Network Security

DNS (In)security

Radboud University, The Netherlands

Spring 2018

A short recap

- Routing means directing (Internet) traffic to its target
- Internet is divided into $\approx 61,000$ Autonomous Systems

A short recap

- Routing means directing (Internet) traffic to its target
- Internet is divided into $\approx 61,000$ Autonomous Systems
- Routing inside an AS uses Interior Gateway Protocols (RIP, OSPF)
- Routing between ASs uses Border Gateway Protocol (BGP)

A short recap

- Routing means directing (Internet) traffic to its target
- Internet is divided into $\approx 61,000$ Autonomous Systems
- Routing inside an AS uses Interior Gateway Protocols (RIP, OSPF)
- Routing between ASs uses Border Gateway Protocol (BGP)
- Large-scale routing attacks are not suitable for homework…

A short recap

- Routing means directing (Internet) traffic to its target
- Internet is divided into $\approx 61,000$ Autonomous Systems
- Routing inside an AS uses Interior Gateway Protocols (RIP, OSPF)
- Routing between ASs uses Border Gateway Protocol (BGP)
- Large-scale routing attacks are not suitable for homework…
- Smaller-scale attacks:

A short recap

- Routing means directing (Internet) traffic to its target
- Internet is divided into $\approx 61,000$ Autonomous Systems
- Routing inside an AS uses Interior Gateway Protocols (RIP, OSPF)
- Routing between ASs uses Border Gateway Protocol (BGP)
- Large-scale routing attacks are not suitable for homework…
- Smaller-scale attacks:
  - Source-routing attacks

A short recap

- Routing means directing (Internet) traffic to its target
- Internet is divided into $\approx 61,000$ Autonomous Systems
- Routing inside an AS uses Interior Gateway Protocols (RIP, OSPF)
- Routing between ASs uses Border Gateway Protocol (BGP)
- Large-scale routing attacks are not suitable for homework...
- Smaller-scale attacks:
  - Source-routing attacks
  - ICMP redirect attacks

A short recap

- Routing means directing (Internet) traffic to its target
- Internet is divided into $\approx 61,000$ Autonomous Systems
- Routing inside an AS uses Interior Gateway Protocols (RIP, OSPF)
- Routing between ASs uses Border Gateway Protocol (BGP)
- Large-scale routing attacks are not suitable for homework…
- Smaller-scale attacks:
  - Source-routing attacks
  - ICMP redirect attacks
  - Rogue DHCP (attacks not only routing)

A short recap

- Routing means directing (Internet) traffic to its target
- Internet is divided into $\approx 61,000$ Autonomous Systems
- Routing inside an AS uses Interior Gateway Protocols (RIP, OSPF)
- Routing between ASs uses Border Gateway Protocol (BGP)
- Large-scale routing attacks are not suitable for homework…
- Smaller-scale attacks:
  - Source-routing attacks
  - ICMP redirect attacks
  - Rogue DHCP (attacks not only routing)
- Firewalls are concepts to separate networks

A short recap

- Routing means directing (Internet) traffic to its target
- Internet is divided into $\approx 61,000$ Autonomous Systems
- Routing inside an AS uses Interior Gateway Protocols (RIP, OSPF)
- Routing between ASs uses Border Gateway Protocol (BGP)
- Large-scale routing attacks are not suitable for homework...
- Smaller-scale attacks:
  - Source-routing attacks
  - ICMP redirect attacks
  - Rogue DHCP (attacks not only routing)
- Firewalls are concepts to separate networks
- Common firewall concept: Packet filtering (`iptables`)

A short recap

- Routing means directing (Internet) traffic to its target
- Internet is divided into $\approx 61,000$ Autonomous Systems
- Routing inside an AS uses Interior Gateway Protocols (RIP, OSPF)
- Routing between ASs uses Border Gateway Protocol (BGP)
- Large-scale routing attacks are not suitable for homework...
- Smaller-scale attacks:
  - Source-routing attacks
  - ICMP redirect attacks
  - Rogue DHCP (attacks not only routing)
- Firewalls are concepts to separate networks
- Common firewall concept: Packet filtering (`iptables`)
- `iptables` can also be used for NAT (and port forwarding)

A short recap

- Routing means directing (Internet) traffic to its target
- Internet is divided into $\approx 61,000$ Autonomous Systems
- Routing inside an AS uses Interior Gateway Protocols (RIP, OSPF)
- Routing between ASs uses Border Gateway Protocol (BGP)
- Large-scale routing attacks are not suitable for homework...
- Smaller-scale attacks:
  - Source-routing attacks
  - ICMP redirect attacks
  - Rogue DHCP (attacks not only routing)
- Firewalls are concepts to separate networks
- Common firewall concept: Packet filtering (`iptables`)
- `iptables` can also be used for NAT (and port forwarding)
- Tunneling can be used to circumvent a packet filter

DNS and domain names

- So far: Configure hostname/IP pairs in `/etc/hosts`
- Important for local configuration (and overrides), but does not scale

DNS and domain names

- So far: Configure hostname/IP pairs in `/etc/hosts`
- Important for local configuration (and overrides), but does not scale
- More flexible solution: *Domain Name System (DNS)*
- Idea: Query a server for a domain name, receive answer
- DNS typically uses UDP on port 53

DNS and domain names

- So far: Configure hostname/IP pairs in `/etc/hosts`
- Important for local configuration (and overrides), but does not scale
- More flexible solution: *Domain Name System (DNS)*
- Idea: Query a server for a domain name, receive answer
- DNS typically uses UDP on port 53
- Domain names have a hierarchy (different levels separated by '.')

