DNSSEC and Unbound

Outline

- DNS Threads
- what is DNSSEC
- Validating DNSSEC in the Internet
- The Unbound validating resolver
- installing Unbound
- Unbound configuration
- testing DNSSEC

DNS Cache Spoofing
Episode I

the Kaspureff attacks
12. July 1997

MEN&MICE

The Kashpureff Attack

- In July, 1997, Eugene Kashpureff used a direct triggered cache poisoning attack against the InterNIC's web site

evil resolver

ISP resolving DNS Server

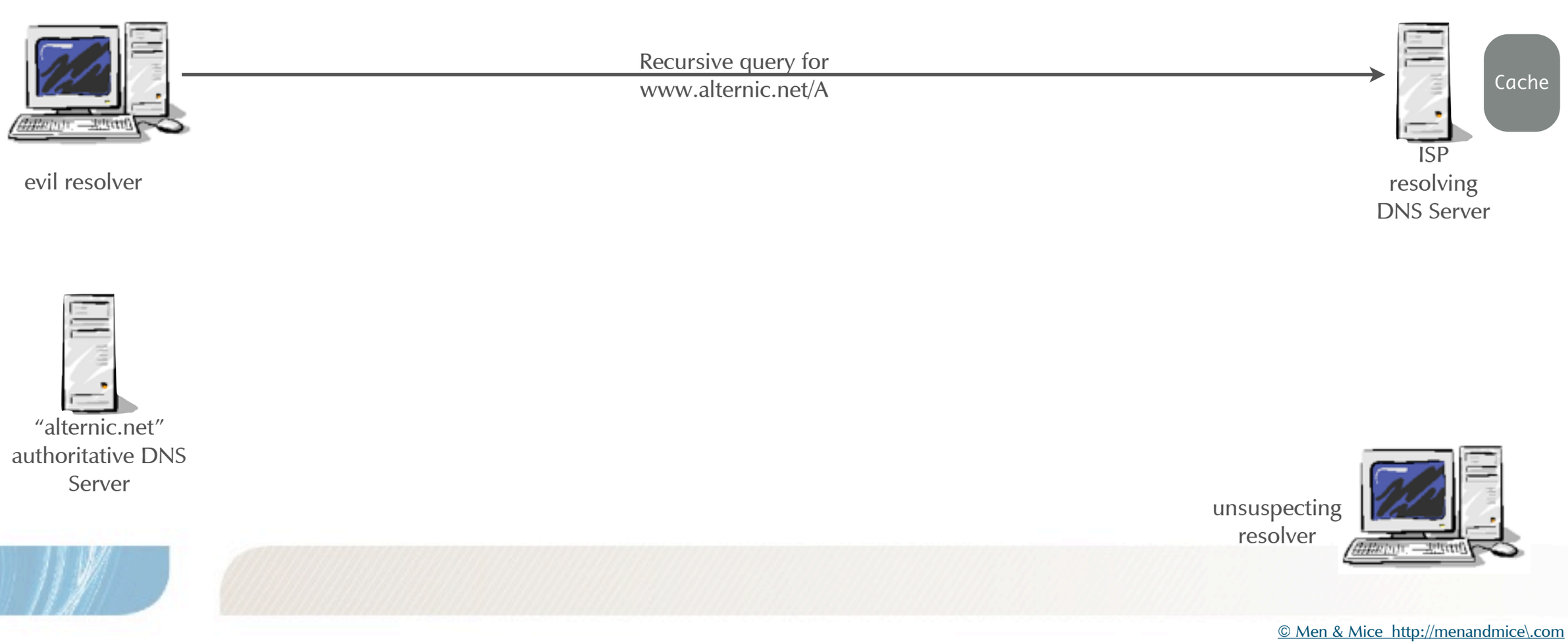
Cache

"alternic.net" authoritative DNS Server

unsuspecting resolver

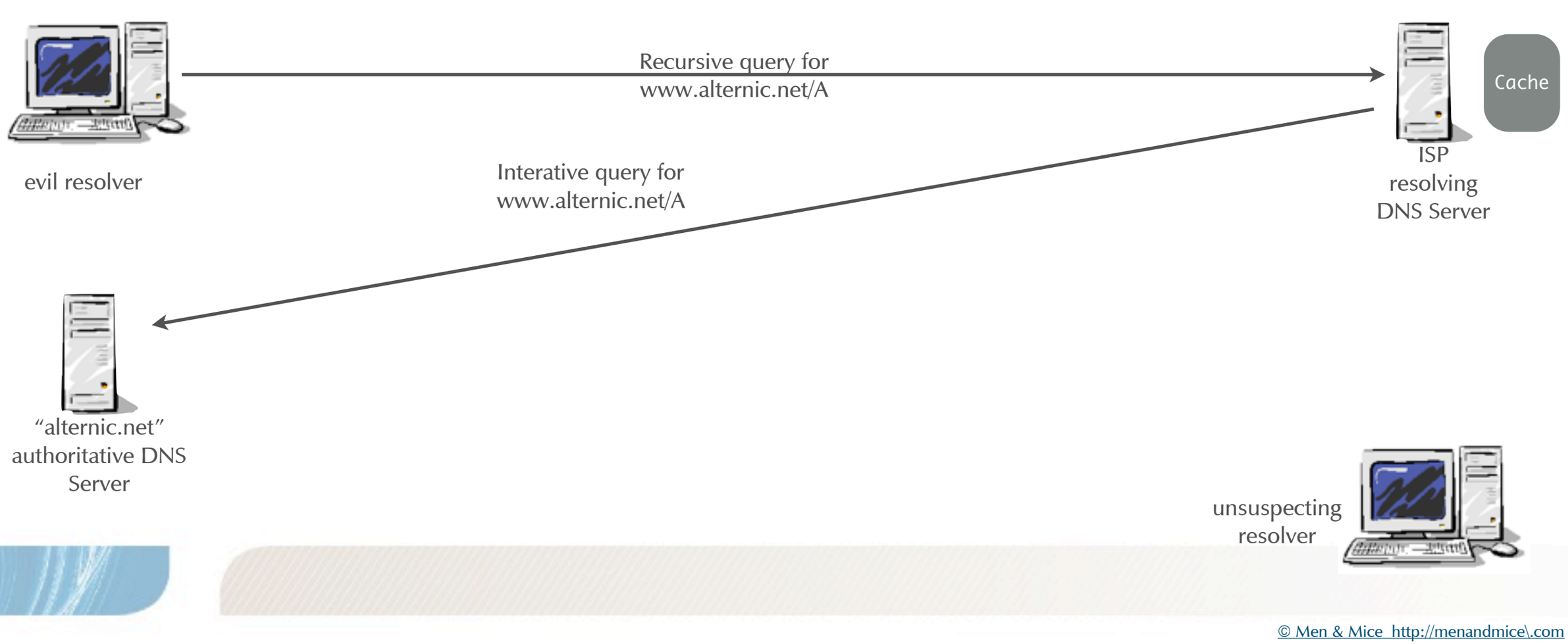
The Kashpureff Attack

- In July, 1997, Eugene Kashpureff used a direct triggered cache poisoning attack against the InterNIC's web site



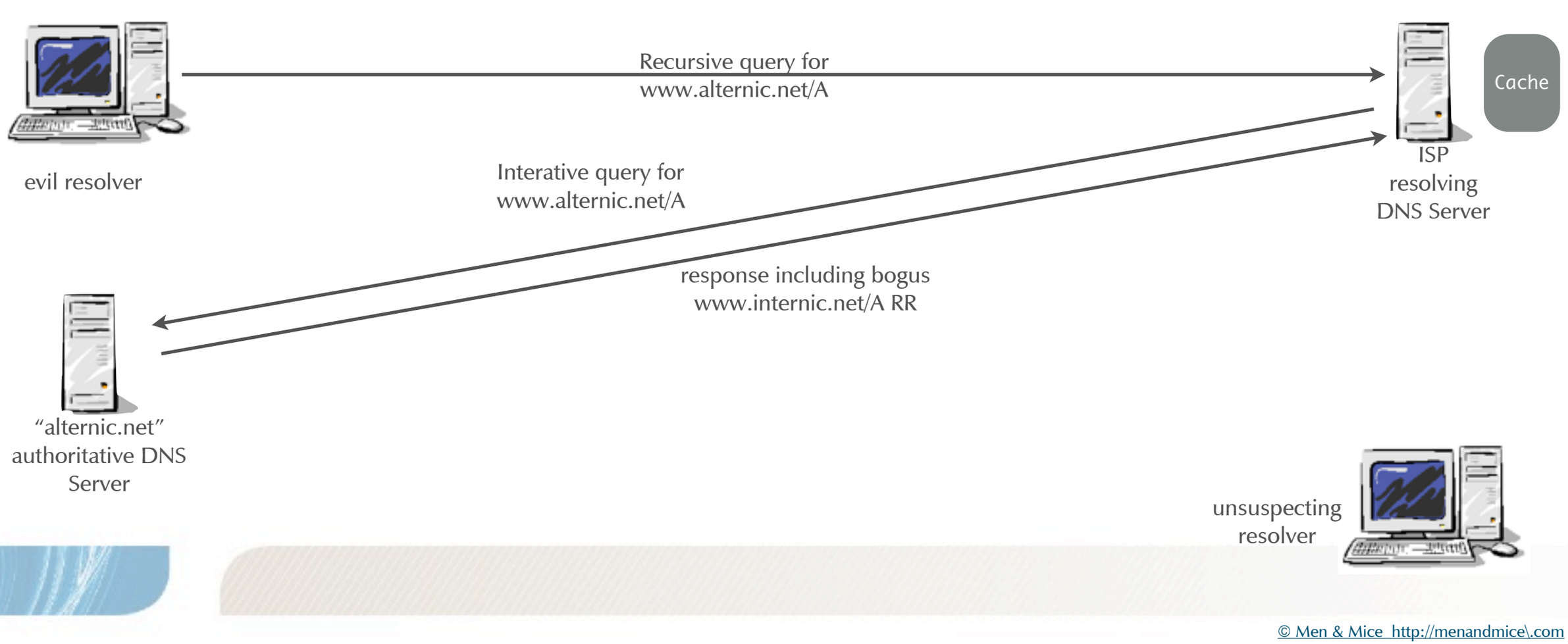
The Kashpureff Attack

- In July, 1997, Eugene Kashpureff used a direct triggered cache poisoning attack against the InterNIC's web site



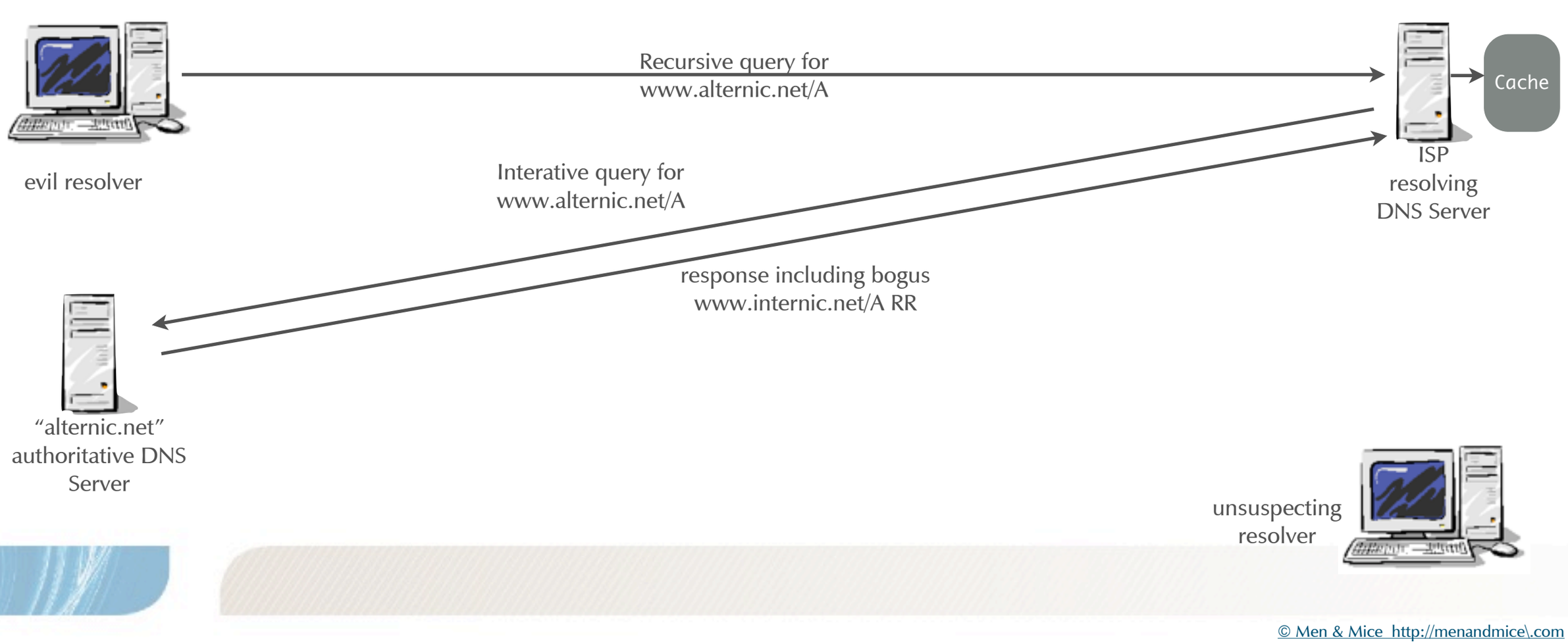
The Kashpureff Attack

- In July, 1997, Eugene Kashpureff used a direct triggered cache poisoning attack against the InterNIC's web site



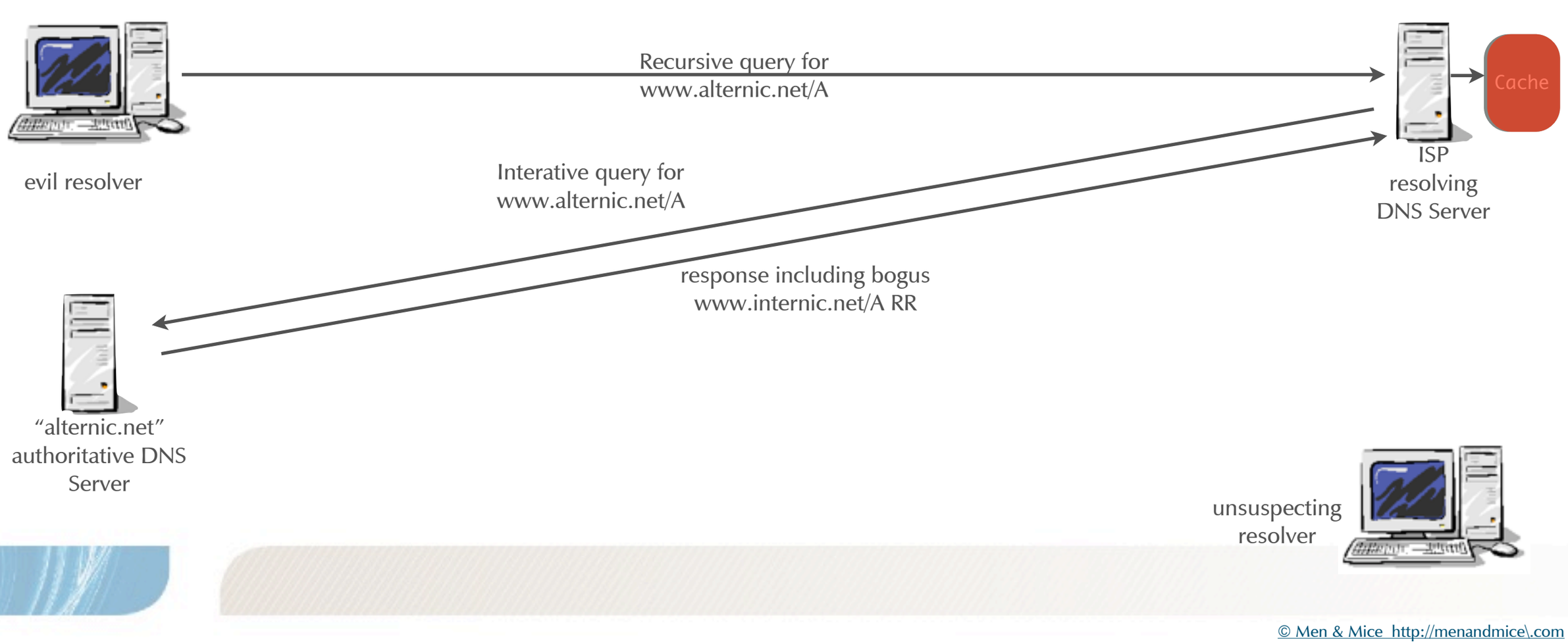
The Kashpureff Attack

- In July, 1997, Eugene Kashpureff used a direct triggered cache poisoning attack against the InterNIC's web site



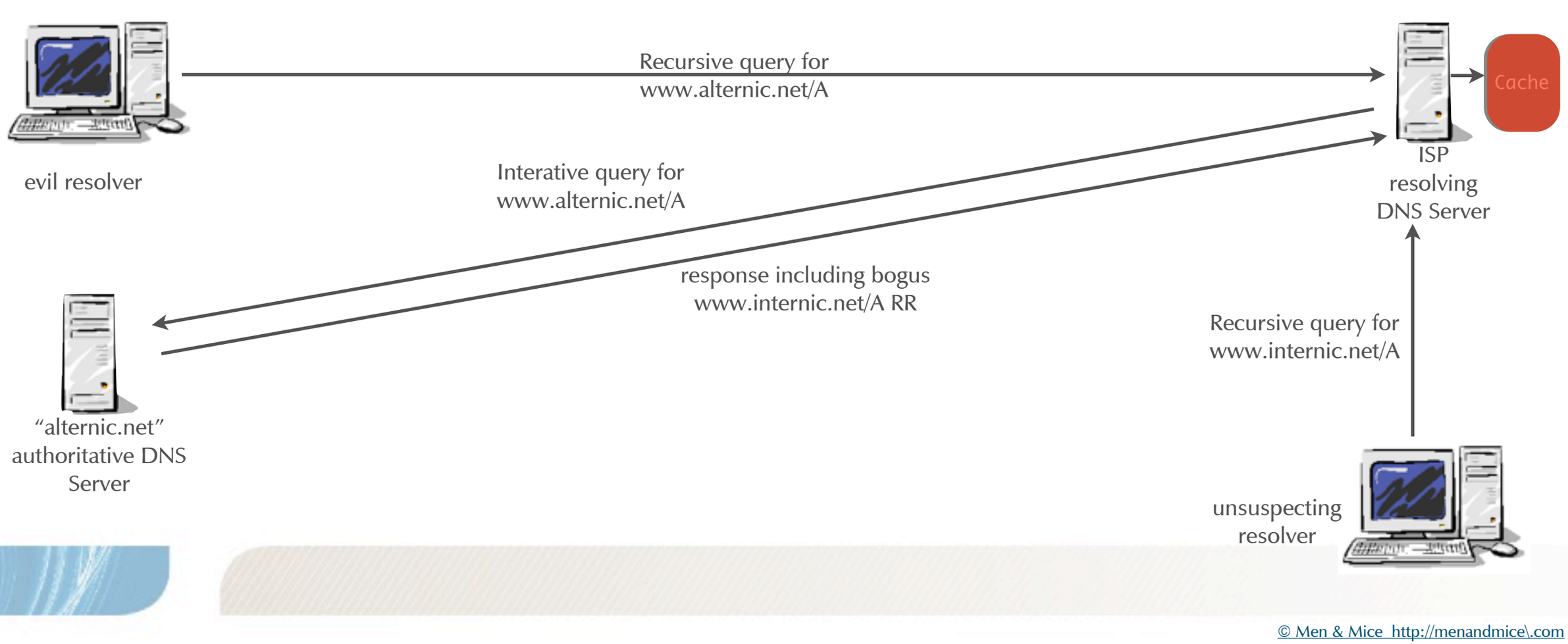
The Kashpureff Attack

- In July, 1997, Eugene Kashpureff used a direct triggered cache poisoning attack against the InterNIC's web site



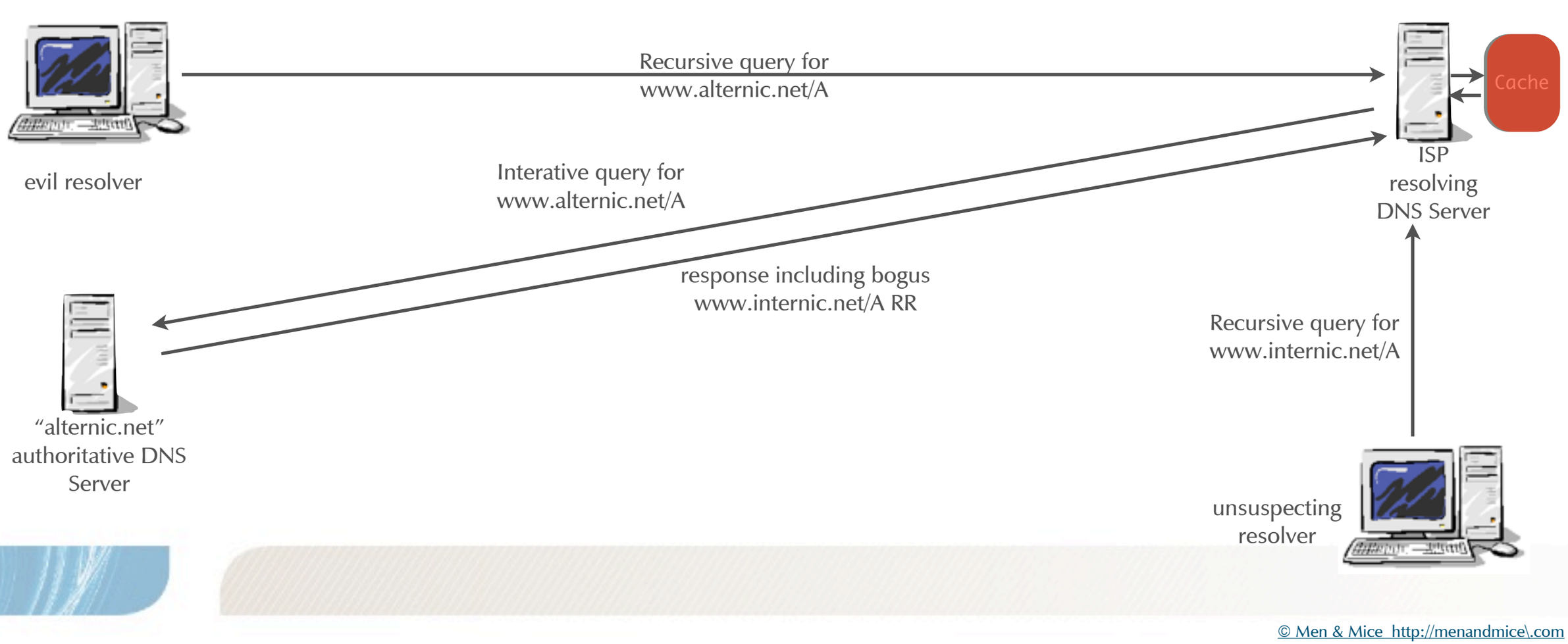
The Kashpureff Attack

- In July, 1997, Eugene Kashpureff used a direct triggered cache poisoning attack against the InterNIC's web site



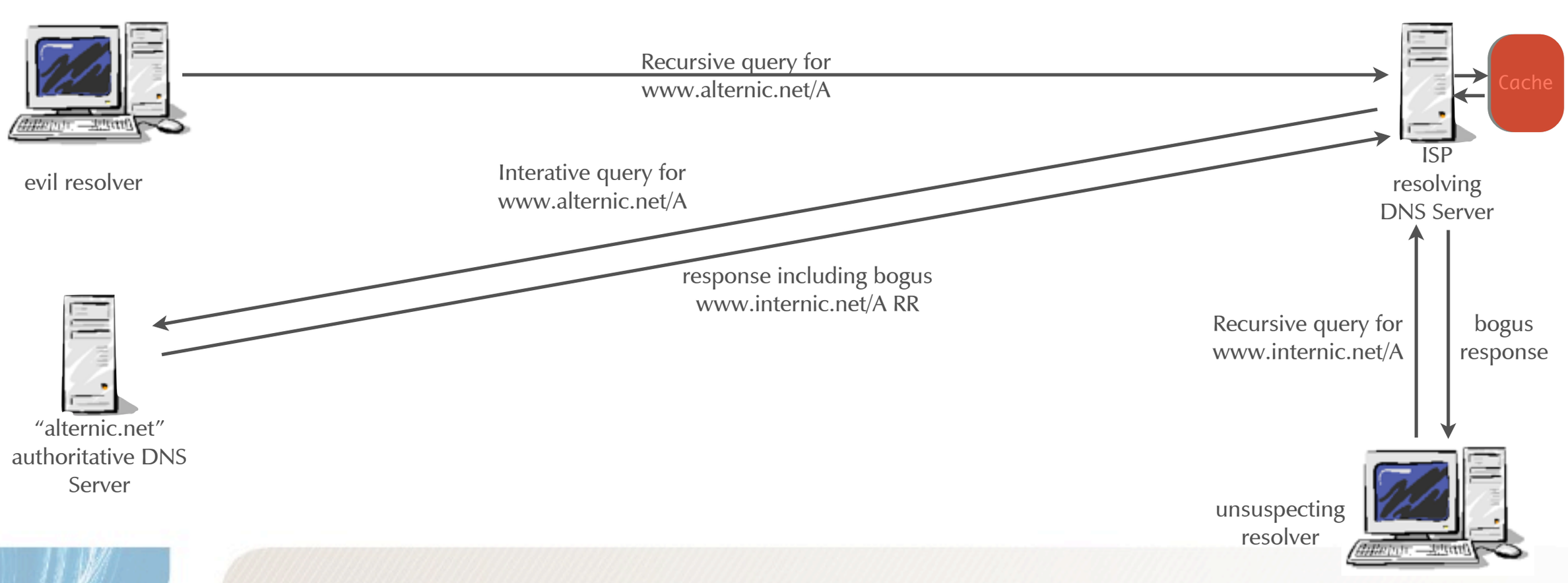
The Kashpureff Attack

- In July, 1997, Eugene Kashpureff used a direct triggered cache poisoning attack against the InterNIC's web site



The Kashpureff Attack

- In July, 1997, Eugene Kashpureff used a direct triggered cache poisoning attack against the InterNIC's web site



DNS 'bailiwick' checking

- The problem:
  - The Kashpureff attack has been possible because DNS Servers were accepting arbitrary information from the additional section of the DNS answer

DNS 'bailiwick' checking

- The fix
  - The credibility checking when replacing cache entries
  - Check for "in bailiwick" in response data. Answer records must be from the same domain as the requested name.

```
$ dig @ns1.example.com www.example.com
;; ANSWER SECTION:
www.example.com.    120      IN    A    192.0.2.10

;; AUTHORITY SECTION:
example.com.  86400    IN    NS   ns1.example.com.
example.com.  86400    IN    NS   ns2.example.com.

;; ADDITIONAL SECTION:
ns1.example.com.    604800   IN    A    192.0.2.120
ns2.example.com.    604800   IN    A    192.0.2.130
www.mybank.com.     604800   IN    A    1.2.3.4
```

DNS 'bailiwick' checking

- The fix
  - The credibility checking when replacing cache entries
  - Check for "in bailiwick" in response data. Answer records must be from the same domain as the requested name.

```
$ dig @ns1.example.com www.example.com
;; ANSWER SECTION:
www.example.com.    120      IN    A    192.0.2.10

;; AUTHORITY SECTION:
example.com.  86400    IN    NS   ns1.example.com.
example.com.  86400    IN    NS   ns2.example.com.

;; ADDITIONAL SECTION:
ns1.example.com.    604800   IN    A    192.0.2.120
ns2.example.com.    604800   IN    A    192.0.2.130
www.mybank.com.     604800   IN    A    1.2.3.4
```

DNS 'bailiwick' checking

- The fix
  - The credibility checking when replacing cache entries
  - Check for "in bailiwick" in response data. Answer records must be from the same domain as the requested name.

```
$ dig @ns1.example.com www.example.com
;; ANSWER SECTION:
www.example.com.    120      IN    A    192.0.2.10

;; AUTHORITY SECTION:
example.com.  86400    IN    NS   ns1.example.com.
example.com.  86400    IN    NS   ns2.example.com.

;; ADDITIONAL SECTION:
ns1.example.com.    604800   IN    A    192.0.2.120
ns2.example.com.    604800   IN    A    192.0.2.130
www.mybank.com.     604800   IN    A    1.2.3.4
```

DNS 'bailiwick' checking

- The fix
  - The credibility checking when replacing cache entries
  - Check for "in bailiwick" in response data. Answer records must be from the same domain as the requested name.

Data not in 'bailiwick' will not be accepted

```
$ dig @ns1.example.com www.example.com
;; ANSWER SECTION:
www.example.com.    120      IN    A    192.0.2.10

;; AUTHORITY SECTION:
example.com.  86400    IN    NS   ns1.example.com.
example.com.  86400    IN    NS   ns2.example.com.

;; ADDITIONAL SECTION:
ns1.example.com.    604800   IN    A    192.0.2.120
ns2.example.com.    604800   IN    A    192.0.2.130
www.mybank.com.     604800   IN    A    1.2.3.4
```

DNS 'bailiwick' checking

- The fix
  - The credibility checking when replacing cache entries
  - Check for "in bailiwick" in response data. Answer records must be from the same domain as the requested name.

Data not in 'bailiwick' will not be accepted

```
$ dig @ns1.example.com www.example.com
;; ANSWER SECTION:
www.example.com.    120      IN    A    192.0.2.10

;; AUTHORITY SECTION:
example.com.  86400    IN    NS   ns1.example.com.
example.com.  86400    IN    NS   ns2.example.com.

;; ADDITIONAL SECTION:
ns1.example.com.    604800   IN    A    192.0.2.120
ns2.example.com.    604800   IN    A    192.0.2.130
www.mybank.com.     604800   IN    A    1.2.3.4
```

DNS Cache Spoofing
Episode II

the Amit Klein findings
March-June 2007

Message ID Guessing

- The DNS message's message ID field is only 16 bits long
  - And the "randomizer" of some name-servers is not truly random
    - It's worse in BIND before 8.2
    - Though it's better in BIND 8.2 and later versions with use-id-pool set and in versions of BIND 9
    - It is only real random in BIND version from end of 2007 on

Message ID Guessing

- Any name server that receives a query from another name server knows

- The source port it's using for queries
- The message ID it used at some point in time
- One query it's currently working on (Query Domainname and Query Record Type)

The Amit Klein findings (1)

- In 2007 Amit Klein found that the randomizers used in most DNS Servers are not truly random: The next message ID's could be pre-calculated

evil resolver

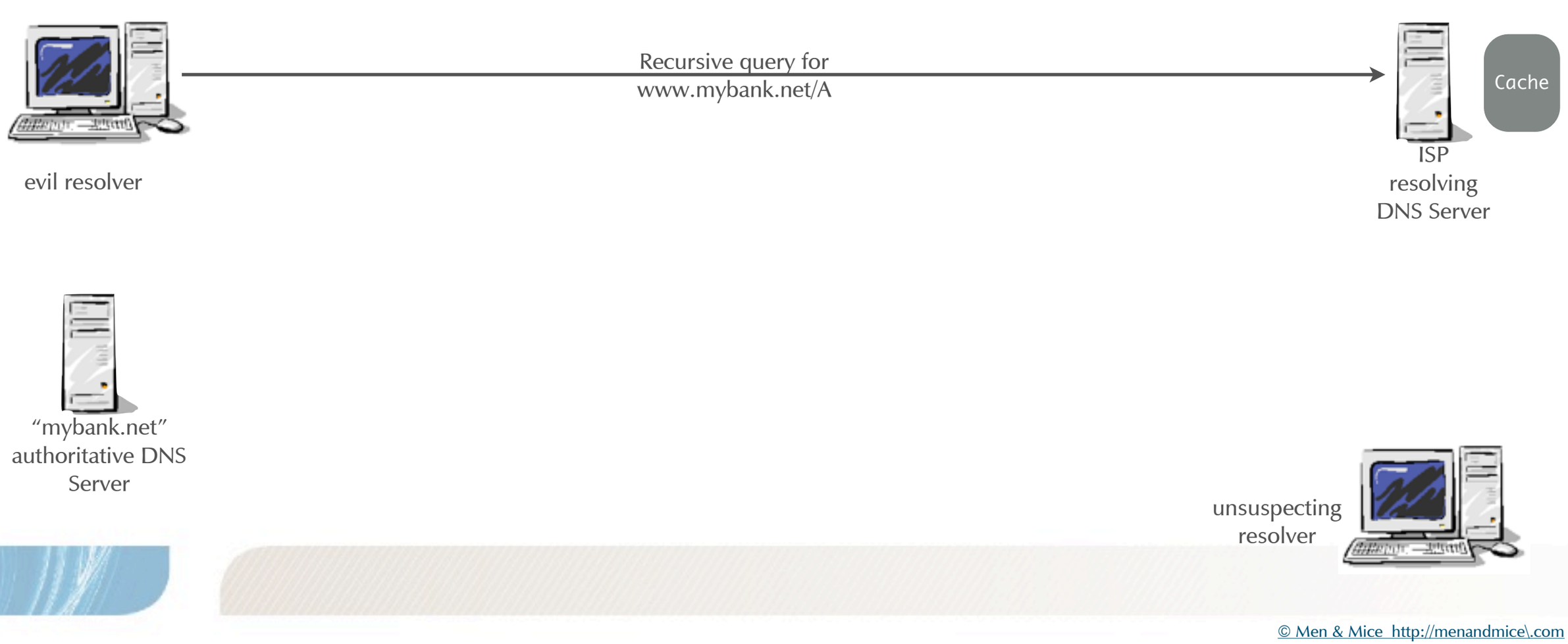
ISP
resolving
DNS Server

"mybank.net"
authoritative DNS
Server

unsuspecting
resolver

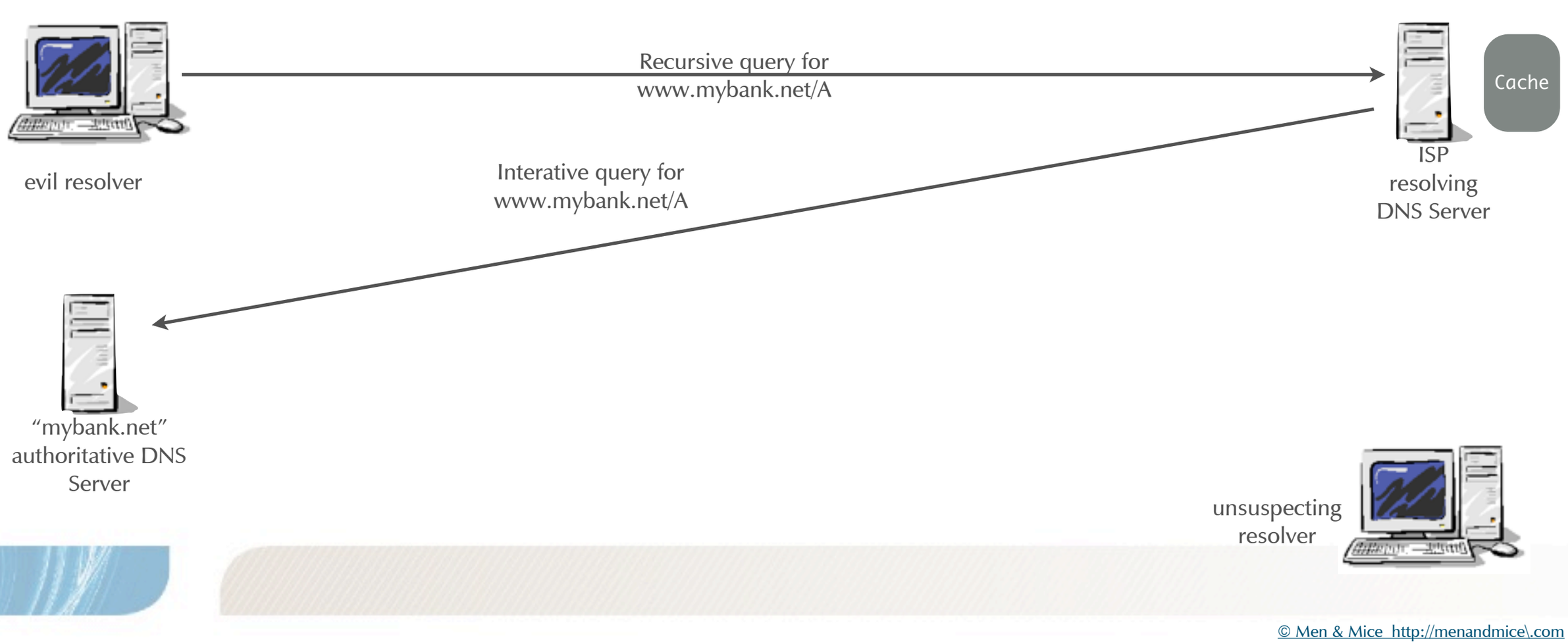
The Amit Klein findings (1)

