StrRat Malware: RAT With Ransomware Module
Executive Summary

StrRat is a Java-Based remote access trojan\(^1\) (RAT) that is delivered by email spam. The malware can not only perform credential theft\(^2\) and allow remote administration of the systems but can also behave as ransomware. This wave of email spam spreading the trojan was discovered by the Microsoft Security Intelligence\(^3\) (MSI) team and they described it as a “massive email campaign”. The campaign was observed on 20 May 2021 and reported in a series of tweets but a similar strain of malware was seen in the wild as earlier as 6 June 2020.

StrRAT remote access malware allows threat actors to perform attacks such as credential theft, installation of key loggers\(^4\) and offers complete remote access of the infected systems—typical characteristics of RAT. The malware also has a functionality to download additional modules from the command and control framework and execute on the infected system.

StrRat malware also offers ransomware features that are not common to other malwares in this category. Earlier versions of this malware only changed the extension of files to .crimson without actually encrypting them. Later actual encryption capabilities were added to the ransomware and the sample we analyzed makes use of AES Encryption\(^5\) Standard to encrypt files and append “.crimson” to them.

The Rat is spread via email spam, phishing techniques and other social engineering attack vectors. The RAT was made with the intention of targeting enterprise users as well as other users. Once inside the network malware acts as a good medium to carry out further attacks.

Description

To start the attack chain, threat actors used compromised email accounts of several different emails to send malspam with the subject line “Outgoing Payments”. Some emails also used highly specific subjects in order to phish the user in.

The campaign included various emails that made use of social engineering techniques\(^6\) based around payment receipt to trick the user into opening a malicious pdf file. The attached pdf in all these cases, was designed in such a way that on opening it, it connects to a domain that hosts a malicious javascript file that downloads StrRat malware onto the system.
The malware downloaded on the system is jar file and not much obfuscated and on execution it will download all dependencies required to execute attacks such as collecting browser passwords, running remote commands on PowerShell, logging keystrokes etc.

The malware currently requires the java runtime environment to be installed on the infected system in order to execute properly. Other than that the malware does not employ any obfuscation technique but the demo version of Allatori Java Obfuscator. Thus the reverse engineering was easy and variants of this malware would be seen from several threat actors in the future.

**Detailed Static Analysis**

**Analysis of Malicious JavaScript File**

**File identification**

**MD5**: 5e1cbb6566f677da1d920c9d22f59bd7  
**Sha1**: 32e31e15fe42e4cb9e2a03698a5c7bc38631eb6  
**Sha256**: a1660698b8655cb721f71693178d2804caba8aa12c3a25446c6f7c665a30f2ff  
**Sha512**: 6ffa6cb201b2c20403214072b41-cafc0e3c3fefe642d475679f63b60b8cd607e090586ece4e86958337638fd87004df20e4f9946ac00fd29482aa0a951dedd66  
**SSDeep**: 3072:z3hdaQo7VJH7cElWrnLNF9CX1UbPU/If7z8gPXP68:LaQo37TlcEX1UbQlH8g/PV

**File Size**: 138549 Bytes

**Virus Total**

https://www.virustotal.com/gui/-file/a1660698b8655cb721f71693178d2804caba8aa12c3a25446c6f7c665a30f2ff/behavior/Lastline

**Malware Bazaar**

https://malshare.com/sample.php?action=detail&hash=5e1cbb6566f677da1d920c9d22f59bd7

