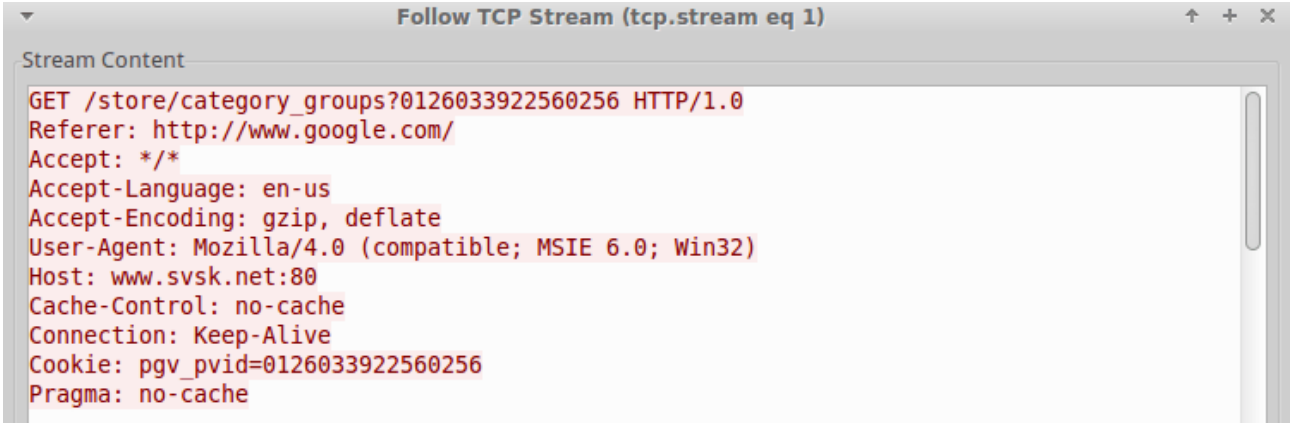

Figure 5: PGV_PVID variant C2 beacon

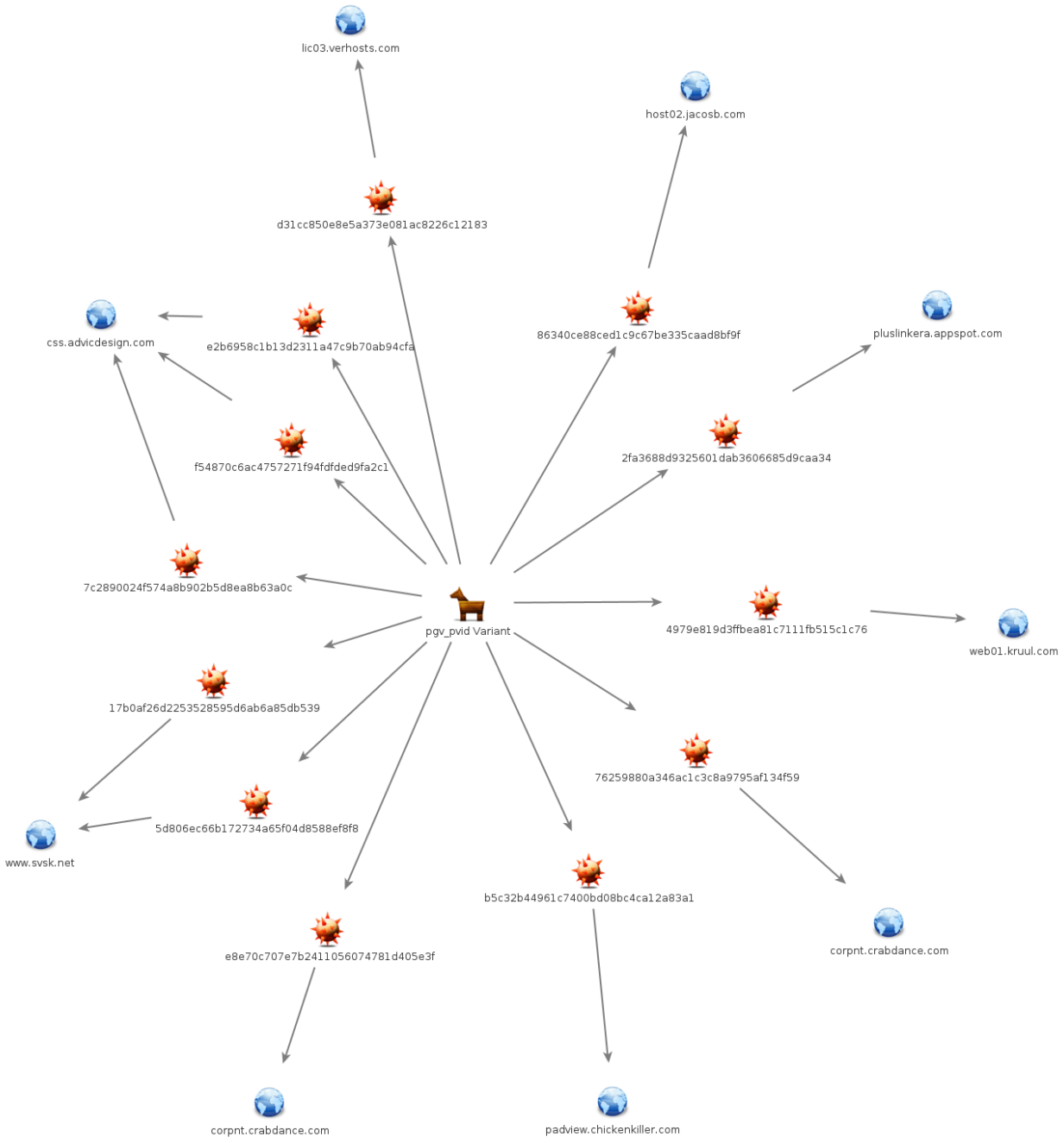

Figure 6: Maltego graph of PGV_PVID variants

In numerous samples in at least the PGV_PVID family, the domain www.nsa[.]org[.]cn may be found encoded separate from the typical 5-byte padded obfuscated strings. For example, in sample 76259880a346ac1c3c8a9795af134f59 the following string is embedded in an encoded state using XOR key 0x90: [hxxp://www[.]nsa.org[.]cn/pwninfo[.]php]. This domain appears in an article citing a July 2, 2015 internal Department of Homeland Security report claiming that it was used in one of the breaches listed in the internal report [5]. We have been unable to confirm whether or not the domain appeared in the cited report, nor were we able to determine which breach the domain was related to, if any.

UID_SID Variant

Similar to the last variant, we refer to this one based on the Cookie variables that are used in the C2 beacons (Fig. 7,8). Unlike the last cluster however, this variant appears to have been used in an extensive DDNS cluster of infrastructure dating back to at least 2013. Tool usage in this cluster includes Gh0st, PlugX, Jolob, and Bergard (Fig. 9).