- In 2007 Amit Klein found that the randomizers used in most DNS Servers are not truly random: The next message ID's could be pre-calculated



The Amit Klein findings (1)

- In 2007 Amit Klein found that the randomizers used in most DNS Servers are not truly random: The next message ID's could be pre-calculated



The Amit Klein findings (2)

- In 2007 Amit Klein found that the randomizers used in most DNS Servers are not truly random: The next message ID's could be pre-calculated

evil resolver

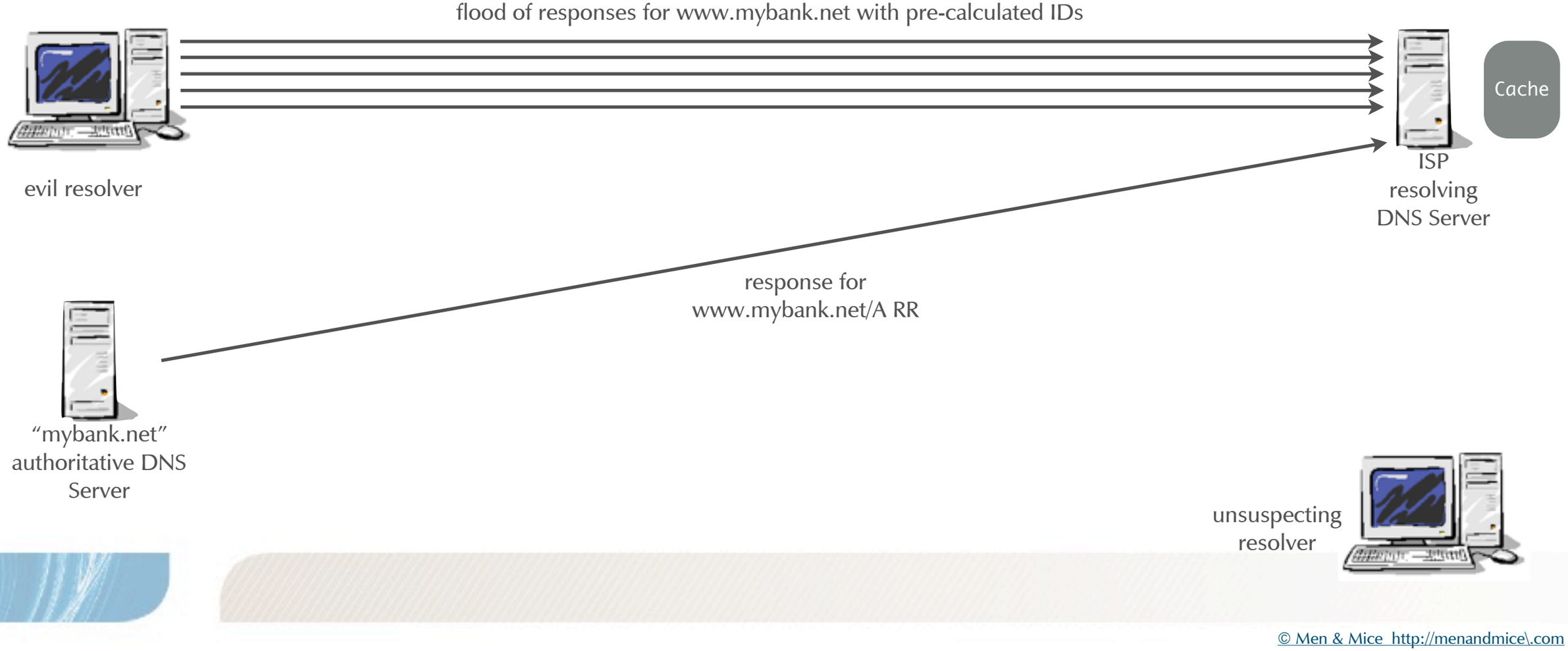
ISP
resolving
DNS Server

"mybank.net"
authoritative DNS
Server

unsuspecting
resolver

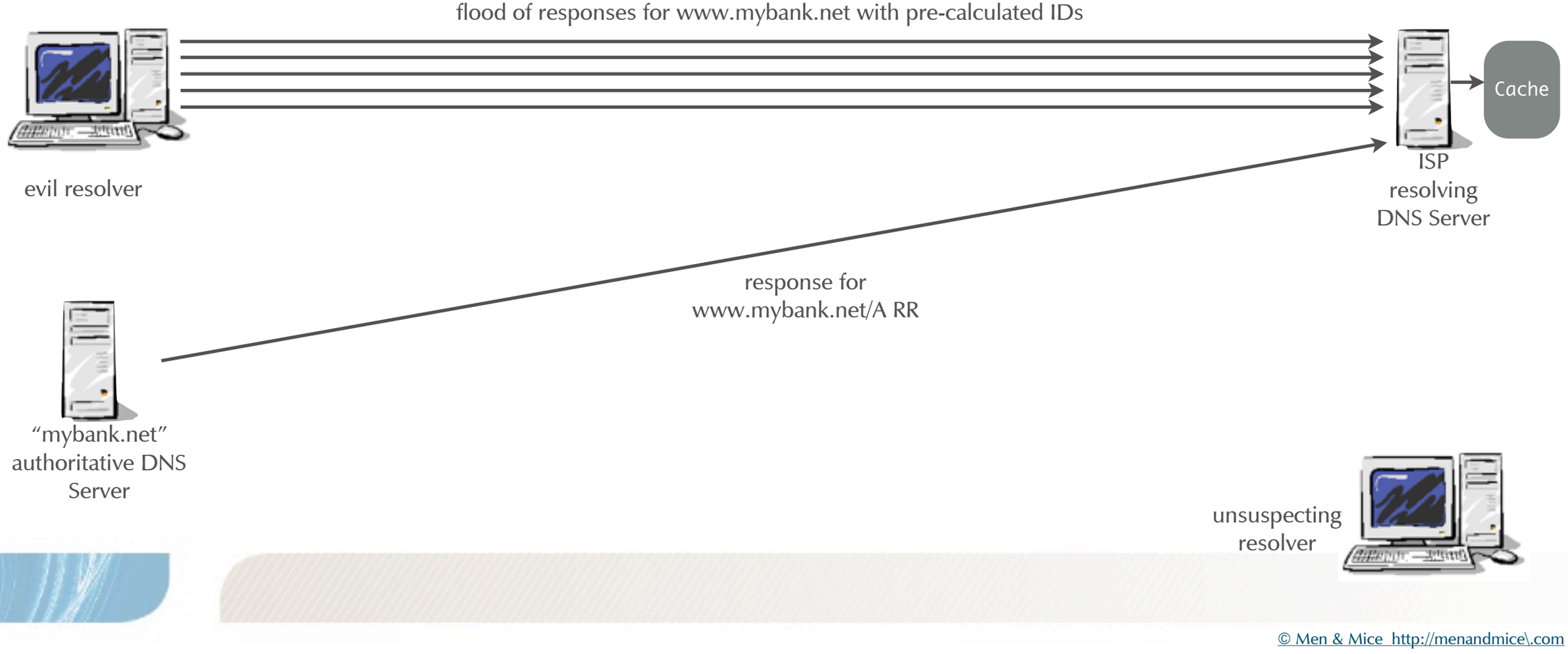
The Amit Klein findings (2)

- In 2007 Amit Klein found that the randomizers used in most DNS Servers are not truly random: The next message ID's could be pre-calculated



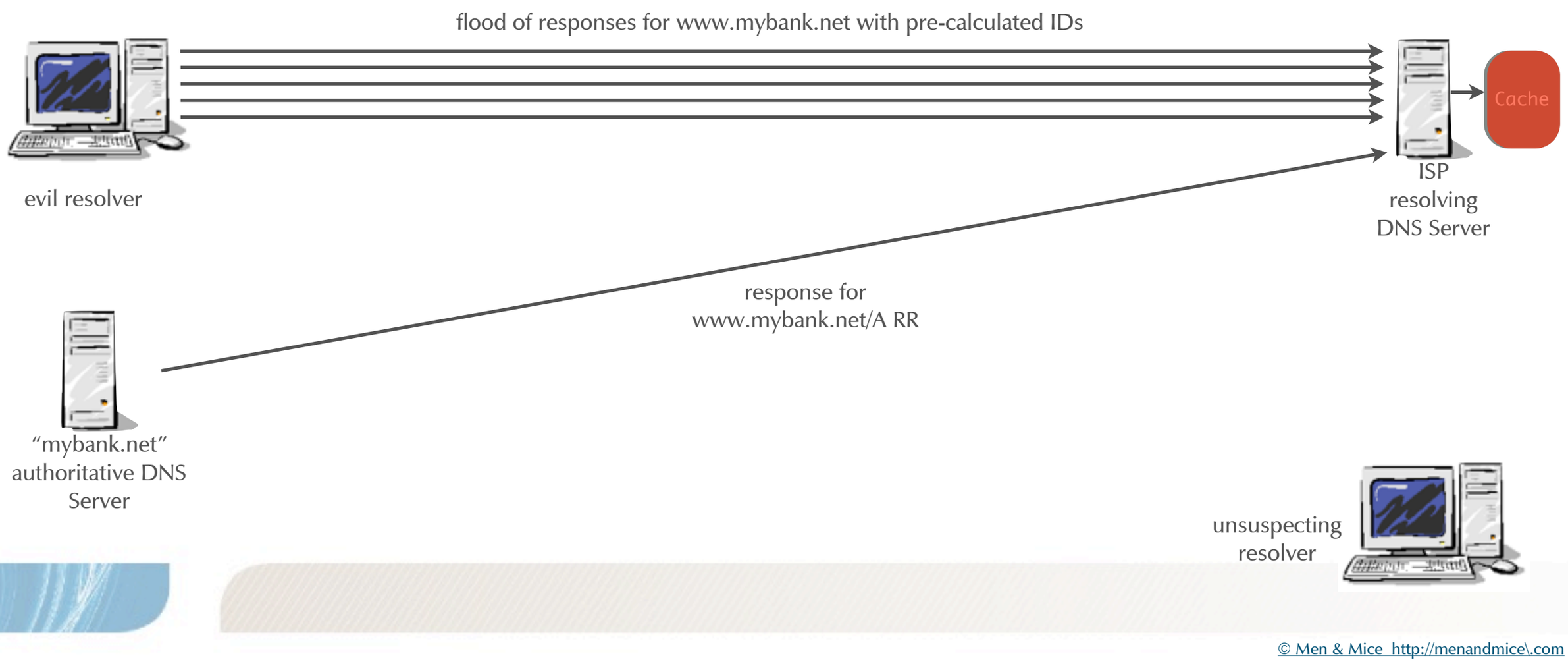
The Amit Klein findings (2)

- In 2007 Amit Klein found that the randomizers used in most DNS Servers are not truly random: The next message ID's could be pre-calculated



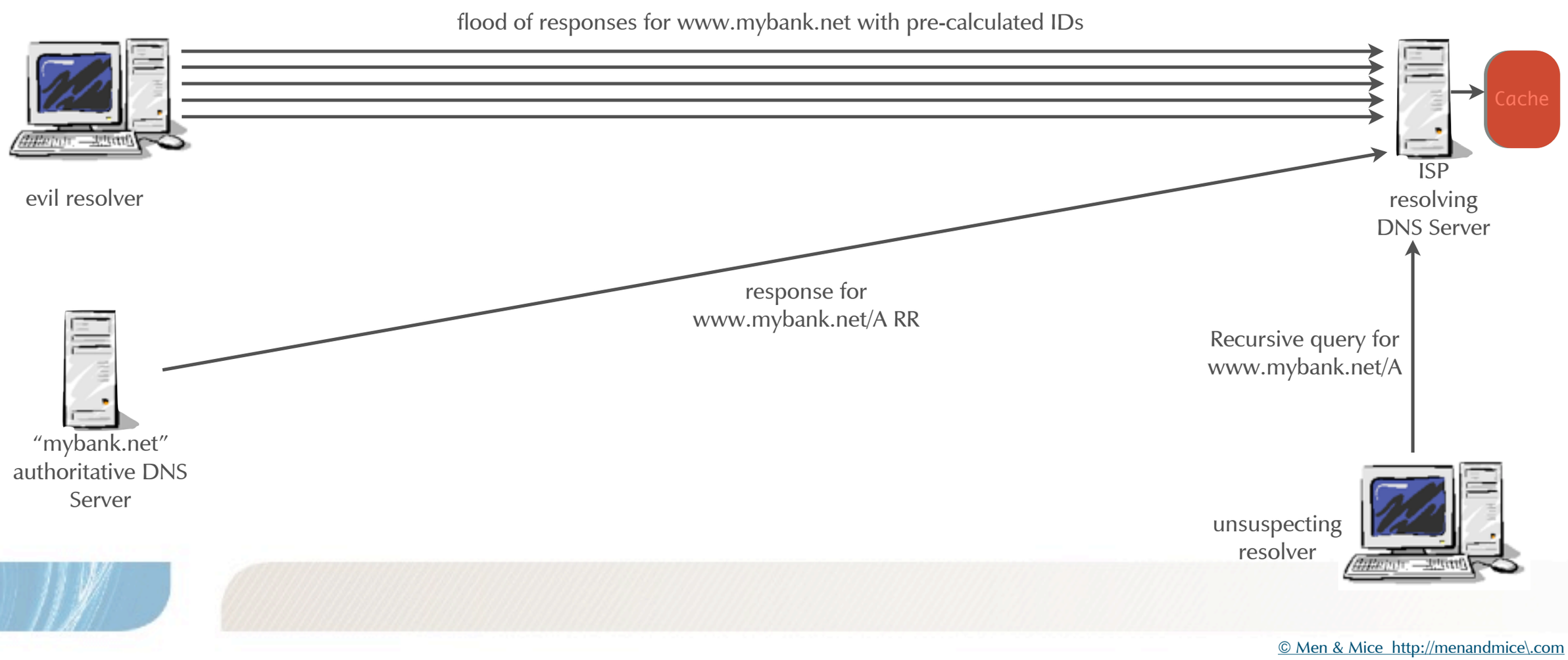
The Amit Klein findings (2)

- In 2007 Amit Klein found that the randomizers used in most DNS Servers are not truly random: The next message ID's could be pre-calculated



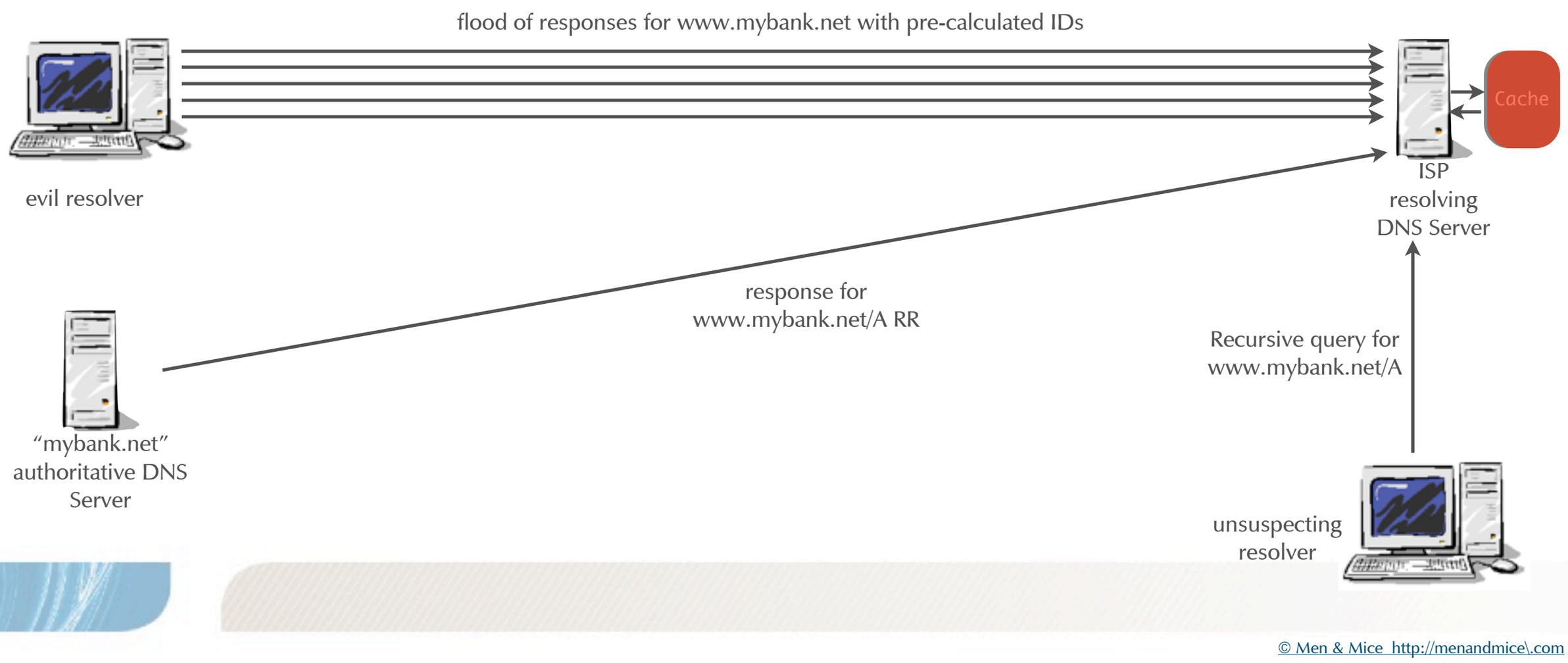
The Amit Klein findings (2)

- In 2007 Amit Klein found that the randomizers used in most DNS Servers are not truly random: The next message ID's could be pre-calculated



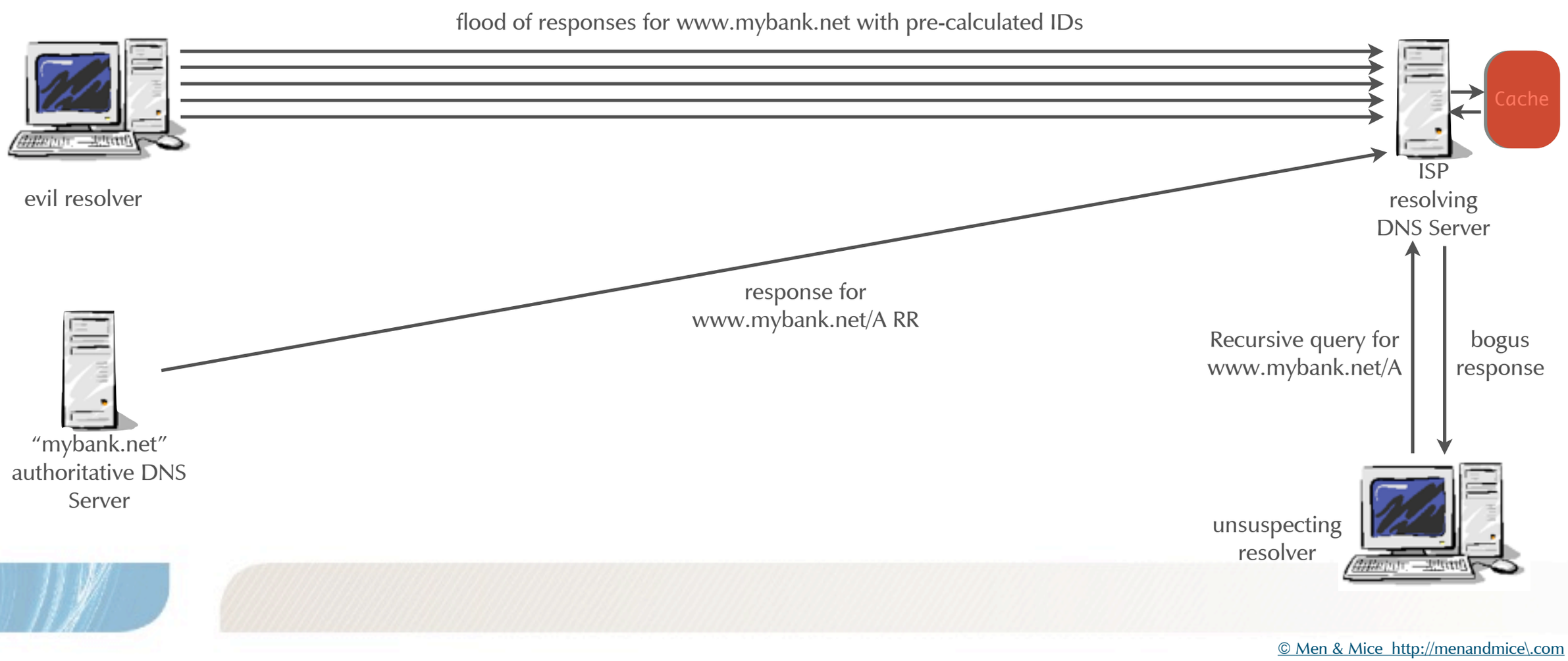
The Amit Klein findings (2)

- In 2007 Amit Klein found that the randomizers used in most DNS Servers are not truly random: The next message ID's could be pre-calculated



The Amit Klein findings (2)

- In 2007 Amit Klein found that the randomizers used in most DNS Servers are not truly random: The next message ID's could be pre-calculated



Bad randomizer

- The problem
  - The Query ID (QID) of DNS messages were not really random
    - They could be pre-calculated
- The fix
  - Better Randomizer code in the DNS Servers

DNS Cache Spoofing
Episode III

the Dan Kaminsky findings
March-August 2008

MEN&MICE

The Dan Kaminsky findings

- Internet security researcher Dan Kaminsky found a way to spoof DNS Server caches even if the QID is truly random
  - By making the target DNS have many open outstanding queries for a domain that is 'in bailiwick' of the domain to be spoofed
- The problem
  - Even if the 16bit QID is truly random, a carefully crafted attack can fool the DNS Servers safety checks

The Dan Kaminsky findings (1)

evil
web-server

evil resolver

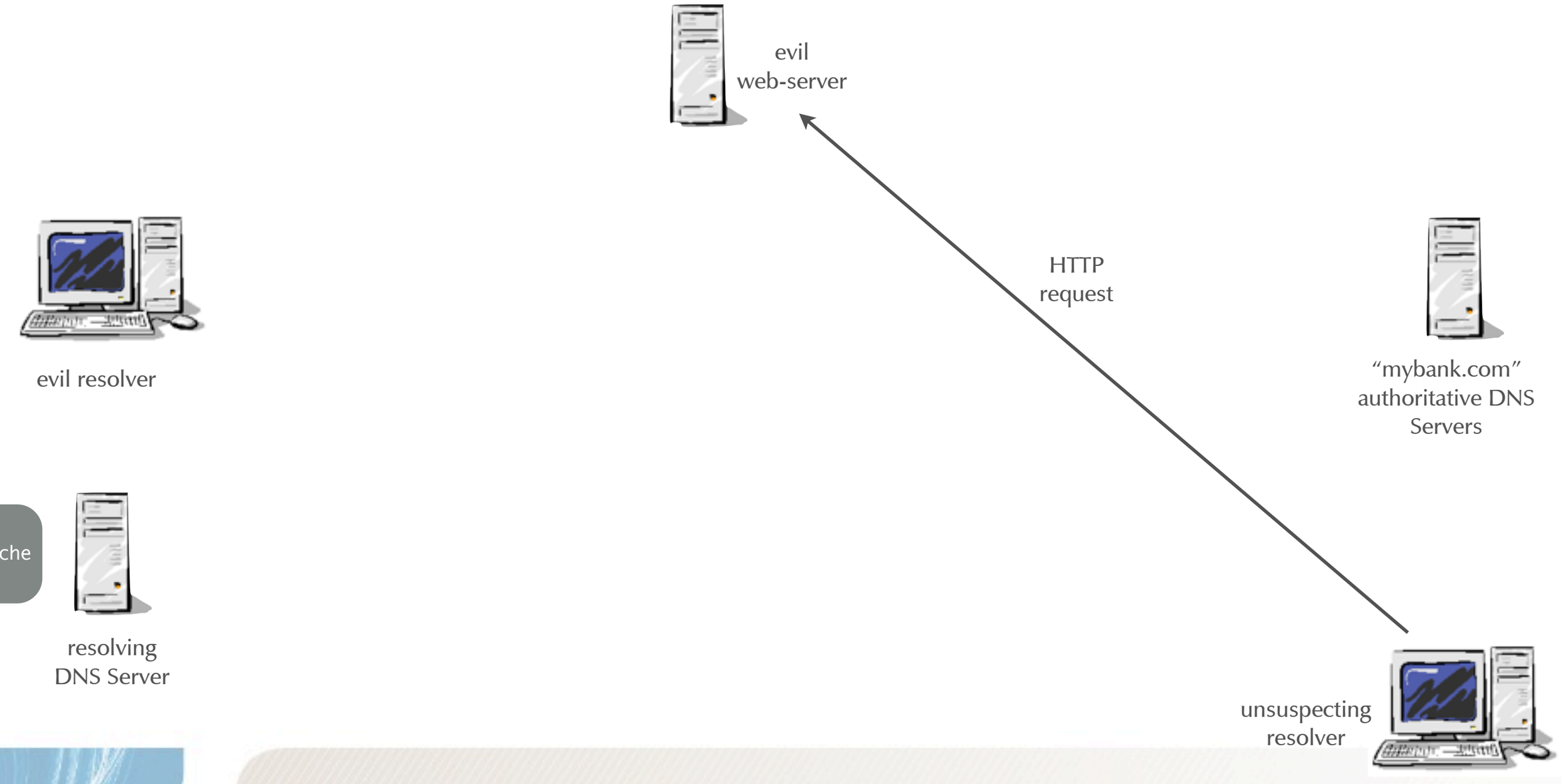
“mybank.com”
authoritative DNS
Servers

Cache

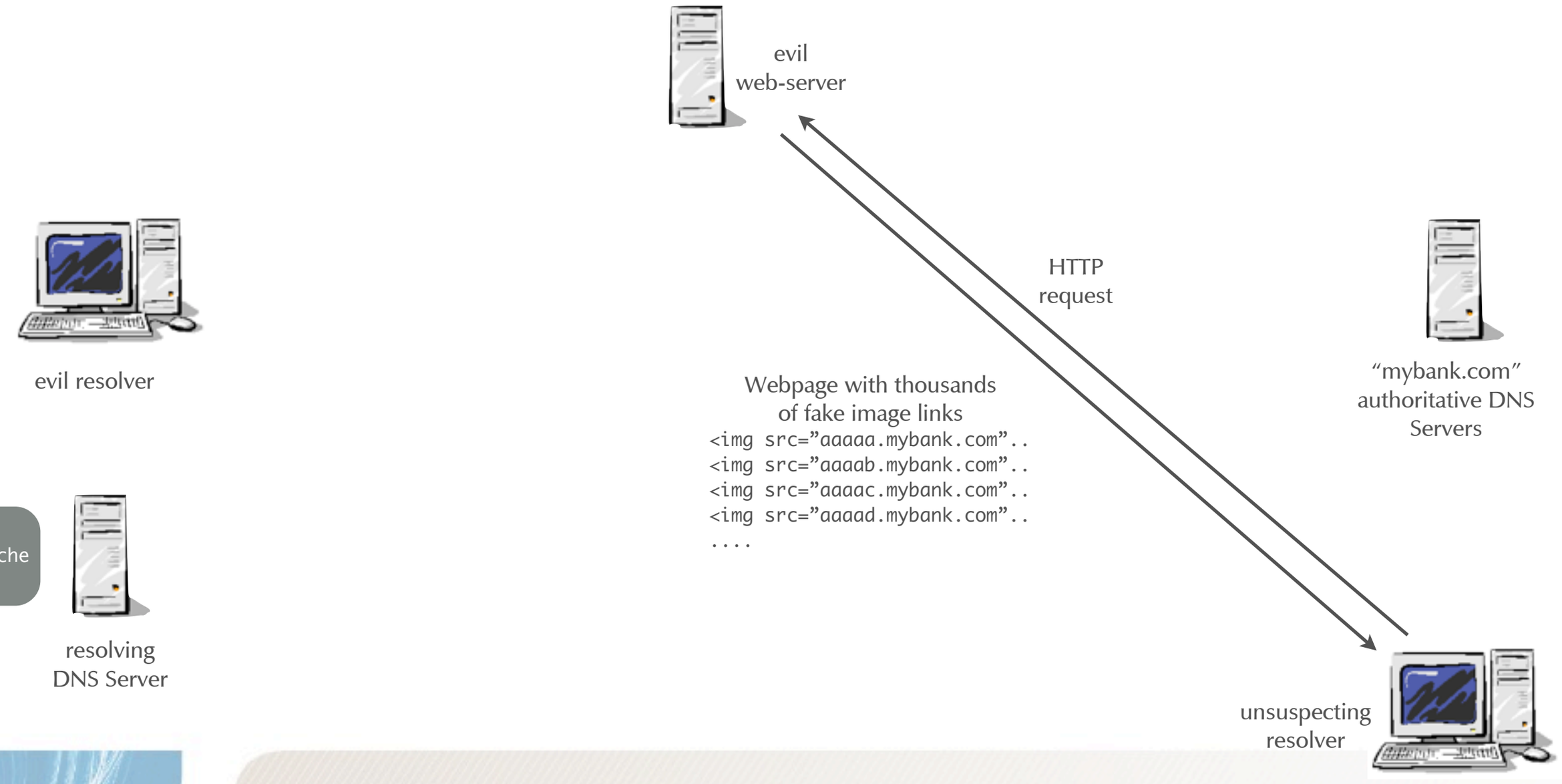
resolving
DNS Server

unsuspecting
resolver

The Dan Kaminsky findings (1)



The Dan Kaminsky findings (1)



The Dan Kaminsky findings (2)

evil
web-server

evil resolver

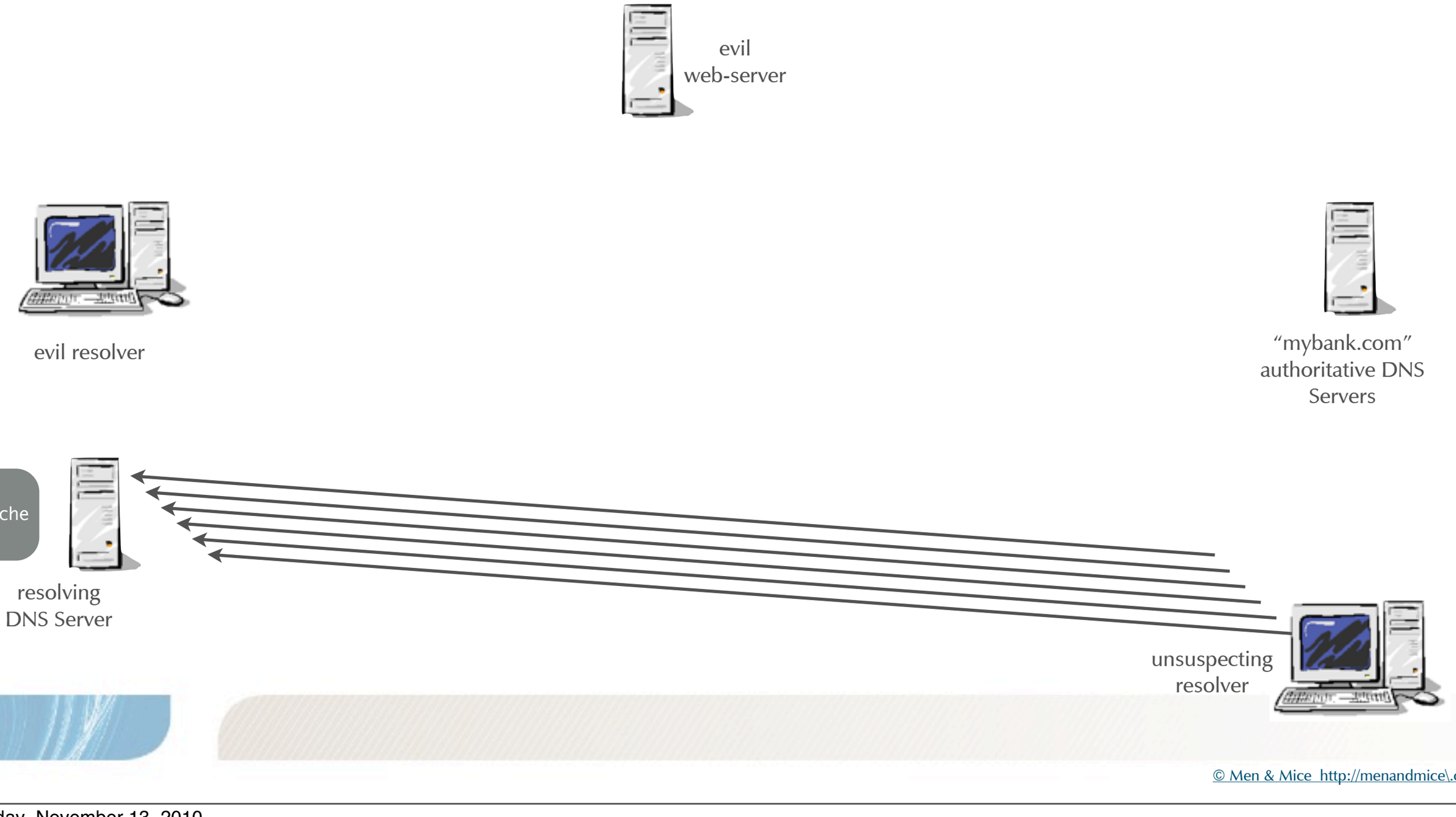
“mybank.com”
authoritative DNS
Servers

Cache

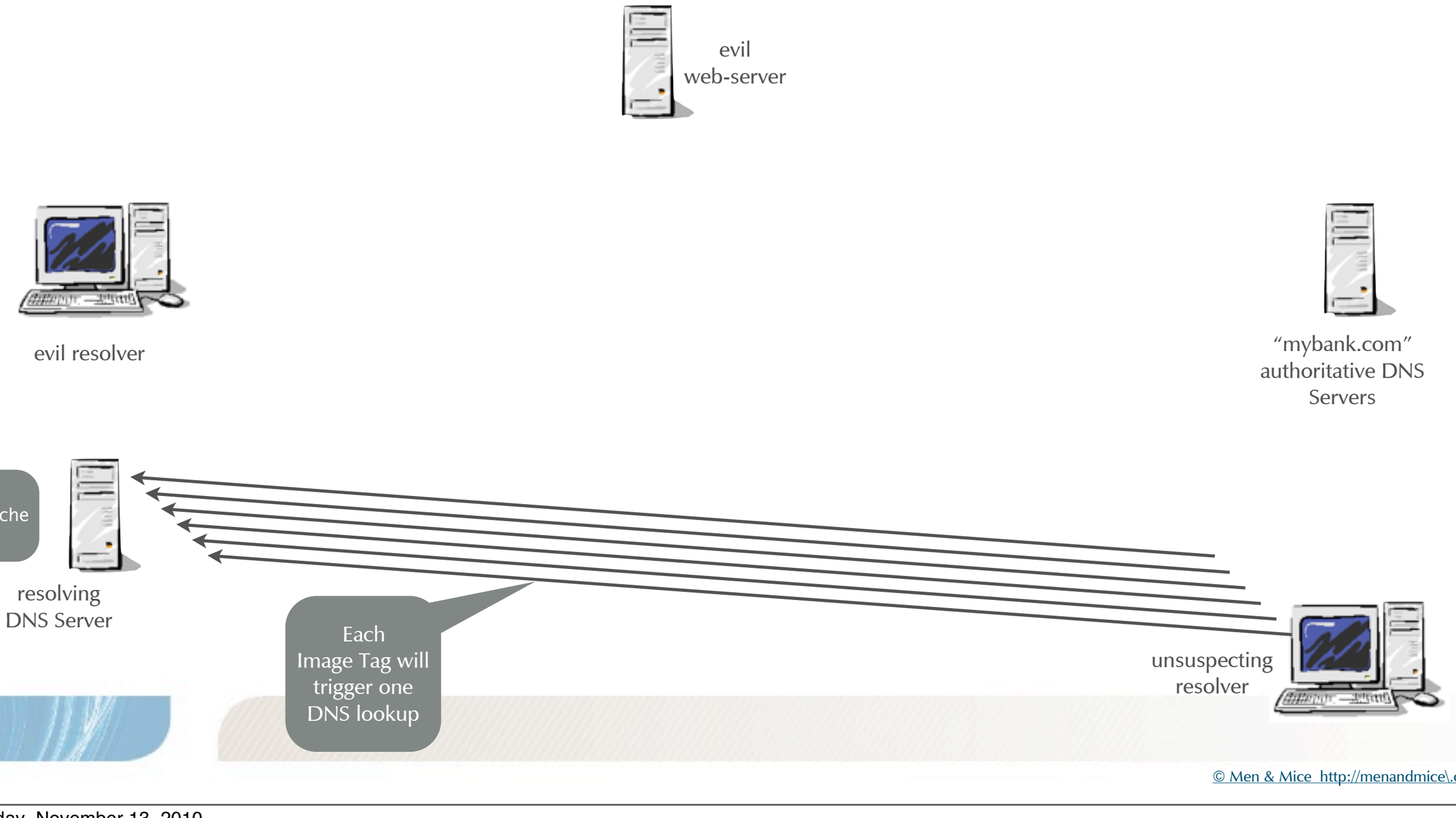
resolving
DNS Server

unsuspecting
resolver

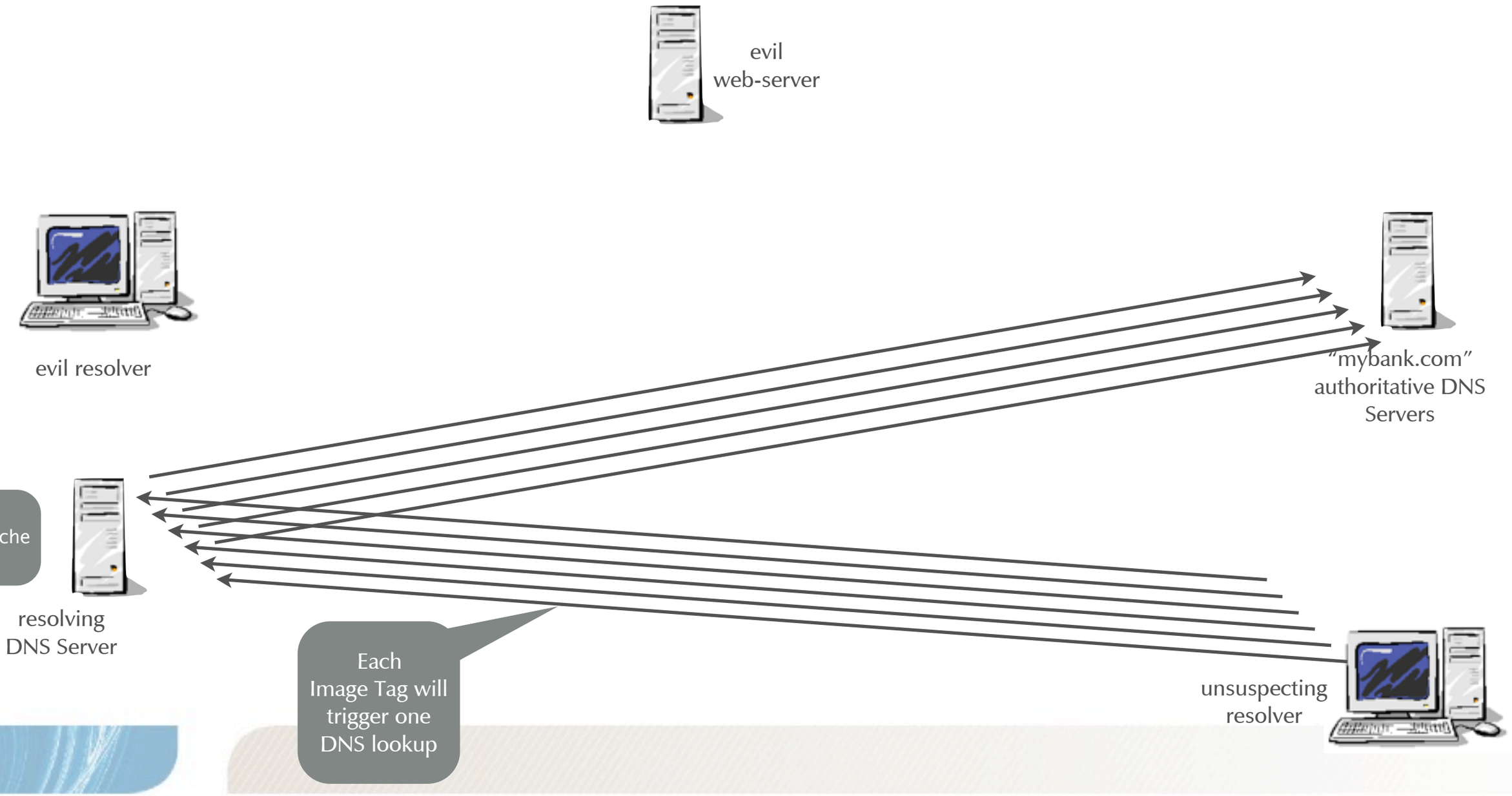
The Dan Kaminsky findings (2)



