Bài Lab Bảo Mật DNS

🔐 *Tự tay cấu hình DNS với CoreDNS + Unbound + mô phỏng tấn công DNS Cache Poisoning + triển khai DNSSEC*

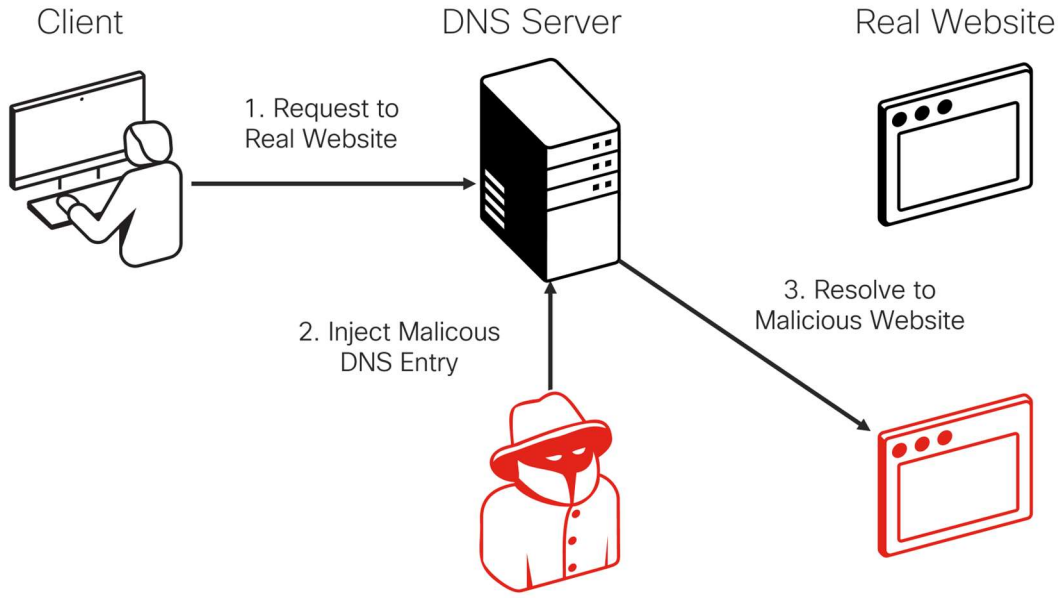
Hiểu đúng bản chất DNS và nguy cơ bị tấn công

DNS (Domain Name System) giúp chuyển đổi tên miền dễ nhớ (như www.cisco.com) thành địa chỉ IP (như 203.0.113.100). Tuy nhiên, DNS cũng là một mục tiêu hấp dẫn cho tấn công "DNS Cache Poisoning", nơi attacker chèn bản ghi giả vào cache của resolver và đánh lừa client truy cập vào trang web giả mạo.

Ví dụ: bạn gõ www.cisco.com nhưng bị dẫn đến trang giả mạo để lấy cắp tài khoản!

Mục tiêu bài lab

- Cấu hình hệ thống DNS với CoreDNS (server) và Unbound (client resolver)
- Phân tích gói tin DNS để hiểu rõ quá trình phân giải tên miền
- Mô phỏng tấn công DNS Cache Poisoning
- Áp dụng các biện pháp phòng vệ: DNSSEC, random hóa truy vấn, quản lý cache
- Triển khai DNSSEC để bảo vệ độ tin cậy của phản hồi DNS



Chuẩn bị Lab

Môi trường gồm:
- 2 máy Linux Alpine (Client và Server)
- 1 router làm NAT (IGW)
- Switch kết nối nội bộ

Cấu hình IP nội bộ: 192.168.1.0/24

Xác minh kết nối Internet:
```bash
ping 8.8.8.8
```

Nếu dùng Cisco Modeling Labs (CML), ping có thể bị chặn — bỏ qua.