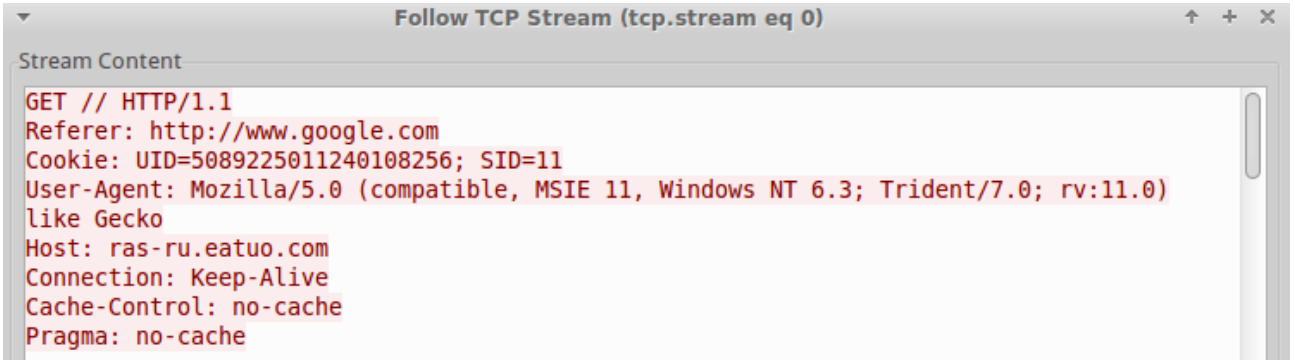

Figure 7: UID_SID variant GET C2 beacon

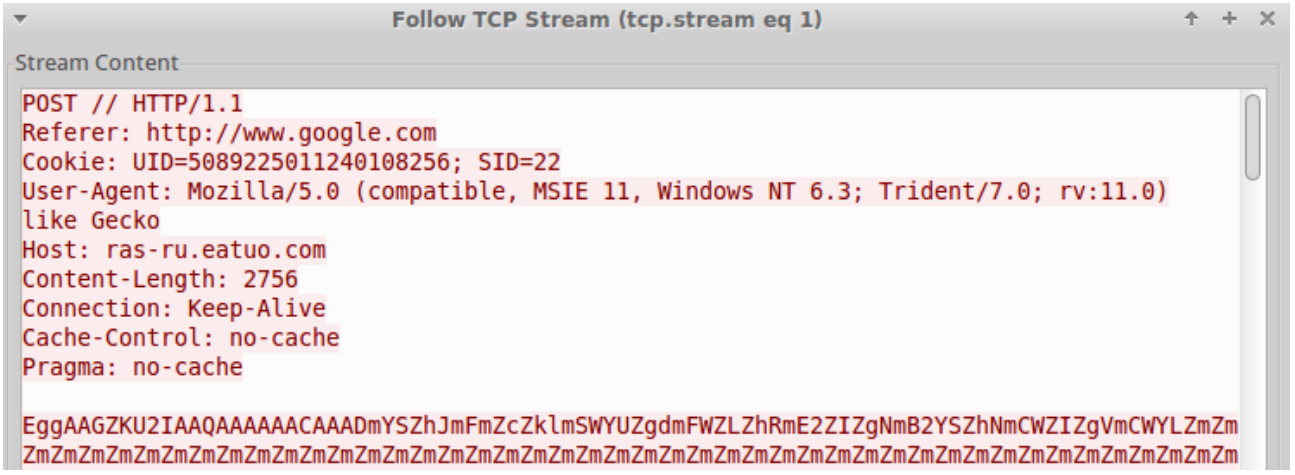

Figure 8: UID_SID variant POST C2 beacon

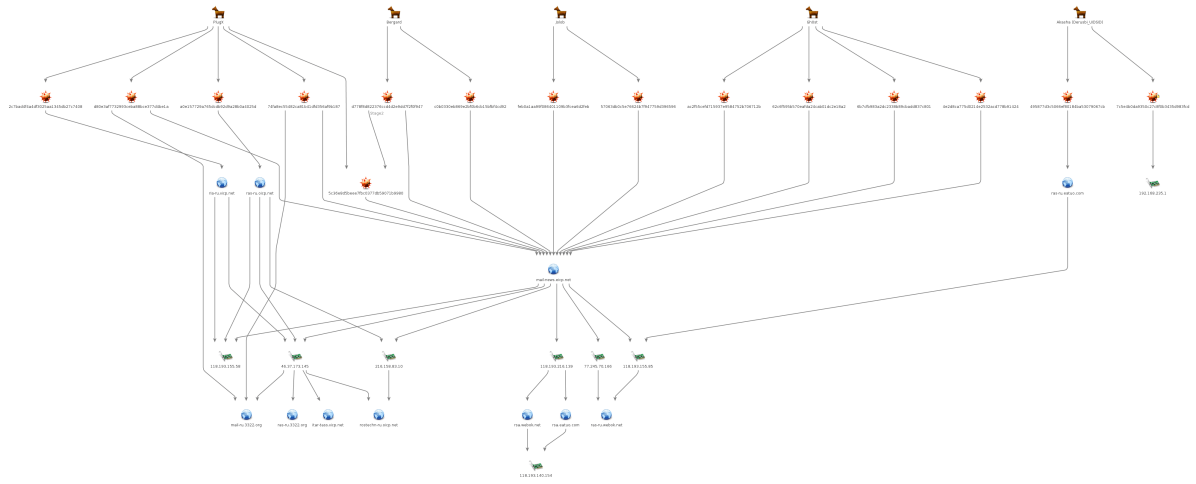

Figure 9: Maltego graph of UID_SID variant and related activity

In one instance related to this cluster of activity, we observed Bergard (md5: d778f8d822376ccd4d2e9dd7f2f0f947) receive instructions from its C2 to retrieve a PNG file (Fig. 10) containing an encoded PlugX payload (md5: 5c36e8d5beee7fbc0377db59071b9980). For an explanation of values received in the instructions from C2, refer to Table 1.