**Malicious File Summary**

The malware attack works in two stages: first the malicious javascript file acts as a dropper and downloads a jar file as ‘%appdata%\Chriso09 Stub.jar’. The file is not downloaded from anywhere but is stored inside the javascript source as base64 encoded. The javascript file was obfuscated but it was not hard to understand what was happening.
The javascript makes a string array called "-_0xfe1d7e" that was not initialized in the script and was used to call functions inside WScript. Here the variable is renamed to "lookup"

```
// WScript("CreateObject")()[?]
    silver = WScript[lookup(0xc7)][lookup(0x9)],
    siinema = silver['createElement'](lookup(0xcc)),
    siinema[lookup(0xc8)] = lookup(0xba), siinema[lookup(0xbe)] = aso_ibora;

//var wscriptShellObject = WScript("CreateObject")("WScript.Shell")
    var sugar = WScript[lookup(0xc7)][lookup(0x1)];

// sugar[lookup(0xd6)] = 0x1, sugar[lookup(0xb4)][], sugar[lookup(0xb1)][siinema[lookup(0xd3)]];
    var wshShell = WScript[lookup(0xc7)][lookup(0xd0)],

    tempdir = wshShell[lookup(0xc3)][lookup(0xc4)],
    appdatadir = wshShell[lookup(0xc3)]('%appdata%'),
    path = 'Chriso09\x20Stub.jar',
    //is temp is always false
    is_temp = [];

    //thus second condition will be executed
    //same as path = appdatadir + '\x5c' + path;
    is_temp ? path = tempdir + '\x5c' + path : path = appdatadir + '\x5c' + path;
```
Malicious File Summary

The malware is written in Java and executed as a jar file. Since the malware is a jar file, it was trivial to extract the entire source code of the malware. The malware was obfuscated with nothing but the Allatori Java Obfuscator.

afshsghsh.java       fhfhjfg.java       sbsbgsgsrg.java
agaflas.java          fhjttjg.java       sbsgssdfg.java
bmcvbmfd.java         fileHandler.java   sdf sdfd.java
bncbndfhd.java        GDI32.java         sf srsbds.java
bsgshsbs.java         ghdfdndfn.java    sg htrt.java
cbnfdhnh.java         ghsghgf.java      sgsfgghh.java
cbnhg.java            gnhsghbnb.java    sgsgshsdg.java
ncbcnmgm.java         gmgmgmgm.java     shshhndn.java
ncnbdjy.java          gsnthtgsb.java    ssdsgsbh.java
dfghdmnc.java         HBrowserNativeApis.java  sstydgn.java
dfhdndfg.java         hghteedr.java     thrhrth.java
dfhttegd.java         Kernel32.java      tthtyrths.java
dgdfndnbnc.java       Main.java          User32.java
nhhgdgdh.java         mddgghdsh.java    WinGDI.java
dhfgfgh.java          mfgfhnb.java      xbcvcul.java
dncbnf.java           ncbndfjg.java     xbcvcJava
ndngdgdh.java         ncdgfhbun.java    xcdgghdgdn.java
dsdsgdfge.java        ncnndfjg.java     xncxbc.java
ertdbdth.java         nddfjdntj.java    xndfdgd.java
fdghdmnh.java         ndgdffhj.java     xngfghgj.java
fgfhnbnc.java         ngdbnb.java       xnncxvb.java
fgsbsfgsb.java        ndffdfnd.java     xvbxbg.java
fgssbg.java           sabretb.java      y.java

The use of cfr Java decompiler was made to get this source code. One can see that most of the classes are obfuscated but few classes such User32, Kernel32, HBrowserNativeApis and GDI32 maintained their full name.

When the malware is run, it clearly shows that it was obfuscated using the Allatori demo version. The malware is feature rich and doesn't do much in order to make the reverse engineering process difficult. The malware requires a stable internet connection in order to execute properly as it downloads many dependencies and uses many pastebins to exploit the system properly.
Also the malware is noisy and doesn't care much about hiding and on execution, starts to download dependencies from the internet. This process is repeated until these dependencies are downloaded successfully.