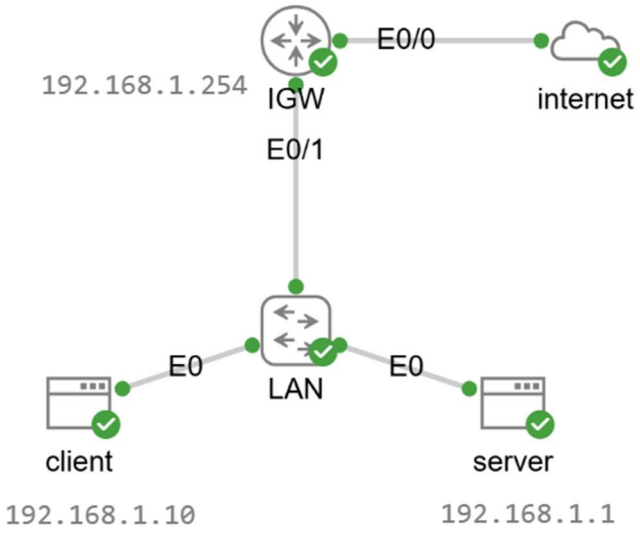
Cài đặt CoreDNS (DNS Server)

Trên máy Server:
```bash
sudo apk update
sudo apk add coredns
```

Cấu hình Corefile:
```bash
cat <<EOF | sudo tee /etc/coredns/Corefile
. {
   forward . 8.8.8.8 {
   }
   cache 30
   log
}
EOF
```

Khởi chạy CoreDNS:
```bash
sudo coredns -conf /etc/coredns/Corefile
```

Cài đặt Unbound (DNS Resolver trên client)



Trên máy Client:
```bash
sudo apk update
sudo apk add unbound
```

Tạo file cấu hình:
```bash
cat <<EOF | sudo tee /etc/unbound/unbound.conf
server:
forward-zone:
  name: "."
  forward-addr: 192.168.1.1
EOF
```

Khởi động Unbound:
```bash
sudo rc-service unbound start
```

Cấu hình hệ điều hành sử dụng localhost làm DNS:
```bash
echo "nameserver 127.0.0.1" | sudo tee /etc/resolv.conf
```

Kiểm tra hoạt động DNS

```bash
nslookup www.cisco.com
ping www.cisco.com
```

Bạn sẽ thấy quá trình phân giải tên miền, từ www.cisco.com chuyển sang một chuỗi các canonical name rồi cuối cùng thành địa chỉ IP thực tế.

Phần tiếp theo: Tấn công & Phòng thủ

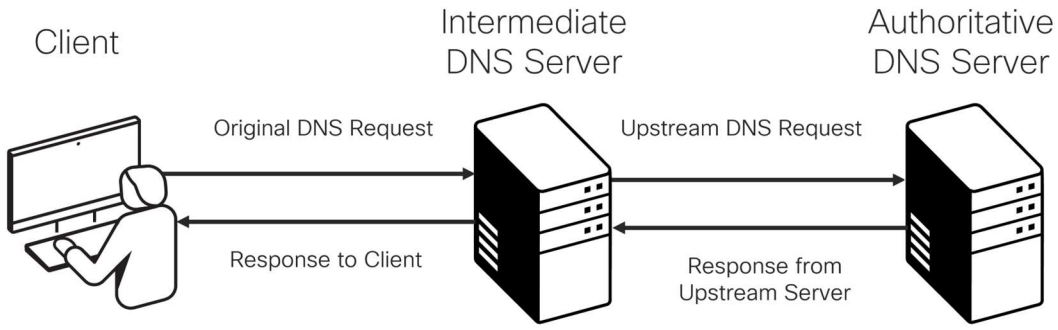
- Mô phỏng cache poisoning bằng cách tiêm bản ghi DNS giả
- Triển khai DNSSEC để xác minh tính hợp lệ phản hồi DNS
- Phân tích log + gói tin thực tế để hiểu chi tiết cách DNSSEC ngăn chặn giả mạo

Kết luận

DNS là nền tảng của Internet nhưng lại rất dễ bị khai thác nếu không cấu hình và bảo vệ đúng cách. Hãy thử lab này để hiểu thật sự cách mà DNS hoạt động — và phòng vệ trước khi bị tấn công.

Mổ Xẻ DNS Query, DNS Response và Tấn Công Cache Poisoning – Mô phỏng thực tế với CoreDNS & Unbound

Một trong những lỗ hổng dễ bị khai thác nhất trong hạ tầng mạng doanh nghiệp chính là DNS Cache Poisoning. Chỉ một bản ghi giả mạo, toàn bộ truy cập của người dùng có thể bị chuyển hướng đến trang giả mạo. Trong bài lab này, chúng ta sẽ vừa học vừa thực hành mô phỏng từ đầu đến cuối quá trình truy vấn DNS, quan sát qua Wireshark/Packet Capture, và cuối cùng là giả lập tấn công cache poisoning trên CoreDNS. Đây là nội dung cực kỳ bổ ích cho anh em học MCSA, Azure, AWS hoặc triển khai lab bảo mật mạng.