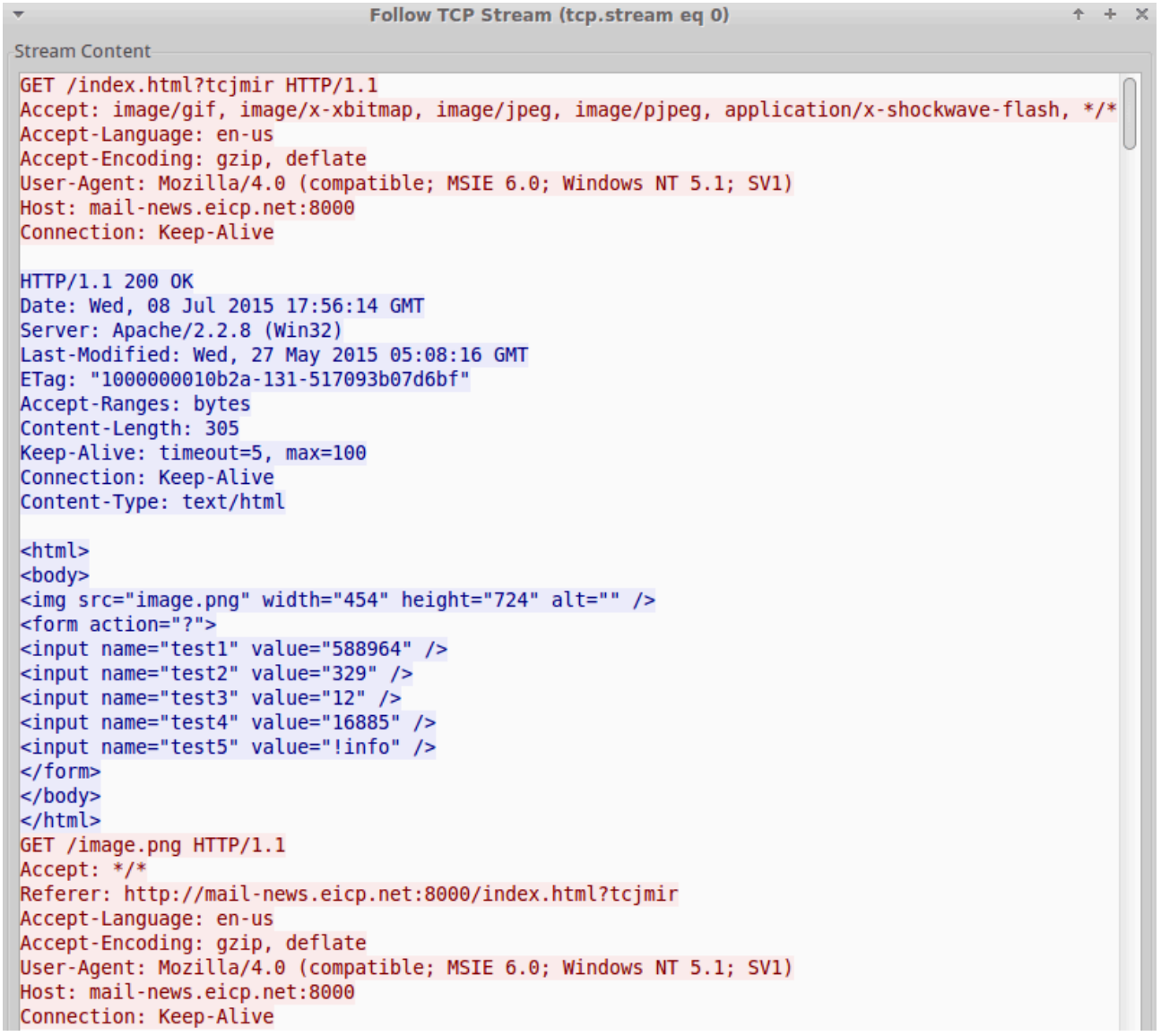

Figure 10: Bergard retrieving commands and encoded payload

Table 1: Description of Bergard C2 response

Item	**Description**
src=	Location of encoded payload

test1	Payload offset
test2	Payload end
test3	XOR key
test4	Unknown
test5	Command. Supported commands: !info, !axel, !exec

TXER

While hunting for additional payloads that may be related to the previously mentioned Bergard samples, we came across what appears to be a malicious payload that utilizes the Tox protocol [6] to connect to a controller. In the two samples we found, they both contained identical string encoding to previous samples that we analyzed (Fig. 11). This payload is still being analyzed to determine its full capabilities, however it appears to at least be capable of receiving and executing additional payloads. We may provide an update at a later time once this payload is fully understood.

Figure 11: TXER string encoding and decoded strings

Bassos Campaign

During our research, we came across a recent campaign that is connected to a different currently active and ongoing campaign with potentially thousands of compromised victims. We are referring to this campaign as

Bassos. We first discovered this threat after analyzing a compressed archive that was uploaded to VirusTotal (VT) on January 5th, 2016. Several items of importance were located in this archive, including VBS downloaders, a Java backdoor, and two Trojans containing identical string encoding with 5-byte padding: a custom Trojan (aka, CustomTCP) that beacons to port 22 and a new Trojan we refer to as Rekaf (Table 2).

Table 2. Description of archive contents

Filename	**MD5 Hash**	**Description**
tmp.vbs	9fc086b05787fb2e6c201de63e6e0698	VBS Downloader. Payload: likely CustomTCP