In order to deobfuscate the malware, you can use online available tools such as Java DeObfuscator or use a custom tool like we did in this case. After deobfuscating the malware it becomes trivial to understand what the malware is doing. Once done use the cfr to decompile the jar file to java source code.

```java
java -jar /opt/cfr/cfr-0.151.jar --outputpath sample2/ output.jar
```

Malware starts by downloading the dependencies off the internet and makes some requests to pastebins which now have been removed and after it pings what seems to be a c2 server for keeping track of all the computers compromised.
Once that is all done then the malware will try to create persistence on the computer so it can survive reboots and can exist on the computer for a long period of time. The malware creates a scheduled task with the name of skype. The file is also run in the current execution environment before the persistence is done.

```java
if (object2[7].equals("true")) {
    Runtime.getRuntime().exec(new StringBuilder().insert(0,
        "cmd /c schtasks /create /sc minute /mo 30 /tn Skype /tr
        ").append(((String)object12).append("\n")).toString());
}
```

Further malware also creates registry based persistence by modifying the following hives:

- `<HKCU>\Software\Microsoft\Windows\CurrentVersion\Run\Chris09 Stub`
- `<HKLM>\Software\Microsoft\Windows\CurrentVersion\Run\Chris09 Stub`

The malware is well written and allows an attacker to do whatever he/she wants on the system. Thus it would be foolish technique if during incident response one only checks for this scheduled task for removing persistence.

The feature rich backdoor of the malware allows it to be a versatile tool after being installed on the victim machine. The malware even has a feature to reach out to the http://ip-api.com/json/ in order to confirm the timezone and report back.

The working of the backdoor is simple. All the information about c2 and other data that can be used to trace back the threat actor is stored inside a file "resources/config.txt". This file is encrypted with an AES scheme and uses the password "strigo" for decryption.
Other features of the malware that it offers

**Rat Features**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>reboot</td>
<td>Reboots the infected system</td>
</tr>
<tr>
<td>shutdown</td>
<td>Shuts down the infected System</td>
</tr>
<tr>
<td>uninstall</td>
<td>Removes the persistence of the malware by removing the persistence</td>
</tr>
<tr>
<td>disconnect</td>
<td>Kills the connection</td>
</tr>
<tr>
<td>down-n-exec</td>
<td>Downloads a file from the given ip and executes it</td>
</tr>
<tr>
<td>update</td>
<td>Kill the connection, then run the file in the start menu.</td>
</tr>
<tr>
<td>up-n-exec</td>
<td>Runs a file given by name. Has fancy features to identify the extension and then run it cmd.exe</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>remote-cmd</td>
<td>Runs the given command with cmd</td>
</tr>
<tr>
<td>powershell</td>
<td>Runs command with powershell</td>
</tr>
<tr>
<td>file-manager</td>
<td>Gives commands to navigate and make changes to file system</td>
</tr>
<tr>
<td>keylogger</td>
<td>Logs keystrokes and sends them back</td>
</tr>
<tr>
<td>o-keylogger</td>
<td>Starts offline keylogger</td>
</tr>
<tr>
<td>process</td>
<td>Dumps running processes</td>
</tr>
<tr>
<td>startup-list</td>
<td>WMI dump a list of autorun entries</td>
</tr>
<tr>
<td>remote-screen</td>
<td>Remote control the infected computer</td>
</tr>
<tr>
<td>rev-proxy</td>
<td>Start a reverse proxy connection</td>
</tr>
<tr>
<td>hrdp-new</td>
<td>Downloads the tool called HRDPInst.exe</td>
</tr>
<tr>
<td>hrdp-res</td>
<td>Downloads the tools called HRDPInst.exe for specific users.</td>
</tr>
<tr>
<td>chrome-pass</td>
<td>Dumps the chrome saved password</td>
</tr>
<tr>
<td>foxmail-pass</td>
<td>Dumps foxmail credentials</td>
</tr>
<tr>
<td>outlook-pass</td>
<td>Dumps Outlook Credentials</td>
</tr>
<tr>
<td>fox-pass</td>
<td>Dumps firefox credentials</td>
</tr>
<tr>
<td>tb-pass</td>
<td>Dumps thunderbird credentials</td>
</tr>
<tr>
<td>ie-pass</td>
<td>Dumps Internet Explorer credentials</td>
</tr>
<tr>
<td>all-pass</td>
<td>Extracts all credentials</td>
</tr>
<tr>
<td>chk-priv</td>
<td>Check if running as administrator user</td>
</tr>
<tr>
<td>req-priv</td>
<td>Tries to run as administrator user</td>
</tr>
<tr>
<td>rw-encrypt</td>
<td>Encrypts the filesystem with AES</td>
</tr>
<tr>
<td>rw-decrypt</td>
<td>Decrypts the filesystem with AES</td>
</tr>
<tr>
<td>show-msg</td>
<td>Shows message with notepad.exe</td>
</tr>
</tbody>
</table>

