dns_sec

June 13, 2024

1 Network Security — Exercise 5: Domain Name System Security & Network Firewalls

Noah Link, Jan Pfeifer, Julian Weske
Date: 20.06.2024

1.1 Domain Name System Security (Time spent: xx h)

1.1.1 Investigating Legal Websites

Initializing the project and dnsdbq worked just fine:

```
./dnsdbq -u circl -n www.heise.de
;; record times: 2023-10-08 18:53:22 .. 2024-06-12 16:29:03 (~247d 21h 35m)
;; count: 369
193.99.144.85  A  www.heise.de

... skipped ...

;; record times: 2024-01-21 15:44:12 .. 2024-06-11 23:03:38 (~142d 7h 19m)
;; count: 7
www.heise.de  PTR  85.144.99.193.in-addr.arpa
```

1.1.2 Passive DNS

- **What is passive DNS?**
  Passive DNS is a method of collecting, storing, and analyzing DNS query and response data from recursive DNS servers, allowing for historical tracking of domain-to-IP mappings without actively querying DNS servers.
- **How can investigations in cybercrime benefit from passive DNS analysis?**
  - Identifying historical domain-IP mappings to uncover the infrastructure used by cybercriminals.
  - Tracking changes in domain associations, which can reveal patterns of malicious activity and help in mapping out criminal networks.
- **Name two factors that the quality of passive DNS analysis, i.e., the number of returned results, depends on.**
  - The volume and diversity of DNS data sources contributing to the passive DNS database.

– The time span over which DNS data has been collected and stored.

1.2 IT Security News

[ ]:
```
import subprocess
import re


def query_dnsdbq(query_type, query_value):
    command = ["./source_dnsdbq/dnsdbq", "-u", "circl", f"-{query_type}",␣
 ↪query_value]

    try:
        result = subprocess.run(command, capture_output=True, text=True,␣
 ↪check=True)
        ip_addresses = re.findall(r"\b(?:\d{1,3}\.){3}\d{1,3}\b", result.stdout)

        return ip_addresses
    except subprocess.CalledProcessError as e:
        print(f"Error running dnsdbq: {e}")
        print(f"Standard Error: {e.stderr}")
        return None


def query_ptr_records(ip_address):
    command = ["./source_dnsdbq/dnsdbq", "-u", "circl", "-r", ip_address]

    try:
        result = subprocess.run(command, capture_output=True, text=True,␣
 ↪check=True)
        fqdns = re.findall(r"(\S+)\s+PTR\s+", result.stdout)

        return fqdns
    except subprocess.CalledProcessError as e:
        print(f"Error running dnsdbq for PTR records: {e}")
        print(f"Standard Error: {e.stderr}")
        return None


def query_aaaa_records(ip_address):
    command = ["./source_dnsdbq/dnsdbq", "-u", "circl", "-r", ip_address]

    try:
        result = subprocess.run(command, capture_output=True, text=True,␣
 ↪check=True)
        ipv6_addresses = re.findall(
            r"\b(?:[0-9a-fA-F]{1,4}:){7}[0-9a-fA-F]{1,4}\b", result.stdout
```

```
        )

        return ipv6_addresses
    except subprocess.CalledProcessError as e:
        print(f"Error running dnsdbq for AAAA records: {e}")
        print(f"Standard Error: {e.stderr}")
        return None
```

1. How many IPv4 addresses exist for this site? List all addresses (i.e., A records) you found:

[ ]:
```
# 1.1.2.1
query_type = "n"
query_value = "www.heise.de"
ipv4_addresses = query_dnsdbq(query_type, query_value)

if ipv4_addresses:
    print(f"Number of IPv4 addresses found for {query_value}:␣
 ↪{len(ipv4_addresses)}")
    print("IPv4 Addresses:")
    for address in ipv4_addresses:
        print(address)
else:
    print("No IPv4 addresses found or there was an error.")
```

```
Number of IPv4 addresses found for www.heise.de: 7
IPv4 Addresses:
193.99.144.85
5.8.0.0
2.0.0.0
2.7.7.7
7.7.7.1
2.0.2.0
85.144.99.193
```