DNS and domain names

- So far: Configure hostname/IP pairs in `/etc/hosts`
- Important for local configuration (and overrides), but does not scale
- More flexible solution: *Domain Name System (DNS)*
- Idea: Query a server for a domain name, receive answer
- DNS typically uses UDP on port 53
- Domain names have a hierarchy (different levels separated by '.')
- Highest domain: root domain (empty string)

DNS and domain names

- So far: Configure hostname/IP pairs in `/etc/hosts`
- Important for local configuration (and overrides), but does not scale
- More flexible solution: *Domain Name System (DNS)*
- Idea: Query a server for a domain name, receive answer
- DNS typically uses UDP on port 53
- Domain names have a hierarchy (different levels separated by '.')
- Highest domain: root domain (empty string)
- Next highest: *top-level domains (TLDs)*, e.g., `.nl`, `.org`, `.sucks`

DNS and domain names

- So far: Configure hostname/IP pairs in `/etc/hosts`
- Important for local configuration (and overrides), but does not scale
- More flexible solution: *Domain Name System (DNS)*
- Idea: Query a server for a domain name, receive answer
- DNS typically uses UDP on port 53
- Domain names have a hierarchy (different levels separated by '.')
- Highest domain: root domain (empty string)
- Next highest: *top-level domains (TLDs)*, e.g., `.nl`, `.org`, `.sucks`
- Administration of top-level domains by Internet Corporation for Assigned Names and Numbers (ICANN)
- Administrations of domains below a TLD by registries, e.g., Stichting Internet Domeinregistratie Nederland (SIDN) for `.nl`

DNS and domain names

- So far: Configure hostname/IP pairs in `/etc/hosts`
- Important for local configuration (and overrides), but does not scale
- More flexible solution: *Domain Name System (DNS)*
- Idea: Query a server for a domain name, receive answer
- DNS typically uses UDP on port 53
- Domain names have a hierarchy (different levels separated by '.')
- Highest domain: root domain (empty string)
- Next highest: *top-level domains (TLDs)*, e.g., `.nl`, `.org`, `.sucks`
- Administration of top-level domains by Internet Corporation for Assigned Names and Numbers (ICANN)
- Administrations of domains below a TLD by registries, e.g., Stichting Internet Domeinregistratie Nederland (SIDN) for `.nl`
- DNS servers are typically resposible for one specific domain (DNS zone)

DNS servers and requests

- Two kind of DNS servers: recursive and authoritative
- Recursive servers (or DNS caches)
  - forward requests to other servers,
  - remember (cache) the reply for a certain amount of time

DNS servers and requests

- Two kind of DNS servers: recursive and authoritative
- Recursive servers (or DNS caches)
  - forward requests to other servers,
  - remember (cache) the reply for a certain amount of time
- Authoritative servers are responsible for a certain domain (or DNS zone) and
  - know the hosts in their domain,
  - know the authoritative DNS servers of their subdomains

DNS servers and requests

- Two kind of DNS servers: recursive and authoritative
- Recursive servers (or DNS caches)
  - forward requests to other servers,
  - remember (cache) the reply for a certain amount of time
- Authoritative servers are responsible for a certain domain (or DNS zone) and
  - know the hosts in their domain,
  - know the authoritative DNS servers of their subdomains
- Two types of requests: recursive or iterative

DNS servers and requests

- Two kind of DNS servers: recursive and authoritative
- Recursive servers (or DNS caches)
  - forward requests to other servers,
  - remember (cache) the reply for a certain amount of time
- Authoritative servers are responsible for a certain domain (or DNS zone) and
  - know the hosts in their domain,
  - know the authoritative DNS servers of their subdomains
- Two types of requests: recursive or iterative
- Recursive request (to a DNS cache): give me the answer or an error

DNS servers and requests

- Two kind of DNS servers: recursive and authoritative
- Recursive servers (or DNS caches)
  - forward requests to other servers,
  - remember (cache) the reply for a certain amount of time
- Authoritative servers are responsible for a certain domain (or DNS zone) and
  - know the hosts in their domain,
  - know the authoritative DNS servers of their subdomains
- Two types of requests: recursive or iterative
- Recursive request (to a DNS cache): give me the answer or an error
- Iterative request (to an authoritative server): give me the answer or tell me who might know

DNS example

- You try to access `sandor.cs.ru.nl`, send request to DNS cache (e.g., 131.174.117.20)

DNS example

- You try to access `sandor.cs.ru.nl`, send request to DNS cache (e.g., 131.174.117.20)
- 131.174.117.20 may know the DNS server for *top-level domain* `.nl`:

```
ns1.dns.nl    193.176.144.5
```

DNS example

- You try to access `sandor.cs.ru.nl`, send request to DNS cache (e.g., 131.174.117.20)
- 131.174.117.20 may know the DNS server for *top-level domain* `.nl`:

  `ns1.dns.nl    193.176.144.5`

- 131.174.117.20 asks `ns1.dns.nl` for `ru.nl` nameserver:

  `ns1.surfnet.nl    192.87.106.101`

DNS example

- You try to access `sandor.cs.ru.nl`, send request to DNS cache (e.g., 131.174.117.20)
- 131.174.117.20 may know the DNS server for *top-level domain* `.nl`:

  `ns1.dns.nl    193.176.144.5`

- 131.174.117.20 asks `ns1.dns.nl` for `ru.nl` nameserver:

  `ns1.surfnet.nl    192.87.106.101`

- 131.174.117.20 asks `ns1.surfnet.nl` for `cs.ru.nl` nameserver:

  `ns2.science.ru.nl    131.174.16.133`

DNS example

- You try to access `sandor.cs.ru.nl`, send request to DNS cache (e.g., 131.174.117.20)
- 131.174.117.20 may know the DNS server for *top-level domain* `.nl`:

  `ns1.dns.nl    193.176.144.5`

- 131.174.117.20 asks `ns1.dns.nl` for `ru.nl` nameserver:

  `ns1.surfnet.nl    192.87.106.101`

- 131.174.117.20 asks `ns1.surfnet.nl` for `cs.ru.nl` nameserver:

  `ns2.science.ru.nl    131.174.16.133`

- 131.174.117.20 asks `ns2.science.ru.nl` for `sandor.cs.ru.nl` IP address:

  `sandor.cs.ru.nl    131.174.142.4`

DNS example

- You try to access `sandor.cs.ru.nl`, send request to DNS cache (e.g., 131.174.117.20)
- 131.174.117.20 may know the DNS server for *top-level domain* `.nl`:

  `ns1.dns.nl    193.176.144.5`

- 131.174.117.20 asks `ns1.dns.nl` for `ru.nl` nameserver:

  `ns1.surfnet.nl    192.87.106.101`

- 131.174.117.20 asks `ns1.surfnet.nl` for `cs.ru.nl` nameserver:

  `ns2.science.ru.nl    131.174.16.133`

- 131.174.117.20 asks `ns2.science.ru.nl` for `sandor.cs.ru.nl` IP address:

  `sandor.cs.ru.nl    131.174.142.4`

- 131.174.117.20 tells your client (e.g., SSH client) the IP address of `sandor.cs.ru.nl`

DNS entry types

Type	Meaning
A	Address record: returns a 32-bit IP address, used to map hostnames to addresses
NS	Nameserver: Lists the authoritative nameservers of a DNS zone
CNAME	Canonical Name: Assigns a hostname alias to a hostname
SOA	"Start Of Authority": Lists authoritative information about the zone: primary DNS server, mail address of administrator (with @ replaced by a .), serial number, refresh times and timeouts.
MX	Mail Exchanger: Gives a mail server responsible for the domain
TXT	Text field: Originally arbitrary human-readable text, today often used for machine-readable data

DNS entry types

- Four sections in a DNS reply:
  - The QUESTION SECTION (repetition of the question)
  - The ANSWER SECTION
  - The AUTHORITY SECTION
  - The ADDITIONAL SECTION
- ADDITIONAL SECTION is particularly important for *glue records*: communicate IP addresses of authoritative DNS servers

`resolv.conf`, `dig`, and `whois`

- The list of (recursive) nameservers to access is in `/etc/resolv.conf`
- It's typically dynamically updated from DHCP information

`resolv.conf`, `dig`, and `whois`

- The list of (recursive) nameservers to access is in `/etc/resolv.conf`
- It's typically dynamically updated from DHCP information
- This is another attack vector for **rogue DHCP**!

`resolv.conf`, `dig`, and `whois`

- The list of (recursive) nameservers to access is in `/etc/resolv.conf`
- It's typically dynamically updated from DHCP information
- This is another attack vector for **rogue DHCP**!
- Command-line tool to request DNS information: `dig`, examples:
  - Find IP address of `sandor.cs.ru.nl`

    ```
    dig sandor.cs.ru.nl
    ```

`resolv.conf`, `dig`, and `whois`

- The list of (recursive) nameservers to access is in `/etc/resolv.conf`
- It's typically dynamically updated from DHCP information
- This is another attack vector for **rogue DHCP**!
- Command-line tool to request DNS information: `dig`, examples:
  - Find IP address of `sandor.cs.ru.nl`
    ```
    dig sandor.cs.ru.nl
    ```
  - Ask `ns1.dns.nl` for `ru.nl` autoritative DNS servers of `ru.nl`:
    ```
    dig @ns1.dns.nl ru.nl NS
    ```

resolv.conf, dig, and whois

- The list of (recursive) nameservers to access is in `/etc/resolv.conf`
- It's typically dynamically updated from DHCP information
- This is another attack vector for **rogue DHCP**!
- Command-line tool to request DNS information: `dig`, examples:
  - Find IP address of `sandor.cs.ru.nl`

    ```
    dig sandor.cs.ru.nl
    ```

  - Ask `ns1.dns.nl` for `ru.nl` autoritative DNS servers of `ru.nl`:

    ```
    dig @ns1.dns.nl ru.nl NS
    ```

  - Ask `ns1.science.ru.nl` for all information of `science.ru.nl`

    ```
    dig @ns1.science.ru.nl science.ru.nl ANY
    ```

`resolv.conf`, `dig`, and `whois`

- The list of (recursive) nameservers to access is in `/etc/resolv.conf`
- It's typically dynamically updated from DHCP information
- This is another attack vector for **rogue DHCP**!
- Command-line tool to request DNS information: `dig`, examples:
  - Find IP address of `sandor.cs.ru.nl`
    ```
    dig sandor.cs.ru.nl
    ```
  - Ask `ns1.dns.nl` for `ru.nl` autoritative DNS servers of `ru.nl`:
    ```
    dig @ns1.dns.nl ru.nl NS
    ```
  - Ask `ns1.science.ru.nl` for all information of `science.ru.nl`
    ```
    dig @ns1.science.ru.nl science.ru.nl ANY
    ```
  - Reverse lookup hostname for 131.174.142.4:
    ```
    dig -x 131.174.142.4
    ```

`resolv.conf`, `dig`, and `whois`

- The list of (recursive) nameservers to access is in `/etc/resolv.conf`
- It's typically dynamically updated from DHCP information
- This is another attack vector for **rogue DHCP**!
- Command-line tool to request DNS information: `dig`, examples:
  - Find IP address of `sandor.cs.ru.nl`
    ```
    dig sandor.cs.ru.nl
    ```
  - Ask `ns1.dns.nl` for `ru.nl` autoritative DNS servers of `ru.nl`:
    ```
    dig @ns1.dns.nl ru.nl NS
    ```
  - Ask `ns1.science.ru.nl` for all information of `science.ru.nl`
    ```
    dig @ns1.science.ru.nl science.ru.nl ANY
    ```
  - Reverse lookup hostname for 131.174.142.4:
    ```
    dig -x 131.174.142.4
    ```
- Find out about ICANN registration information of a domain: `whois`, e.g.:
  ```
  whois cryptojedi.org
  ```

The DNS root servers

- Whenever a DNS server does not know the authoritative DNS servers of a Domain, it asks the *DNS root servers*
- DNS root servers are extremely critical piece of Internet infrastructure
- How many are there?

The DNS root servers

- Whenever a DNS server does not know the authoritative DNS servers of a Domain, it asks the *DNS root servers*
- DNS root servers are extremely critical piece of Internet infrastructure
- How many are there? Answer: 13

The DNS root servers

- Whenever a DNS server does not know the authoritative DNS servers of a Domain, it asks the *DNS root servers*
- DNS root servers are extremely critical piece of Internet infrastructure
- How many are there? Answer: 13
- Names of these servers: `a.root-servers.net` ... `m.root-servers.net`
- Those servers are actually highly redundant, some even distributed over the globe
- Example: K-root server, run by RIPE-NCC
  - Used to be in Amsterdam
  - Now at 18 different locations

The DNS root servers

DNS root servers hit by largest DDoS ever

News By Oct. 23, 2002 12:38 pm

The largest Distributed Denial of Service (DDoS) attack in history went largely unnoticed by the general public on October 21, 2002, but it was almost a disaster, say several Internet backbone operators.