The malware is sophisticated and provides several features and things that it can do on infected computers. But detecting and removing the malware is easy. Microsoft 365 defender provides coordinated defense against these kinds of threats. Thus relying on Microsoft defender for Office 365 protection against malicious emails can work to great extent in order to stop these kinds of attacks. Further this advisory offers some detection rules that can be used in order to detect and remove malware from infected systems.
Indicator of Compromise

URLs Called

- https://github.com/kristian/system-hook/releases/download/3.5/system-hook-3.5.jar
- https://pastebin.com/raw/Jdnx8jdg
- https://pastebin.com/raw/NAEwG3TH
- https://repo1.maven.org/maven2/net/java/dev/jna/jna/5.5.0/jna-5.5.0.jar
- https://repo1.maven.org/maven2/net/java/dev/jna/jna-platform/5.5.0/jna-platform-5.5.0.jar
- http://wshsoft.company/multrdp.jpg
- http://www.allatori.com

IP’s Reached Out

- 199.232.196.209
140.82.121.3

WHOIS record (June 7, 2021)

**Clean text**

NetRange: 140.82.112.0 - 140.82.127.255
CIDR: 140.82.112.0/20
NetName: GITHUB
NetHandle: NET-140-82-112-0-1
Parent: NET410 (NET-140-0-0-0-0)
NetType: Direct Assignment
OriginAS: AS36459
Organization: GitHub, Inc. (GITHUB)
RegDate: 2018-04-25
Updated: 2019-06-20
Ref: https://rdap.arin.net/registry/ip/140.82.112.0
OrgName: GitHub, Inc.
OrgId: GITHUB
Address: 88 Colin P Kelly Jr Street
City: San Francisco
StateProv: CA
PostalCode: 94107
Country: US
RegDate: 2012-10-22
Updated: 2021-05-20
Comment: https://github.com
Comment: Please contact us directly for matters pertaining to abuse.
Comment: Urgent matters including DDoS are handled 24x7.
Ref: https://rdap.arin.net/registry/entity/GITHUB

185.199.111.154

WHOIS record (June 7, 2021)

**Clean text**

% Information related to '185.199.108.0 - 185.199.111.255'
% Abuse contact for '185.199.108.0 - 185.199.111.255' is 'abuse@github.com'

inetnum: 185.199.108.0 - 185.199.111.255
netname: US-GITHUB-20170413
country: US
org: ORG-GI58-RIPE
admin-c: GA9828-RIPE
technical: NO1444-RIPE
status: ALLOCATED PA
mnt-by: RIPE-NCC-HM-MNT
mnt-by: us-github-1-mnt
created: 2017-04-13T15:36:35Z
last-modified: 2018-12-14T10:48:39Z
source: RIPE
**WHOIS record (June 7, 2021)**