I. Quá trình DNS Query và DNS Response diễn ra như thế nào?

1. Mô hình mạng Lab:
- Client: 192.168.1.10 – có cài Unbound làm DNS Resolver.
- Server: 192.168.1.1 – chạy CoreDNS, forward truy vấn lên DNS công cộng.
- Public DNS: 8.8.8.8 (hoặc 208.67.222.222 tùy môi trường).

2. Quy trình phân giải DNS:

Khi client cần truy cập www.ietf.org, chuỗi sự kiện xảy ra:
- Client → Unbound (192.168.1.1): gửi query loại A (IPv4).
- Unbound → CoreDNS (nếu chưa có cache): forward tiếp lên DNS công cộng.
- CoreDNS → 8.8.8.8: truy vấn đích.
- 8.8.8.8 trả lời: IP hợp lệ, kèm RRSIG nếu có DNSSEC.
- CoreDNS trả về cho Unbound → Client nhận kết quả: lưu cache nếu được.

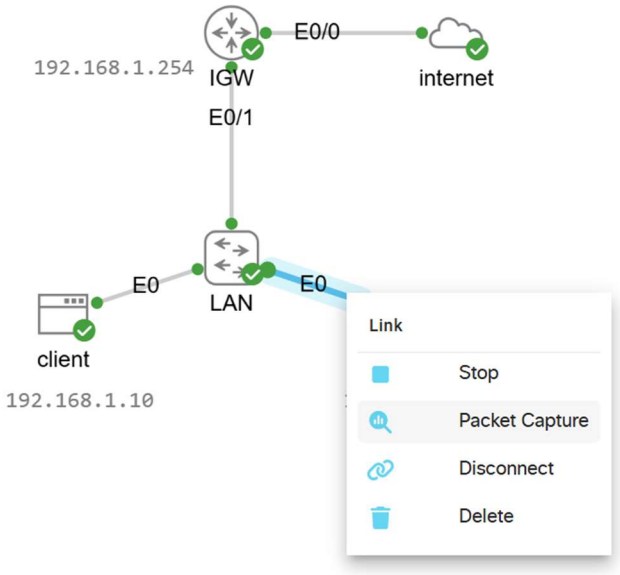
Ví dụ lệnh trên client:
nslookup -querytype=A www.ietf.org
Kết quả:
Name: www.ietf.org

Address: 104.16.44.99
Address: 104.16.45.99

3. Phân tích gói tin:

Dùng tính năng Packet Capture của Cisco Modeling Labs để phân tích:
- Bắt gói DNS trên link E0 từ server → switch.
- Nhập dns trong filter.
- Click từng packet để thấy chi tiết: Query ID, UDP source port, response từ 8.8.8.8, A record trả về.

II. Thực hành mô phỏng DNS Cache Poisoning



DNS Cache Poisoning là gì?

Là kỹ thuật tiêm bản ghi DNS giả vào bộ nhớ đệm của resolver, khiến client nhận IP sai, có thể bị chuyển hướng tới website giả mạo/phishing.

1. Cơ chế tấn công cổ điển:

- Dự đoán transaction ID và source port.
- Gửi hàng loạt phản hồi giả mạo trước khi phản hồi hợp lệ đến.
- Nếu resolver nhận phản hồi sai, nó lưu lại và trả cho các client sau.

2. Mô phỏng tấn công đơn giản:

Thay vì fake packet, ta chỉnh file cấu hình CoreDNS để trả về IP giả.

Các bước thực hiện:
- Dừng CoreDNS: Ctrl + C
- Sửa Corefile:

sudo vi /etc/coredns/Corefile
Nội dung:

```
. {
  hosts {
     192.0.2.100 www.ietf.org
     fallthrough
  }
  forward . 8.8.8.8 {
  }
  cache 30
  log
}
```

- Hoặc dùng một lệnh duy nhất để tạo:

```
cat <<EOF | sudo tee /etc/coredns/Corefile
. {
  hosts {
     192.0.2.100 www.ietf.org
     fallthrough
  }
  forward . 8.8.8.8 {
  }
  cache 30
  log
}
EOF
```

- Khởi động lại CoreDNS:

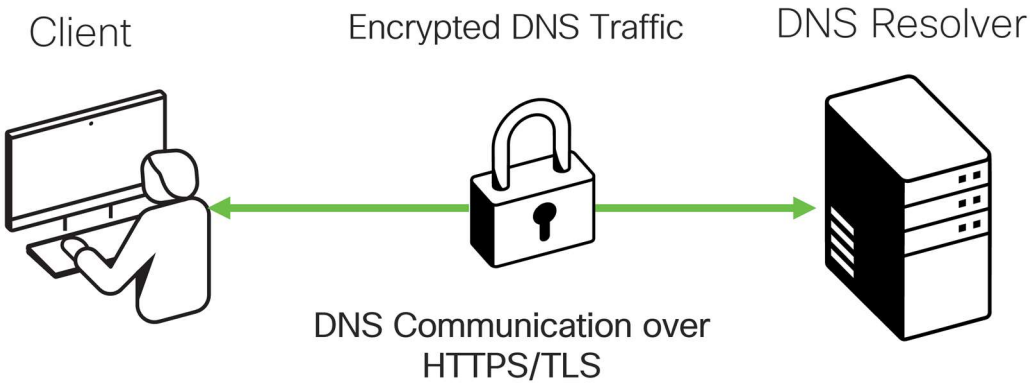
```
sudo coredns -conf /etc/coredns/Corefile
```

- Truy vấn lại từ client:

```
nslookup -querytype=A www.ietf.org
```

Kết quả:

```
Name: www.ietf.org
Address: 192.0.2.100
```



III. Kiến thức thực chiến dành cho bạn

- CoreDNS và Unbound là hai công cụ phổ biến để thiết lập môi trường DNS trong nội bộ và kiểm thử bảo mật.
- Việc phân tích packet DNS giúp bạn hiểu chi tiết về transaction ID, UDP port, và cách thức giao tiếp của recursive resolver.
- Việc chỉnh sửa file cấu hình DNS chính là cách mà attacker dùng để "tiêm độc" vào quá trình phân giải tên miền.

IV. Gợi ý nâng cao

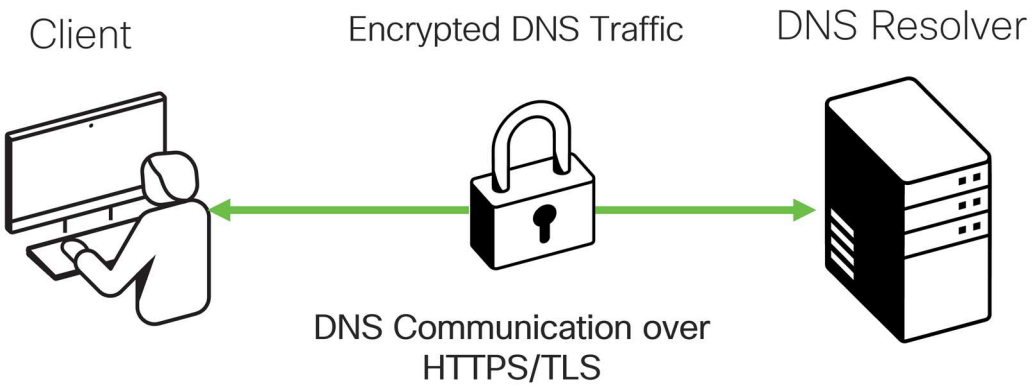
- Bật DNSSEC validation ở Unbound để kiểm thử chữ ký RRSIG.
- Mô phỏng DNS over HTTPS (DoH) để kiểm thử bảo mật nâng cao.
- Viết script giám sát DNS logs để phát hiện truy vấn bất thường từ client.

Nếu bạn đang học MCSA, Azure Fundamentals, AWS, hoặc làm quản trị hệ thống, đừng bỏ qua việc nắm chắc nguyên lý DNS – không chỉ để vận hành, mà còn để bảo vệ người dùng nội bộ khỏi bị điều hướng độc hại.

Bảo vệ hệ thống DNS khỏi tấn công DNS Cache Poisoning bằng DNSSEC

DNS Cache Poisoning là gì?

Khi client gửi truy vấn tên miền (như www.cisco.com) đến máy chủ DNS trung gian, nếu phản hồi trả về bị giả mạo (spoofed) và được lưu vào bộ nhớ đệm (cache), thì tất cả truy vấn sau đó đều sẽ bị chuyển hướng sai. Hậu quả? Người dùng truy cập vào trang web giả mạo, nơi kẻ tấn công có thể đánh cắp mật khẩu, cookie, session hoặc cài mã độc vào hệ thống.