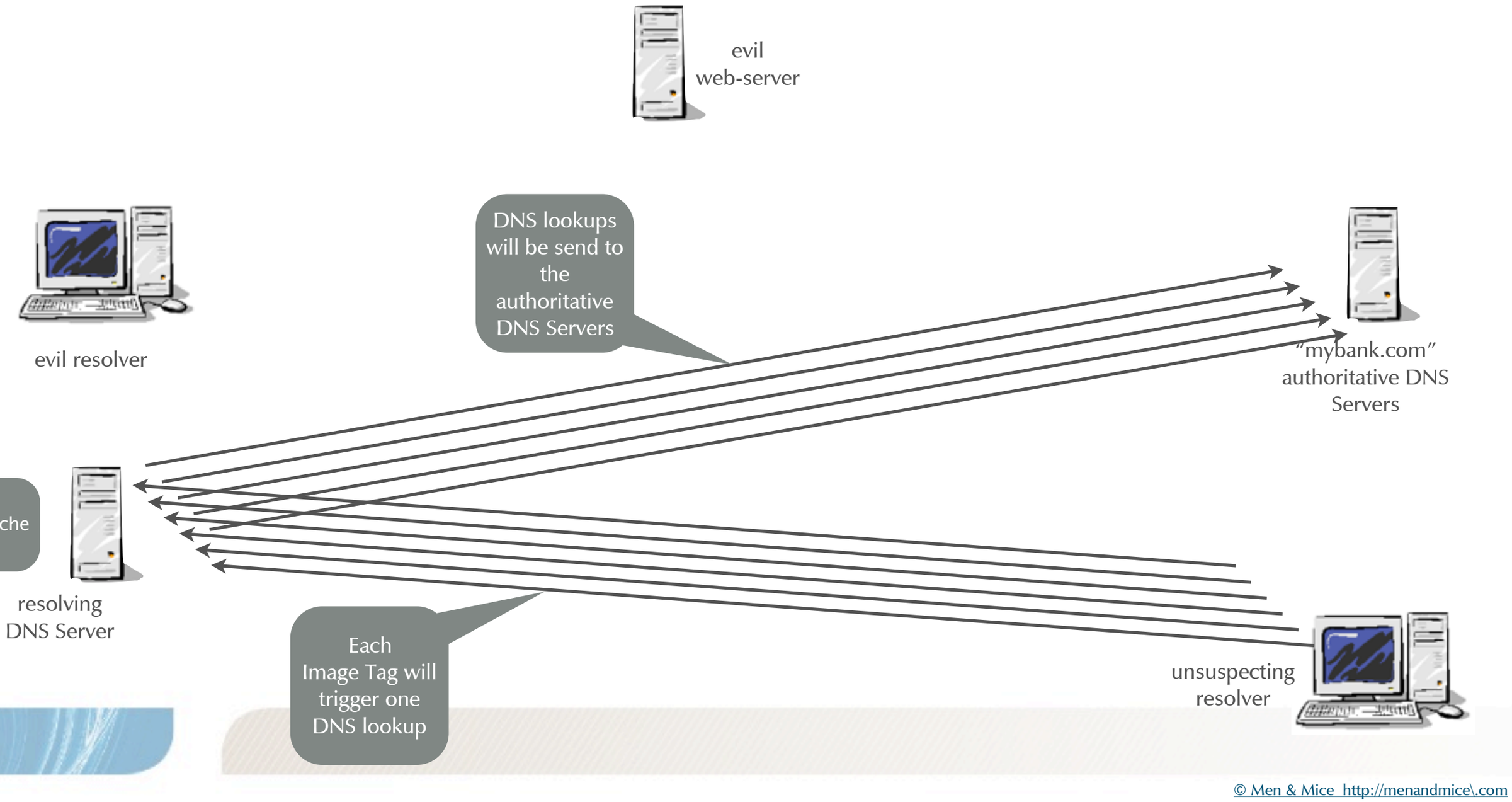
The Dan Kaminsky findings (2)



The Dan Kaminsky findings (2)



The Dan Kaminsky findings (2)



The Dan Kaminsky findings (3)

evil
web-server

evil resolver

“mybank.com”
authoritative DNS
Servers

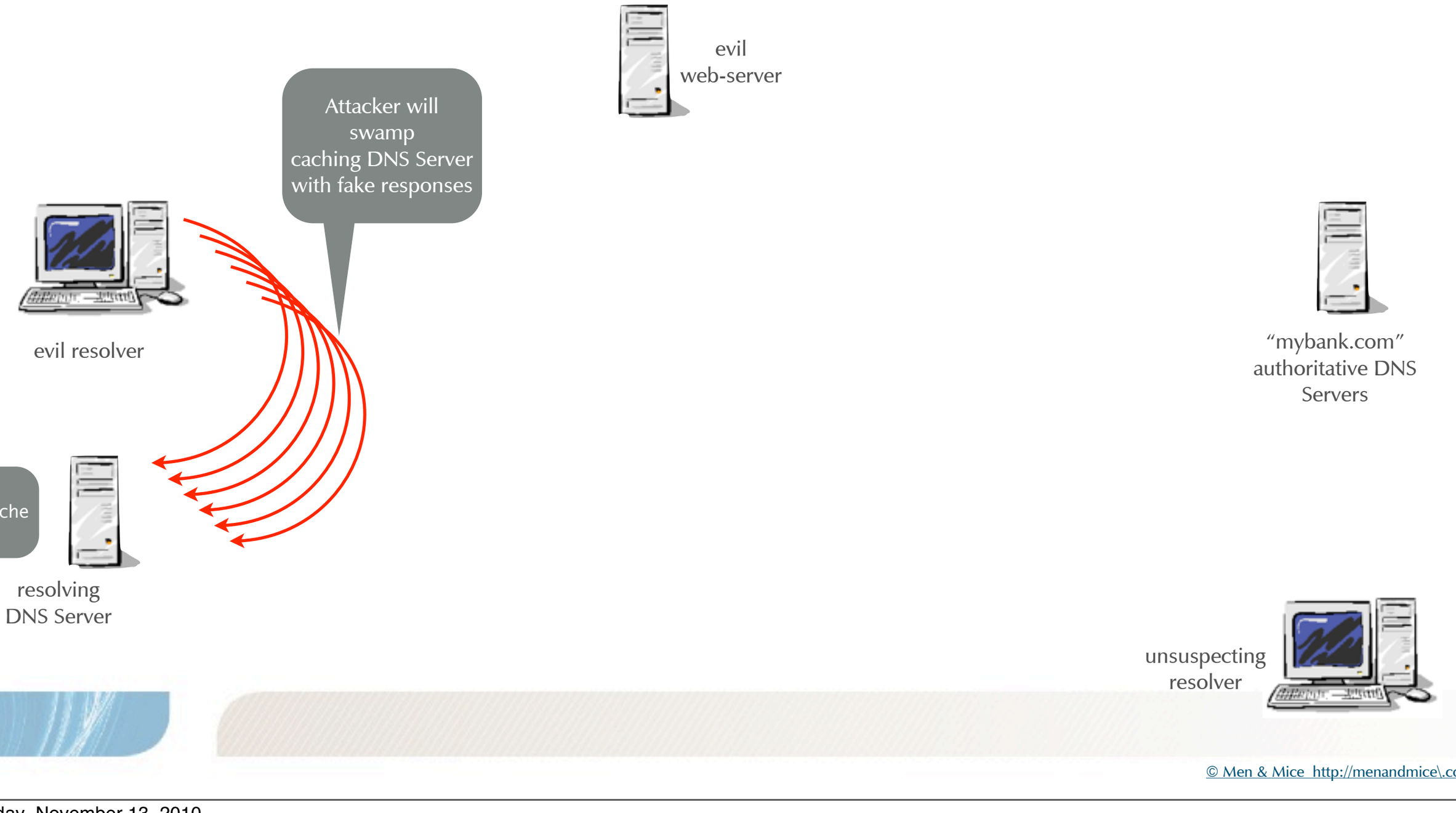
Cache

resolving
DNS Server

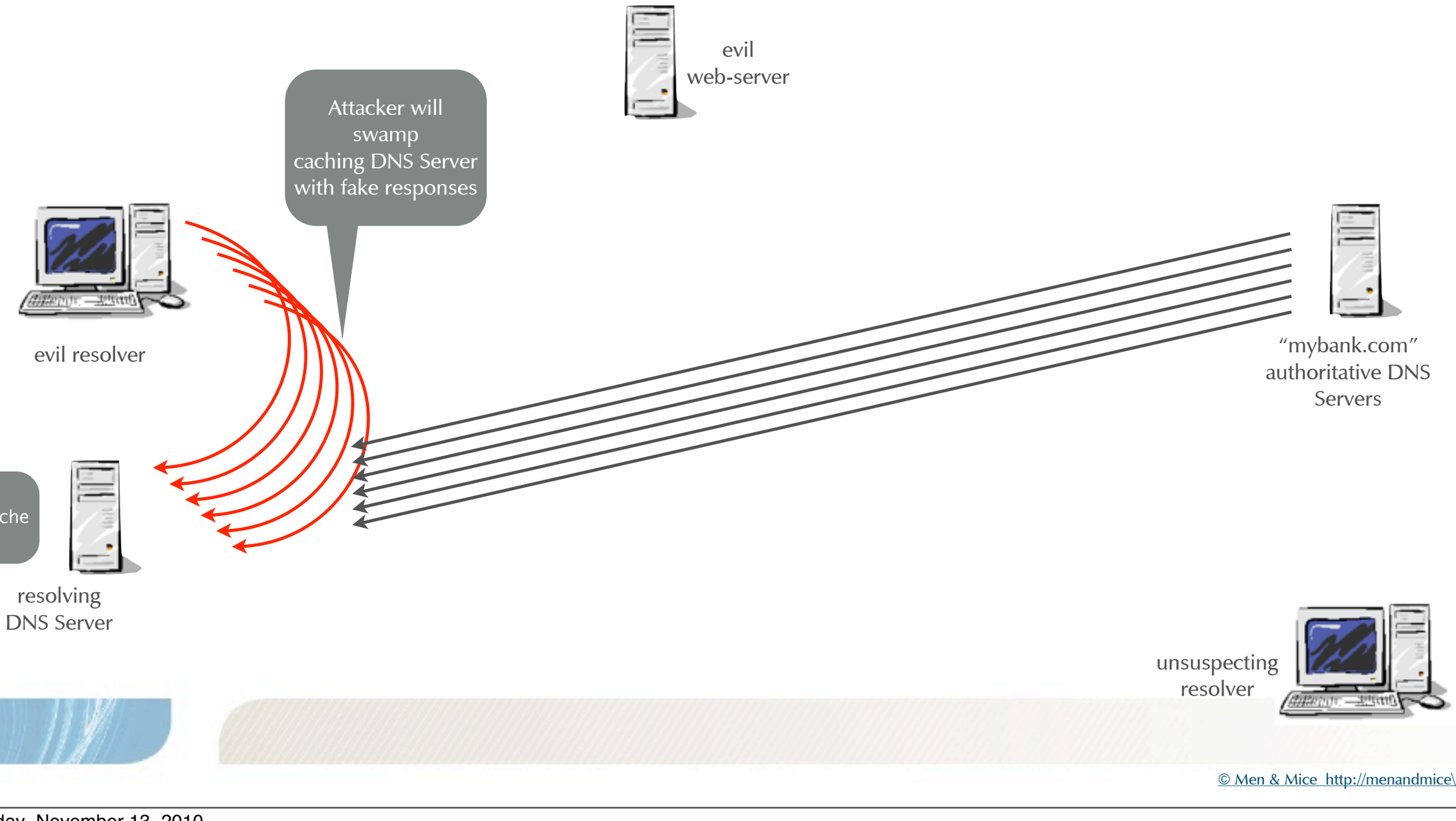
unsuspecting
resolver

The Dan Kaminsky findings (3)

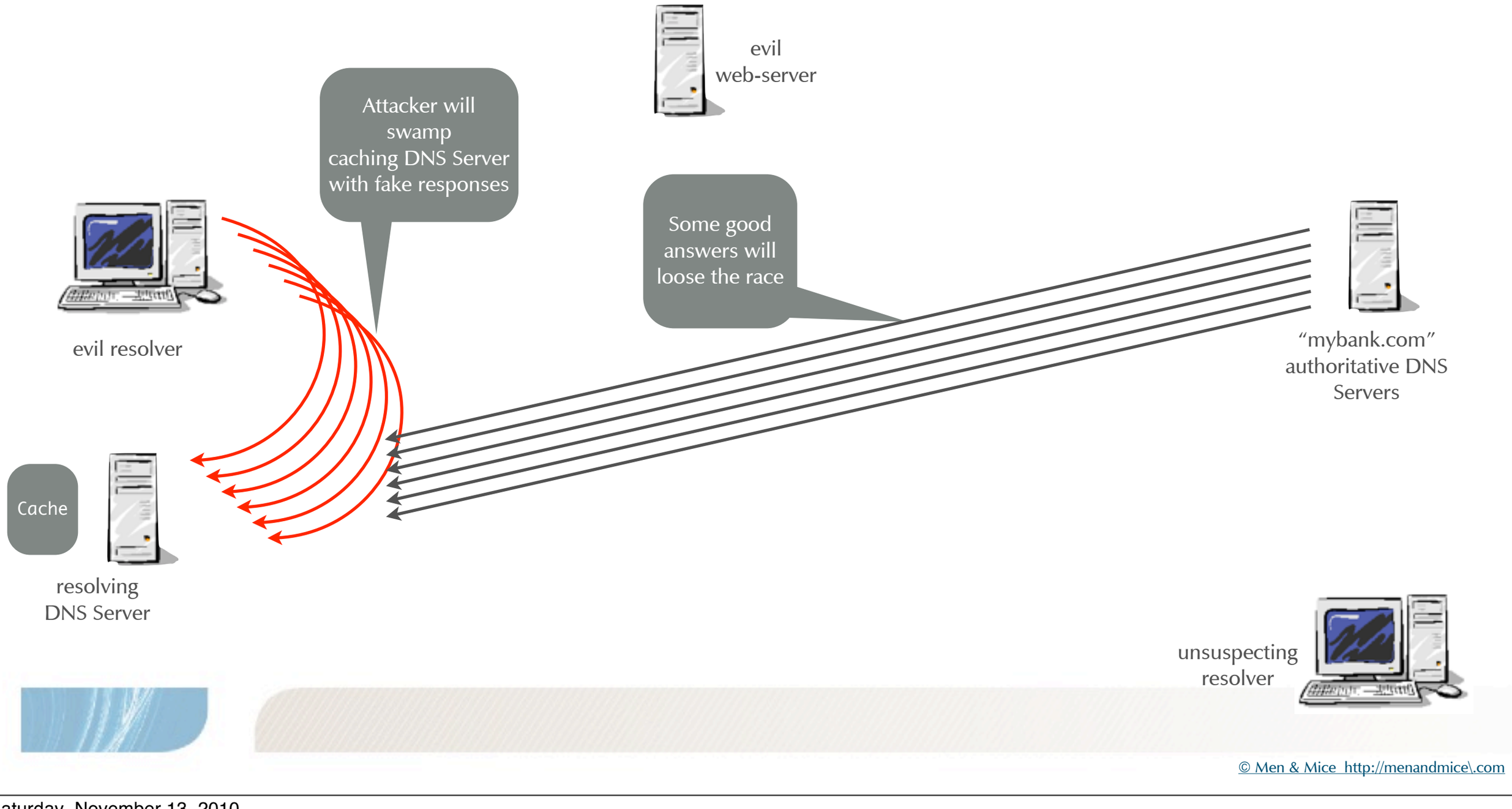
The Dan Kaminsky findings (3)



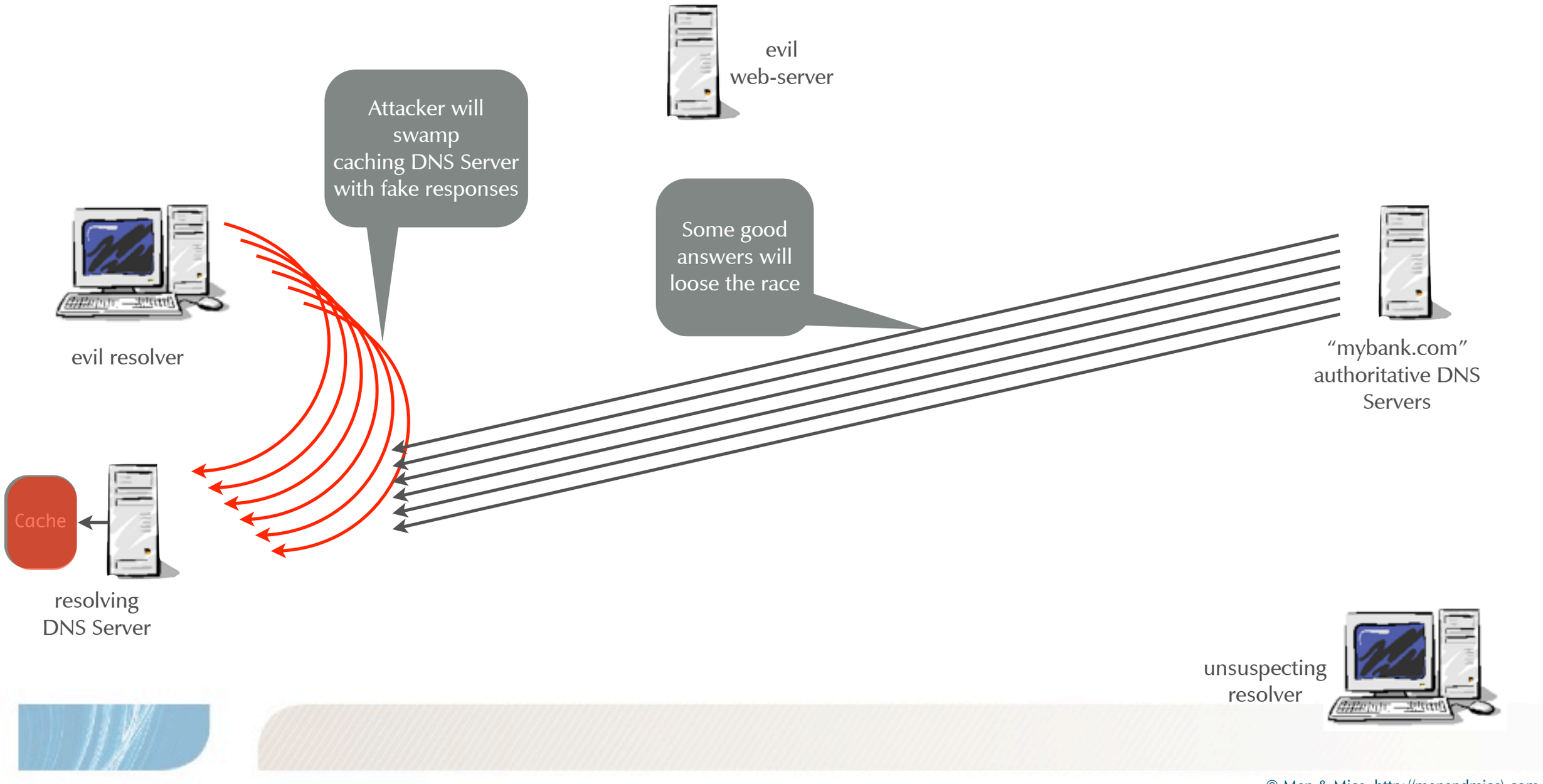
The Dan Kaminsky findings (3)



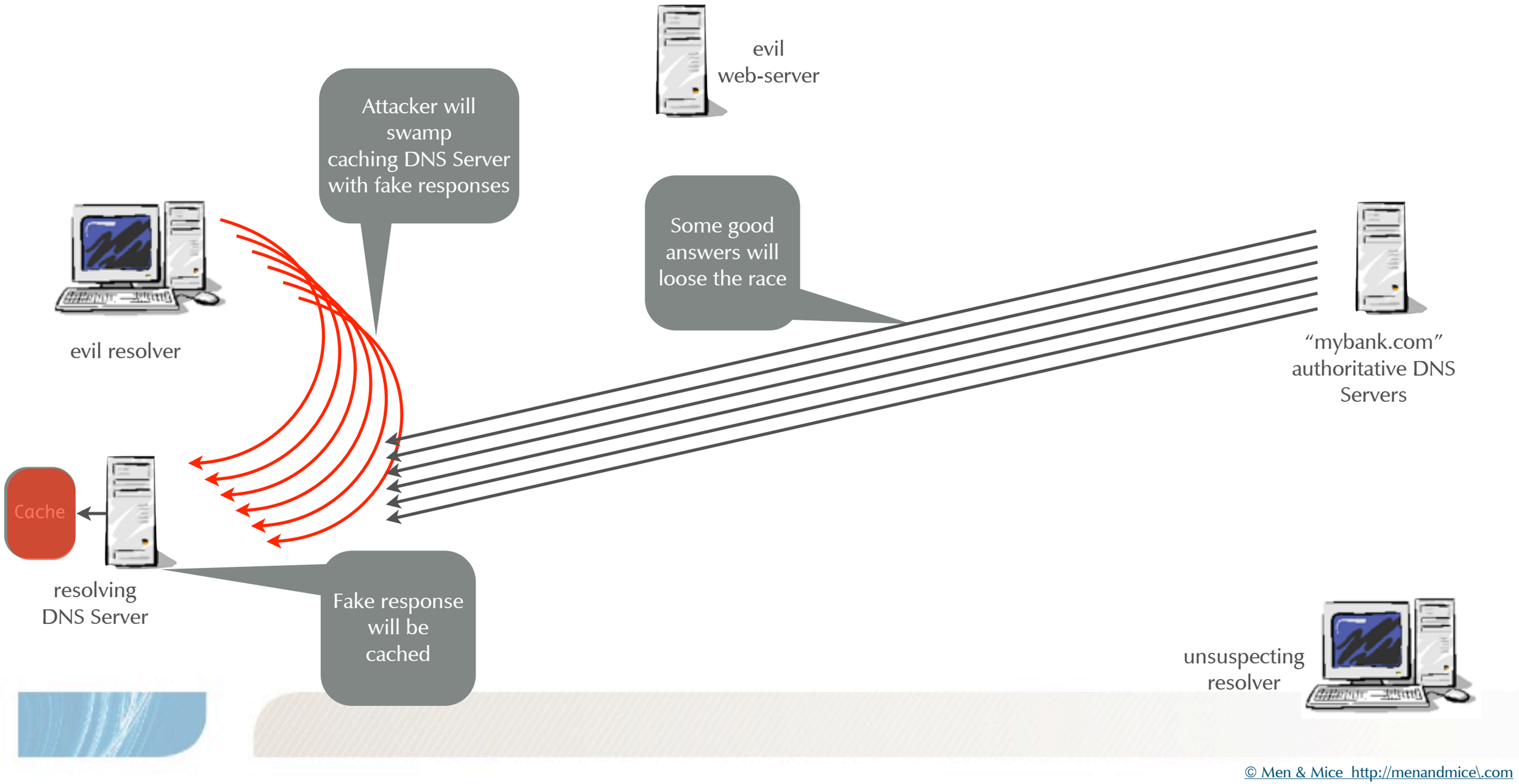
The Dan Kaminsky findings (3)



The Dan Kaminsky findings (3)



The Dan Kaminsky findings (3)



The Dan Kaminsky findings (3)

evil
web-server

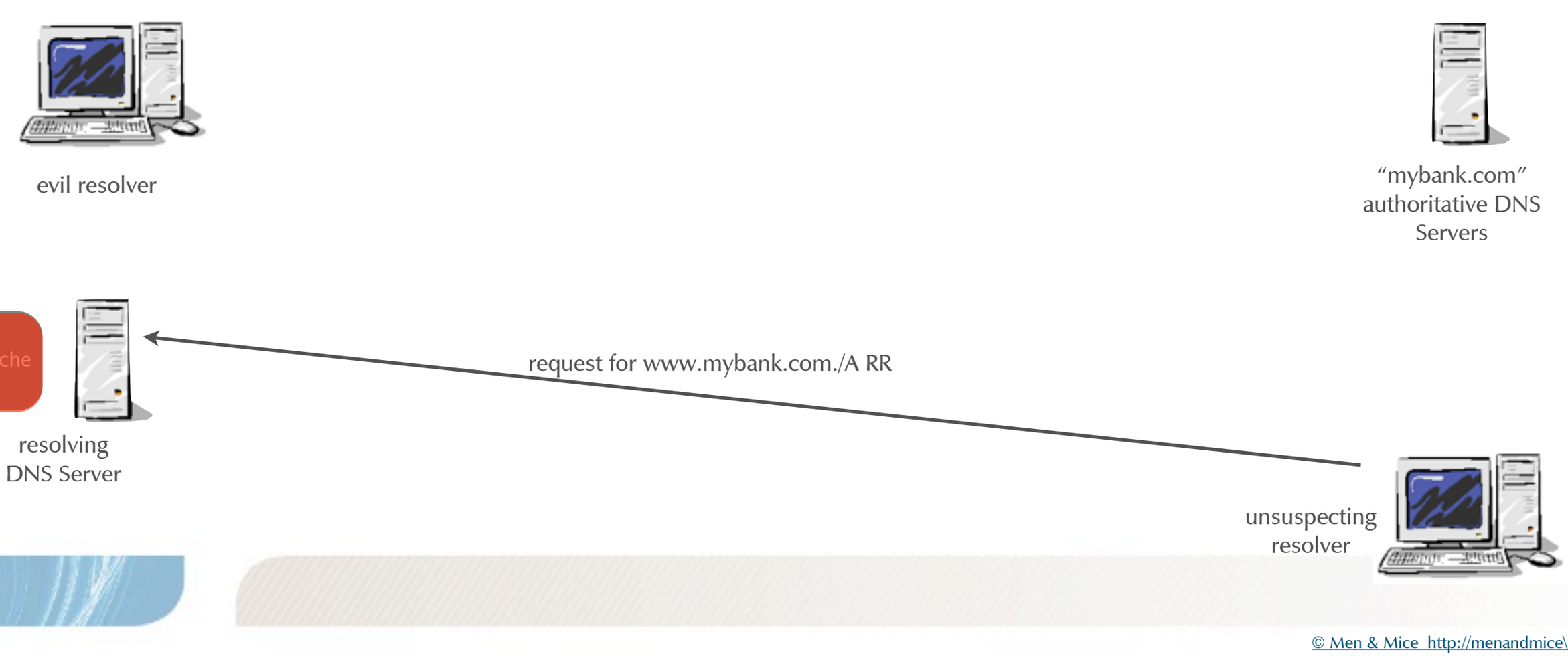
evil resolver

“mybank.com”
authoritative DNS
Servers

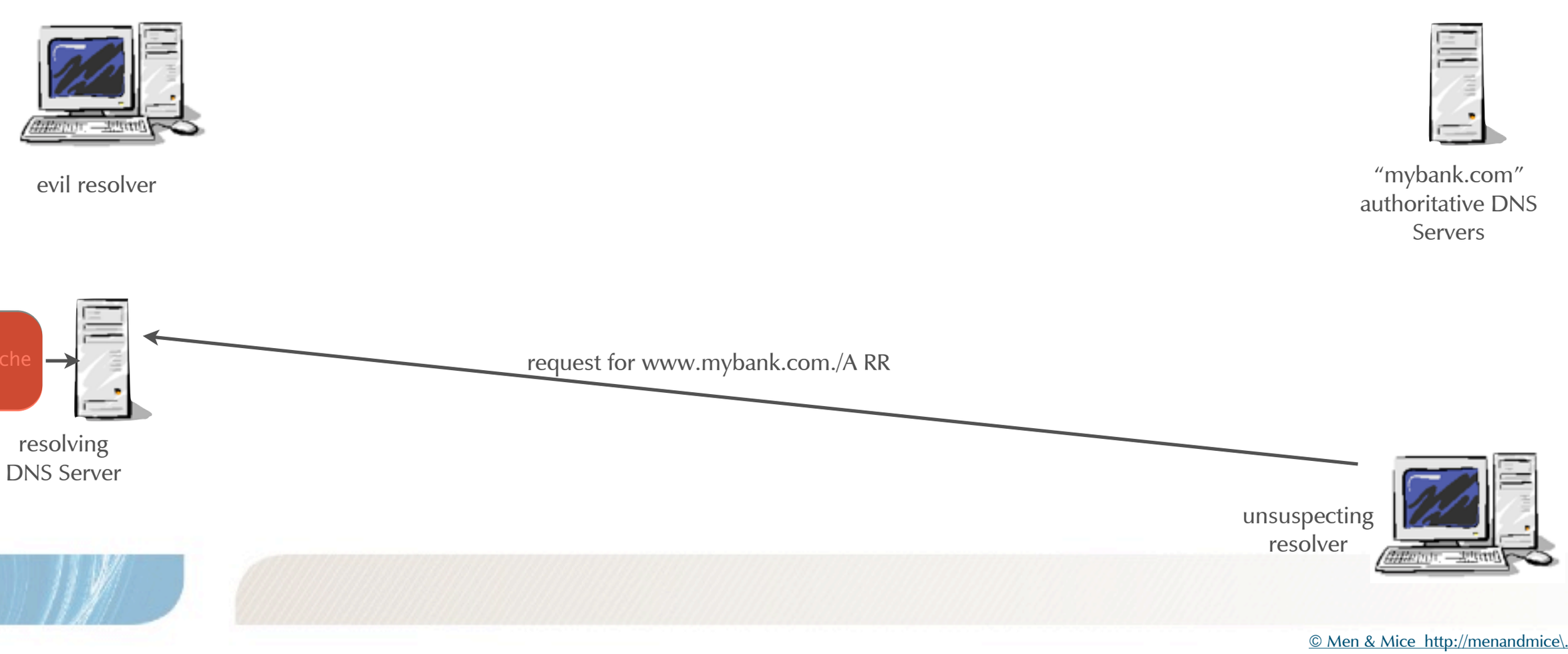
resolving
DNS Server

unsuspecting
resolver

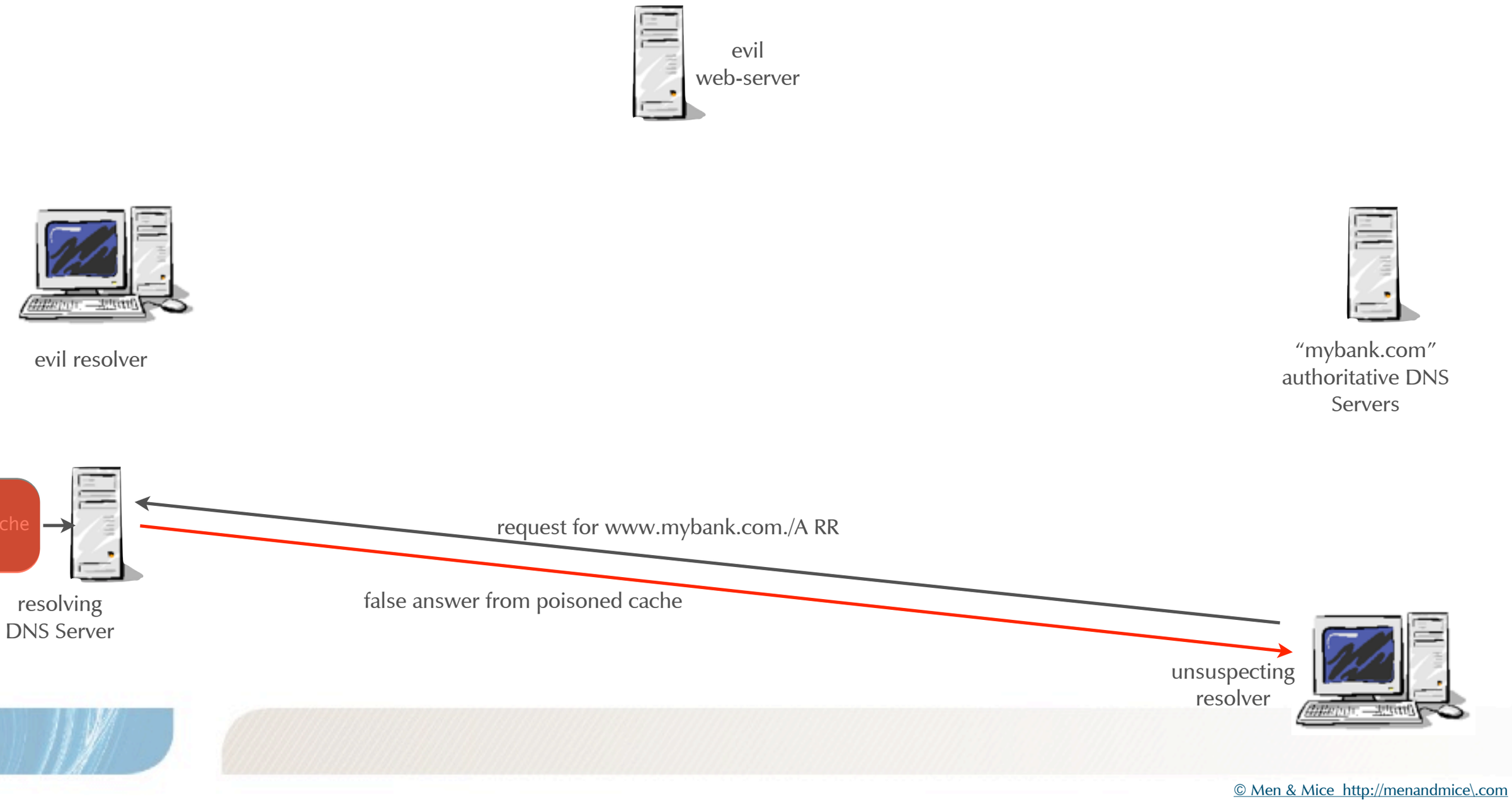
The Dan Kaminsky findings (3)