mc.vbs	5029b0d6f6621bf8e8f524fcea69d2b8	VBS Downloader. Payload: Rekaf
McAltLib.dll	b06a3a9744e9d4c059422e7ad729ef90	CustomTCP Trojan
dbgeng.dll	2123c5c24d8c06a10807458630751ded	Rekaf
tk.jar	9d863756a69401765252f5133023240c	Java Backdoor

Although we did not discover these samples in the wild, they potentially provide a glimpse into how the custom downloader and Rekaf are delivered to victims. Both VBS scripts operate very similarly, with one primary difference being that the mc.vbs script contains status reporting functionality to the following domain: www.jweblogic.com (Fig. 12) as well as checks if the “360rp” service is running (if found, it will not continue).

Figure 12: VBS Downloader reporting system information to status server

An additional, slightly different version of mc.vbs was also found (md5: 9f47f04aa9eb72f749cbf4bb7e40c446). Of the payload locations observed in the VBS scripts, we were able to retrieve payloads only from 210.181.184.64. We were also able to retrieve mcs.exe (md5: 1501eed51578e795af7f2f5fb3078178) and

McAltLib.dll (md5: 26e863f917da0b3f7a48304eb6d1b1d3) from 218.54.139.20, however we found no VBS script referencing that download location. The exact same or similar mcs.exe/McAltLib.dll combination was also likely hosted on 42.200.18.194 [2] (Fig. 13).

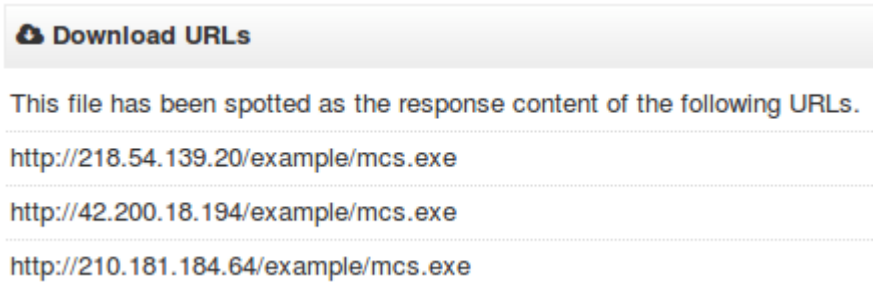

Figure 13: Locations hosting 1501eed51578e795af7f2f5fb3078178 reported by VT

All three locations appear to be legitimately compromised websites in addition to hosting Jboss Application Server (JAS). It is our hypothesis that these legitimate compromised sites were all compromised sometime after early November using CVE-2015-7501 [7,8] and publicly available exploit code [9]. Several items support this hypothesis:

- Many samples have compile times between November and now
- Many samples first appeared in our sample exchange and on VT between November and now
- CVE-2015-7501 received significant media attention in early November
- The java backdoor contained in the archive found on VT appeared in public exploit code for CVE-2015-7501 [10]
- The archive found on VT contains JAS logs of what appears to be a vulnerable version of JAS
- Jboss themed C2 infrastructure

Without having analyzed the compromised servers, nor observing the first stage in any of the campaigns, it is impossible to know for certain if that is the vector of compromise for these legitimate websites. A broad overview of this campaign is provided in Maltego graph form in Figure 14.

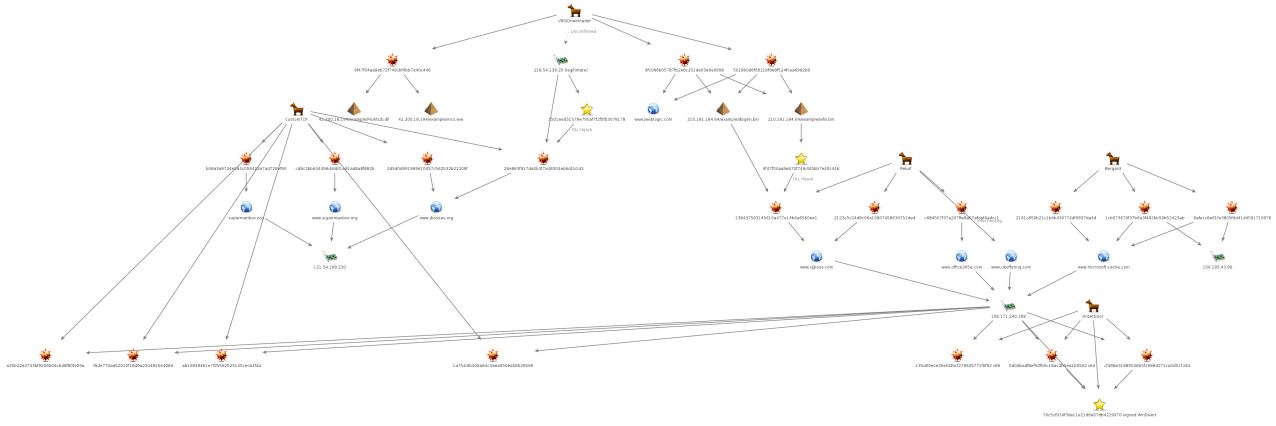

Figure 14: Maltego graph of Bassos campaign

Rekaf

Like the custom downloader/McAltLib.dll payload, Rekaf/dbgeng.dll utilizes a signed, legitimate executable (iefix.exe) and DLL Search Order Hijacking [11] for execution. Upon successful execution, Rekaf first collects various system information in the format found in Figure 15. As shown in the image, this is likely the first version of Rekaf according to the ver0.0.1 indicator that is hardcoded in the sample.

```
Version: ver0.0.1
CPU: [removed]
ComputerName: [removed]
UserName: [removed]
OsVersion: [removed]
CodePage: [removed]
LanIP: [removed];
ConnectUrl: http://www.xjjboss.com/index.php
TrojanPath: [removed]\dbgeng.dll
AVInfo: [removed];
TimeInfo: SystemDate=[removed] BootDate=[removed] RunTime=[removed]
```

Figure 15: Rekaf collected information exfiltrated to C2

Prior to submitting the collected information to C2, the information is encoded with the first byte in the MAC address, which is appended after the keyid variable in the URI (Fig. 16).

Figure 16: Rekaf HTTP POST C2 Beacon

If the server responds with an encoded string “Login Server Success” then the Rekaf bot will continue to perform HTTP GET requests to the server until it receives a command. Supported commands are listed in Table 3. Rekaf also stores detailed debug information in the following file appended to the result of WinAPI GetTempPath: MSHelper.bin. We have observed the adversaries retrieve this log from an infected machine with a timezone of GMT+8 on at least one occasion using the download command.

Table 3: Rekaf supported commands and descriptions

Command	Description

upload	Download file from C2
download	Upload file to C2
cmdshell	Perform issued command using cmd.exe
update	Retrieve an updated payload

Lastly, Rekaf contains several encoded PEs embedded inside itself which may be dropped and executed in certain situations. Their functionality is still being analyzed and will not be covered in this report.