Chiến lược phòng chống DNS Cache Poisoning

- DNSSEC – Gốc rễ của sự tin cậy:
  Sử dụng chữ ký số để đảm bảo tính toàn vẹn và xác thực của bản ghi DNS. Nếu chữ ký không hợp lệ, resolver sẽ từ chối phản hồi.

- DNS over TLS (DoT) hoặc DNS over HTTPS (DoH):
  Mã hóa truy vấn DNS để ngăn kẻ tấn công đọc hoặc thay đổi dữ liệu khi đang truyền.

- Random hóa truy vấn:
  Thêm tính ngẫu nhiên cho source port và transaction ID.

- TTL ngắn và flush cache định kỳ:
  Giảm thời gian sống của bản ghi DNS.

- Giới hạn quyền và theo dõi log:
  Hạn chế người chỉnh sửa zone file, audit log thường xuyên.

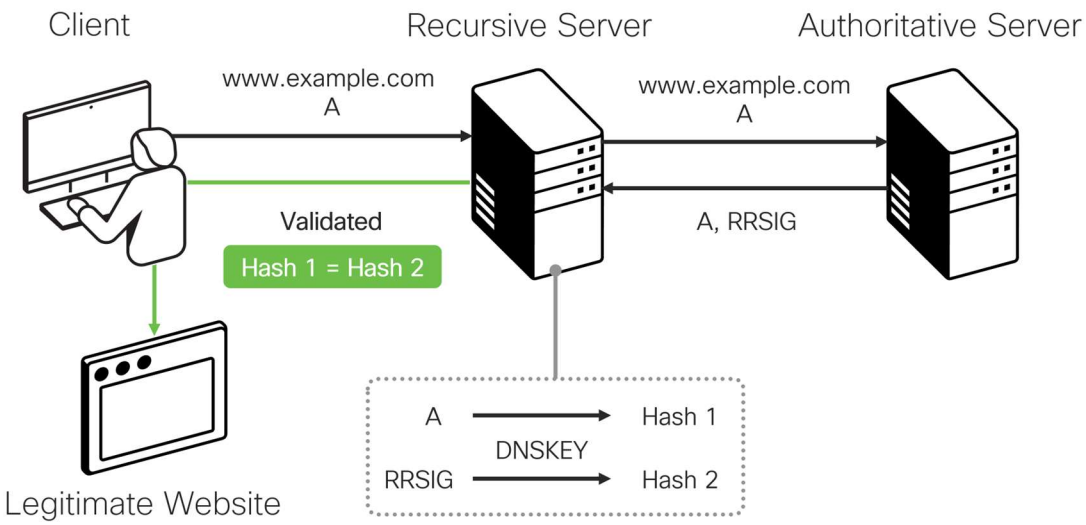
Triển khai DNSSEC với Unbound trên máy khách

Bước 1: Tải Trust Anchor
sudo mkdir -p /var/lib/unbound
sudo chown unbound:unbound /var/lib/unbound
sudo unbound-anchor -a /var/lib/unbound/root.key

Bước 2: Cấu hình unbound.conf
sudo tee /etc/unbound/unbound.conf <<EOF
server:
  auto-trust-anchor-file: "/var/lib/unbound/root.key"
forward-zone:
  name: "."
  forward-addr: 192.168.1.1
EOF

Bước 3: Khởi động lại Unbound
sudo rc-service unbound restart

Bước 4: Cài dig để kiểm tra DNSSEC
sudo apk add bind-tools
dig +dnssec @127.0.0.1 www.example.com



Kiểm chứng hiệu quả của DNSSEC

Giả lập truy vấn với một bản ghi đã bị chỉnh sửa (VD: www.ietf.org) và xem phản hồi:
dig +dnssec @127.0.0.1 www.ietf.org
Kết quả trả về SERVFAIL => DNSSEC đã chặn phản hồi độc hại thành công.

Kết luận cho cộng đồng MCSA / Azure / AWS

DNSSEC không phải là một tính năng xa lạ – nó là một chuẩn bắt buộc nếu bạn muốn vận hành một hệ thống DNS an toàn trong môi trường on-prem, hybrid cloud hoặc production trên AWS, Azure. Một dòng cấu hình đúng = Ngăn cả mạng khỏi bị tấn công âm thầm