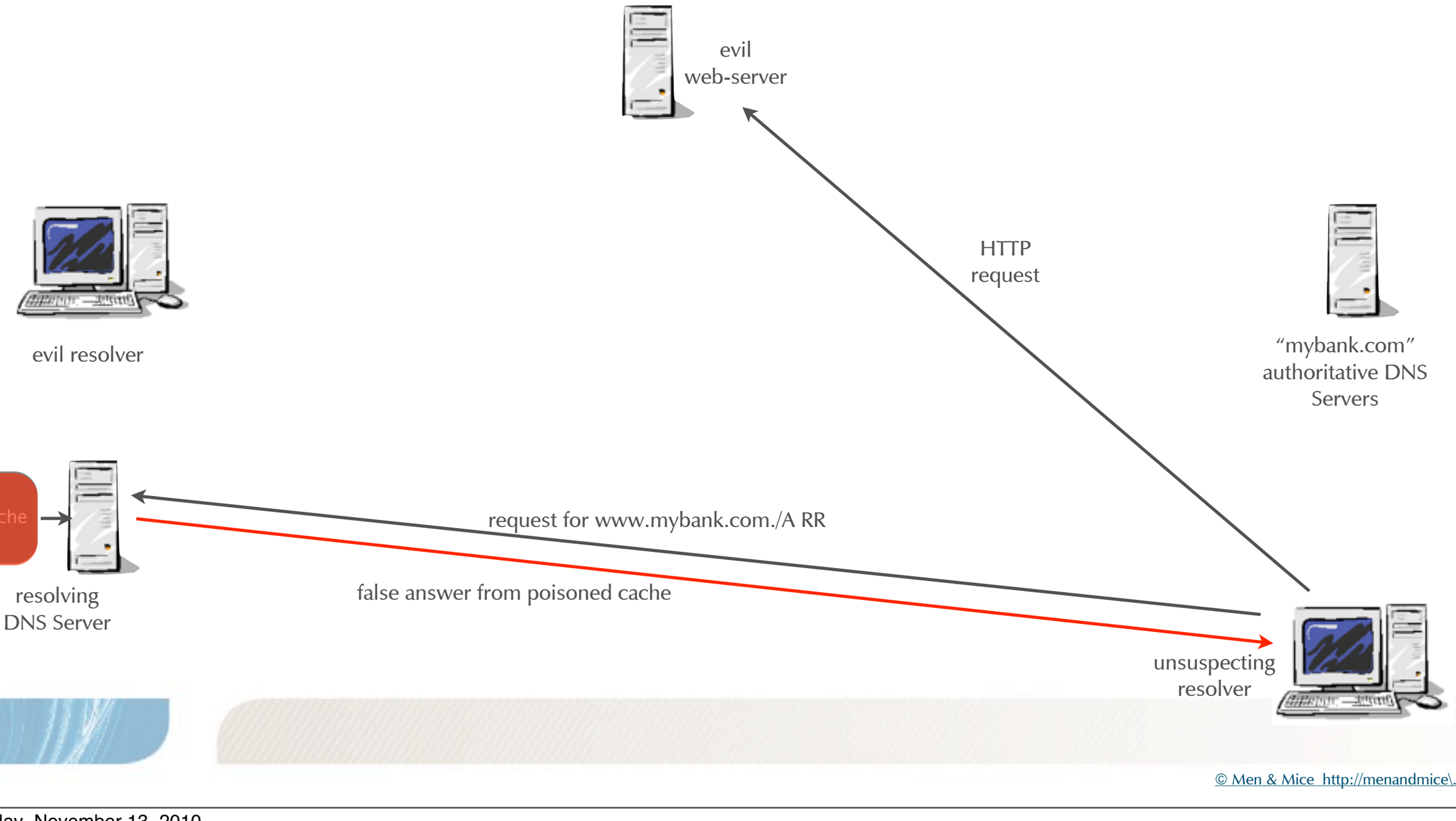
The Dan Kaminsky findings (3)



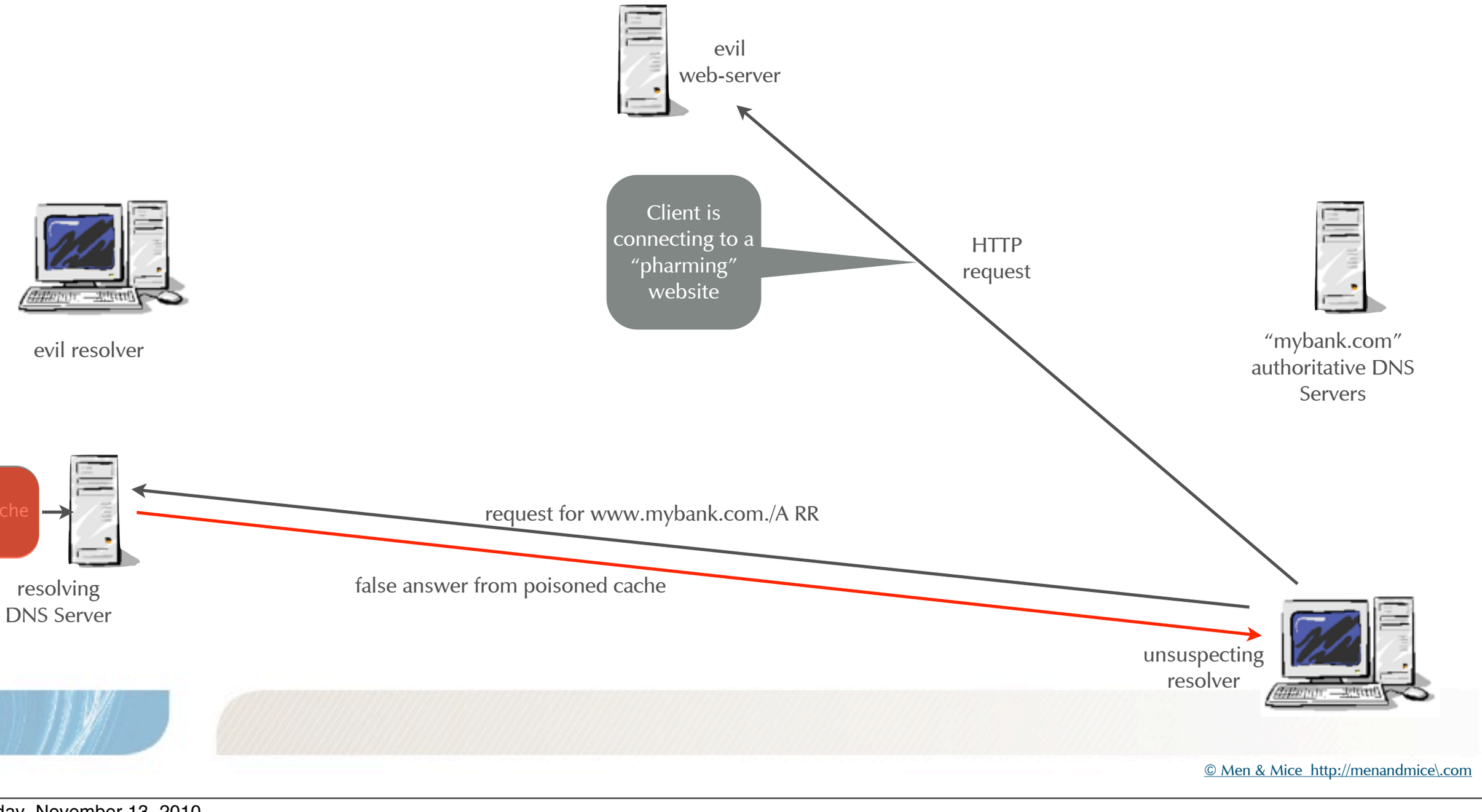
The Dan Kaminsky findings (3)



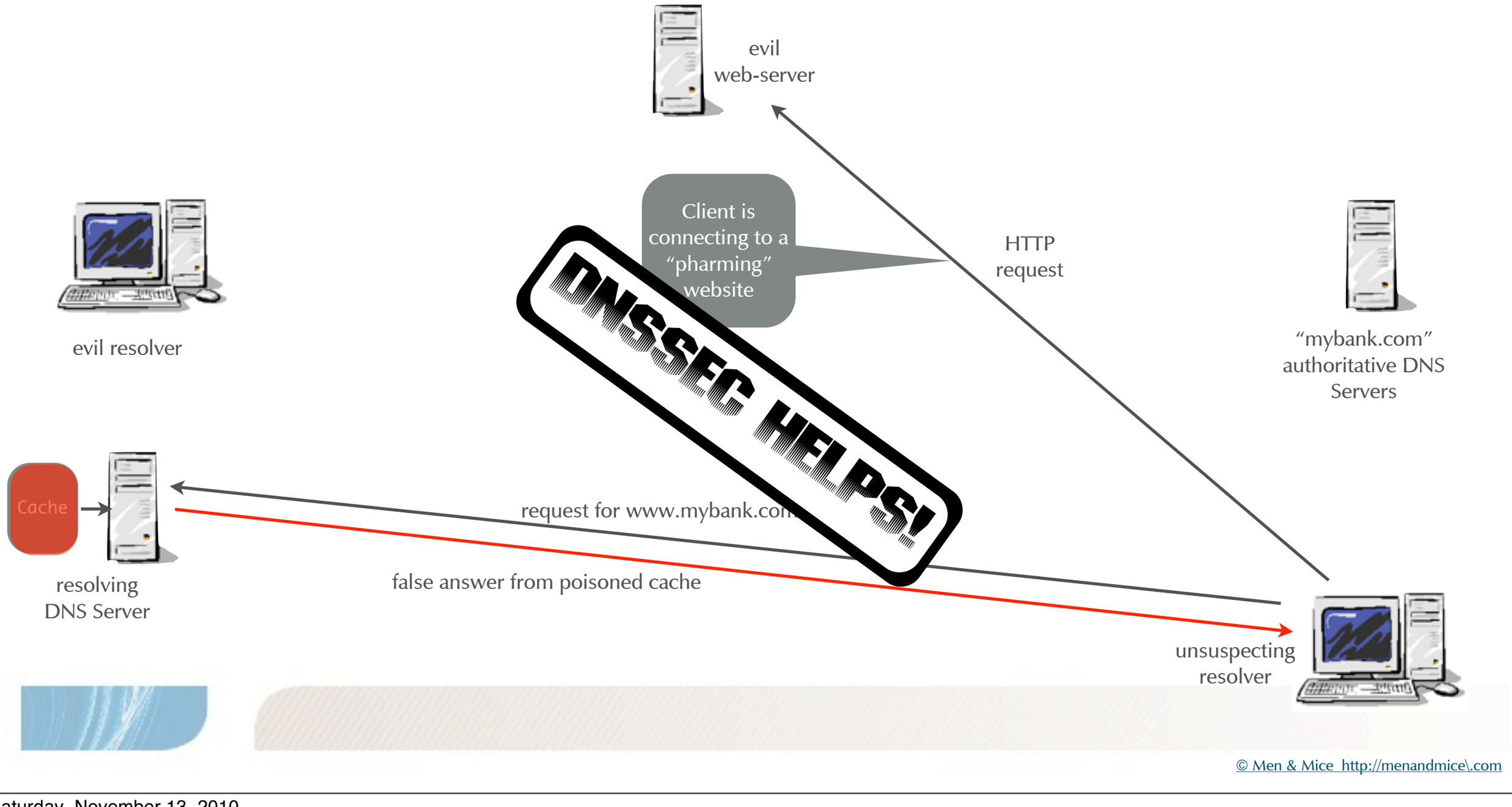
The Dan Kaminsky findings (3)



The Dan Kaminsky findings (3)



The Dan Kaminsky findings (3)



the Dan Kaminsky “bug”

- Attackers try to overwrite or place a NS record in the cache

```
;; ANSWER SECTION:
aaaa.mybank.com.    120    IN    A    1.2.3.4

;; AUTHORITY SECTION:
mybank.com.         86400  IN    NS   ns1.mybank.com.
mybank.com.         86400  IN    NS   ns2.mybank.com.

;; ADDITIONAL SECTION:
ns1.mybank.com.     604800 IN    A    192.0.2.20
ns2.mybank.com.     604800 IN    A    192.0.2.30
```

the Dan Kaminsky "bug"

- Attackers try to overwrite or place a NS record in the cache

```
;; ANSWER SECTION:
aaaa.mybank.com.    120    IN    A    1.2.3.4

;; AUTHORITY SECTION:
mybank.com.         86400  IN    NS   ns1.mybank.com.
mybank.com.         86400  IN    NS   ns2.mybank.com.

;; ADDITIONAL SECTION:
ns1.mybank.com.     604800 IN    A    192.0.2.20
ns2.mybank.com.     604800 IN    A    192.0.2.30
```

high TTL for maximum damage

the Dan Kaminsky "bug"

- Attackers try to overwrite or place a NS record in the cache

```
;; ANSWER SECTION:
aaaa.mybank.com.    120    IN    A    1.2.3.4

;; AUTHORITY SECTION:
mybank.com.         86400  IN    NS   ns1.mybank.com.
mybank.com.         86400  IN    NS   ns2.mybank.com.

;; ADDITIONAL SECTION:
ns1.mybank.com.     604800 IN    A    192.0.2.20
ns2.mybank.com.     604800 IN    A    192.0.2.30
```

high TTL for maximum damage

Here is the fake data

The Dan Kaminsky findings

- The patch
  - Add more randomization bits
    - UDP Source Port randomization
    - Other tricks and enhancements that will add more random bits to the inter-DNS-Server communication
- The Fix
  - Deploy and use of DNSSEC (in a large scale)

Patching the Dan Kaminsky "bug"

- UDP Source port randomization will only "raise the bar" for the attacker
  - From about 10 seconds to about 10 hours for an successful attack (if no 'per host' rate limit is configured on the inbound network)
- After configuring/updating a DNS Server, test if the source ports appear random in the Internet
  - Some Firewalls or NAT devices can "undo" the UDP Source Port randomization

Patching the Dan Kaminsky "bug"

- DNS Server behind a Firewall are **not** immune against cache poisoning attacks

- Test the UDP Port randomization for the **whole** network device chain from the DNS Server to the outside Internet

```
# dig +short txt porttest.dns-oarc.net
porttest.y.x.w.v.u.t.s.r.q.p.o.n.m.l.k.j.i.h.g.f.e.d.c.b.a
.pt.dns-oarc.net.

"192.0.2.10 is GREAT: 26 queries in 4.5 seconds from 26
ports with std dev 17383"
```

Preventing Cache Poisioning

- There is only one real “fix” available: DNSSEC

“Men in the middle” attacks

- DNS Messages can be intercepted “enroute” and can be changed or altered

- “Men in the middle” attacks are easy with plain DNS
  - DNS UDP communication is “stateless”
  - Each DNS packet (query and answer) contains a full header and the query section

Men in the middle attack

- an attacker en-route can change DNS data unnoticed

authoritative DNS Server

ISP resolving DNS Server

Cache

attacker

client resolver

Men in the middle attack

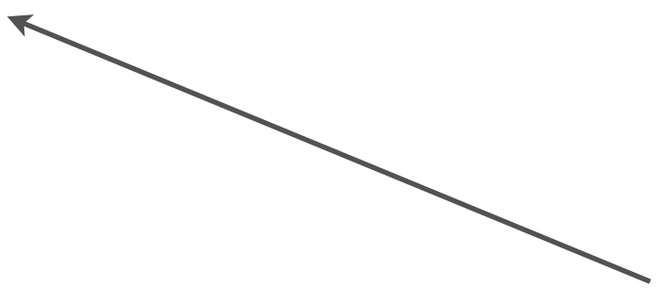
- an attacker en-route can change DNS data unnoticed

authoritative DNS Server

ISP resolving DNS Server

attacker

Cache



client resolver

Men in the middle attack

- an attacker en-route can change DNS data unnoticed

authoritative DNS Server

ISP resolving DNS Server

attacker

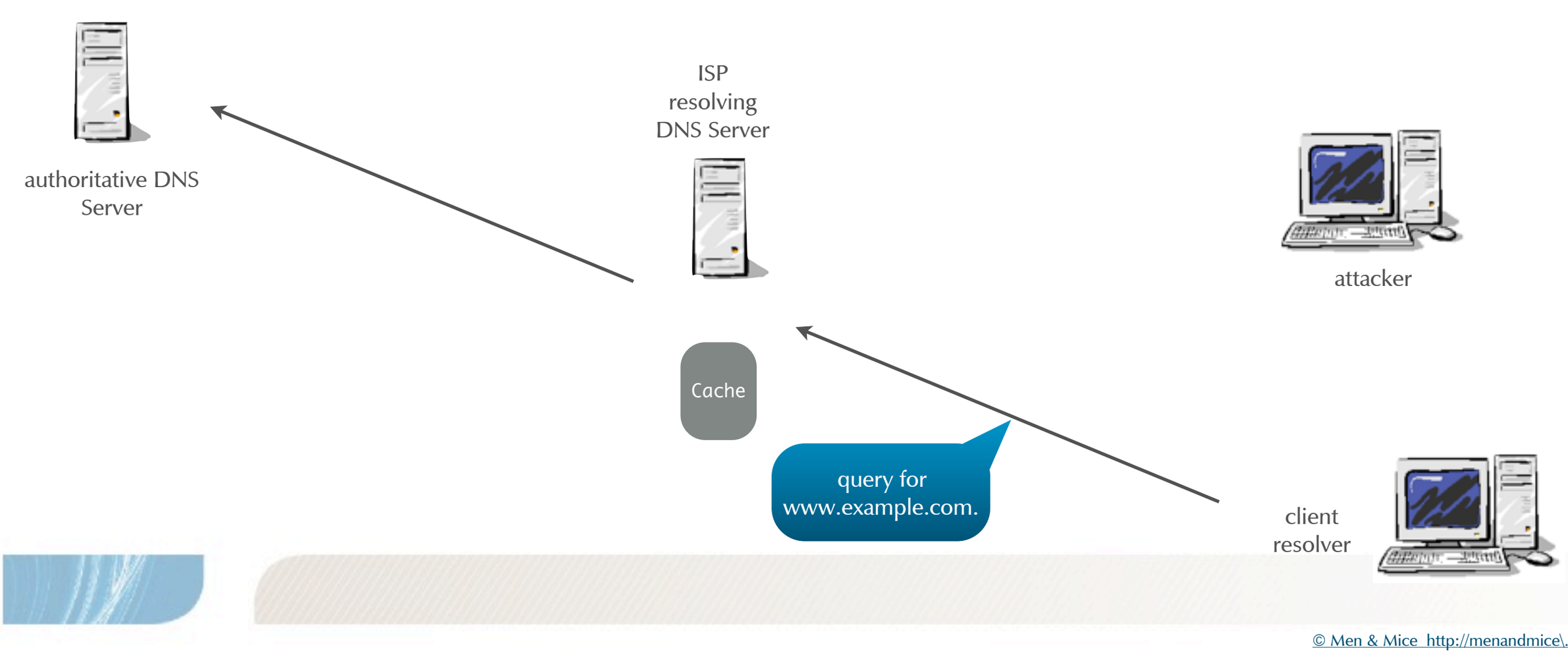
Cache

query for www.example.com.