Monitoring Rekaf C2: 108.171.240.208

Upon discovering Rekaf, we began passively monitoring the C2 to gain insight into their operation. During our monitoring, we were able to collect information related to botnet volume in addition to a timeline of certain actions the adversaries conducted. While monitoring the Rekaf C2 we observed over two thousand unique infections. Figure 17 shows a frequency graph of new and duplicate infections occurring by date. Two dates, 12/22/15 and 1/19/16, have enormous peaks that could have occurred for a number of reasons including:

- 12/22/15 was the beginning of a new campaign that may have stretched through to 12/30/15
- 1/19/16 (and possibly 12/22/15): the botnet was updated, re-initialized, or C2 data was wiped

There could be another explanation for the peaks, however due to our monitoring beginning in the middle of a campaign our picture of the C2 may not be complete.

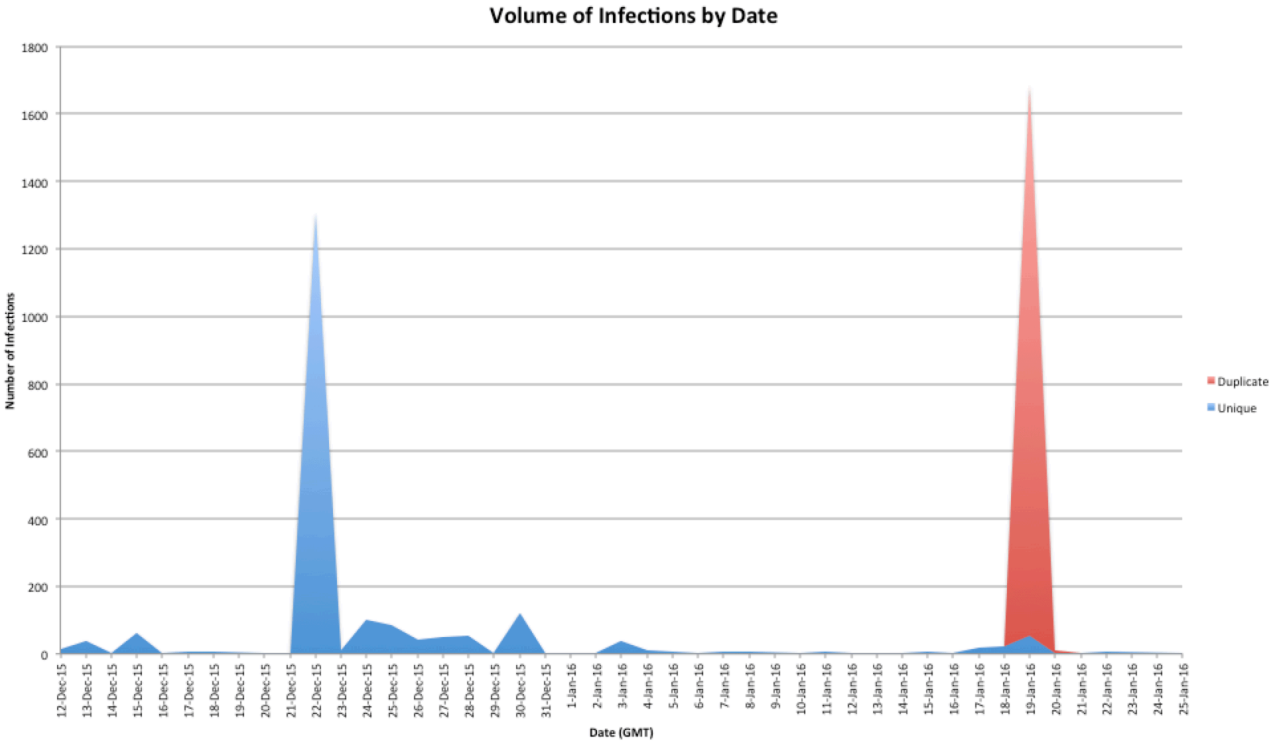

Figure 17: Volume graph of Rekaf infections by date (GMT)

In addition to monitoring infection volume, we also compiled a list of various actions we observed the adversaries take including: executables placed onto C2, executed commands sent to bots, large infection spikes, and compiled timestamps for related executables (Fig. 18).