Around 5:00 P.M. Eastern time, the root servers that handle domain name resolution for all top-level domains on the Internet were subjected to an hour-long attack by thousands of "zombie" computers–PCs that have been co-opted by a hacker into participating in an attack without the knowledge of the PC owner. Of the 13 root servers in existence, only four were able to keep operating during the attack. Had the attack continued for much longer, experts say, the remaining servers may have been overwhelmed, effectively strangling the entire root Domain Name Server (DNS) system. Although many ISPs and companies maintain

http://www.geek.com/news/dns-root-servers-hit-by-largest-ddos-ever-550549/

The DNS root servers

Topic: *Security* *Follow via:*

DNS root server attack launched from Germany

Summary: *According to a published report, the recent attack against the DNS root servers was launched from a host server in Germany that controlled millions of zombie machines in South Korea*

By Ryan Naraine for Zero Day | February 22, 2007 -- 09:41 GMT

Follow @ryanaraine | *Get the ZDNet Security newsletter now*

The distributed denial-of-service attack against the DNS root servers earlier this month was launched from a host server in Germany that controlled millions of zombie machines in South Korea, according to a report in The Korea Times.

Details of the cross-continent attack, which almost took out three of the 13 official root DNS servers are beginning to surface with South Korea's ministry of information and communication confirming that a host server in Coburg, Germany ordered hijacked Windows machines in Korea to stage the attacks.

http://www.zdnet.com/blog/security/dns-root-server-attack-launched-from-germany/50

The DNS root servers

PC NEWS / OPINIONS / FEATURES / DEALS / HOW-TO / BUSINESS / SUBSC

ALL REVIEWS | LAPTOPS / TABLETS / PHONES / APPS / SOFTWARE / SECURITY

Home / Reviews / Software / Security / Anonymous' 'Operation Blackout' Goes Dark; DNS Just Fine

Anonymous' 'Operation Blackout' Goes Dark; DNS Just Fine

BY DAVID MURPHY   MARCH 31, 2012 01:08PM EST   7 COMMENTS

It doesn't appear as if Web browsing will be affected today, as Anonymous is likely not launching an attack on the Web's root DNS servers.

1.9K SHARES

http://www.pcmag.com/article2/0,2817,2402469,00.asp

DNS tunneling

- Firewalls may block anything, but typically not DNS

DNS tunneling

- Firewalls may block anything, but typically not DNS
- Idea: set up authoritative DNS server for some subdomain `tunnel.mydomain.nl`
- Encode SSH traffic as DNS requests to this server
- Tunnel SSH traffic through DNS

DNS tunneling

- Firewalls may block anything, but typically not DNS
- Idea: set up authoritative DNS server for some subdomain `tunnel.mydomain.nl`
- Encode SSH traffic as DNS requests to this server
- Tunnel SSH traffic through DNS
- This is slow (small payload, UDP is not reliable)

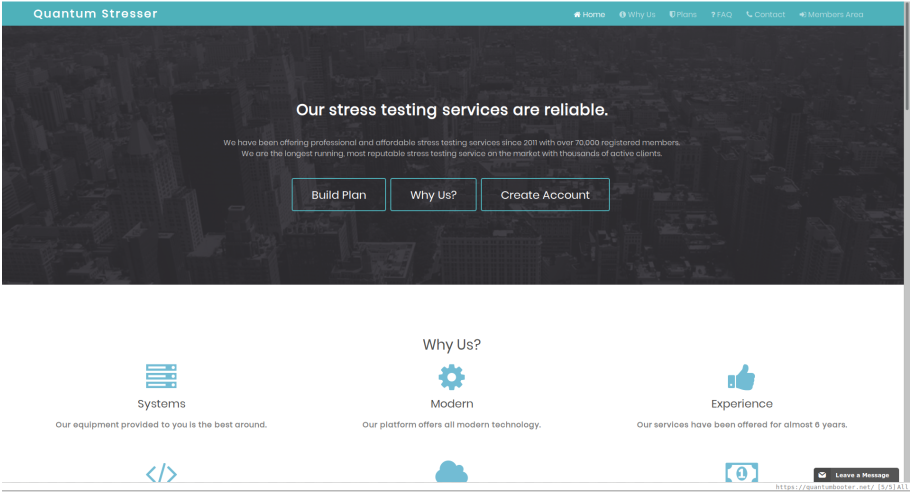
DNS tunneling

- Firewalls may block anything, but typically not DNS
- Idea: set up authoritative DNS server for some subdomain `tunnel.mydomain.nl`
- Encode SSH traffic as DNS requests to this server
- Tunnel SSH traffic through DNS
- This is slow (small payload, UDP is not reliable)
- Ready-made client/server: `ozymandns` by Kaminsky: http://dankaminsky.com/2004/07/29/51/
- Tutorial for DNS tunneling (with `ozymandns`): http://dnstunnel.de/

DNS DDoS amplification

- DNS (typically) uses UDP
- No session establishment: send request, get answer
- Answer can be much larger than the request
- Idea: Spoof IP address of DOS victim in DNS request
- Victim will receive much more data than attacker has to send
- This is called *DNS (D)DOS amplification*

DNS DDoS amplification



DNS DDoS countermeasures?

- Very hard to defend against DDOS (and DNS amplification)
- Can (temporarily) block packets from open DNS servers
- Can (temporarily) block large DNS reply packets
- Can try to filter spoofed IP addresses (“ingres and egress filtering”)

DNS spoofing

- Probably most obvious DNS attack: send wrong answer
- Send wrong answer to client: hit one target
- Send wrong answer to DNS cache: hit many targets
- Answers contain “validity period”
- It’s possible to poison DNS caches for a pretty long time

In the old days

```
$ dig @ns1.attacker.com www.attacker.com
    ;; ANSWER SECTION:
    www.attacker.com.    120      IN    A    123.45.67.8

    ;; AUTHORITY SECTION:
    attacker.com.        86400    IN    NS   ns1.attacker.com.

    ;; ADDITIONAL SECTION:
    ns1.attacker.com.    604800   IN    A    123.45.67.89
    www.target.com.       43200   IN    A     66.66.66.66
```

In the old days

```
$ dig @ns1.attacker.com www.attacker.com
    ;; ANSWER SECTION:
    www.attacker.com.    120      IN    A     123.45.67.8

    ;; AUTHORITY SECTION:
    attacker.com.        86400    IN    NS    ns1.attacker.com.

    ;; ADDITIONAL SECTION:
    ns1.attacker.com.    604800   IN    A     123.45.67.89
    www.target.com.       43200   IN    A      66.66.66.66
```