client resolver

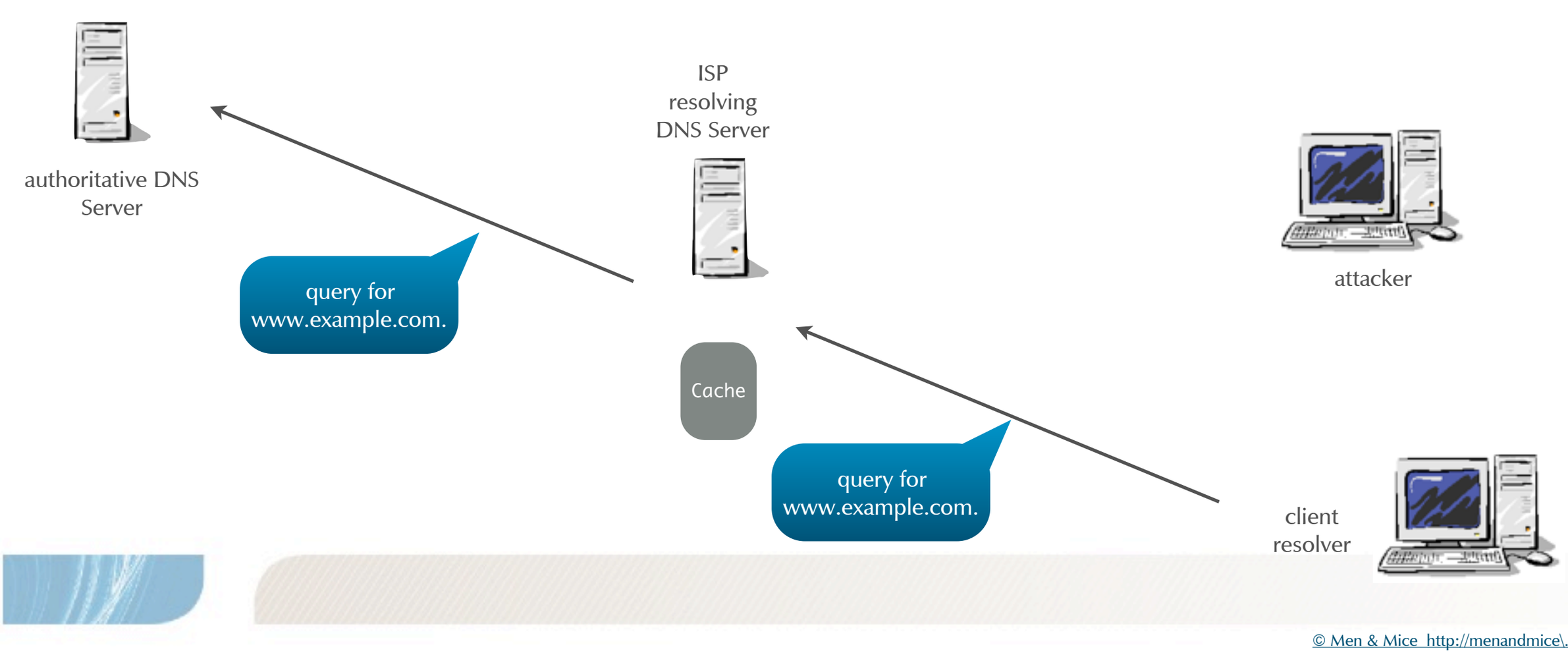
Men in the middle attack

- an attacker en-route can change DNS data unnoticed



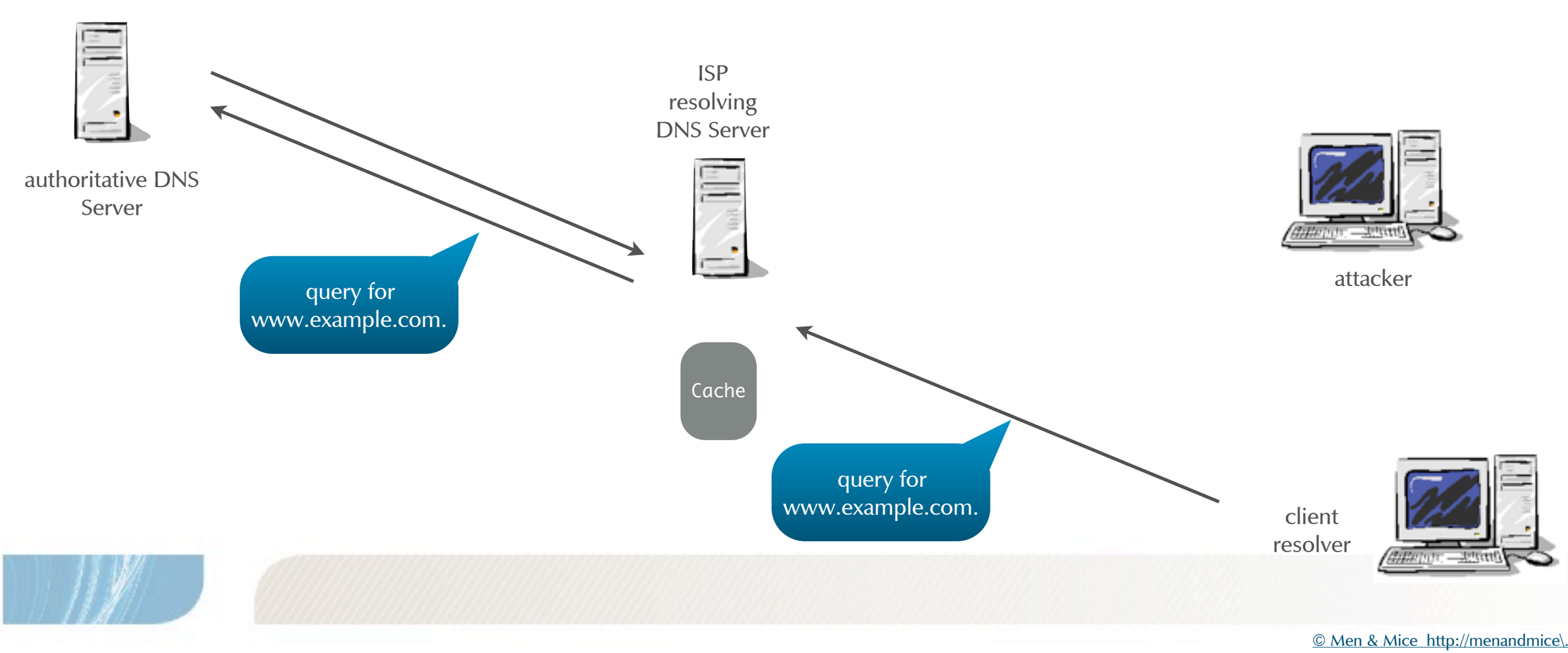
Men in the middle attack

- an attacker en-route can change DNS data unnoticed



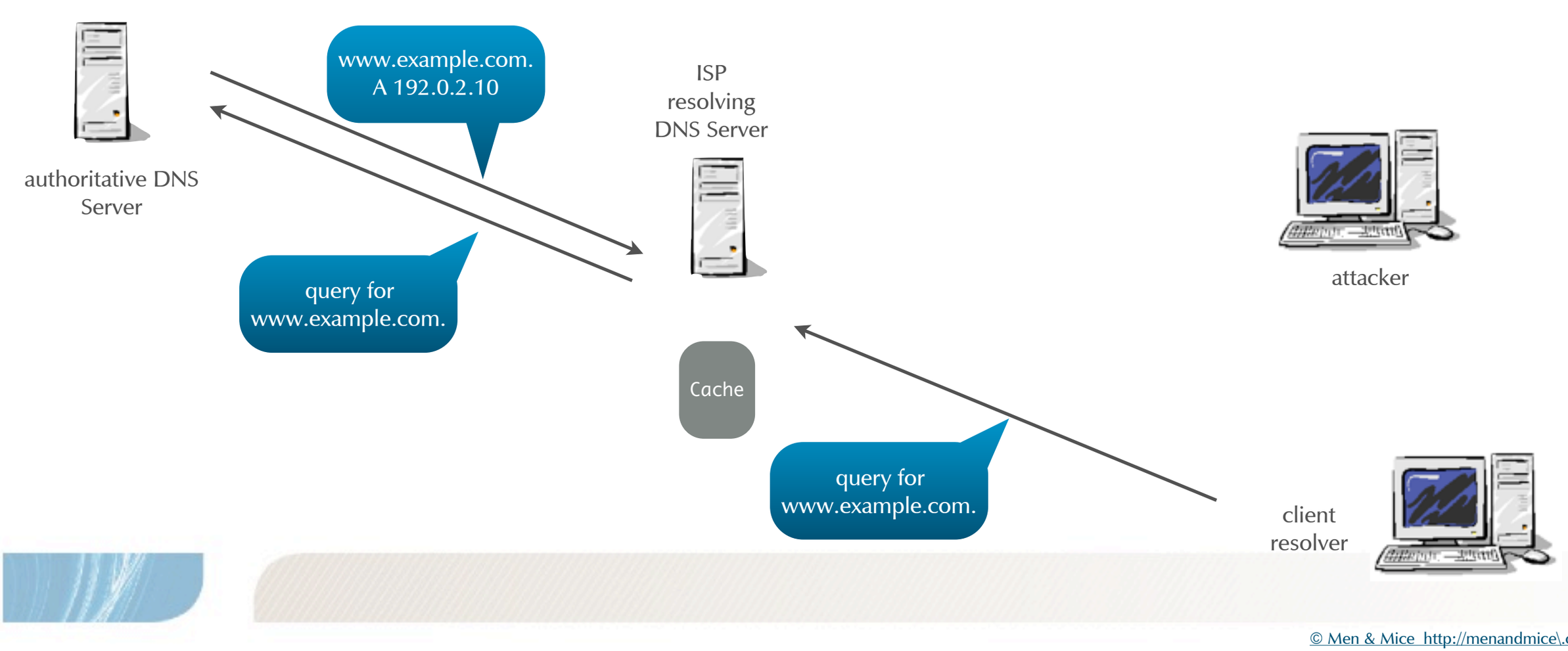
Men in the middle attack

- an attacker en-route can change DNS data unnoticed



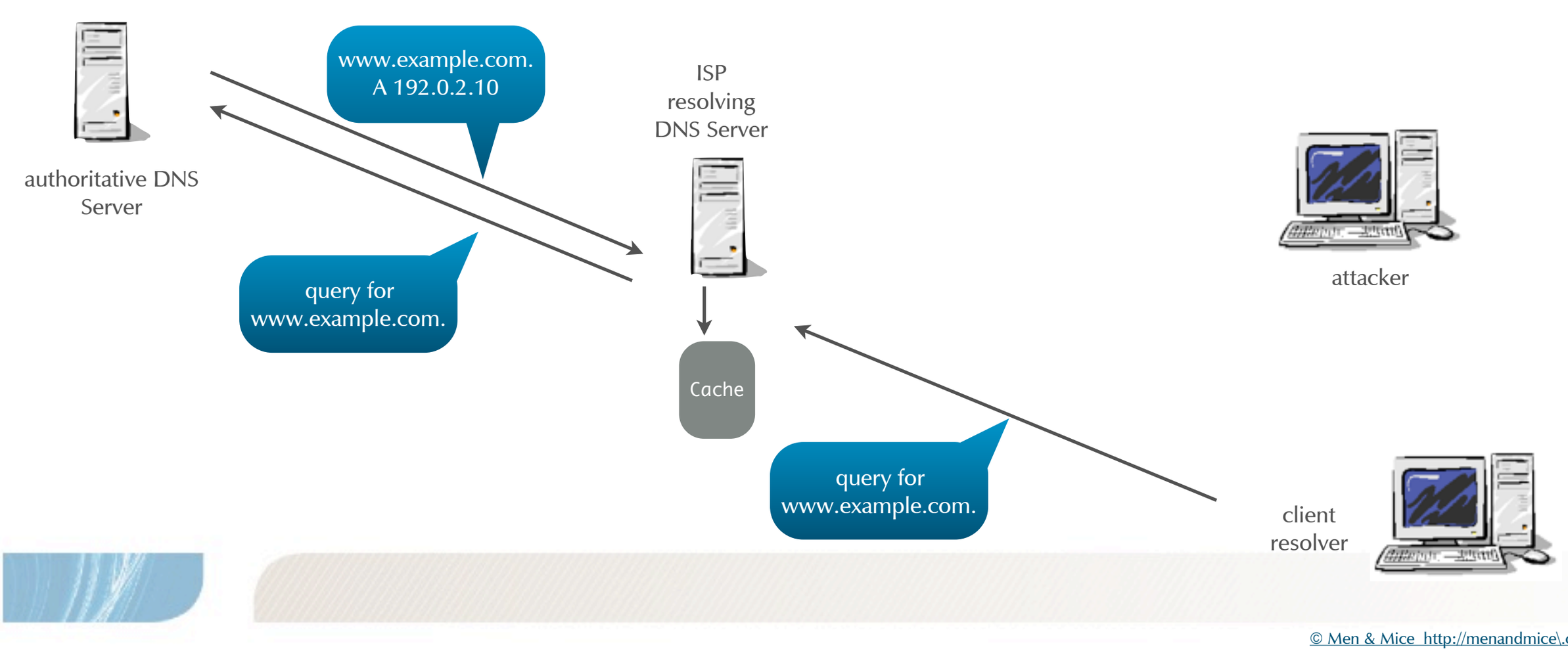
Men in the middle attack

- an attacker en-route can change DNS data unnoticed



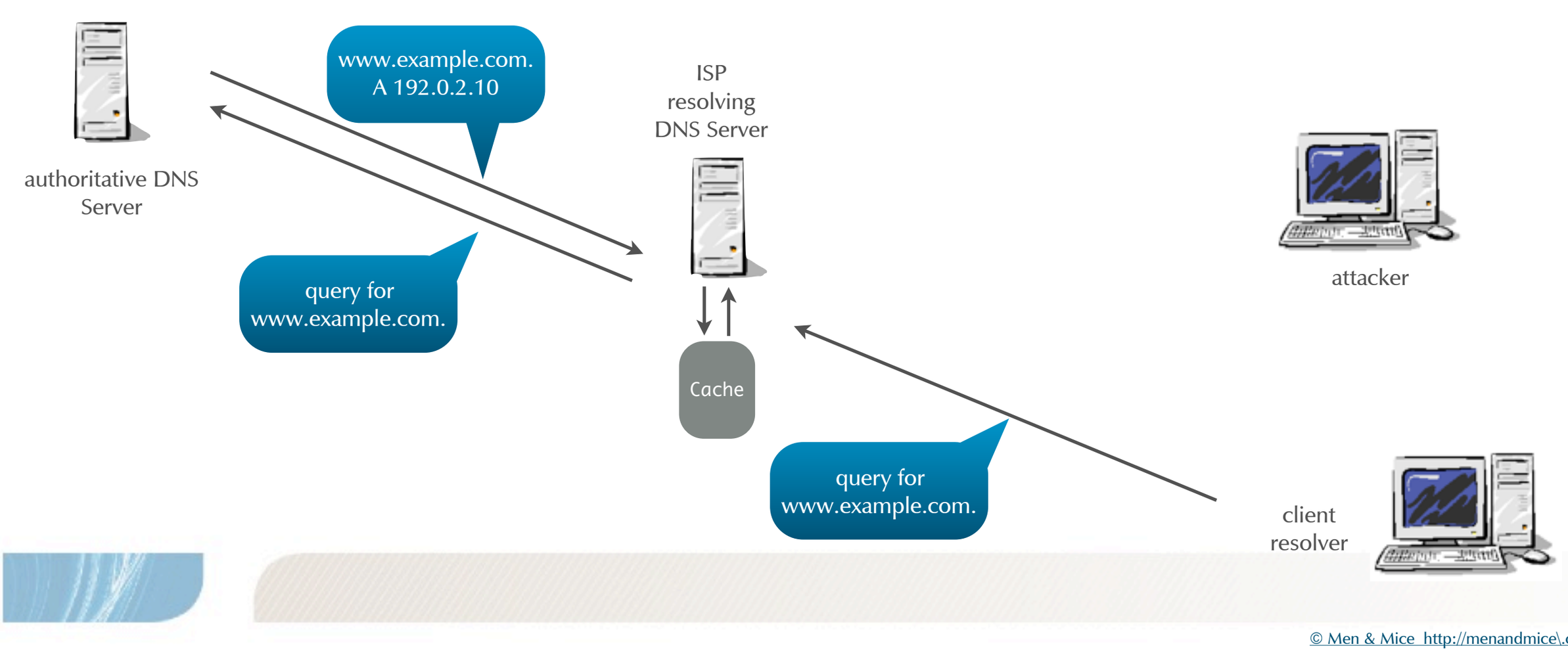
Men in the middle attack

- an attacker en-route can change DNS data unnoticed



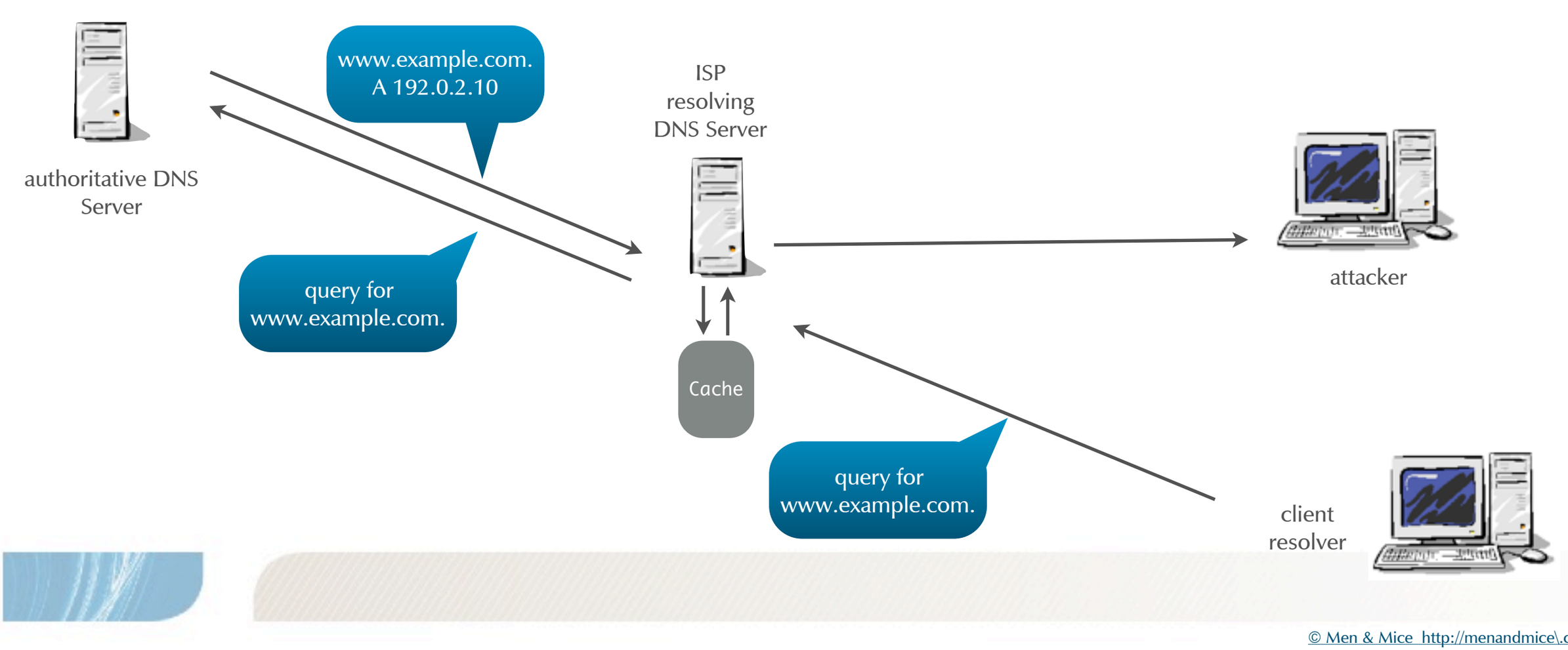
Men in the middle attack

- an attacker en-route can change DNS data unnoticed



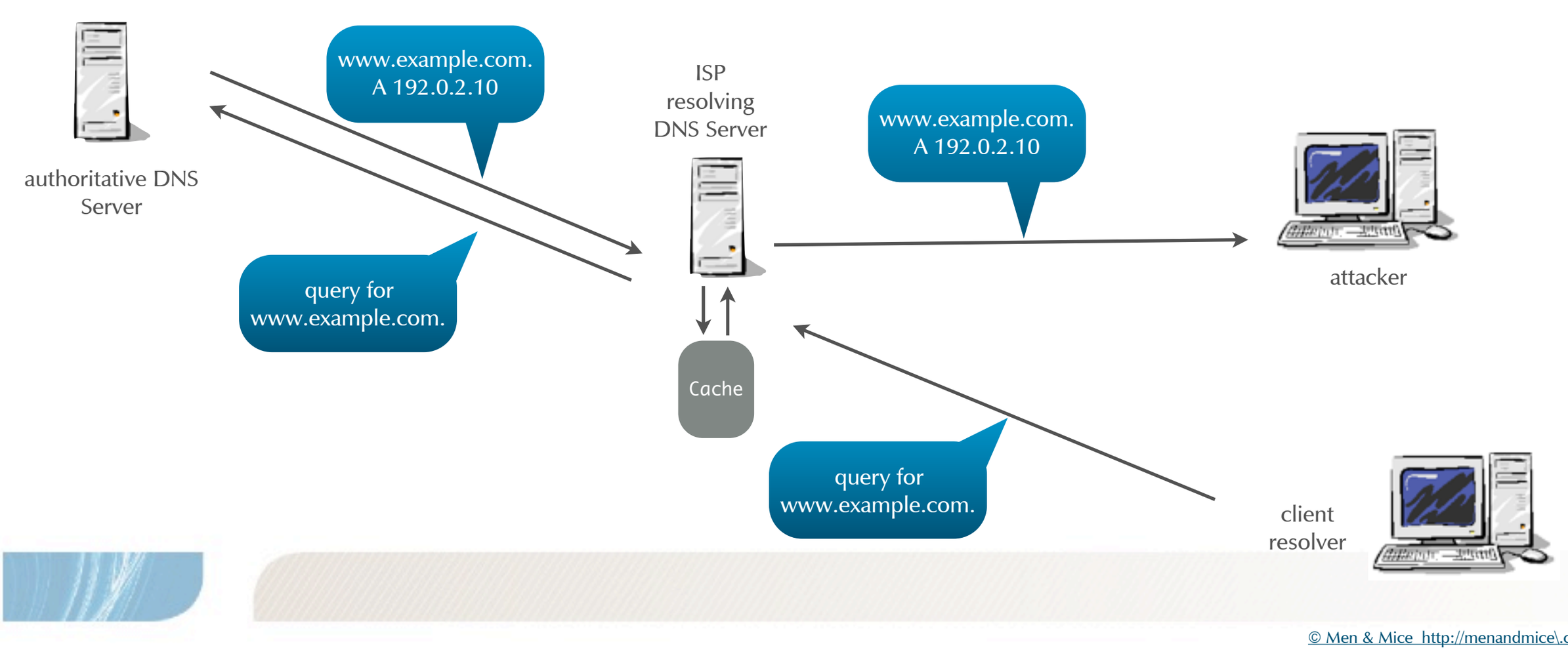
Men in the middle attack

- an attacker en-route can change DNS data unnoticed



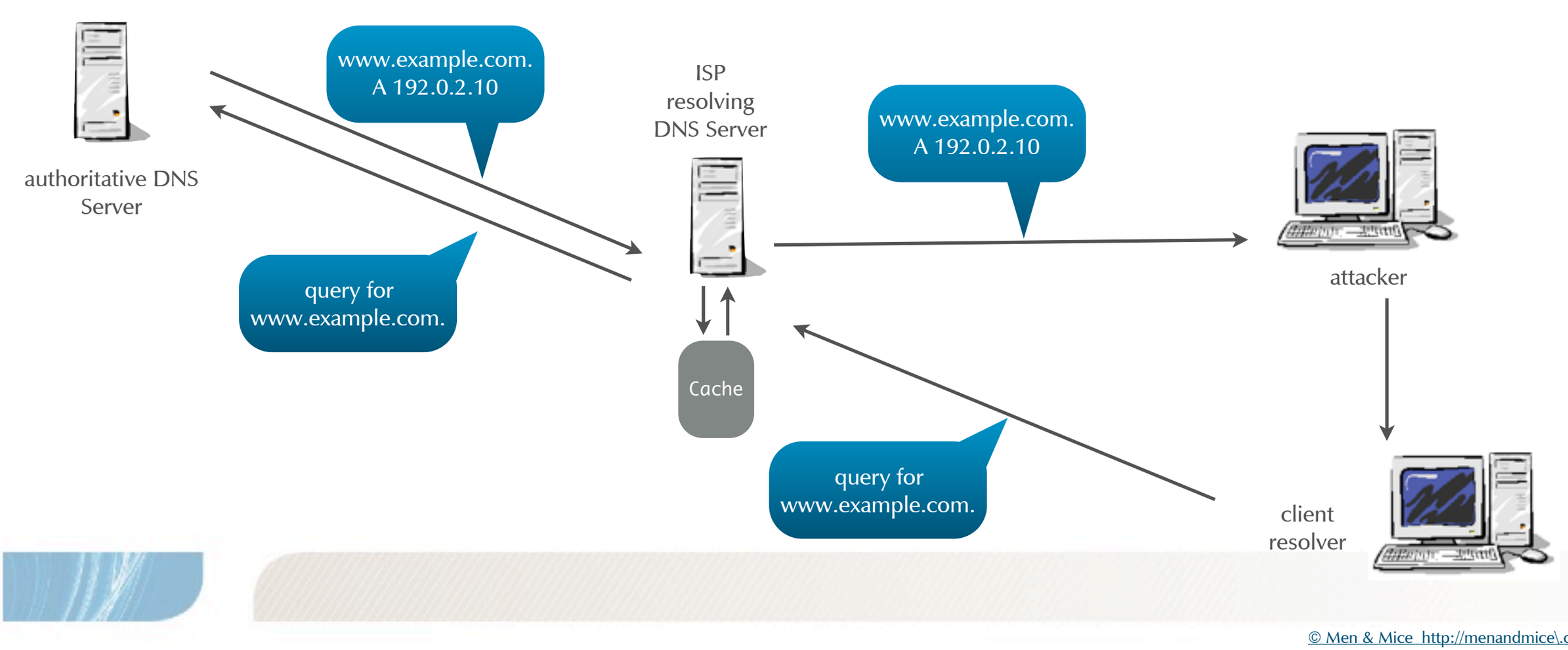
Men in the middle attack

- an attacker en-route can change DNS data unnoticed



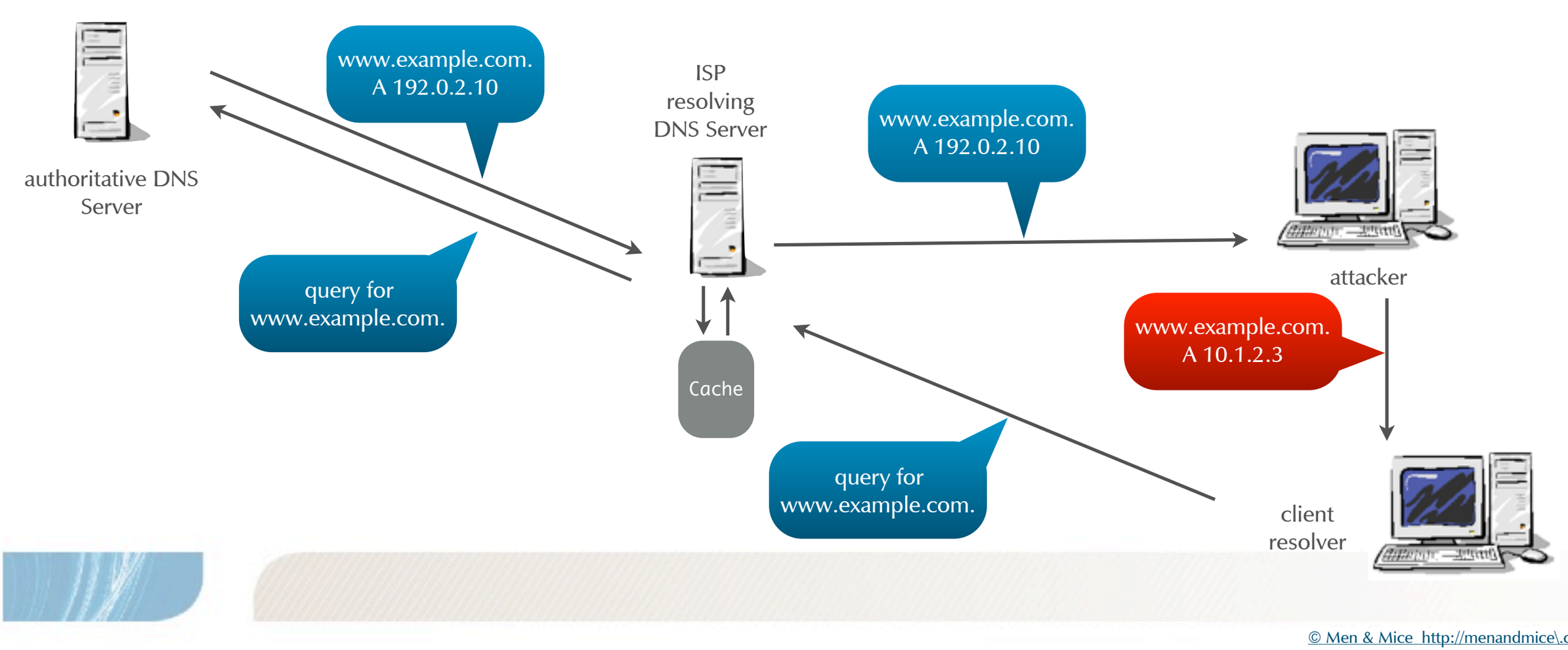
Men in the middle attack

- an attacker en-route can change DNS data unnoticed



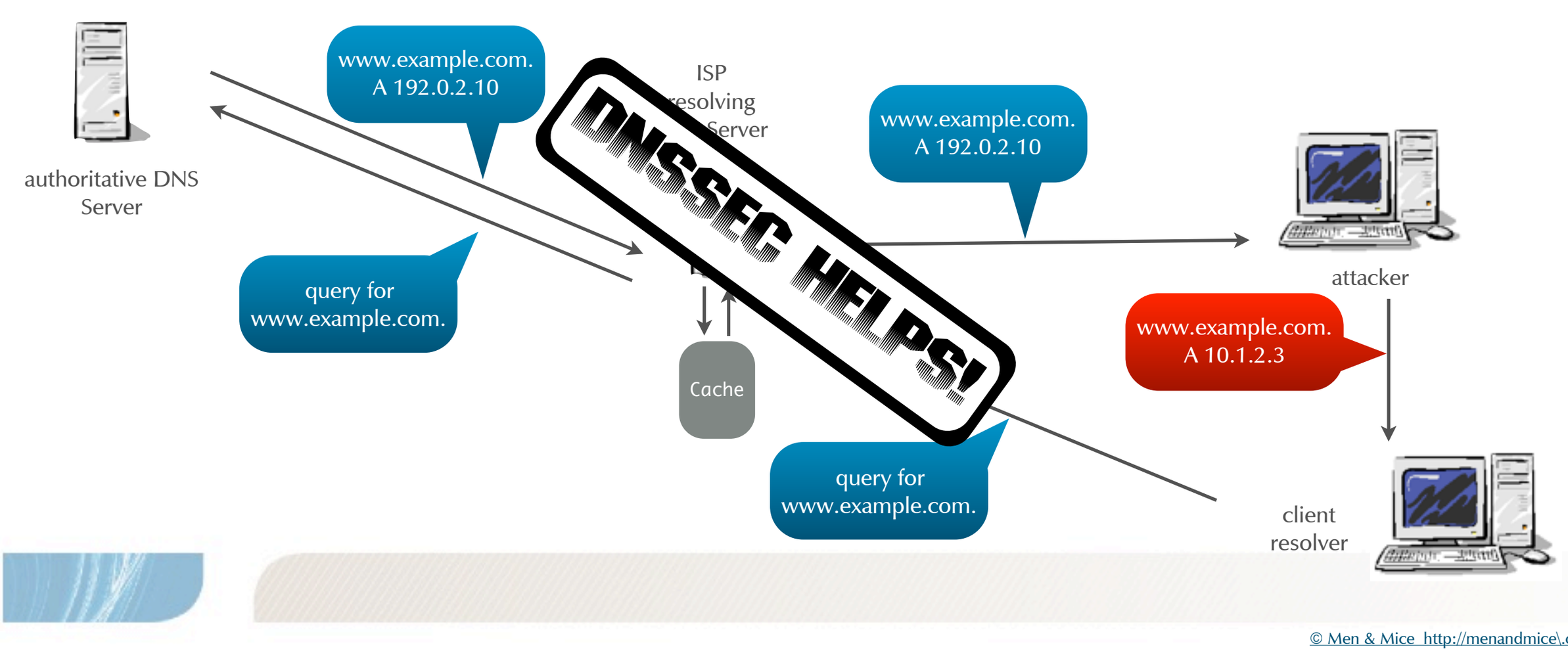
Men in the middle attack

- an attacker en-route can change DNS data unnoticed



Men in the middle attack

- an attacker en-route can change DNS data unnoticed



Betrayal by a trusted name server

- DNS Clients “trust” their local DNS Servers
  - But these DNS Servers can be not-so-trustworthy
    - An attacker can install a rogue DHCP Server and hand out configuration pointing to “pirate” DNS servers
    - An attacker might be able to take over an internal or external caching DNS Server, altering incoming or outgoing data, without anyone noticing (for example in a Hotel Internet Access System)
    - Viruses or Spyware can alter the local resolver configuration...
    - ... or install a small “pirate” DNS Server locally on the client

Betrayal of a trusted name server

- someone in control of an resolving DNS Server has full control over the data returned

authoritative DNS Server

insecure/compromised resolving DNS Server

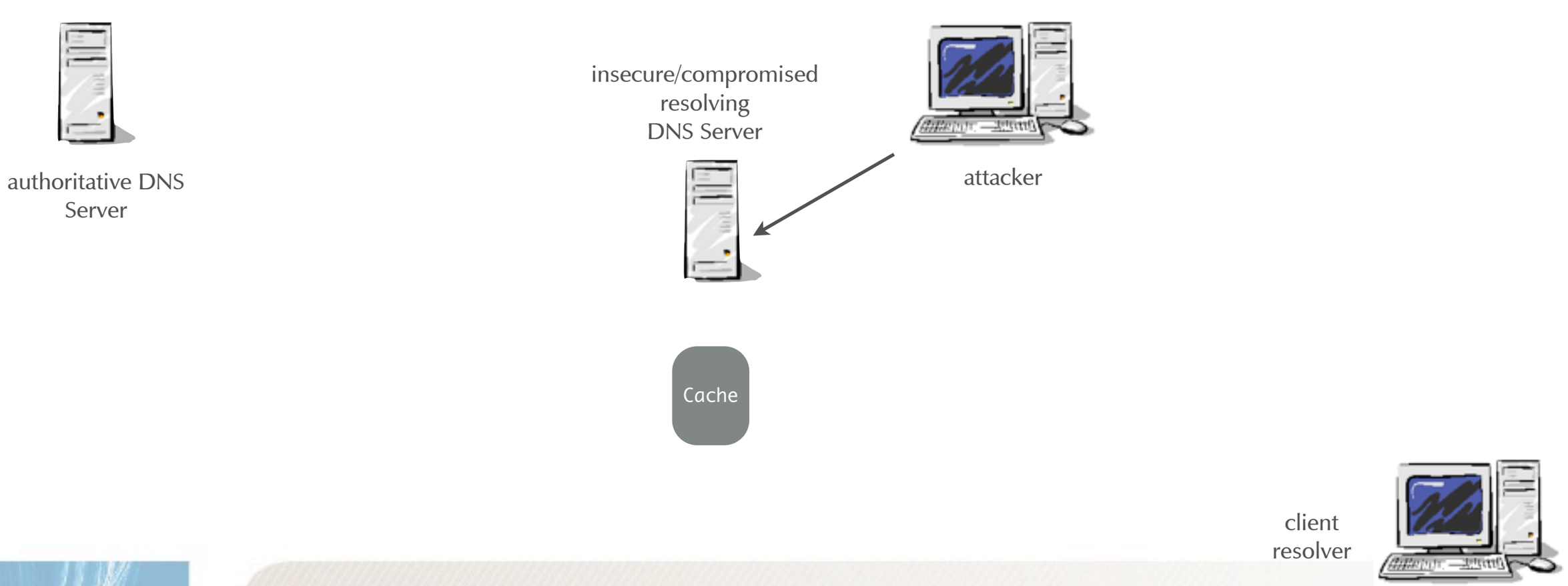
Cache

attacker

client resolver

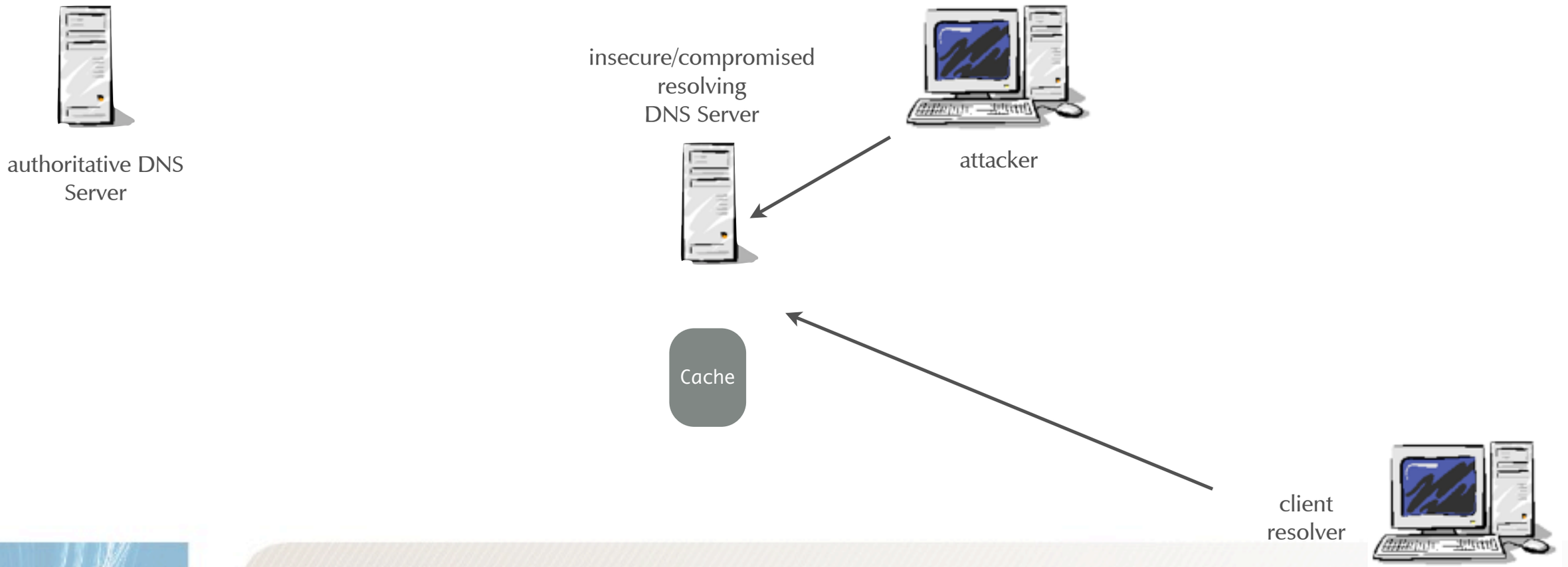
Betrayal of a trusted name server

- someone in control of an resolving DNS Server has full control over the data returned



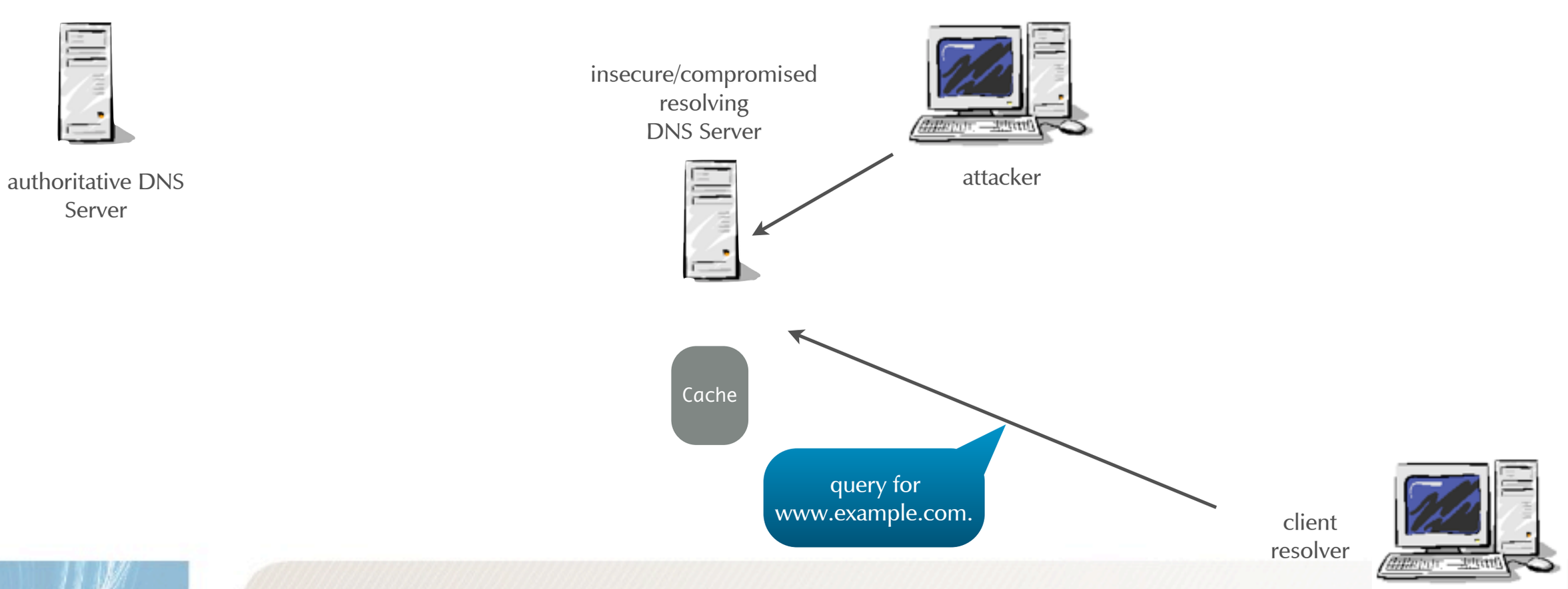
Betrayal of a trusted name server

- someone in control of an resolving DNS Server has full control over the data returned



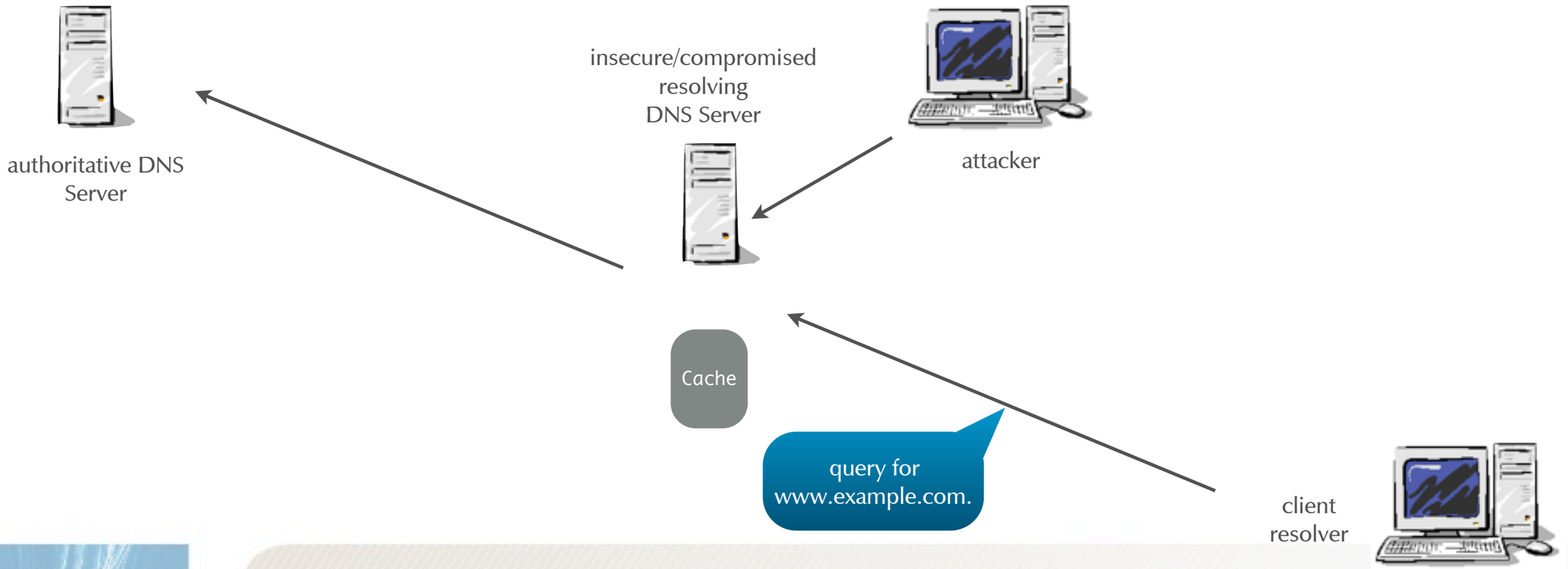
Betrayal of a trusted name server

- someone in control of an resolving DNS Server has full control over the data returned



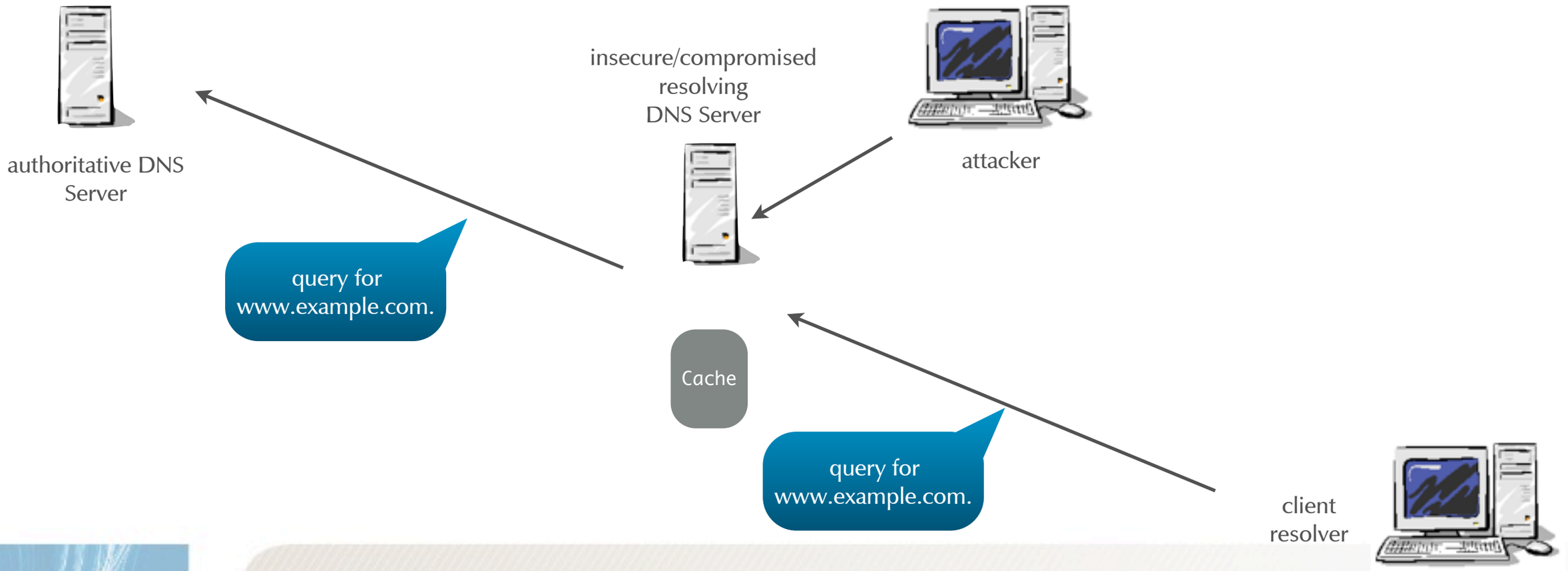
Betrayal of a trusted name server

- someone in control of an resolving DNS Server has full control over the data returned



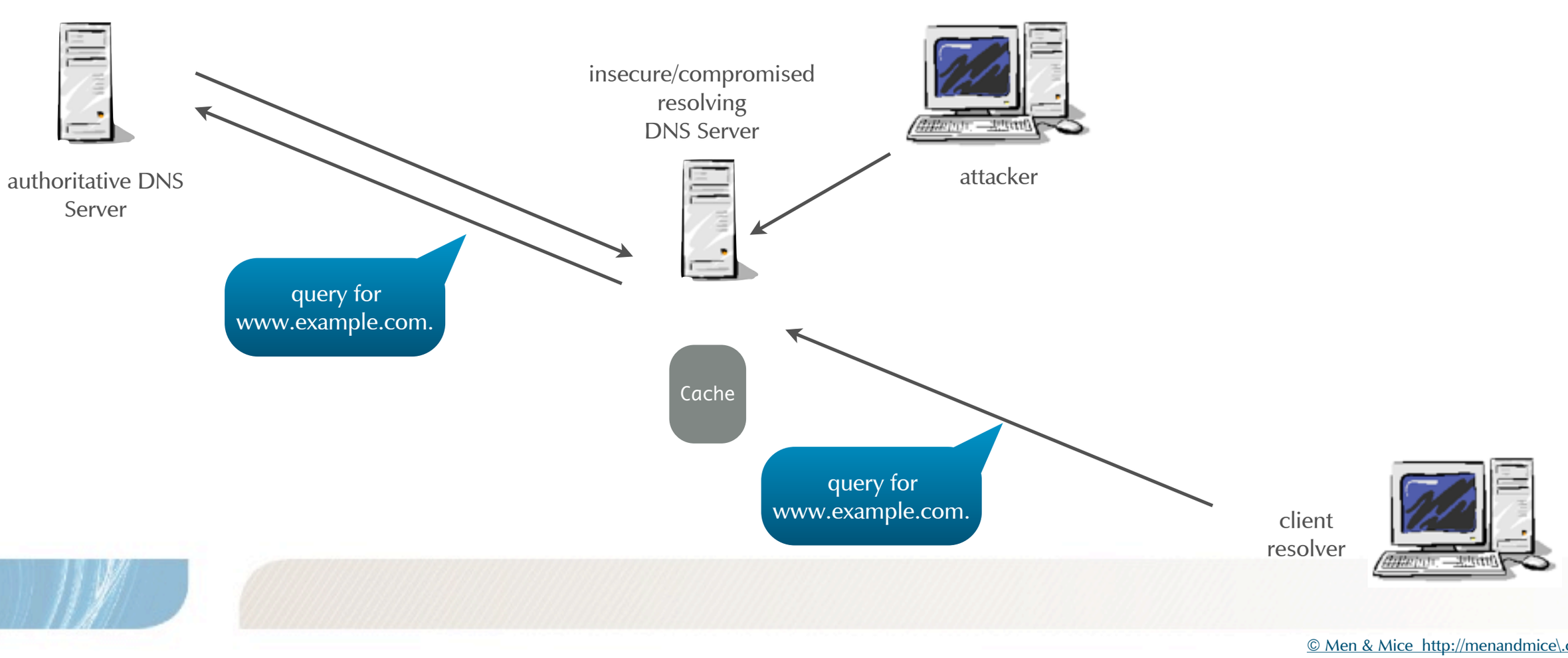
Betrayal of a trusted name server

- someone in control of an resolving DNS Server has full control over the data returned