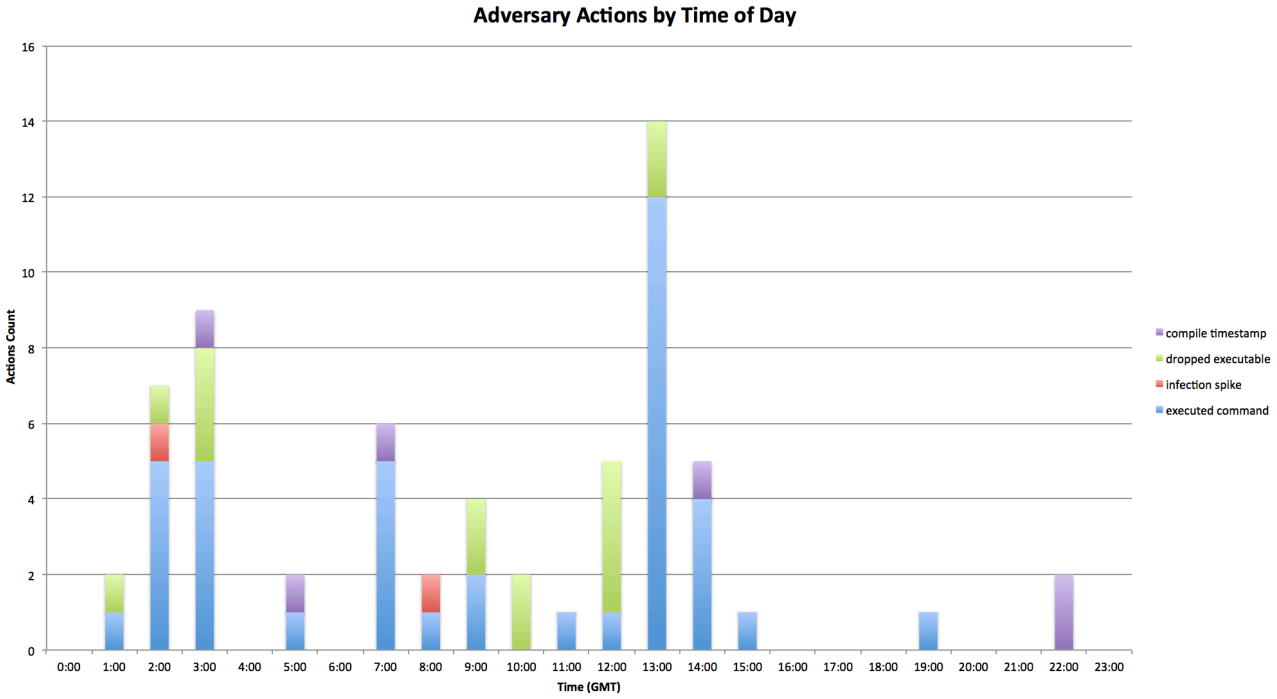

Figure 18: Adversary actions graph by time of day

This is useful from an attribution standpoint as it could potentially reveal a normal working day for the adversary group. Our analysis of the available data suggests that this adversary group is located geographically somewhere in or near a GMT+8 timezone assuming they maintain normal working hours from 9am-10am to 6pm-8pm. An additional result from our analysis is that no observed actions were ever performed on a weekend (GMT), providing further evidence that this is most likely an organized outfit that functions during a standard work week.

Unfortunately the method we used for data collection could have resulted in us analyzing doctored data, including modified PE compile timestamps. Based on observed activity by this adversary group however, at this time we do not believe that timestamps were modified in any of the data we analyzed.

VirdetDoor

While monitoring the Rekaf C2 we observed the adversaries prepare what appears to be a new implant for delivery to already infected bots. We have named this backdoor VirdetDoor as it utilizes a signed WinDivert driver [12] for part of its operation. This executable may take two different command line arguments, -v which enables verbose logging to a file named msocache7.log, and –l [port] which changes the default listener port from 6666 to the specified port. The payload first listens for the string “Aabac” to initiate a connection. This payload is still being analyzed to determine its full capabilities and to confirm that it is in fact malicious in nature. We may provide an update at a later time once this payload is fully understood.

CustomTCP

The CustomTCP Trojan was covered in a recent article and tentatively attributed to C0d0s0, while also explaining the same similarities we discussed earlier to the November 2014 Forbes watering hole attack (aka, Port 22 Variant) [2]. The adversaries controlling the 108.171.240.208 C2 appear to be utilizing this malware extensively, as we observed them prepare four different CustomTCP payloads for delivery to already infected bots. This is in addition to other payloads we discovered while conducting our hunting research.

Attribution

Recently the Bassos campaign has already been attributed to ‘C0d0s0’ (aka, Codoso) [2]. Based on the mail-news.eicp.org cluster of activity, that campaign appeared to have slightly different tactics, techniques, and procedures (TTPs), including potentially target-themed domain infrastructure as well as heavily relying on dynamic DNS for C2 domains. Due to the varying TTPs in infrastructure, we think it is possible that the Bergard and the related toolset could now be shared by multiple adversary groups. If that is the case, observing Bergard usage and related families do not provide clear indication that the C0d0s0 group is involved. Attribution becomes increasingly difficult as adversaries evolve and adapt so we have provided the data available to us to better assist organizations in formulating their own attribution.

Conclusion

The historical analysis conducted by Proofpoint researchers on recently completed campaigns revealed actions by what appear to be well-organized actors consistent with advanced persistent threats. Additionally, the in-depth analysis above on a currently ongoing campaign, Bassos, being conducted by possibly the same actor (‘C0d0s0’,

‘Codoso’) as the Forbes watering hole attack in 2014, provides insight into the adversary's activities and infection volume related to the Bassos campaign. Additionally, the email-news campaign, PGV_PVID campaign, and UID_SID variants may also be linked to C0d0s0 as well. However, the differences in TTP could indicate that the Bergard toolset is now available to multiple adversary groups. Regardless of the exact attribution for the attack, organizations, researchers, and vendors should take note of the growing potential footprint of the Bergard toolset.