**Clean text**

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetRange</td>
<td>67.199.248.0 - 67.199.248.255</td>
</tr>
<tr>
<td>CIDR</td>
<td>67.199.248.0/24</td>
</tr>
<tr>
<td>NetName</td>
<td>BITLY</td>
</tr>
<tr>
<td>NetHandle</td>
<td>NET-67-199-248-0-1</td>
</tr>
<tr>
<td>Parent</td>
<td>NET67 (NET-67-0-0-0-0)</td>
</tr>
<tr>
<td>NetType</td>
<td>Direct Assignment</td>
</tr>
<tr>
<td>OriginAS</td>
<td>AS996982, AS995224, AS36351, AS32787</td>
</tr>
<tr>
<td>Organization</td>
<td>Bitly Inc (BITLY)</td>
</tr>
<tr>
<td>RegDate</td>
<td>2016-05-31</td>
</tr>
<tr>
<td>Updated</td>
<td>2019-03-13</td>
</tr>
<tr>
<td>Ref</td>
<td><a href="https://rdap.arin.net/registry/ip/67.199.248.0">https://rdap.arin.net/registry/ip/67.199.248.0</a></td>
</tr>
<tr>
<td>OrgName</td>
<td>Bitly Inc</td>
</tr>
<tr>
<td>OrgId</td>
<td>BITLY</td>
</tr>
<tr>
<td>Address</td>
<td>139 5th Ave</td>
</tr>
<tr>
<td>City</td>
<td>New York</td>
</tr>
<tr>
<td>StateProv</td>
<td>NY</td>
</tr>
<tr>
<td>PostalCode</td>
<td>10010</td>
</tr>
<tr>
<td>Country</td>
<td>US</td>
</tr>
<tr>
<td>RegDate</td>
<td>2011-11-18</td>
</tr>
<tr>
<td>Updated</td>
<td>2016-04-28</td>
</tr>
<tr>
<td>Ref</td>
<td><a href="https://rdap.arin.net/registry/entity/BITLY">https://rdap.arin.net/registry/entity/BITLY</a></td>
</tr>
</tbody>
</table>

**WHOIS record (June 7, 2021)**

**Clean text**

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetRange</td>
<td>208.95.112.0 - 208.95.115.255</td>
</tr>
<tr>
<td>CIDR</td>
<td>208.95.112.0/22</td>
</tr>
<tr>
<td>NetName</td>
<td>TUTL-5</td>
</tr>
<tr>
<td>NetHandle</td>
<td>NET-208-95-112-0-1</td>
</tr>
<tr>
<td>Parent</td>
<td>NET208 (NET-208-0-0-0-0)</td>
</tr>
<tr>
<td>NetType</td>
<td>Direct Allocation</td>
</tr>
<tr>
<td>OriginAS</td>
<td>AS55334</td>
</tr>
<tr>
<td>Organization</td>
<td>Total Uptime Technologies, LLC (TUTL-1)</td>
</tr>
<tr>
<td>RegDate</td>
<td>2019-07-11</td>
</tr>
<tr>
<td>Updated</td>
<td>2019-07-11</td>
</tr>
<tr>
<td>Comment</td>
<td>Enhanced Internet Delivery / ADC-as-a-Service Platform</td>
</tr>
<tr>
<td>Comment</td>
<td>Abuse or security issues, please contact us at https://</td>
</tr>
<tr>
<td>Ref</td>
<td><a href="https://rdap.arin.net/registry/ip/208.95.112.0">https://rdap.arin.net/registry/ip/208.95.112.0</a></td>
</tr>
<tr>
<td>OrgName</td>
<td>Total Uptime Technologies, LLC</td>
</tr>
<tr>
<td>OrgId</td>
<td>TUTL-1</td>
</tr>
<tr>
<td>Address</td>
<td>PO Box 2228</td>
</tr>
<tr>
<td>City</td>
<td>Skyland</td>
</tr>
<tr>
<td>StateProv</td>
<td>NC</td>
</tr>
<tr>
<td>PostalCode</td>
<td>28776</td>
</tr>
<tr>
<td>Country</td>
<td>US</td>
</tr>
<tr>
<td>RegDate</td>
<td>2011-09-26</td>
</tr>
<tr>
<td>Updated</td>
<td>2017-01-28</td>
</tr>
<tr>
<td>Ref</td>
<td><a href="https://rdap.arin.net/registry/entity/TUTL-1">https://rdap.arin.net/registry/entity/TUTL-1</a></td>
</tr>
</tbody>
</table>
Command Executed