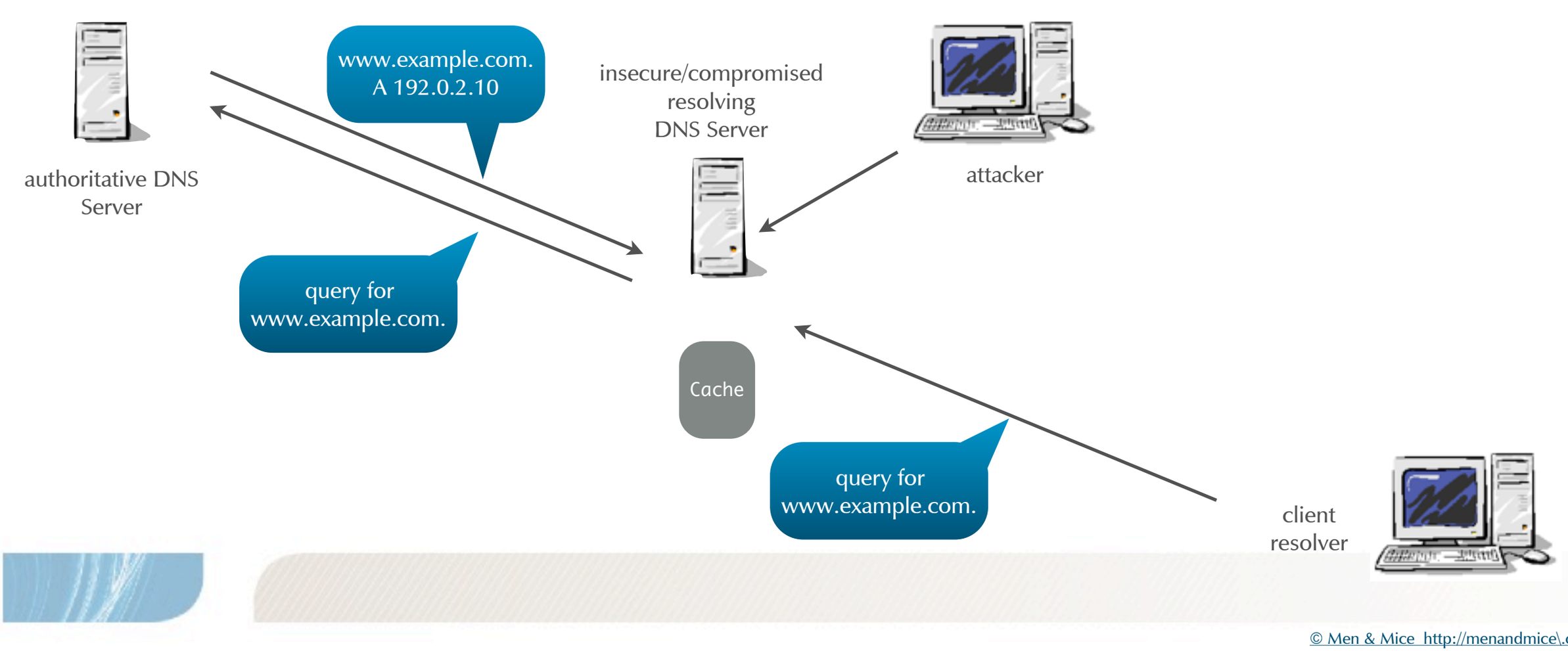
Betrayal of a trusted name server

- someone in control of an resolving DNS Server has full control over the data returned



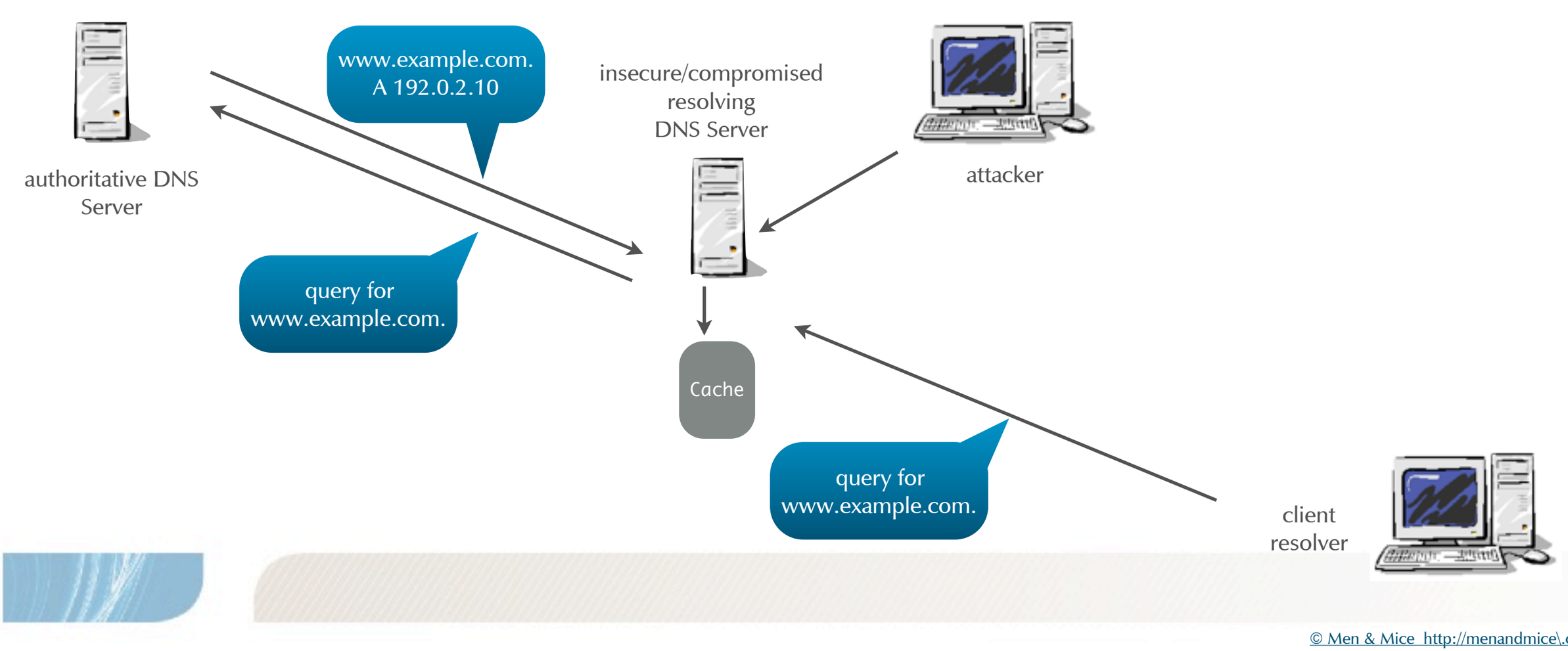
Betrayal of a trusted name server

- someone in control of an resolving DNS Server has full control over the data returned



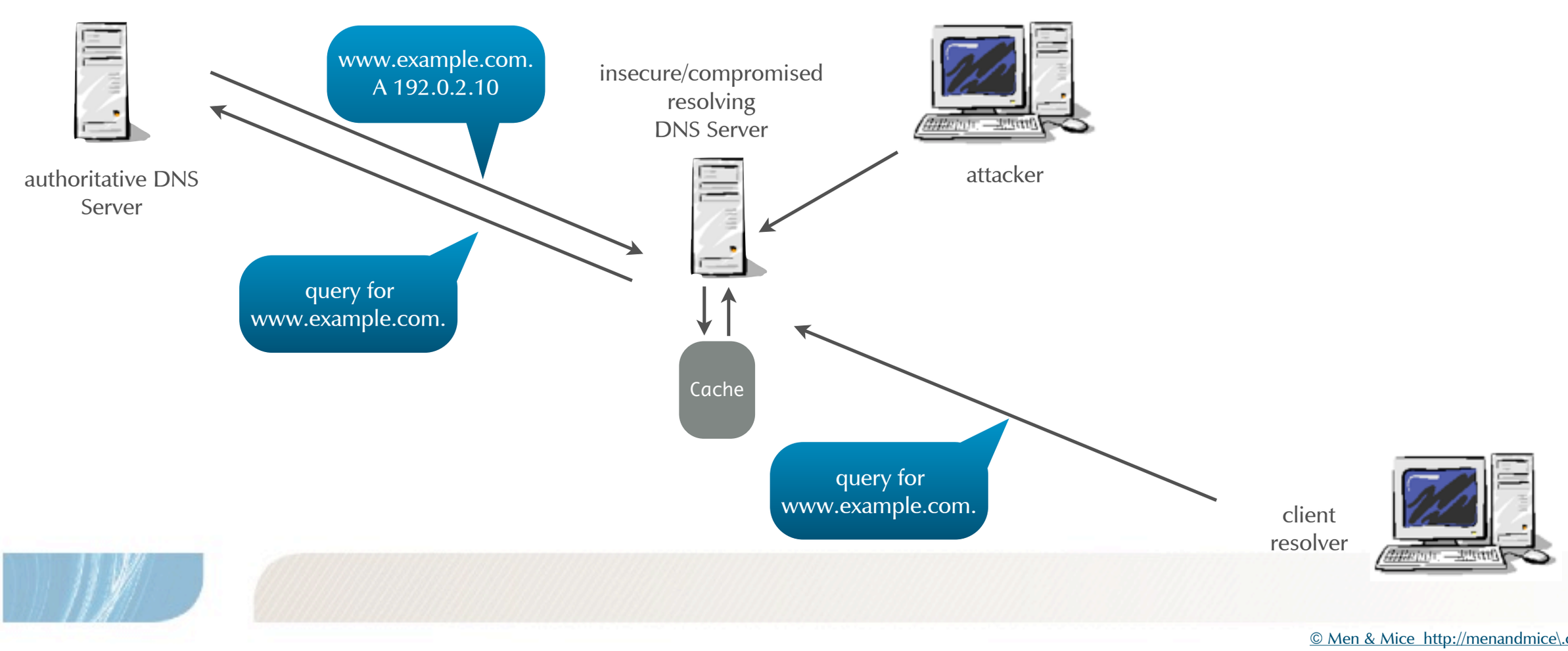
Betrayal of a trusted name server

- someone in control of an resolving DNS Server has full control over the data returned



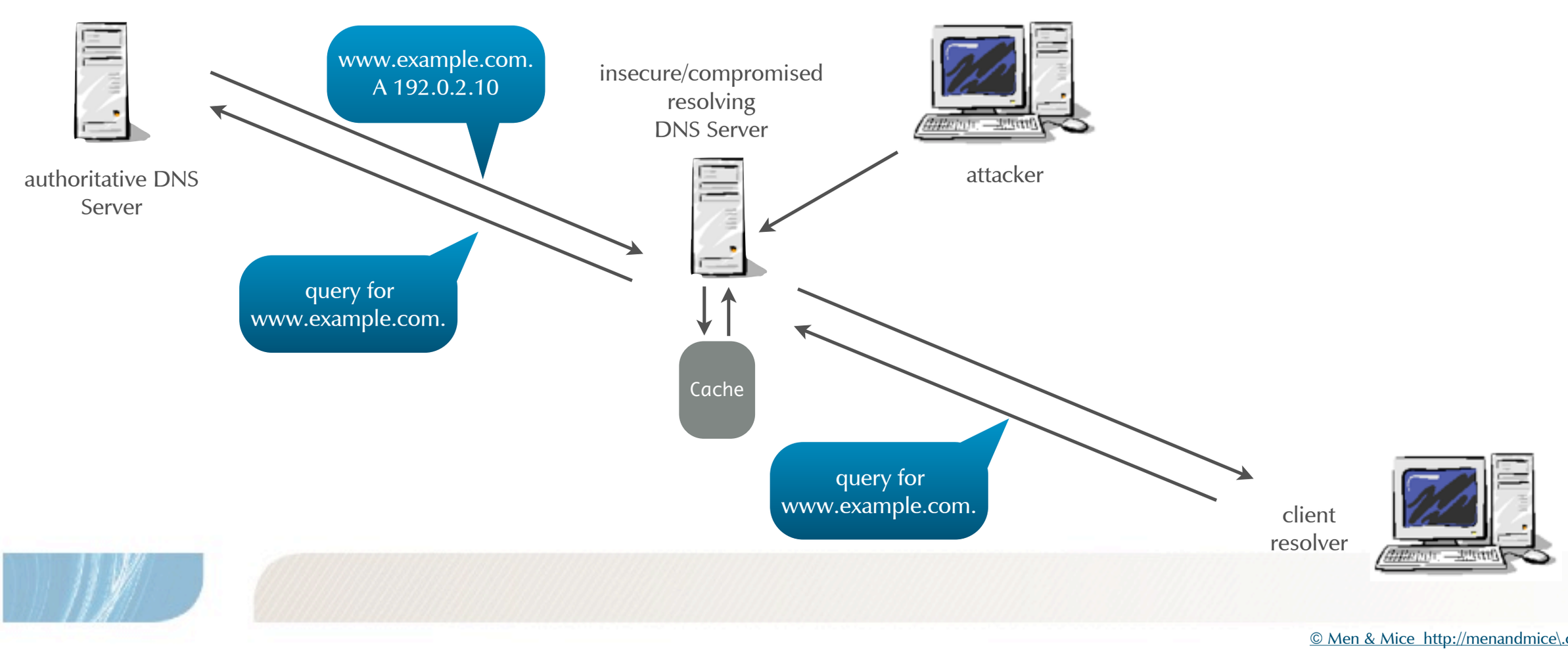
Betrayal of a trusted name server

- someone in control of an resolving DNS Server has full control over the data returned



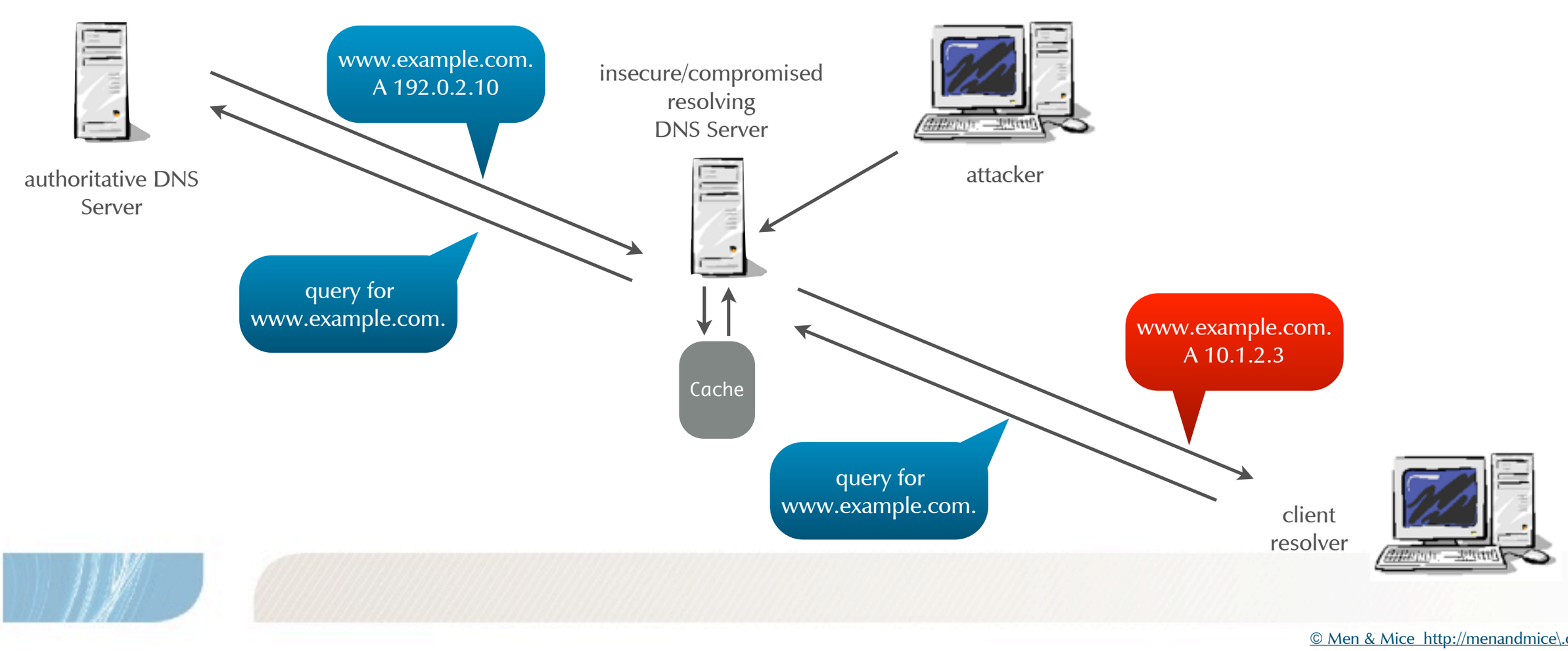
Betrayal of a trusted name server

- someone in control of an resolving DNS Server has full control over the data returned



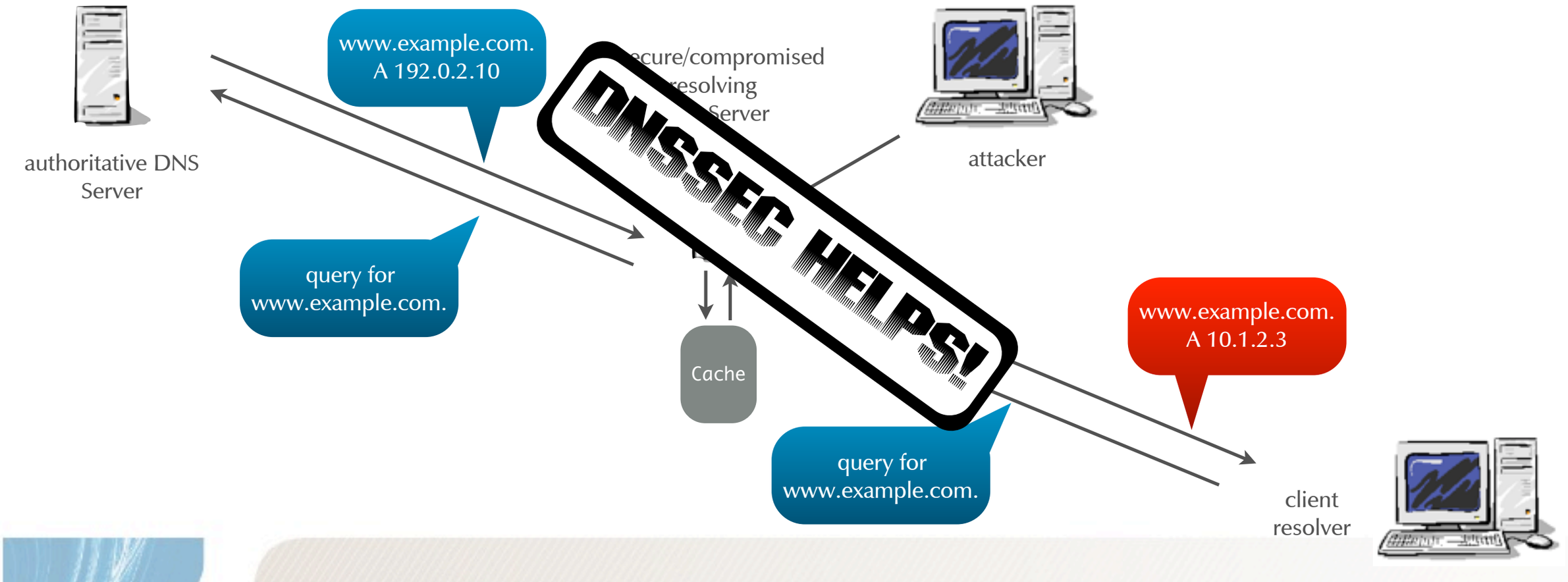
Betrayal of a trusted name server

- someone in control of an resolving DNS Server has full control over the data returned



Betrayal of a trusted name server

- someone in control of an resolving DNS Server has full control over the data returned



Betrayal of a trusted name server

- someone in control of an resolving DNS Server has full control over the data returned

authoritative DNS Server

ISP resolving DNS Server

Cache

client resolver

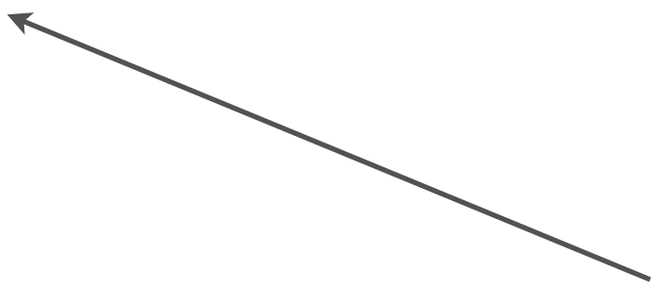
Betrayal of a trusted name server

- someone in control of an resolving DNS Server has full control over the data returned

authoritative DNS Server

ISP resolving DNS Server

Cache



client resolver

Betrayal of a trusted name server

- someone in control of an resolving DNS Server has full control over the data returned

authoritative DNS Server

ISP resolving DNS Server

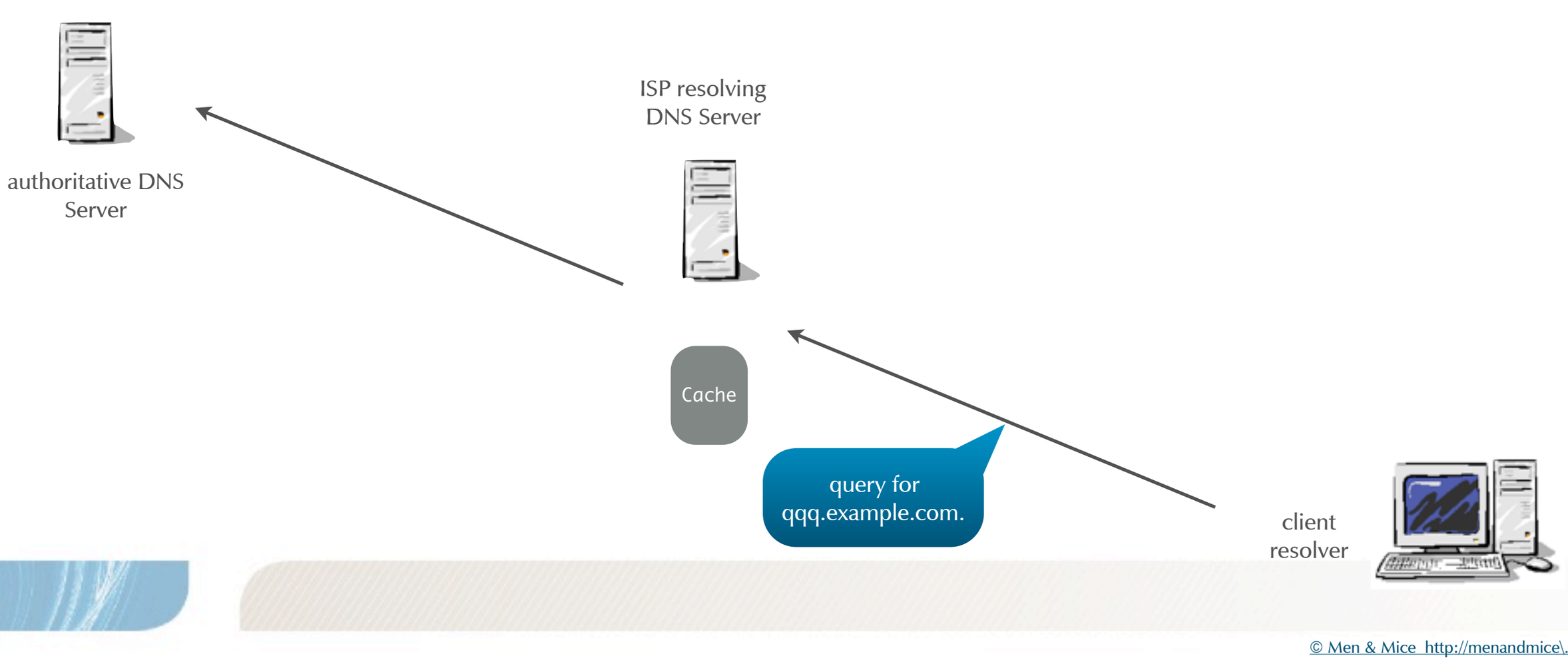
Cache

query for qqq.example.com.

client resolver

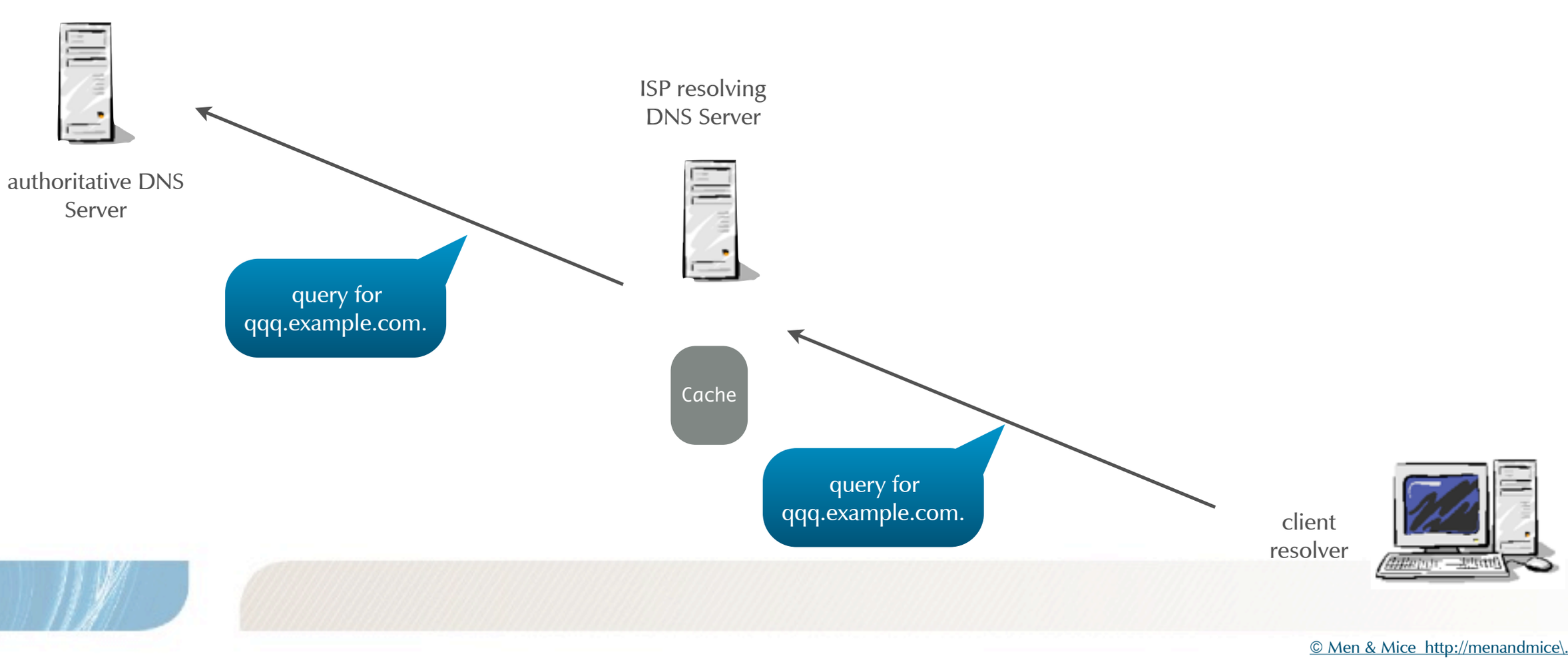
Betrayal of a trusted name server

- someone in control of an resolving DNS Server has full control over the data returned



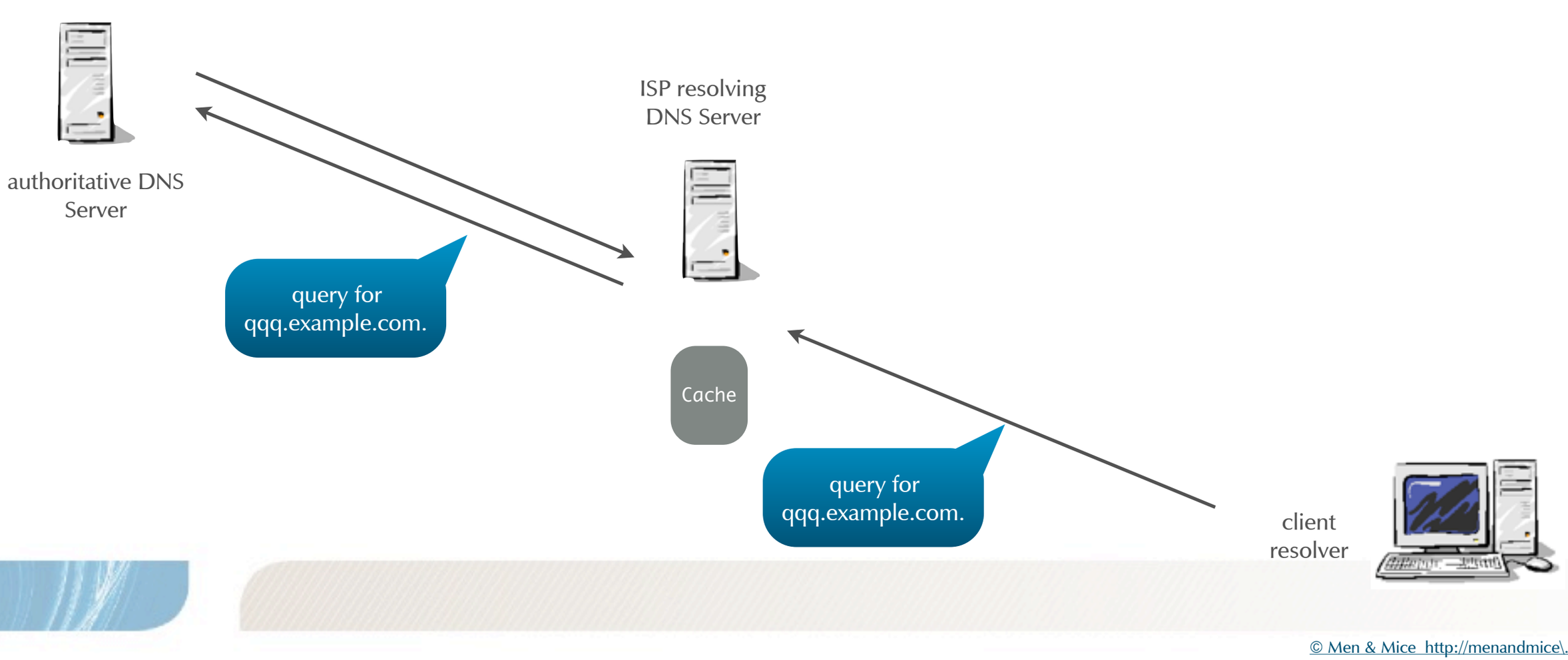
Betrayal of a trusted name server

- someone in control of an resolving DNS Server has full control over the data returned



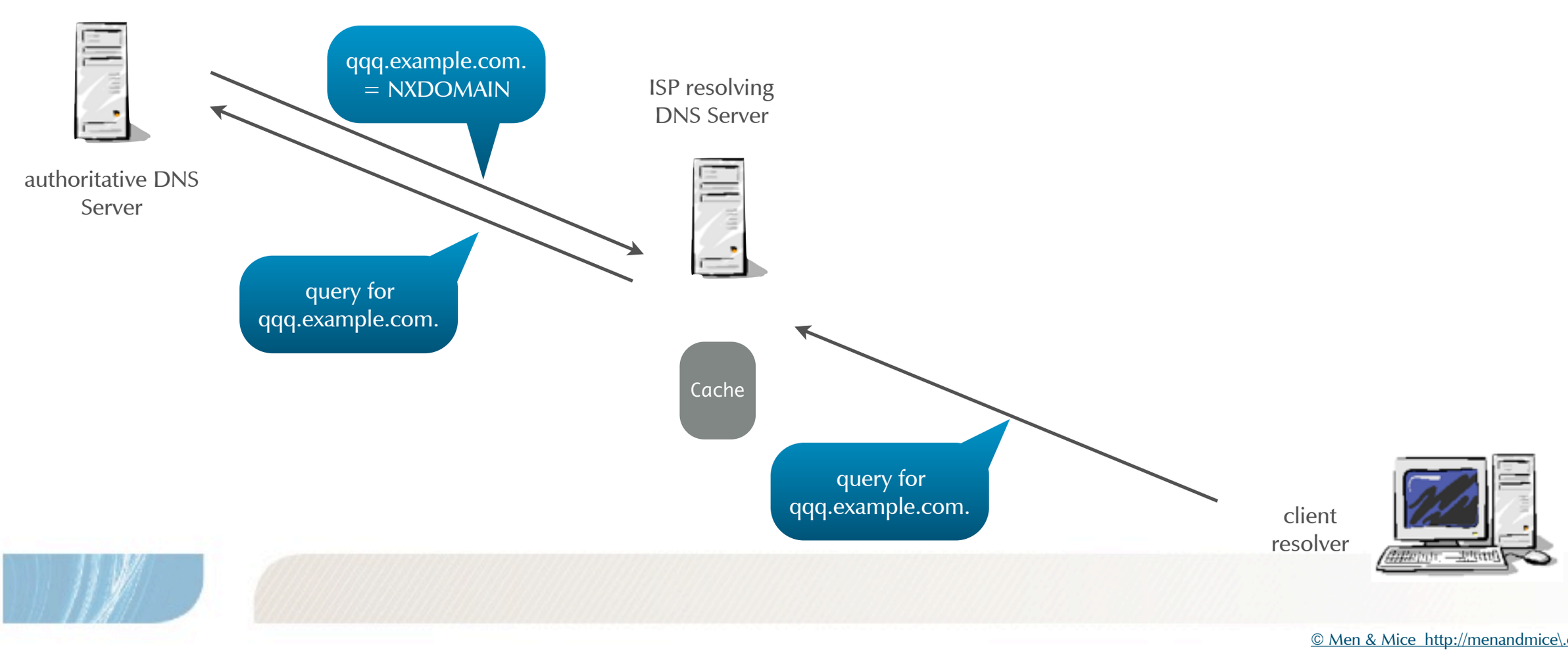
Betrayal of a trusted name server

- someone in control of an resolving DNS Server has full control over the data returned



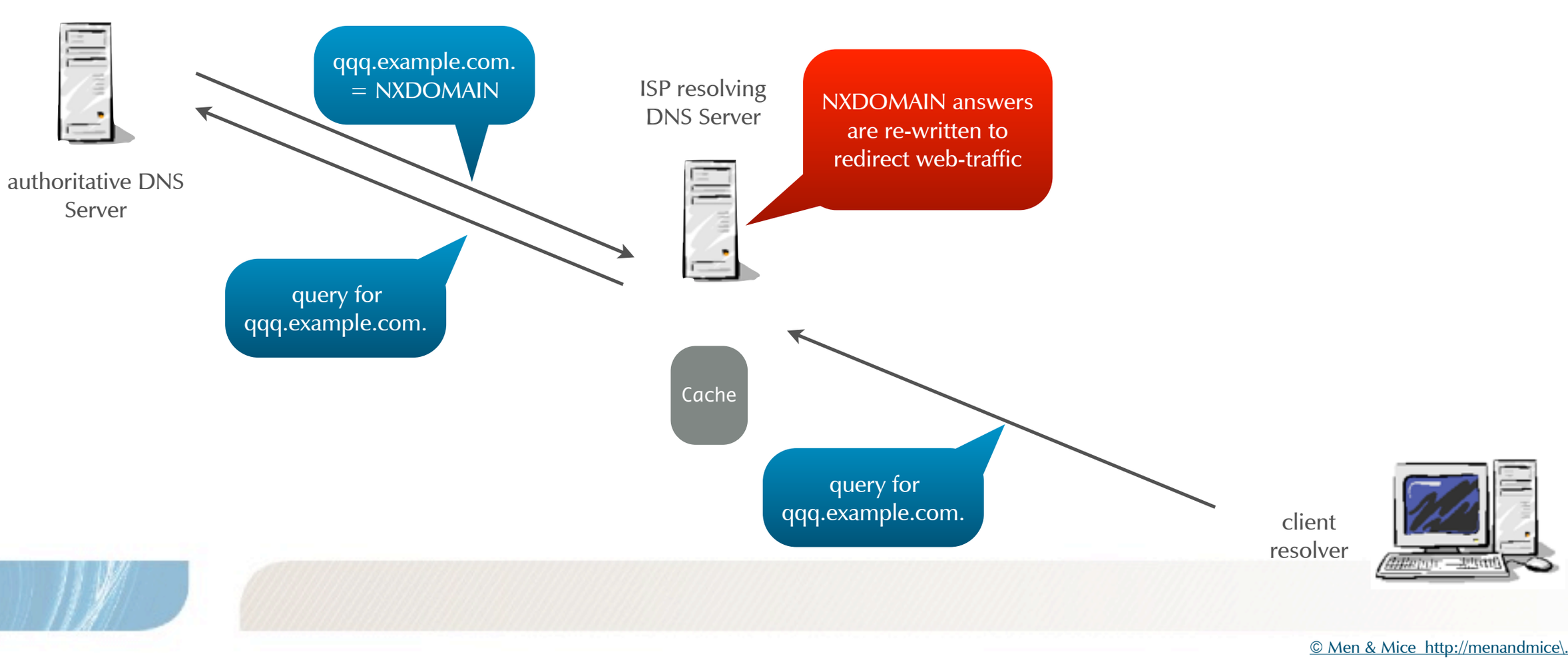
Betrayal of a trusted name server

- someone in control of an resolving DNS Server has full control over the data returned