The bailiwick check

- Idea of the attack: wrong entry for `www.target.com` ends up in cache
- Countermeasure (since 1997): use *bailiwick* check
- Reject ADDITIONAL information if the requested server is not authorized to answer for the domain

Short interlude: A bailiwick

Definition of BAILIWICK

1. the office or jurisdiction of a bailiff
2. a special domain

Source: http://www.merriam-webster.com/dictionary/bailiwick

Short interlude: A bailiwick

Definition of BAILIFF

1. **a:** an official employed by a British sheriff to serve writs and make arrests and executions
   b: a minor officer of some United States courts usually serving as a messenger or usher
2. chiefly British: one who manages an estate or farm

Source: http://www.merriam-webster.com/dictionary/bailiff

The race for the answer

- A client is asking for an IP address; if attacker answers first, he wins

The race for the answer

- A client is asking for an IP address; if attacker answers first, he wins
- Not quite that easy: Request contains 16-bit Query ID (QID)
- DNS reply has to have the right ID
- Attacker has to guess the ID
- This is a bit similar to the TCP ISN in a session-stealing attack

The race for the answer

- A client is asking for an IP address; if attacker answers first, he wins
- Not quite that easy: Request contains 16-bit Query ID (QID)
- DNS reply has to have the right ID
- Attacker has to guess the ID
- This is a bit similar to the TCP ISN in a session-stealing attack
- In the old days use simply increasing IDs: easy for an attacker to figure out
- Nowadays use randomized 16-bit ID

The race for the answer

- A client is asking for an IP address; if attacker answers first, he wins
- Not quite that easy: Request contains 16-bit Query ID (QID)
- DNS reply has to have the right ID
- Attacker has to guess the ID
- This is a bit similar to the TCP ISN in a session-stealing attack
- In the old days use simply increasing IDs: easy for an attacker to figure out
- Nowadays use randomized 16-bit ID
- The attacker can start the race:
  - Lure victim to website at `www.attacker.com`
  - Include picture from `www.target.com`
  - Attacker sees website request, knows that DNS request for `www.target.com` will follow

The race for the answer

- A client is asking for an IP address; if attacker answers first, he wins
- Not quite that easy: Request contains 16-bit Query ID (QID)
- DNS reply has to have the right ID
- Attacker has to guess the ID
- This is a bit similar to the TCP ISN in a session-stealing attack
- In the old days use simply increasing IDs: easy for an attacker to figure out
- Nowadays use randomized 16-bit ID
- The attacker can start the race:
  - Lure victim to website at `www.attacker.com`
  - Include picture from `www.target.com`
  - Attacker sees website request, knows that DNS request for `www.target.com` will follow
- Attacker can send *many packets*
- Attacker can also try to run DOS against real DNS server

Kaminsky's attack (2008)

- Idea: Use website with many links on *subdomains*:

```
<img src="http://aaaa.target.com/image.jpg"/>
<img src="http://aaab.target.com/image.jpg"/>
<img src="http://aaac.target.com/image.jpg"/>
...
```

Kaminsky's attack (2008)

- Idea: Use website with many links on *subdomains*:

```
<img src="http://aaaa.target.com/image.jpg"/>
<img src="http://aaab.target.com/image.jpg"/>
<img src="http://aaac.target.com/image.jpg"/>
...
```

- Victim will request all of those subdomains, race for each query

Kaminsky's attack (2008)

- Idea: Use website with many links on *subdomains*:

```
<img src="http://aaaa.target.com/image.jpg"/>
<img src="http://aaab.target.com/image.jpg"/>
<img src="http://aaac.target.com/image.jpg"/>
...
```

- Victim will request all of those subdomains, race for each query
- Attacker crafts answer packet for each of those requests:

```
;; ANSWER SECTION:
    aaaa.target.com.     120     IN    A    10.10.10.10

    ;; AUTHORITY SECTION:
    target.com.         86400    IN    NS   ns.target.com.

    ;; ADDITIONAL SECTION:
    www.target.com.     604800   IN    A    66.66.66.66
```

Kaminsky's attack (2008)

- Idea: Use website with many links on *subdomains*:

```
<img src="http://aaaa.target.com/image.jpg"/>
<img src="http://aaab.target.com/image.jpg"/>
<img src="http://aaac.target.com/image.jpg"/>
...
```

- Victim will request all of those subdomains, race for each query
- Attacker crafts answer packet for each of those requests:

```
;; ANSWER SECTION:
    aaaa.target.com.    120     IN    A     10.10.10.10

    ;; AUTHORITY SECTION:
    target.com.         86400   IN    NS    ns.target.com.

    ;; ADDITIONAL SECTION:
    www.target.com.     604800  IN    A     66.66.66.66
```

- The client requested the IP address with `target.com` domain
- The answer for `www.target.com` *passes the bailiwick check*!

Kaminsky's attack (2008)

- Idea: Use website with many links on *subdomains*:

```
<img src="http://aaaa.target.com/image.jpg"/>
<img src="http://aaab.target.com/image.jpg"/>
<img src="http://aaac.target.com/image.jpg"/>
...
```

- Victim will request all of those subdomains, race for each query
- Attacker crafts answer packet for each of those requests:

```
;; ANSWER SECTION:
    aaaa.target.com.    120     IN    A    10.10.10.10

    ;; AUTHORITY SECTION:
    target.com.        86400    IN    NS   ns.target.com.

    ;; ADDITIONAL SECTION:
    www.target.com.    604800   IN    A    66.66.66.66
```

- The client requested the IP address with `target.com` domain
- The answer for `www.target.com` *passes the bailiwick check*!
- The value 604800 defines the validity period of the information: 7 days

Impact of Kaminsky's attack

SECURITYWEEK NETWORK: Information Security News | Infosec Island | Suits and Spooks

SECURITY WEEK

INTERNET AND ENTERPRISE SECURITY NEWS, INSIGHTS & ANA Subscribe (Free) | Security White Papers | IC

Malware & Threats Cybercrime Mobile & Wireless Risk & Compliance Security Architecture Manag