Indicators of Compromise (IOC’s)

TXER

16652d4213991ae58e268ae03a4c4e97

e81f9dadbdc7eea937e586afc9fb59f8

PGV_PVID Samples

5d806ec66b172734a65f04d8588ef8f8

17b0af26d2253528595d6ab6a85db539

b5c32b44961c7400bd08bc4ca12a83a1

86340ce88ced1c9c67be335caad8bf9f

e8e70c707e7b2411056074781d405e3f

2fa3688d9325601dab3606685d9caa34

d31cc850e8e5a373e081ac8226c12183

e2b6958c1b13d2311a47c9b70ab94cfa

7c2890024f574a8b902b5d8ea8b63a0c

f54870c6ac4757271f94fdfded9fa2c1

4979e819d3ffbea81c7111fb515c1c76

76259880a346ac1c3c8a9795af134f59

PGV_PVID Domains

www.svsk.net

padview.chickenkiller.com

host02.jacosb.com

corpnt.crabdance.com

pluslinkera.appspot.com

lic03.verhosts.com

css.advicdesign.com

web01.kruul.com

corpnt.crabdance.com

Mail-news campaign

Bergard samples

d778f8d822376ccd4d2e9dd7f2f0f947

c0b0330eb869e2bf0b6cb15bfbf4cd92

UID_SID samples

495877d3c5066ef80184ba53079067cb

7c5e4b0da9350c27c8f0b3435d983fcd

Gh0st samples

ac2f55cefd715937e9584752b706712b

62c6f595b570eafda24cab01dc2e18a2

6b7cfb983a2dc2338b89cbadd837c801

4e2d8ca775d0214e2532acd778b91424

Jolob samples

feb0a1aa99f086401109b3fcea6d2feb

57063db0c5e76624b7f947759d396596

PlugX samples

2c7bad4f4a4df3025aa1345db27c7408

d80e3af7732993ceba88bce377d4be1a

a0e157729a765dcdb92d9a28b0a4025d

74fa8ec55482ca81b41dfd356af9b187

5c36e8d5beee7fbc0377db59071b9980

Confirmed Mail-news Domains

mail-news.eicp.net

mail-news.eicp.net

mail-ru.3322.org

ras-ru.eatuo.com

ria-ru.xicp.net

ras-ru.oicp.net

Bassos Campaign

VBS Downloader

5029b0d6f6621bf8e8f524fcea69d2b8

9f47f04aa9eb72f749cbf4bb7e40c446

9fc086b05787fb2e6c201de63e6e0698

CustomTCP samples

b06a3a9744e9d4c059422e7ad729ef90

2d5d0d991999610457c562532b21209f

26e863f917da0b3f7a48304eb6d1b1d3

a35b22e2743bf9206b06cbd8f80fe29a

ab108484b1e75f5562525145cecb4f4a

1a7fcb0b406a64c3aea050ea58529596

f6de770ad52015f18d0a2344815e408d

aa2c1bdeff0076ccd79d4cb6ae29f1d8

cd8c2bb644496d46bf1e91ad8a8f882b

CustomTCP Domains

supermanbox.org

www.supermanbox.org

www.jbossas.org

Rekaf samples

2123c5c24d8c06a10807458630751ded

138437593145610a477e14b9a6560ee1

c684507f37a207ffe8a67afdaf4adcc1

Rekaf Domains

www.xjjboss.com

www.office365e.com

www.ukoffering.com

Bergard samples

1cb673679f37b6a3f482bb59b52423ab

8afecc8e61fe3805fdd41d4591710976

2161c859b21c1b4b430774df0837da9d

Bergard Domains/IP

www.microsoft-cache.com

106.185.43.96

Java Backdoor

9d863756a69401765252f5133023240c

VirdetDoor

135d00ece30efd46cf279645771f6f92

cf488e31889546b5f1688d271ca0d51f

5d0dbadf8ef50fb6c18ac4b0ea1b5562

Additional CustomTCP Payloads

40a00b89365c739950140697a6474286

ea8545992806966484baafbcdf79bfdc

7ddf02a5afaab8e03ebd9af04b76603a

885c5eb20c3b40eed76fd3c48b912697

a4fe7449dae9a1a38497069c2a574309

d7e2c212ffc8f1639fc0120888ea30cd

Unclustered Bergard

Samples

e5274ff02184a304d45d42ca953148ce

7f466312a3b1176f052f8c05f7781715

Domains

f1a9d91a738041a8.appspot.com

f5310cff818ea0e7.appspot.com

References

1. http://www.isightpartners.com/2015/02/codoso/
2. https://github.com/darienhuss/shotgunyara/blob/master/shotgunyara.py
3. http://freebeacon.com/national-security/opm-hack-part-of-large-scale-cyber-attack-on-personal-data/
4. https://en.wikipedia.org/wiki/Tox_%28protocol%29
5. http://researchcenter.paloaltonetworks.com/2016/01/new-attacks-linked-to-c0d0s0-group/
6. http://foxglovesecurity.com/2015/11/06/what-do-weblogic-websphere-jboss-jenkins-opennms-and-your-application-have-in-common-this-vulnerability/
7. https://access.redhat.com/solutions/2045023
8. https://github.com/Xyntax/JBoss-exp/
9. https://github.com/Xyntax/JBoss-exp/blob/master/iswin.jar
10. https://www.fireeye.com/blog/threat-research/2010/08/dll-search-order-hijacking-revisited.html
11. https://reqrypt.org/windivert.html