- "C:\Program Files\Java\re7\bin\javaw.exe" -jar C:\Users\Lucas\AppData\Local\Temp\SCbjyWcdc.jar
- wscript C:\Users\Lucas\obsrmtvwfv.js
- "C:\Program Files\Java\re7\bin\javaw.exe" -jar C:\Users\Lucas\AppData\Roaming\ebjsencja.txt
- "C:\Program Files\Java\re7\bin\java.exe" -jar C:\Users\Lucas\ebjsencja.txt
- cmd /c schtasks /create /sc minute /mo 30 /tn Skype /tr C:\Users\Lucas\AppData\Roaming\ebjsencja.txt
- "C:\Program Files\Java\re7\bin\java.exe" -jar C:\Users\Lucas\AppData\Roaming\ebjsencja.txt
- schtasks /create /sc minute /mo 30 /tn Skype /tr C:\Users\Lucas\AppData\Roaming\ebjsencja.txt
- cmd.exe /c "wmic /node: /namespace:\"root\cimv2\" path win32_logicaldisk get volumeserialnumber /format:list"
- wmic /node: /namespace:\"root\cimv2\" path win32_logicaldisk get volumeserialnumber /format:list
- cmd.exe /c "wmic /node: /namespace:\"root\cimv2\" path win32_operatingsystem get caption,OSArchitecture /format:list"
- wmic /node: /namespace:\"root\cimv2\" path win32_operatingsystem get caption,OSArchitecture /format:list
Mitigation Strategy And Techniques

Specific To StrRat Malware

Microsoft 365 Defender can be used to defend systems from this stain of malware. Following queries can be used in order to identify malicious behaviour related to StrRat and similar threats.

Inspecting Machines for Malware

```
DeviceProcessEvents
| where InitiatingProcessFileName in~("java.exe", "javaw.exe") and
  InitiatingProcessCommandLine has "roaming"
| where FileName == 'cmd.exe' and ProcessCommandLine has 'path
  antivirusproduct get displayname'
```
Inspecting Emails For Malicious Spam

```sql
EmailUrlInfo
| where UrlDomain has any ('metroscamngltg .co.uk',
  'pg-finacesolutions.co.uk',
  'jpfletcherconsultancy .co.uk',
  'buildersworlinc.co.uk',
  'bentlycon#build .co.uk',
  'alfredoscafeltd.co.uk',
  'zincocorporation.co.uk',
  'playerscireleinc.co.uk',
  'tg-craedinc.co.uk',
  'adamridley.co.uk',
  'we#coa$truedetaxis .co.uk',
  'sivospremiumclub.co.uk',
  'gossyxperience.co.uk',
  'jeffersonsandc .co.uk',
  'fillinaresortsltd .co.uk',
  'tk-consultancyltd.co.uk')
```

Inspecting Machines For Malware Persistence

```sql
DeviceProcessEvents
| where InitiatingProcessFileName in~("java.exe","javaw.exe")
| where FileName == 'cmd.exe' and ProcessCommandLine has_all("schtasks /create", "tn Skype")
```

Intrusion Detection System Rules

**Sslbl: Malicious Ja3 Ssl-client Fingerprint Detected (Tofsee)**

```sql
alert tls any any -> any any (msg:"SSLBL: Malicious JA3 SSL-Client Fingerprint detected (Tofsee)"; ja3_hash;
  content:"a61299f9b501adcf620b9275d79d4ac6"; reference:url,
  sslbl.abuse.ch/ja3-fingerprints/ a61299f9b501adcf620b9275d79d4ac6/;
  sid:906200050; rev:1;)
```
Malware Strrat Cnc Checkin

```
alert tcp $HOME_NET any -> $EXTERNAL_NET any (msg:"ET MALWARE STRRAT CnC Checkin"; flow:established,to_server; dsize:< 300;
content:"ping/7c|STRRAT/7c"; depth:12; fash_pattern; content:"|7c|"
distance: 0; content:"|7c|"; distance: 0; content:"|7c|"; distance: 0;
content:"|7c|"; distance: 0; content:"|7c|"; distance: 0; classtype:command-and-control; sid:2030358; rev:1; metadata:affected_product
Windows_XP_Vista_7_8_10_Server_32_64_Bit, attack_target Client_Endpoint,
created_at 2020_06_19, deployment Perimeter, former_category MALWARE,
malware_family STRRAT, signature_severity Major, updated_at 2020_06_19;)
```