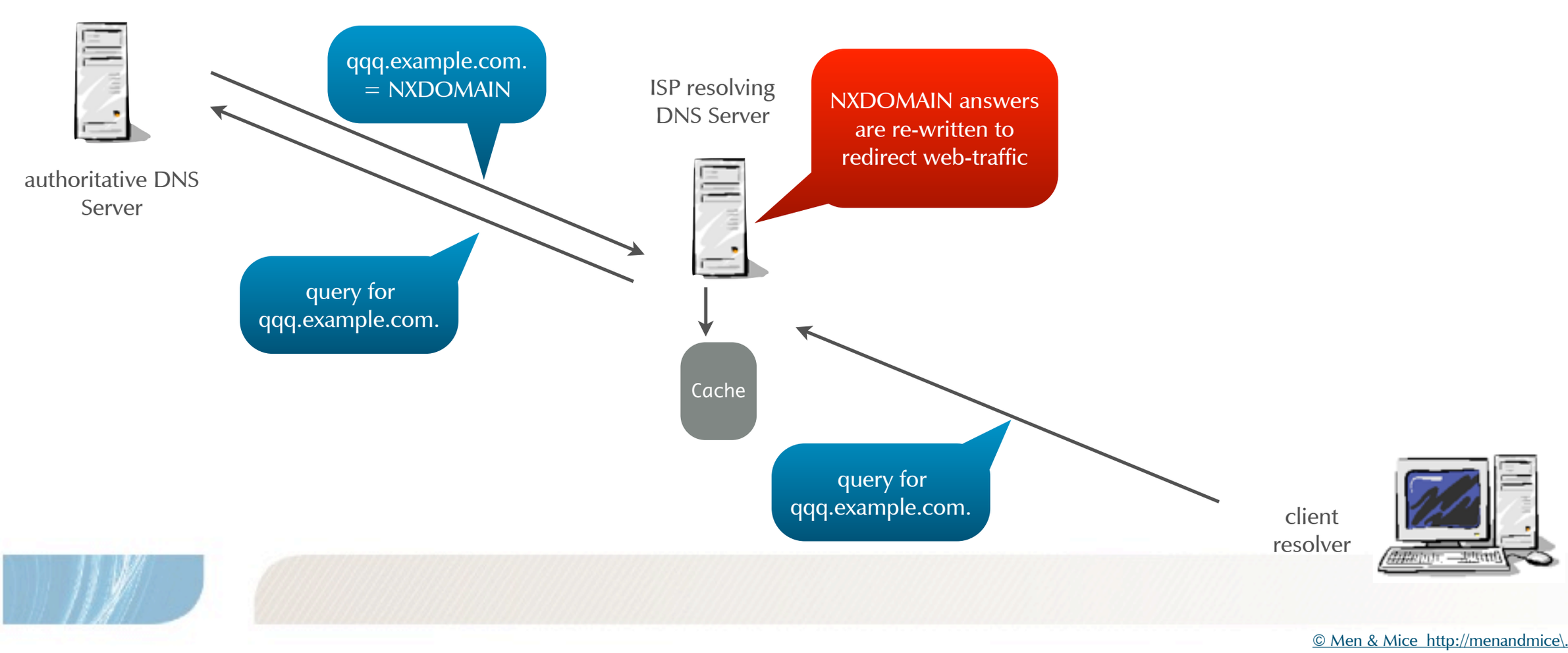
Betrayal of a trusted name server

- someone in control of an resolving DNS Server has full control over the data returned



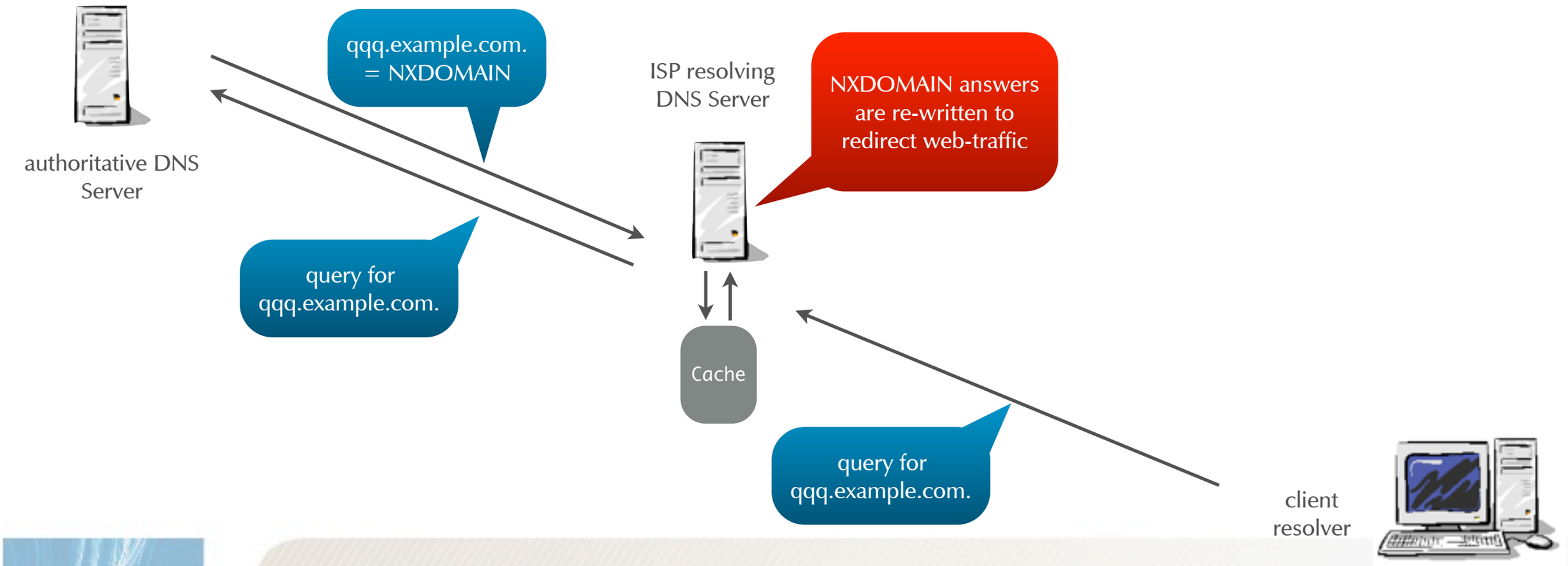
Betrayal of a trusted name server

- someone in control of an resolving DNS Server has full control over the data returned



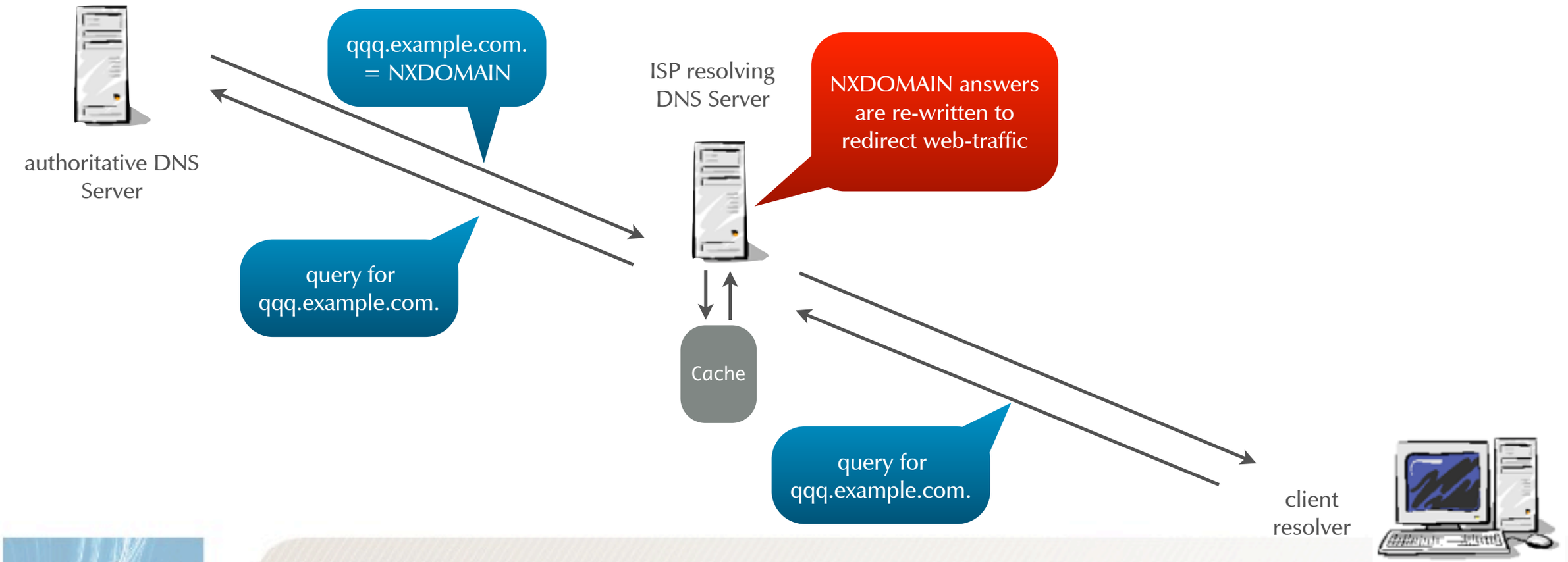
Betrayal of a trusted name server

- someone in control of an resolving DNS Server has full control over the data returned



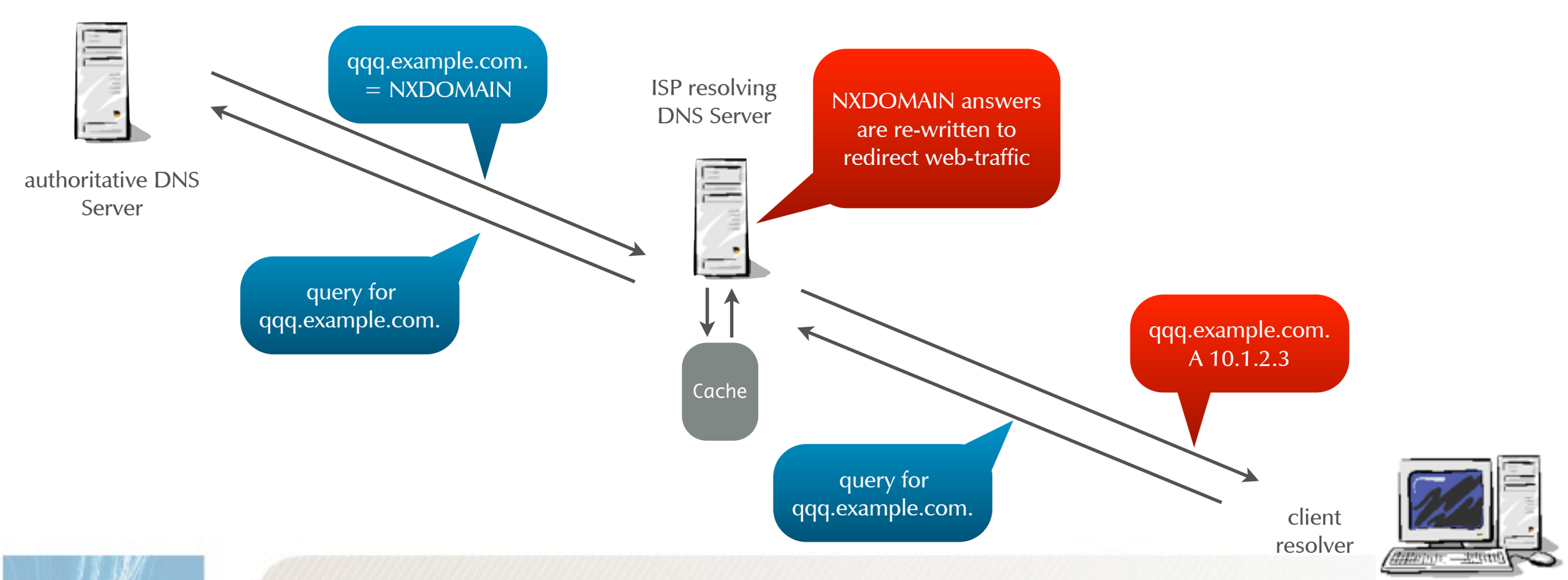
Betrayal of a trusted name server

- someone in control of an resolving DNS Server has full control over the data returned



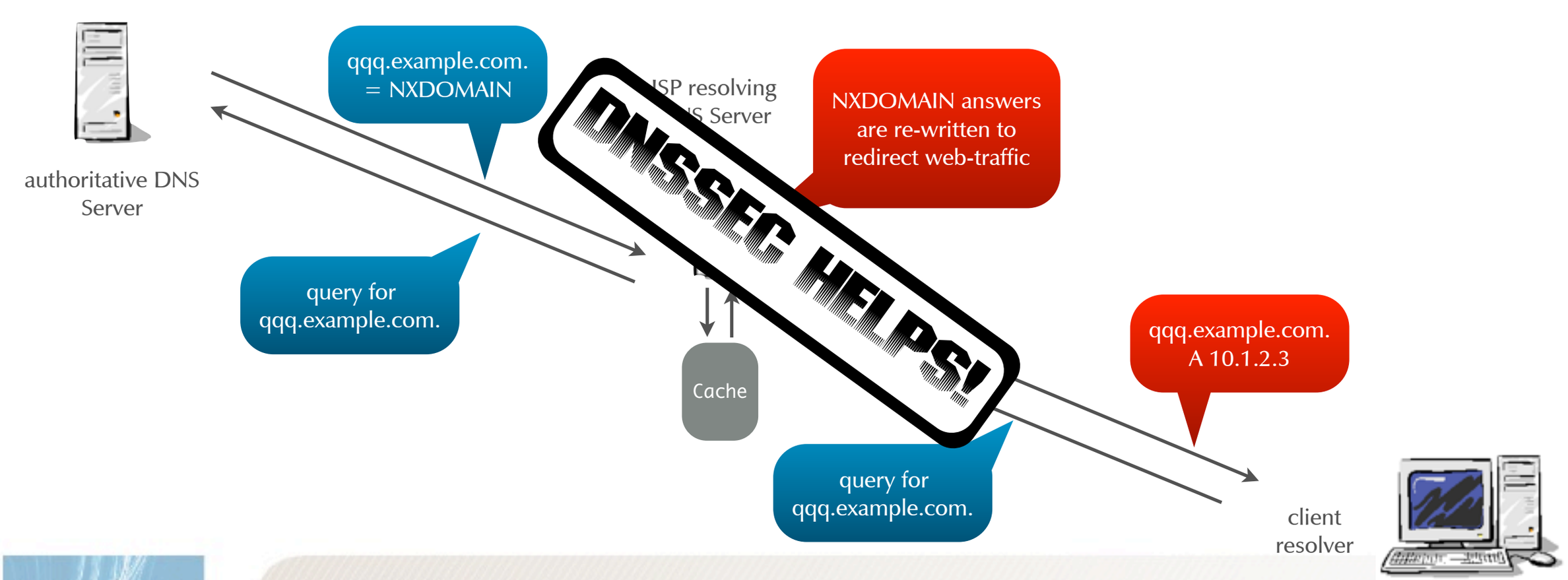
Betrayal of a trusted name server

- someone in control of an resolving DNS Server has full control over the data returned



Betrayal of a trusted name server

- someone in control of an resolving DNS Server has full control over the data returned



attacker changes the local resolver settings

- the local resolver settings are changed without the client user noticing, returning bad data

authoritative DNS Server

attackers resolving DNS Server

attacker

ISP/company resolving DNS Server

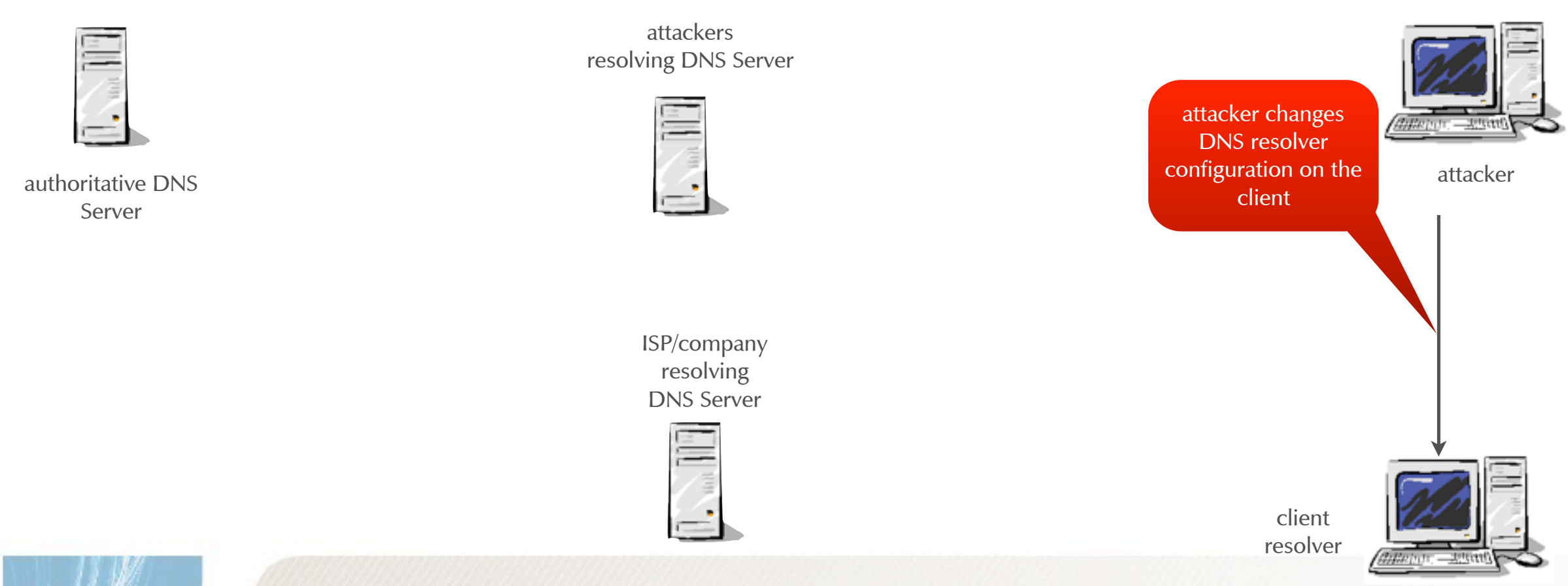
client resolver

attacker changes the local resolver settings

- the local resolver settings are changed without the client user noticing, returning bad data

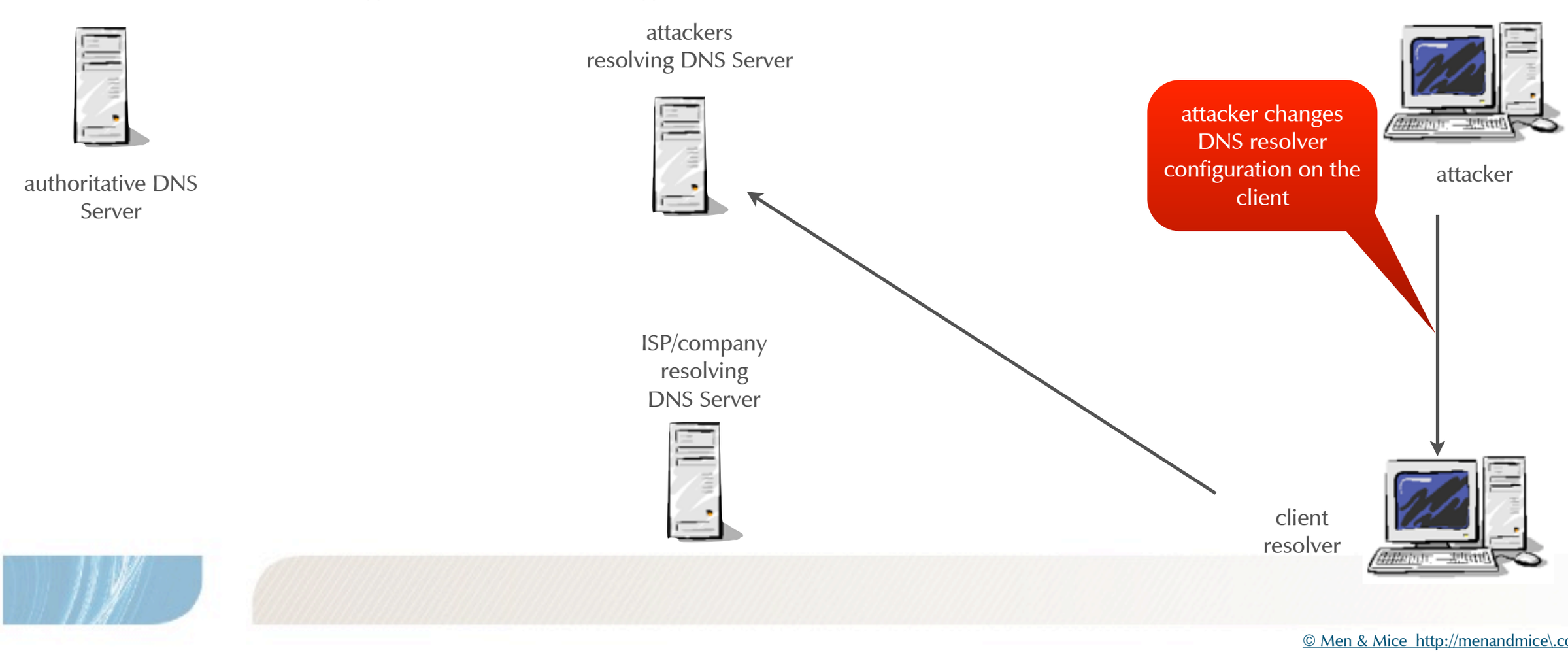
attacker changes the local resolver settings

- the local resolver settings are changed without the client user noticing, returning bad data



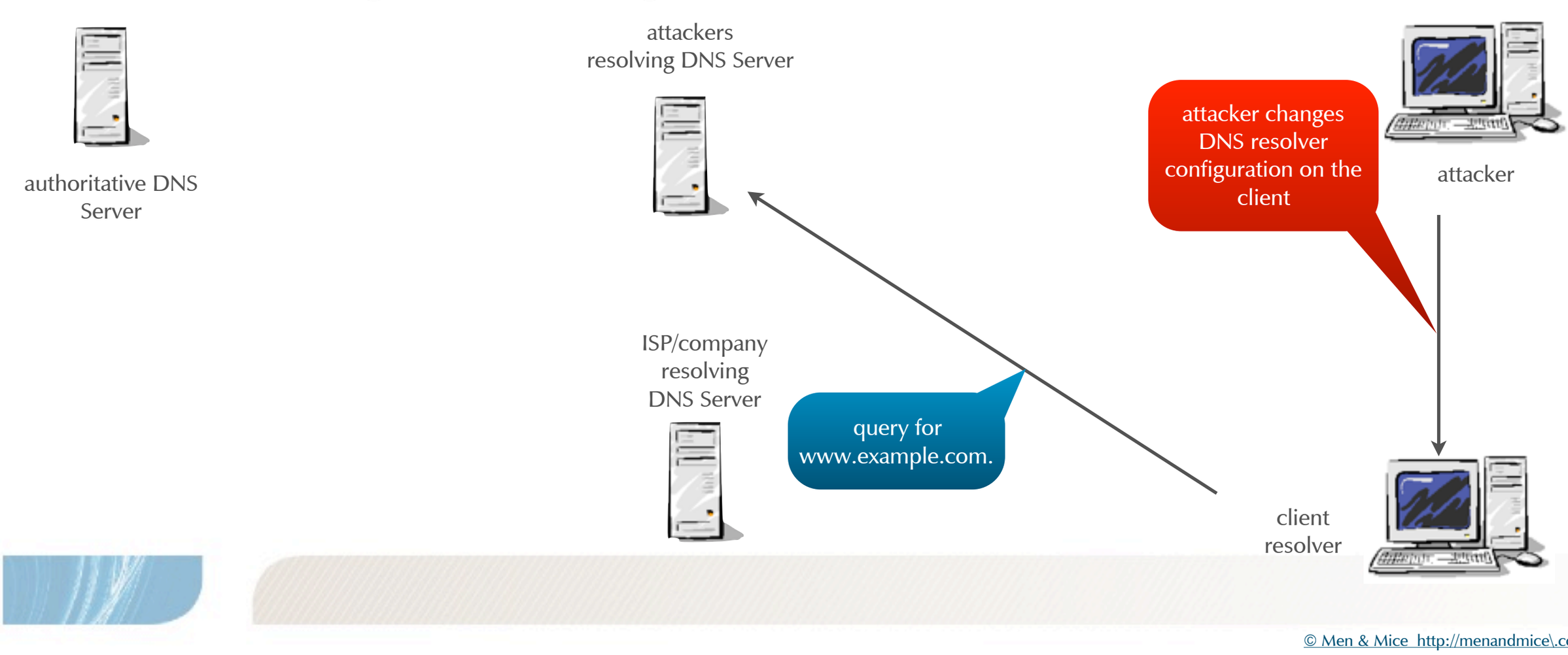
attacker changes the local resolver settings

- the local resolver settings are changed without the client user noticing, returning bad data



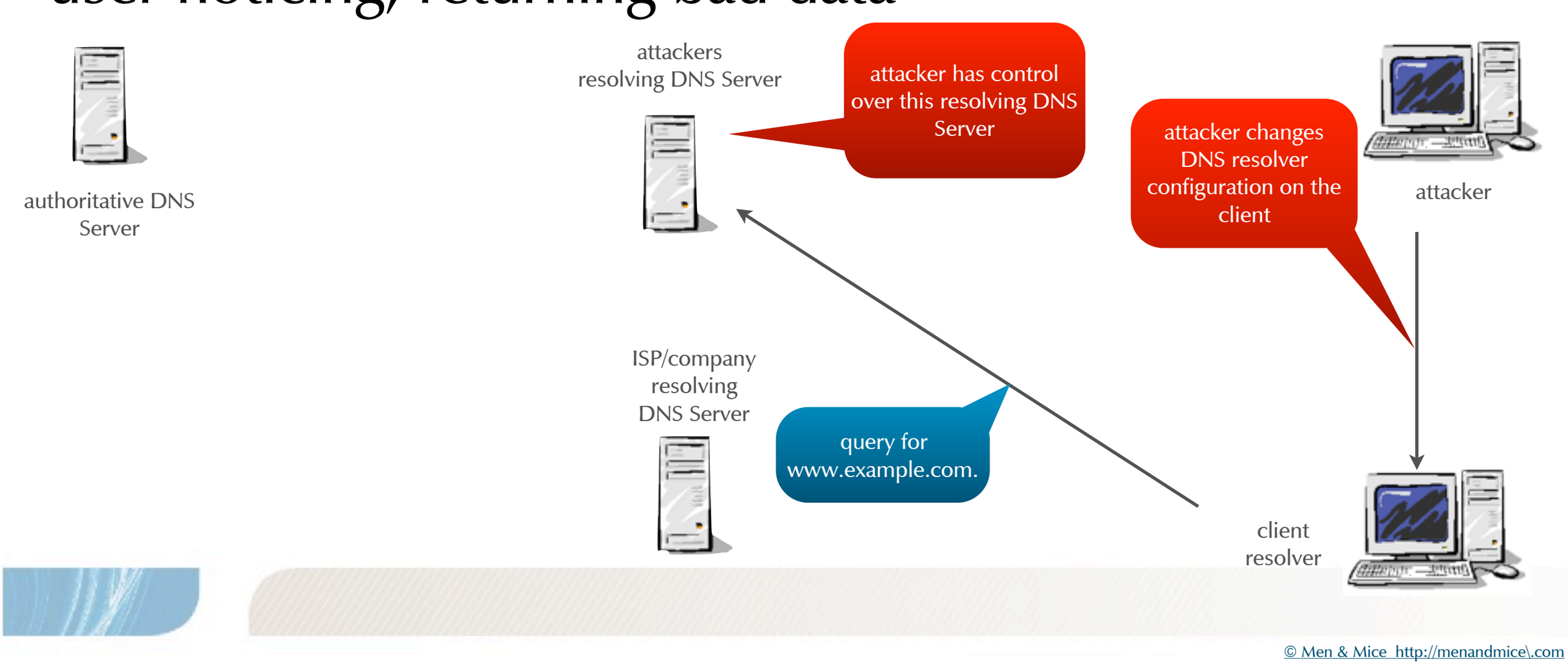
attacker changes the local resolver settings

- the local resolver settings are changed without the client user noticing, returning bad data



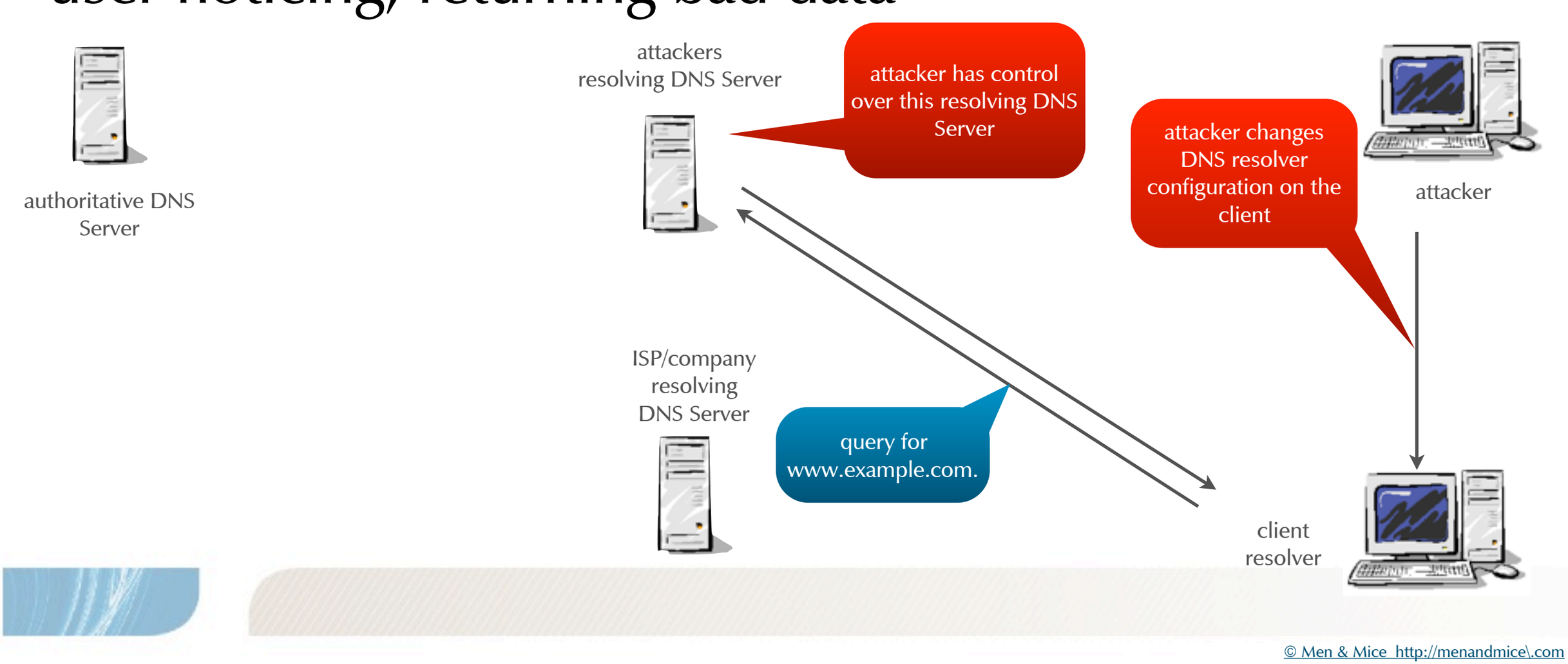
attacker changes the local resolver settings

- the local resolver settings are changed without the client user noticing, returning bad data



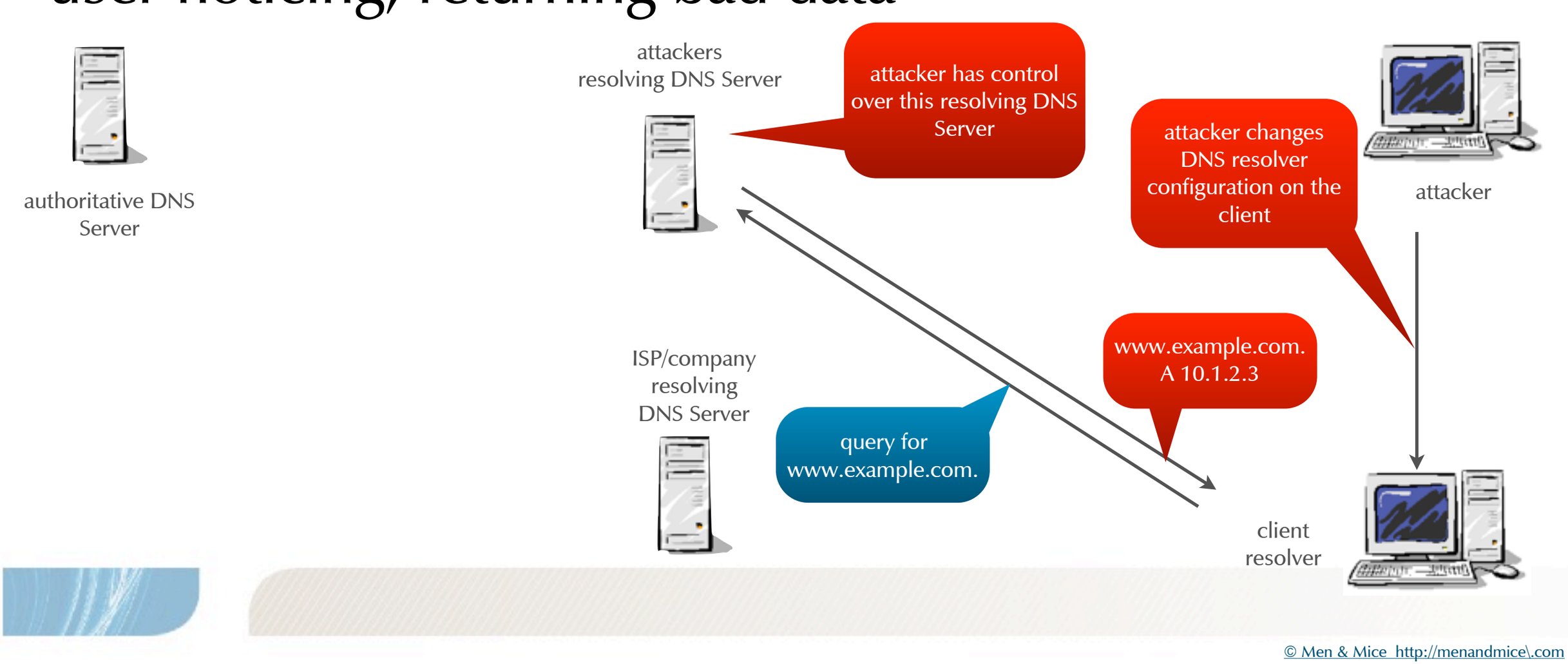
attacker changes the local resolver settings

- the local resolver settings are changed without the client user noticing, returning bad data



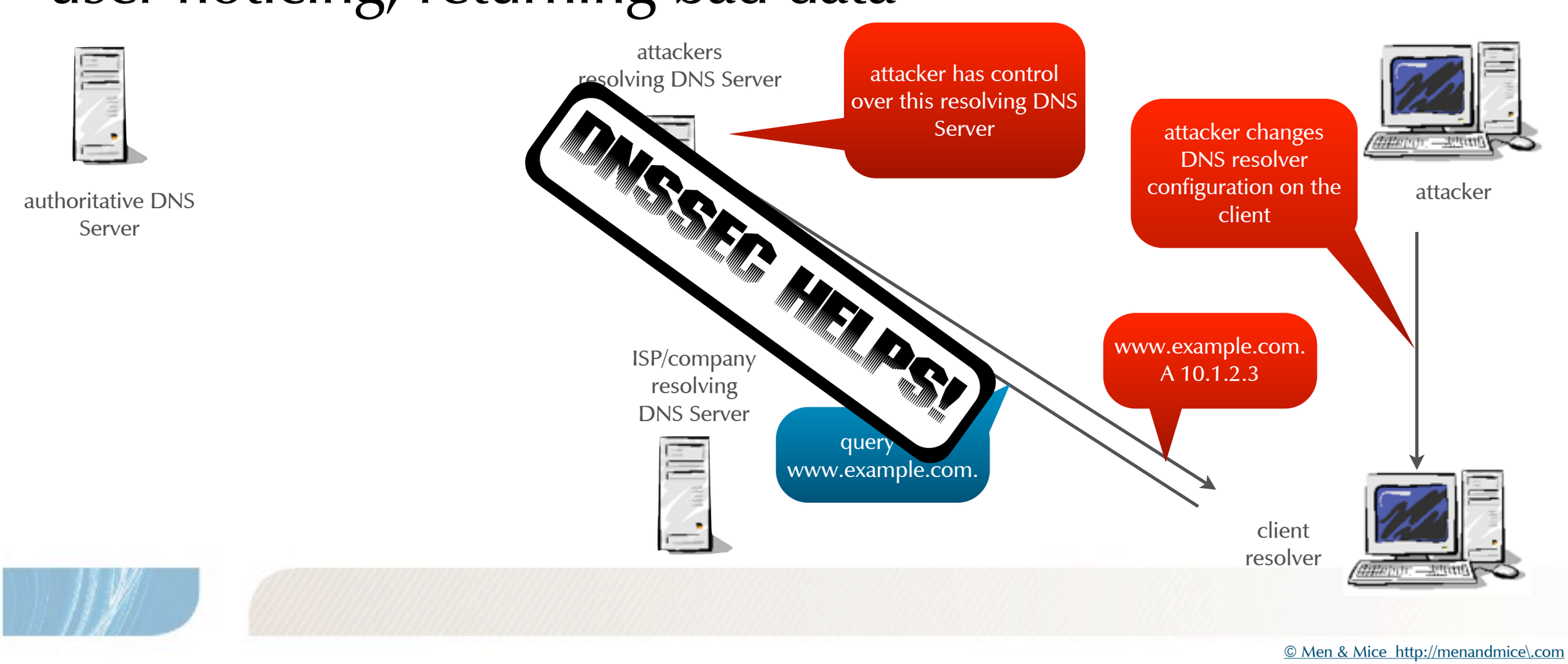
attacker changes the local resolver settings