Home › Network Security

The Top Five Worst DNS Security Incidents

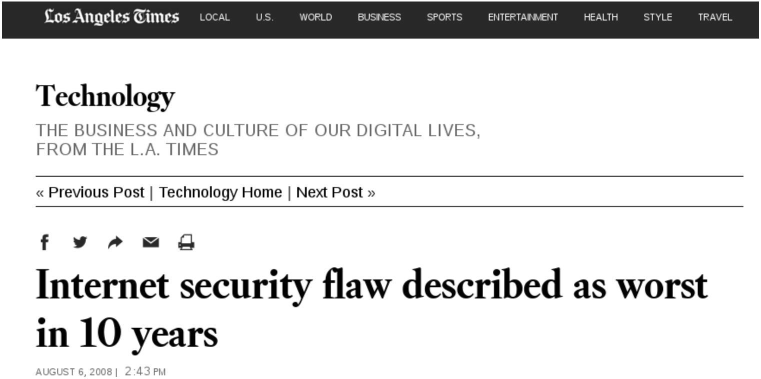
By Ram Mohan on August 11, 2010

Share 2 +1 0 Tweet 22 Recommend RSS

1. "The Kaminsky Bug" puts the whole Internet at risk

Often regarded as possibly the greatest security threat the Internet has ever faced, the so-called "Kaminsky Bug" emerged in July 2008, creating great unease and even greater hype. Researcher Dan Kaminsky discovered that it was easy to exploit a weakness in the DNS and built the software to do it. This weakness would enable malicious hackers to transparently imitate any Web page or e-mail account by poisoning the DNS information cached by Internet service providers.

Impact of Kaminsky's attack



Los Angeles Times LOCAL U.S. WORLD BUSINESS SPORTS ENTERTAINMENT HEALTH STYLE TRAVEL

Technology

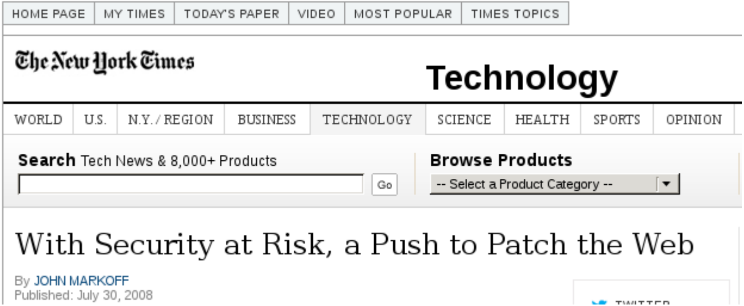
THE BUSINESS AND CULTURE OF OUR DIGITAL LIVES,
FROM THE L.A. TIMES

« Previous Post | Technology Home | Next Post »

Internet security flaw described as worst in 10 years

AUGUST 6, 2008 | 2:43 PM

Impact of Kaminsky's attack



HOME PAGE | MY TIMES | TODAY'S PAPER | VIDEO | MOST POPULAR | TIMES TOPICS

The New York Times

Technology

WORLD | U.S. | N.Y. / REGION | BUSINESS | TECHNOLOGY | SCIENCE | HEALTH | SPORTS | OPINION

Search Tech News & 8,000+ Products

Go

Browse Products

-- Select a Product Category --

With Security at Risk, a Push to Patch the Web

By JOHN MARKOFF
Published: July 30, 2008

http://www.nytimes.com/2008/07/30/technology/30flaw.html?pagewanted=all

Source-port randomization

- Kaminsky's attack hit most big DNS server suites
- `djbdns`, PowerDNS, MaraDNS, and Unbound were not affected

Source-port randomization

- Kaminsky's attack hit most big DNS server suites
- `djbdns`, PowerDNS, MaraDNS, and Unbound were not affected
- Those suites randomized the UDP source port
- Not just 16 bits of entropy to guess for an attacker but 32 bits

Source-port randomization

- Kaminsky's attack hit most big DNS server suites
- `djbdns`, PowerDNS, MaraDNS, and Unbound were not affected
- Those suites randomized the UDP source port
- Not just 16 bits of entropy to guess for an attacker but 32 bits
- Today, all DNS servers randomize the source port

Source-port randomization

- Kaminsky's attack hit most big DNS server suites
- `djbdns`, PowerDNS, MaraDNS, and Unbound were not affected
- Those suites randomized the UDP source port
- Not just 16 bits of entropy to guess for an attacker but 32 bits
- Today, all DNS servers randomize the source port
- Potential problem with NAT: source port is rewritten

Birthday attacks

- Imagine that a DNS server is sending out many *identical requests* (with different source port and QID)
- Attacker spoofs replies with different port+QID combinations
- Any collision with one of the requests wins

Birthday attacks

- Imagine that a DNS server is sending out many *identical requests* (with different source port and QID)
- Attacker spoofs replies with different port+QID combinations
- Any collision with one of the requests wins
- Do servers send out identical requests?
- Some do, e.g., `djbdns`'s `dnscache` (Kevin Day, 2009):
  - Trigger 200 identical queries (default size of query queue)
  - Need to be fast, send these queries before first reply is received
  - Increase attacker's success probability from $1/2^{32}$ to $200/2^{32}$

More randomization?

- The QUESTION section of a DNS request is copied to the reply
- Some bits in the QUESTION section, don't matter:
  `www.ExAMPle.com` is the same as `www.example.com`
- The `0x20` bit changes capitalization of letters

More randomization?

- The QUESTION section of a DNS request is copied to the reply
- Some bits in the QUESTION section, don't matter: `www.ExAMPle.com` is the same as `www.example.com`
- The `0x20` bit changes capitalization of letters
- Idea: Use this bit for extra entropy
- Slight problem: DNS standard does not *require* the QUESTION section to be copied bit-by-bit

More randomization?

- The QUESTION section of a DNS request is copied to the reply
- Some bits in the QUESTION section, don't matter: `www.ExAMPle.com` is the same as `www.example.com`
- The `0x20` bit changes capitalization of letters
- Idea: Use this bit for extra entropy
- Slight problem: DNS standard does not *require* the QUESTION section to be copied bit-by-bit
- Other idea: query repetition (another 32 bits of entropy)
- Adds additional complications (not broadly implemented)
- Bernstein on randomization:

  "It is clear that enough randomization effort would be able to stop all blind forgeries."

The easy way. . .