Yara Rule For Detection

```
import "hash"
rule detect_strrat_malware {
    meta:
        description = "Detects StrRat Malware"
        organisation = "cyberpeace"
        date = "2021-06-7"

    strings:
        $magic = { 50 4b }
        $string1 = "nozocqvdou/resources/neomcrngiw"
        $string2 = "nozocqvdou/Msrppteacsm.class"
        $string3 = "nozocqvdou/resources/neomcrngiw"
        $string4 = "nozocqvdou/Mgutdjgplhq.class"
        $string5 = "nozocqvdou/Mjowcbqlzyz.class"
        $string6 = "nozocqvdou/Msrppteacsm.class"

    condition:
        ($magic and $string1 or $string2 or $string3 or $string4 or
        $string5 or $string6) or
        hash.sha256(0,filesize)=
        "0be8c1fa0f4bdb661b3b738755281d821c53974385c79ddeeeca657d91f937db5"
}
```
General Recommendation

Utilize a Next-Generation Antivirus Solution for Your Enterprise

VirusTotal results clearly show that this malware is easily detected by most industry recommended solutions. Thus using a good AV solution on your computer would easily stop these kinds of attacks.

Continuously Apply Software Updates and Patches When Released

Weaknesses seem when fundamental programming gets out of date. This makes openings in the framework that can become points of passage for a wide range of awful dangers. Along these lines, introducing vital updates when they are conveyed is a significant advance in keeping up legitimate digital cleanliness in your business organization.

There are a couple of sensible practices that ought to be imparted in your association's working environment culture consistently, for example:

1. Not giving log-in certifications to dubious solicitations,
2. Trying not to open messages from new sources,
3. Distinguishing counterfeit marking in messages,
4. Checking the objective of a connection prior to opening it, also, in every case twofold checking the legitimacy of different requests.

Square File Attachments That Are Associated with Malware

Another digital cleanliness practice that you can authorize for the online wellbeing of your business is to hinder email connections containing record augmentations that are most ordinarily connected with malware, to be specific .dll and .exe. Make it one stride further by obstructing augmentations that can't be as expected filtered by your antivirus, for example, .zip. This can forestall an Emotet malware contamination, as the Trojan depends on Dynamic Link Libraries in its assaults.

References

2. https://searchsecurity.techtarget.com/definition/credential-theft
3. https://twitter.com/msftsecintel/status/1395138347601854465


10. https://www.virustotal.com/gui/file/efb6c8d5f71fd6d8f97f71267ecef5b6bb9b881bcca795113bfb05c514ce00f/detection

11. https://www.virustotal.com/gui/file/5d5ed41f5159cf4d0d1f5b235e0b2f510a967022e8537f872a1dee627485cdd77/details

12. https://www.virustotal.com/gui/file/0be8c1fa0f4dbb66183b73b8755281d821c53974385c79ddeca657d91f937db5/detection


14. https://www.virustotal.com/gui/-file/a1660698b8655cb72f71693178d2804caba8aa12c3a25446c6f7c665a30f2ff/detection


Revision Table:

<table>
<thead>
<tr>
<th>SI. NO.</th>
<th>DESCRIPTION</th>
<th>STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>Initial Release</td>
<td>Final</td>
</tr>
</tbody>
</table>

Issued by

Research wing; CyberPeace Foundation
Research Wing, Autobot Infosec Private Ltd.