2. Is this site using IPv6 as well? If yes, list all addresses (i.e., AAAA records) you found.

[ ]:
```
# 1.1.2.2
ipv6_addresses = query_aaaa_records(query_value)

if ipv6_addresses:
    print(f"\nNumber of IPv6 addresses found for {query_value}:␣
 ↪{len(ipv6_addresses)}")
    print("IPv6 Addresses:")
    for address in ipv6_addresses:
        print(address)
else:
    print("No IPv6 addresses found or there was an error.")
```

```
Number of IPv6 addresses found for www.heise.de: 1
IPv6 Addresses:
2a02:2e0:3fe:1001:7777:772e:2:85
```

3. Search for other Fully-Qualified Domain Names (FQDNs) that are also hosted on the first IPv4 address of www.heise.de.

```
[ ]: # 1.1.2.3
if ipv4_addresses:
    first_ipv4_address = ipv4_addresses[0]
    print(
        f"\nFinding other FQDNs hosted on the first IPv4 address␣
 ↪({first_ipv4_address}) of {query_value}:"
    )
    ptr_records = query_ptr_records(first_ipv4_address)
    if ptr_records:
        print(f"\nFQDNs for IP address {first_ipv4_address}:")
        for fqdn in ptr_records:
            print(fqdn)
    else:
        print(f"No PTR records found for IP address {first_ipv4_address}")
```

```
Finding other FQDNs hosted on the first IPv4 address (193.99.144.85) of
www.heise.de:
No PTR records found for IP address 193.99.144.85
```

4. List all FQDNs that share a common IPv6 address with www.heise.de.

```
[ ]: # 1.1.2.4
if ipv6_addresses:
    print(f"\nFinding other FQDNs that share a common IPv6 address with␣
 ↪{query_value}:")
    common_ipv6_addresses = set()
    for ipv6_address in ipv6_addresses:
        associated_fqdns = query_ptr_records(ipv6_address)
        if associated_fqdns:
            common_ipv6_addresses.add(ipv6_address)
            print(f"\nIPv6 Address: {ipv6_address}")
            print("Associated FQDNs:")
            for fqdn in associated_fqdns:
                print(fqdn)

    if not common_ipv6_addresses:
        print("No common IPv6 addresses found with associated FQDNs.")
```

```
Finding other FQDNs that share a common IPv6 address with www.heise.de:
```

```
No common IPv6 addresses found with associated FQDNs.
```

1.2.1 University Website

- How many IPv4 addresses exist for this site? List all addresses (i.e., A records) you found.
- Is this site using IPv6 as well? If yes, list all addresses (i.e., AAAA records) you found.
- Search for other Fully-Qualified Domain Names (FQDNs) that are also hosted on the first IPv4 address of www.heise.de.
- List all FQDNs that share a common IPv6 address with www.heise.de.
- Find more domain names that do not directly resolve to the IP address of www.uni-hamburg.de but also indirectly via the respective CNAME.

1.2.2 Investigating Illegal Websites

- Search for FQDNs that are hosted within the address block 104.28.21.0/24. Describe your working steps.
- Come up with at least four more appropriate search strings for filtering the FQDNs.
- Apply your search strings to the full list of FQDNs and list the filtered names.

1.3 DNS and Firewall Evasion (Time spent: xx h)

1.3.1 DNS Mechanisms and Evasion Techniques

- Why is DNS often used to bypass firewalls, and why is this a popular attack vector?
- Explain the process of how a DNS tunnel works from the client request through to the response.
- Describe a method to further cloak traffic via DNS tunneling. Provide a detailed description and analyze the overhead involved with concrete numbers and percentages.

1.3.2 DoH, DoT Implementation and Analysis

- Implement a subset of DNS over HTTPS (DoH) and DNS over TLS (DoT) to query an A record. Demonstrate the implementation by querying a public DNS server.
- What are the drawbacks of DoH and DoT, and how could these drawbacks be addressed?