- A passive MitM can read DNS requests
- Becoming a passive MitM is not that hard:
  - Sniff WiFi
  - ARP spoofing
  - Be an ISP
  - Be a network administrator in a company
- A DNS attacker can poison a DNS cache
- Affects many more clients than a MitM between clients!

DNS censorship

- DNS can be used for censorship:
  - April 1997: German provider DFN blocks IPs of xs4all.nl
  - German "Zugangserschwerungsgesetz"
  - "Child Sexual Abuse Anti Distribution Filter" (CSAADF) by CIRCAMP used in Denmark, Finland, Italy, Newzealand, Norway, Sweden und der Switzerland
  - Idea in all these cases: "redirect" (spoof) DNS

DNS censorship

- DNS can be used for censorship:
  - April 1997: German provider DFN blocks IPs of xs4all.nl
  - German "Zugangser-schwerungsgesetz"
  - "Child Sexual Abuse Anti Distribution Filter" (CSAADF) by CIRCAMP used in Denmark, Finland, Italy, Newzealand, Norway, Sweden und der Switzerland
  - Idea in all these cases: "redirect" (spoof) DNS
  - Circumvention: Use alternative DNS

Source: http://xkcd.com/1361/

DNSSEC

- Idea: Use cryptographically signed DNS entries
- Initial design decision: sign information offline:
  - No need for expensive public-key crypto for each reply
  - No need to hold the private keys on DNS servers

DNSSEC

- Idea: Use cryptographically signed DNS entries
- Initial design decision: sign information offline:
  - No need for expensive public-key crypto for each reply
  - No need to hold the private keys on DNS servers
- Public keys are authenticated through a chain of trust
- Root of trust: public keys of the DNS root servers

DNSSEC

- Idea: Use cryptographically signed DNS entries
- Initial design decision: sign information offline:
  - No need for expensive public-key crypto for each reply
  - No need to hold the private keys on DNS servers
- Public keys are authenticated through a chain of trust
- Root of trust: public keys of the DNS root servers
- Additional (cryptographic) information in new DNS entry types:
  - RRSIG: DNSSEC signature
  - DNSKEY: public key to verify signature

More amplification!

- DNSSEC does not increase the size of DNS requests
- DNSSEC does significantly increase the size of DNS replies
- Modern DDOS uses DNS+DNSSEC

More amplification!

- DNSSEC does not increase the size of DNS requests
- DNSSEC does significantly increase the size of DNS replies
- Modern DDOS uses DNS+DNSSEC
- RFC 4033: "DNSSEC provides no protection against denial of service attacks. Security-aware resolvers and security-aware name servers are vulnerable to an additional class of denial of service attacks based on cryptographic operations."

DNS zone enumeration

- You want DNS to answer a request for domain names
- You *do not* want to hand out a list of all domain names
- Finding all hosts in a DNS zone is called *zone enumeration*

DNS zone enumeration

- You want DNS to answer a request for domain names
- You *do not* want to hand out a list of all domain names
- Finding all hosts in a DNS zone is called *zone enumeration*
- Problem for DNSSEC: offline-signed answer for *non-existing* entries (negative answer)
- First solution: Don't sign (bad idea, can use for attack)

DNS zone enumeration

- You want DNS to answer a request for domain names
- You *do not* want to hand out a list of all domain names
- Finding all hosts in a DNS zone is called *zone enumeration*
- Problem for DNSSEC: offline-signed answer for *non-existing* entries (negative answer)
- First solution: Don't sign (bad idea, can use for attack)
- Second idea: Sign "There is no name between `smtp.example.com` and `www.example.com`"

DNS zone enumeration

- You want DNS to answer a request for domain names
- You *do not* want to hand out a list of all domain names
- Finding all hosts in a DNS zone is called *zone enumeration*
- Problem for DNSSEC: offline-signed answer for *non-existing* entries (negative answer)
- First solution: Don't sign (bad idea, can use for attack)
- Second idea: Sign "There is no name between `smtp.example.com` and `www.example.com`"
- This trivially allows zone enumeration:
  - Try some hostname, this will give you 1 or 2 valid hostnames
  - Try just before (alphabetically) a valid hostname: find previous
  - Try just after (alphabetically) a valid hostname: find next

DNS zone enumeration

- You want DNS to answer a request for domain names
- You *do not* want to hand out a list of all domain names
- Finding all hosts in a DNS zone is called *zone enumeration*
- Problem for DNSSEC: offline-signed answer for *non-existing* entries (negative answer)
- First solution: Don't sign (bad idea, can use for attack)
- Second idea: Sign "There is no name between `smtp.example.com` and `www.example.com`"
- This trivially allows zone enumeration:
  - Try some hostname, this will give you 1 or 2 valid hostnames
  - Try just before (alphabetically) a valid hostname: find previous
  - Try just after (alphabetically) a valid hostname: find next
- RFC 4033: "DNSSEC introduces the ability for a hostile party to enumerate all the names in a zone by following the NSEC chain."

NSEC3

- Idea: Hash domain names, sign information on gaps between existing *hashes*
- Example:
  - request for (non-existing) `test.example.com`
  - Hash `test.example.com` (with SHA-1), obtain: `401f83bc96721eeeba6f5c1c54cf0a83dc08a30b`
  - Signed answer: "There is no name with hash between `068503358dddd23cf6cf00f5d6ad9a45cd0a8e03` and `512e9305c87f4f1ccdbacb80c559f3dce496ae29`.

NSEC3

- Idea: Hash domain names, sign information on gaps between existing *hashes*
- Example:
  - request for (non-existing) `test.example.com`
  - Hash `test.example.com` (with SHA-1), obtain: `401f83bc96721eeeba6f5c1c54cf0a83dc08a30b`
  - Signed answer: "There is no name with hash between `068503358dddd23cf6cf00f5d6ad9a45cd0a8e03` and `512e9305c87f4f1ccdbacb80c559f3dce496ae29`.
- Problem: Can revert these hashes

NSEC3

- Idea: Hash domain names, sign information on gaps between existing *hashes*
- Example:
  - request for (non-existing) `test.example.com`
  - Hash `test.example.com` (with SHA-1), obtain: `401f83bc96721eeeba6f5c1c54cf0a83dc08a30b`
  - Signed answer: "There is no name with hash between `068503358dddd23cf6cf00f5d6ad9a45cd0a8e03` and `512e9305c87f4f1ccdbacb80c559f3dce496ae29`.
- Problem: Can revert these hashes
- Wait, shouldn't it be hard to compute preimages of hashes?

NSEC3

- Idea: Hash domain names, sign information on gaps between existing *hashes*
- Example:
  - request for (non-existing) `test.example.com`
  - Hash `test.example.com` (with SHA-1), obtain: `401f83bc96721eeeba6f5c1c54cf0a83dc08a30b`
  - Signed answer: "There is no name with hash between `068503358dddd23cf6cf00f5d6ad9a45cd0a8e03` and `512e9305c87f4f1ccdbacb80c559f3dce496ae29`.
- Problem: Can revert these hashes
- Wait, shouldn't it be hard to compute preimages of hashes?
- Well, domain names are not that hard to guess, can just try valid domain names, e.g.

```
www.example.com    068503358dddd23cf6cf00f5d6ad9a45cd0a8e03
smtp.example.com   512e9305c87f4f1ccdbacb80c559f3dce496ae29
```

NSEC3

- Idea: Hash domain names, sign information on gaps between existing *hashes*
- Example:
  - request for (non-existing) `test.example.com`
  - Hash `test.example.com` (with SHA-1), obtain: `401f83bc96721eeeba6f5c1c54cf0a83dc08a30b`
  - Signed answer: "There is no name with hash between `068503358dddd23cf6cf00f5d6ad9a45cd0a8e03` and `512e9305c87f4f1ccdbacb80c559f3dce496ae29`.
- Problem: Can revert these hashes
- Wait, shouldn't it be hard to compute preimages of hashes?
- Well, domain names are not that hard to guess, can just try valid domain names, e.g.

```
www.example.com    068503358dddd23cf6cf00f5d6ad9a45cd0a8e03
smtp.example.com   512e9305c87f4f1ccdbacb80c559f3dce496ae29
```

- Software by Niederhagen: Try 6000 billion hashes per week on NVIDIA GTX295 GPU
- This is *much* faster than trying domain names through DNS queries

More DNSSEC problems

- Second implication of offline-signed records: *replay attacks*
- Attack scenario:
  - Company runs server `www.example.com` at 123.45.67.89
  - DNS server sends signed name resolution for this name/IP, attacker records it
  - Company moves or changes provider, now `www.example.com` is at 98.76.54.32
  - Attacker replays name resolution to 123.45.67.89

More DNSSEC problems

- Second implication of offline-signed records: *replay attacks*
- Attack scenario:
  - Company runs server `www.example.com` at 123.45.67.89
  - DNS server sends signed name resolution for this name/IP, attacker records it
  - Company moves or changes provider, now `www.example.com` is at 98.76.54.32
  - Attacker replays name resolution to 123.45.67.89
- DNSSEC uses bleeding-edge crypto (1024-bit RSA)

More DNSSEC problems

- Second implication of offline-signed records: *replay attacks*
- Attack scenario:
  - Company runs server `www.example.com` at 123.45.67.89
  - DNS server sends signed name resolution for this name/IP, attacker records it
  - Company moves or changes provider, now `www.example.com` is at 98.76.54.32
  - Attacker replays name resolution to 123.45.67.89
- DNSSEC uses bleeding-edge crypto (1024-bit RSA)
- DNSSEC does not encrypt queries; from RFC 4033:
  "Due to a deliberate design choice, DNSSEC does not provide confidentiality"

DNSCurve

- Alternative to DNSSEC proposed by Bernstein: DNSCurve
- Idea is to encrypt and authenticate DNS traffic (not sign records)
- The idea is a bit similar to SSL/TLS (next lecture)

DNSCurve

- Alternative to DNSSEC proposed by Bernstein: DNSCurve
- Idea is to encrypt and authenticate DNS traffic (not sign records)
- The idea is a bit similar to SSL/TLS (next lecture)
- DNSCurve does not have the problems that come with offline signing:
  - No zone enumeration
  - No replay attacks

DNSCurve

- Alternative to DNSSEC proposed by Bernstein: DNSCurve
- Idea is to encrypt and authenticate DNS traffic (not sign records)
- The idea is a bit similar to SSL/TLS (next lecture)
- DNSCurve does not have the problems that come with offline signing:
  - No zone enumeration
  - No replay attacks
- It also has other advantages over DNSSEC:
  - Much stronger (and faster) crypto
  - Much more limited amplification issues (replies grow, but so do requests)
  - Confidentiality of DNS requests (encryption)

DNSCurve

- Alternative to DNSSEC proposed by Bernstein: DNSCurve
- Idea is to encrypt and authenticate DNS traffic (not sign records)
- The idea is a bit similar to SSL/TLS (next lecture)
- DNSCurve does not have the problems that come with offline signing:
  - No zone enumeration
  - No replay attacks
- It also has other advantages over DNSSEC:
  - Much stronger (and faster) crypto
  - Much more limited amplification issues (replies grow, but so do requests)
  - Confidentiality of DNS requests (encryption)
- Potential disadvantage of DNSCurve: crypto keys need to be on DNS servers

DNSCurve

- Alternative to DNSSEC proposed by Bernstein: DNSCurve
- Idea is to encrypt and authenticate DNS traffic (not sign records)
- The idea is a bit similar to SSL/TLS (next lecture)
- DNSCurve does not have the problems that come with offline signing:
  - No zone enumeration
  - No replay attacks
- It also has other advantages over DNSSEC:
  - Much stronger (and faster) crypto
  - Much more limited amplification issues (replies grow, but so do requests)
  - Confidentiality of DNS requests (encryption)
- Potential disadvantage of DNSCurve: crypto keys need to be on DNS servers
- Addional disadvantage: It's much easier to deploy than DNSSEC, does not create as many jobs for consultants

More reading...

- Dan Bernstein about DNSCurve (and DNSSEC vulnerabilities):
  - http://dnscurve.org/
  - http://cr.yp.to/talks/2010.12.28/slides.pdf
  - Updated: http://cr.yp.to/talks/2016.12.08/slides-djb-20161208-dnssec-a4.pdf
- Dan Kaminsky's answer:
  http://dankaminsky.com/2011/01/05/djb-ccc/

  "DNSSEC Is Not Necessarily An Offline Signer – In Fact, It Works Better Online!"
- Dan Bernstein's answer:
  http://marc.info/?l=djbdns&m=129434351607605&w=2