- the local resolver settings are changed without the client user noticing, returning bad data



attacker changes the local resolver settings

- the local resolver settings are changed without the client user noticing, returning bad data



Attack on authoritative data

- attackers can us security issues to “break in” the DNS Server to alter DNS content
  - exploit security issues
    - in the operating system
    - in the DNS Server software
    - in other network software running (ssh, syslog, ...)

attack on an authoritative DNS Server

- an attacker changes the authoritative data on the DNS Server

authoritative DNS Server

resolving DNS Server

Cache

attacker

client resolver

attack on an authoritative DNS Server

- an attacker changes the authoritative data on the DNS Server

authoritative DNS Server

attacker

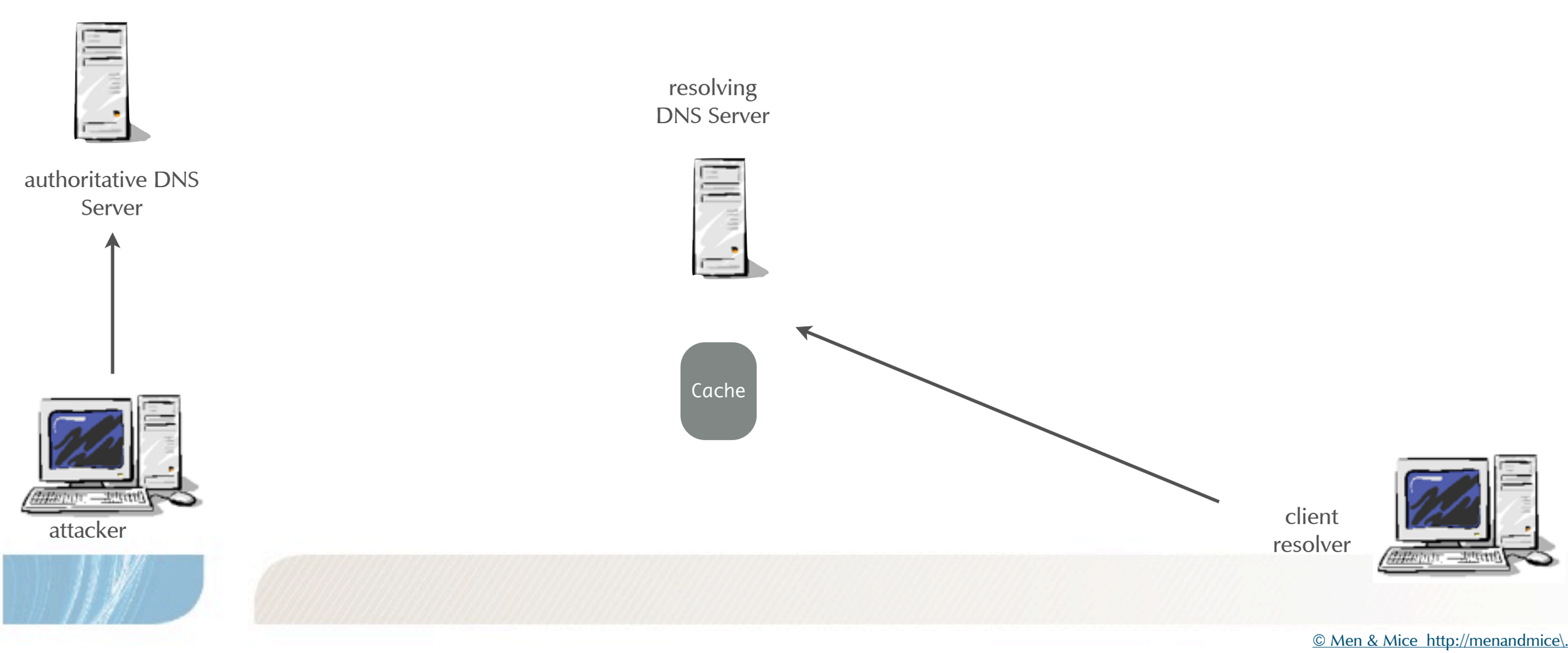
resolving DNS Server

Cache

client resolver

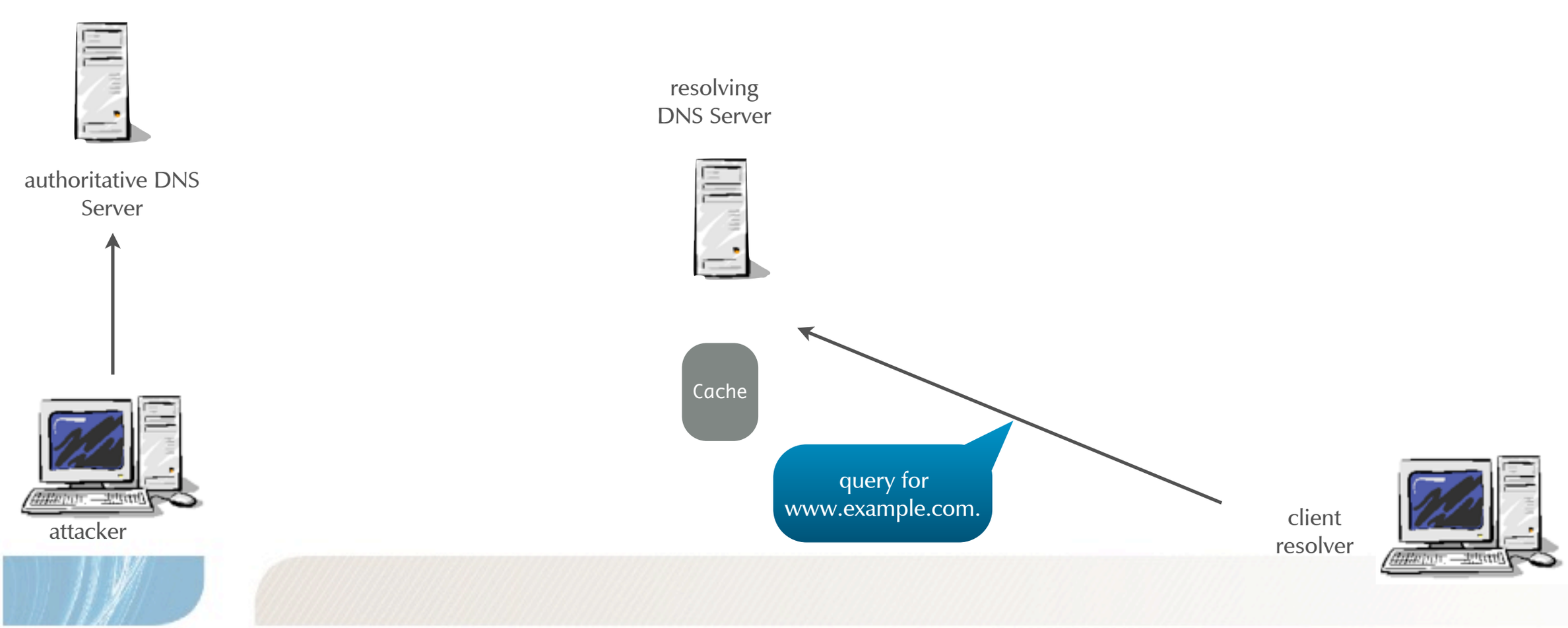
attack on an authoritative DNS Server

- an attacker changes the authoritative data on the DNS Server



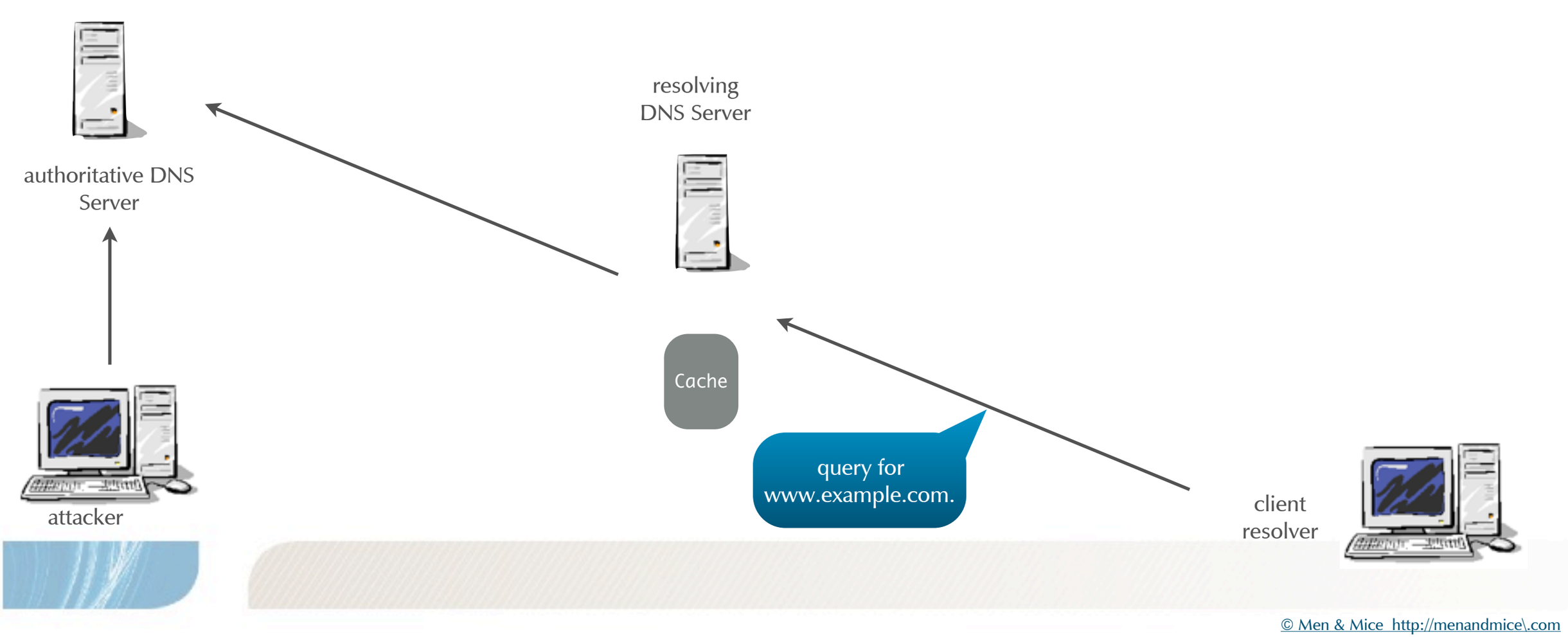
attack on an authoritative DNS Server

- an attacker changes the authoritative data on the DNS Server



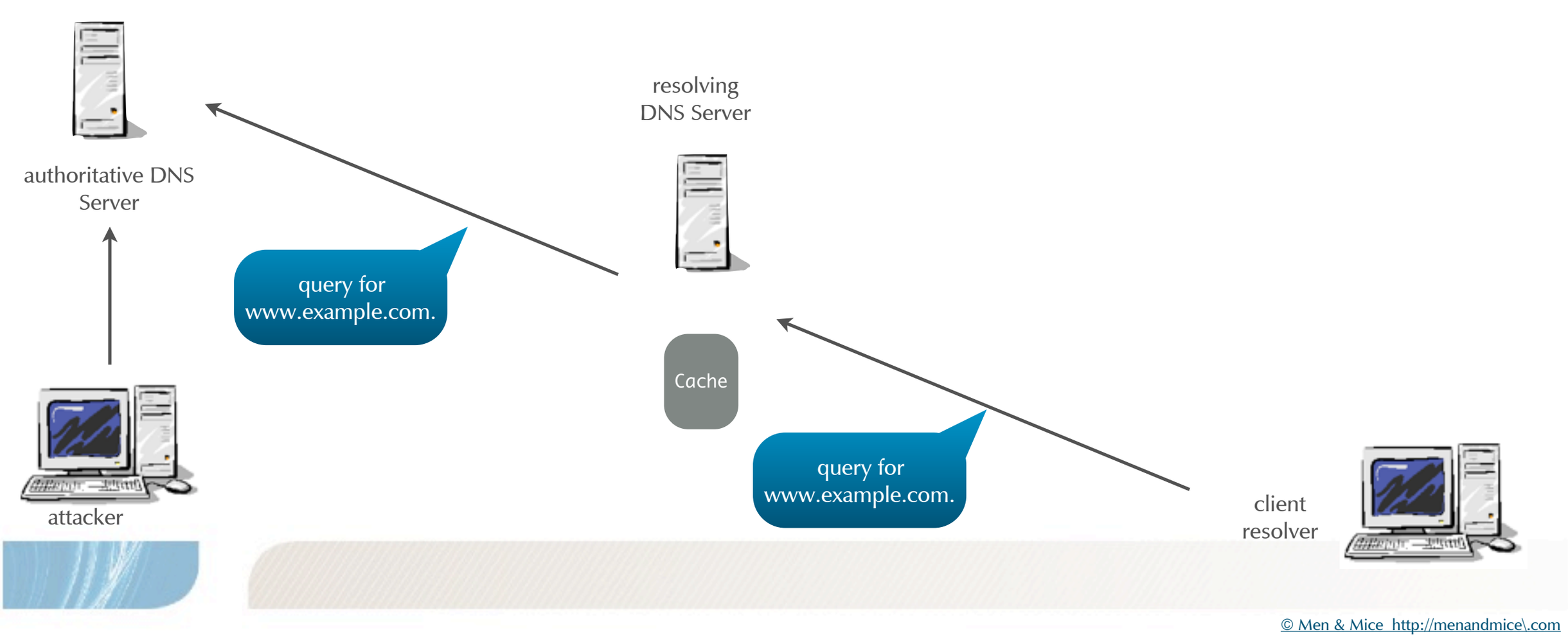
attack on an authoritative DNS Server

- an attacker changes the authoritative data on the DNS Server



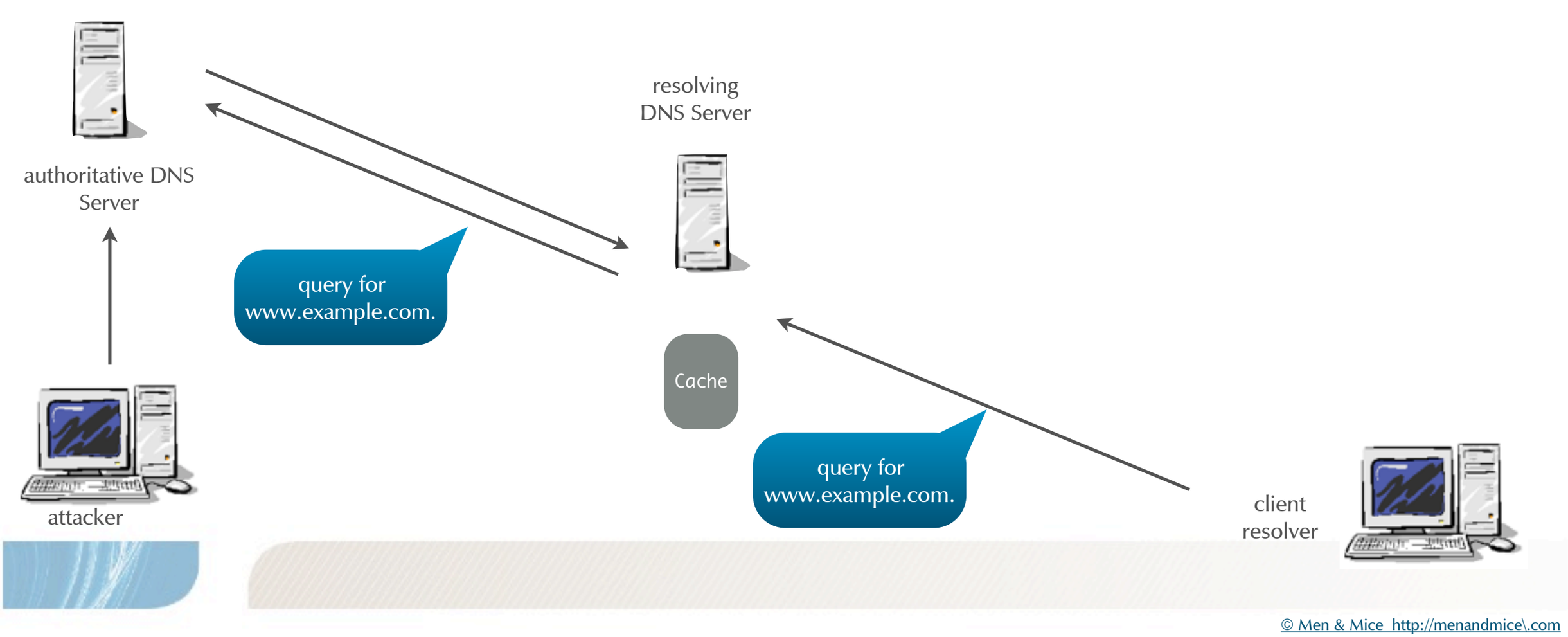
attack on an authoritative DNS Server

- an attacker changes the authoritative data on the DNS Server



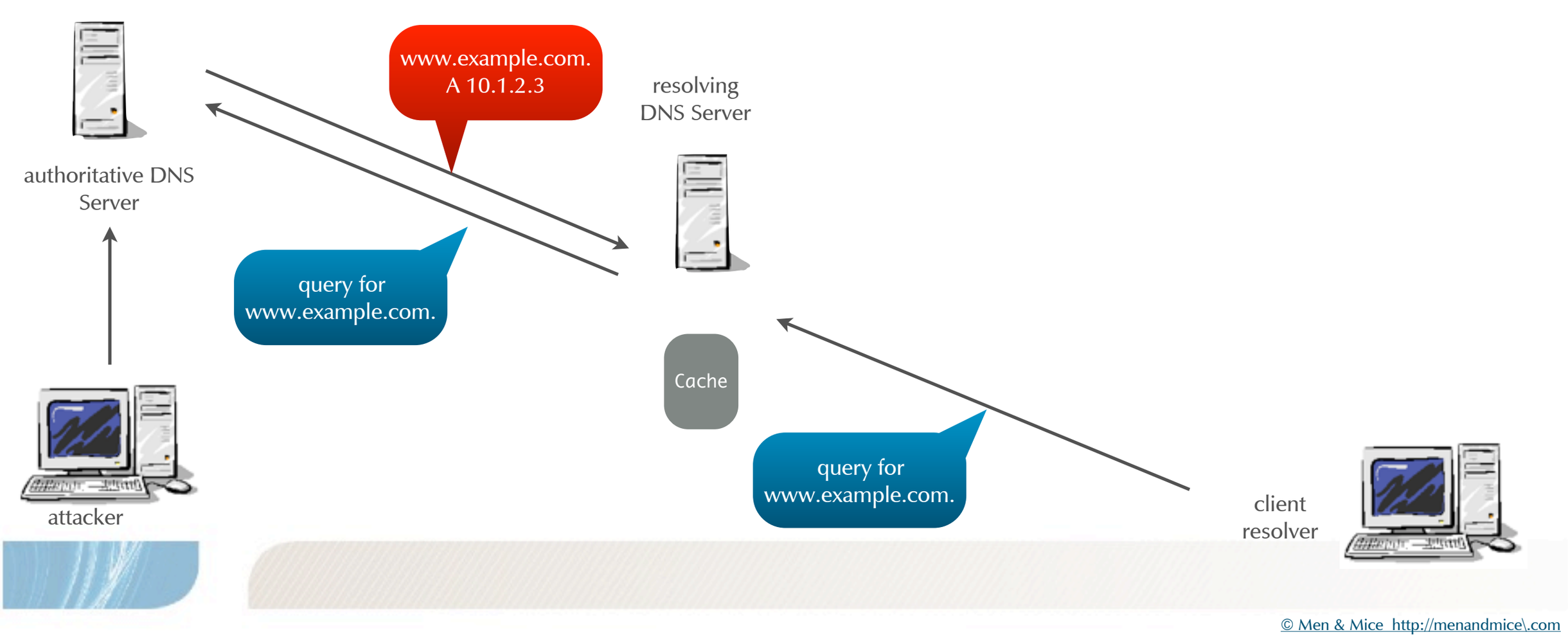
attack on an authoritative DNS Server

- an attacker changes the authoritative data on the DNS Server



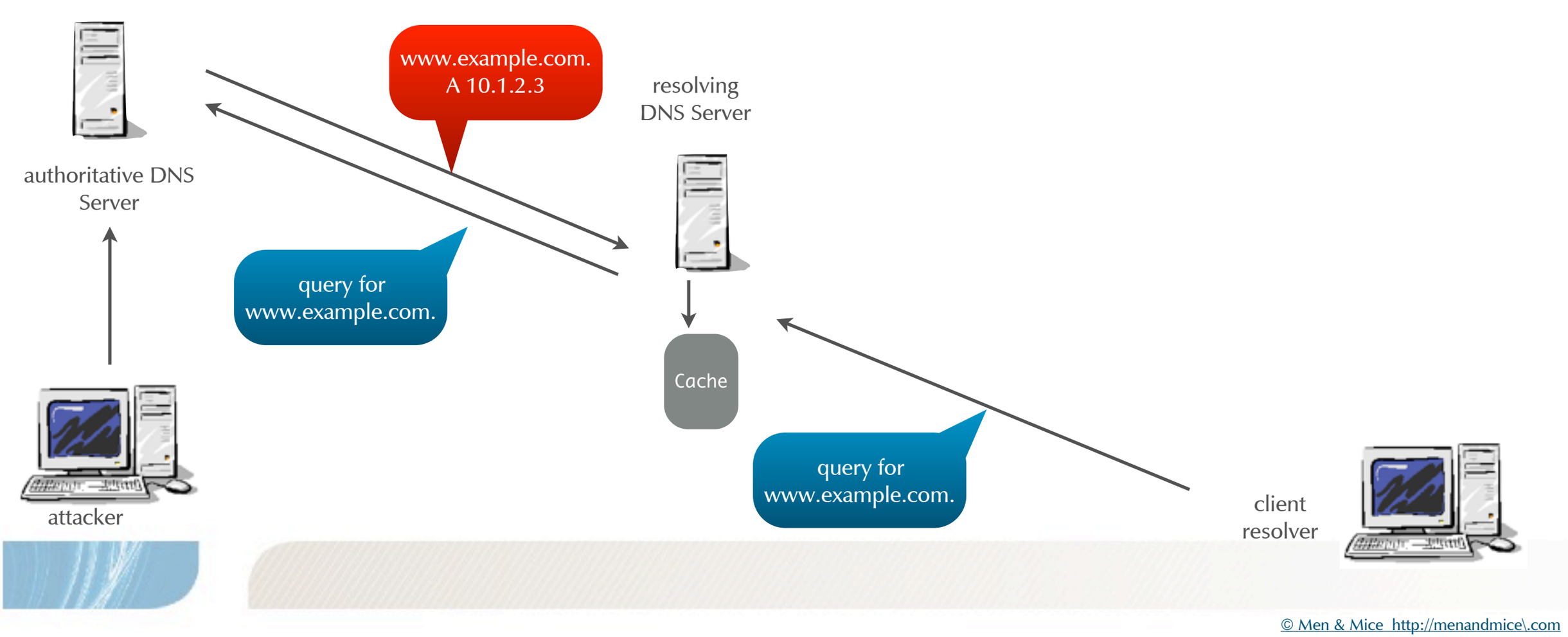
attack on an authoritative DNS Server

- an attacker changes the authoritative data on the DNS Server



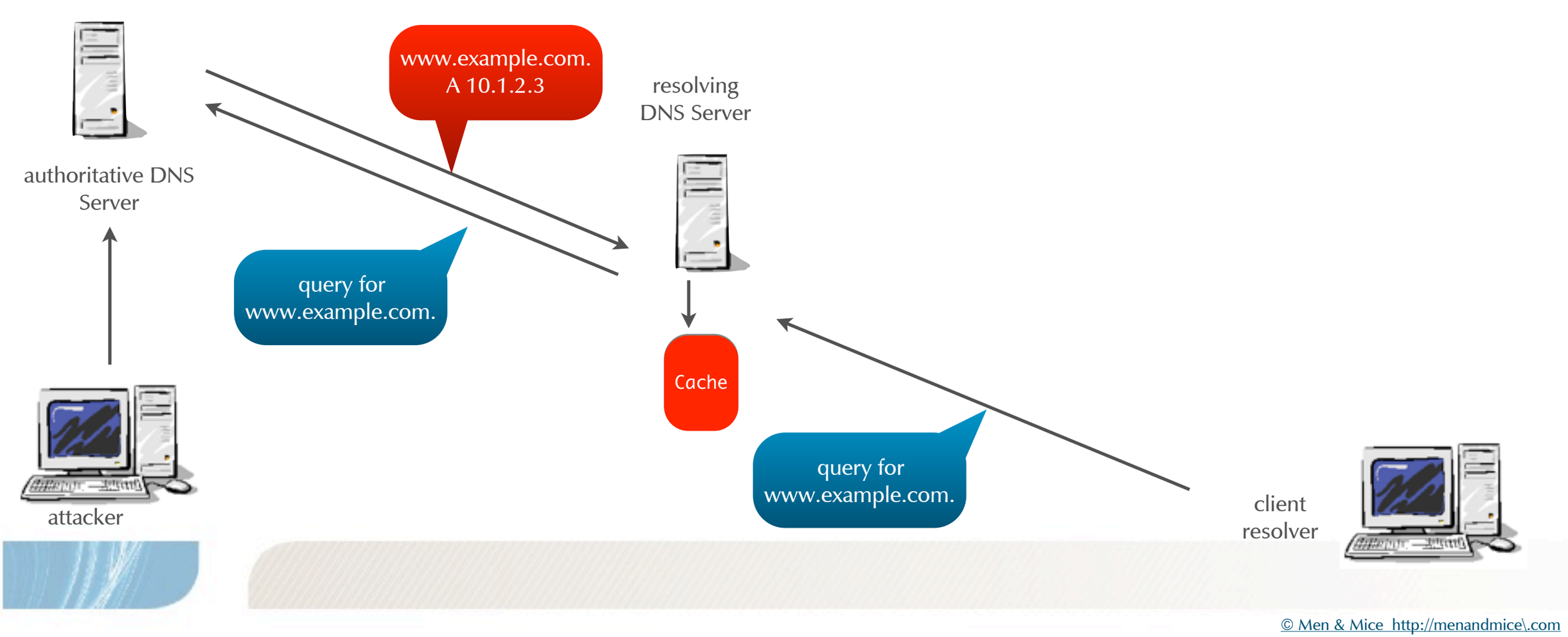
attack on an authoritative DNS Server

- an attacker changes the authoritative data on the DNS Server



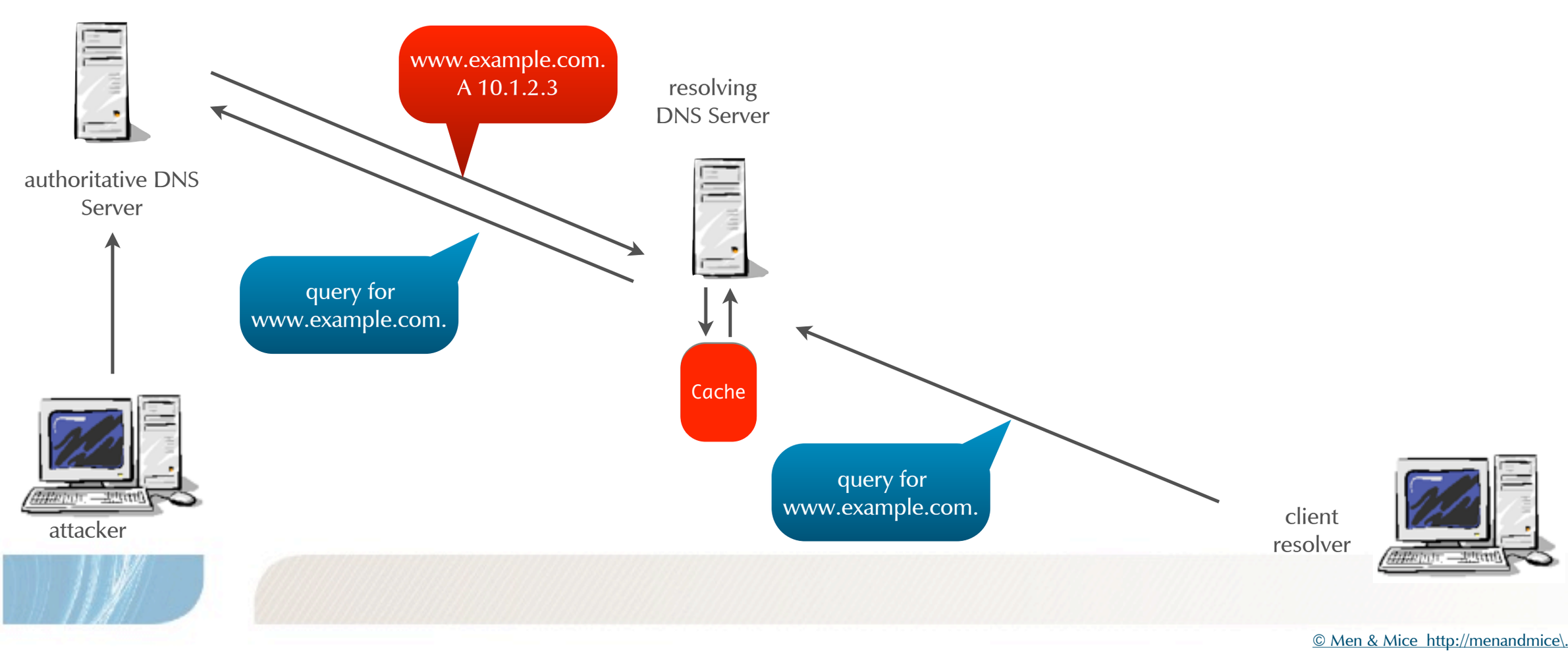
attack on an authoritative DNS Server

- an attacker changes the authoritative data on the DNS Server



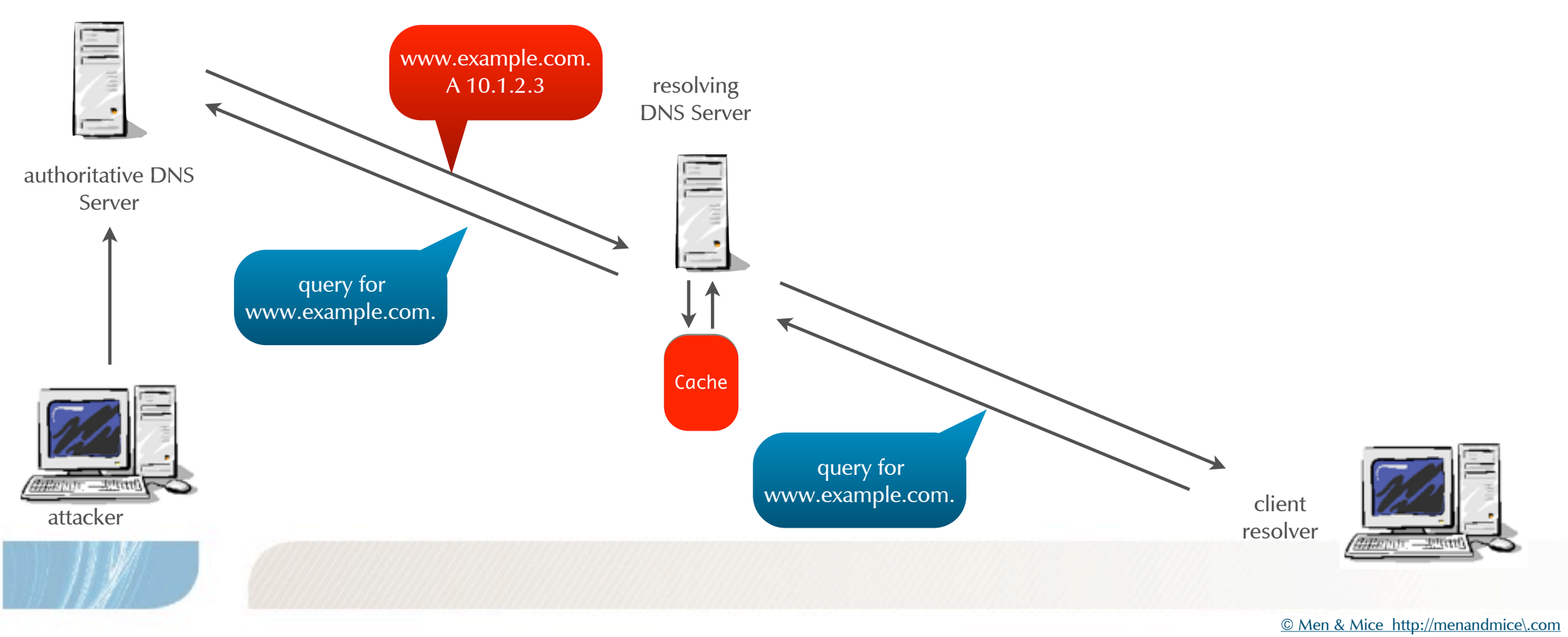
attack on an authoritative DNS Server

- an attacker changes the authoritative data on the DNS Server



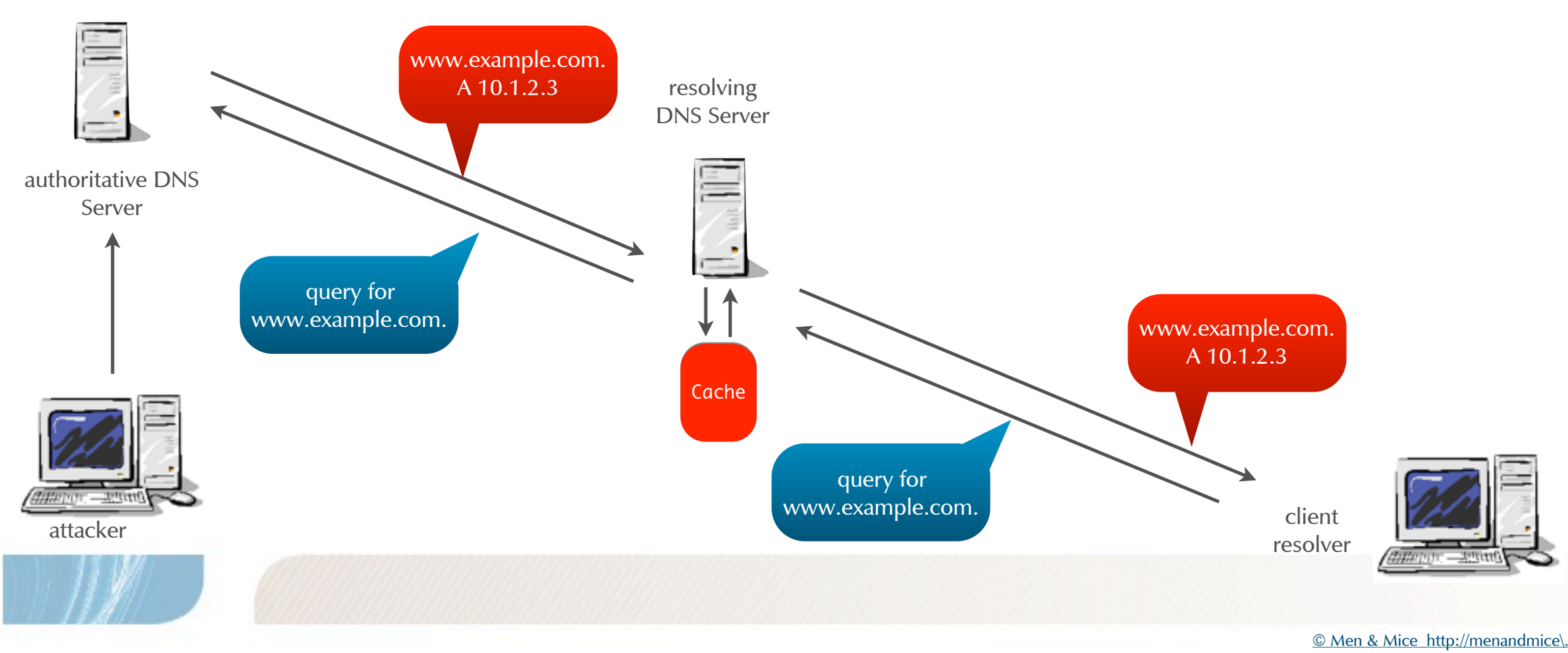
attack on an authoritative DNS Server

- an attacker changes the authoritative data on the DNS Server



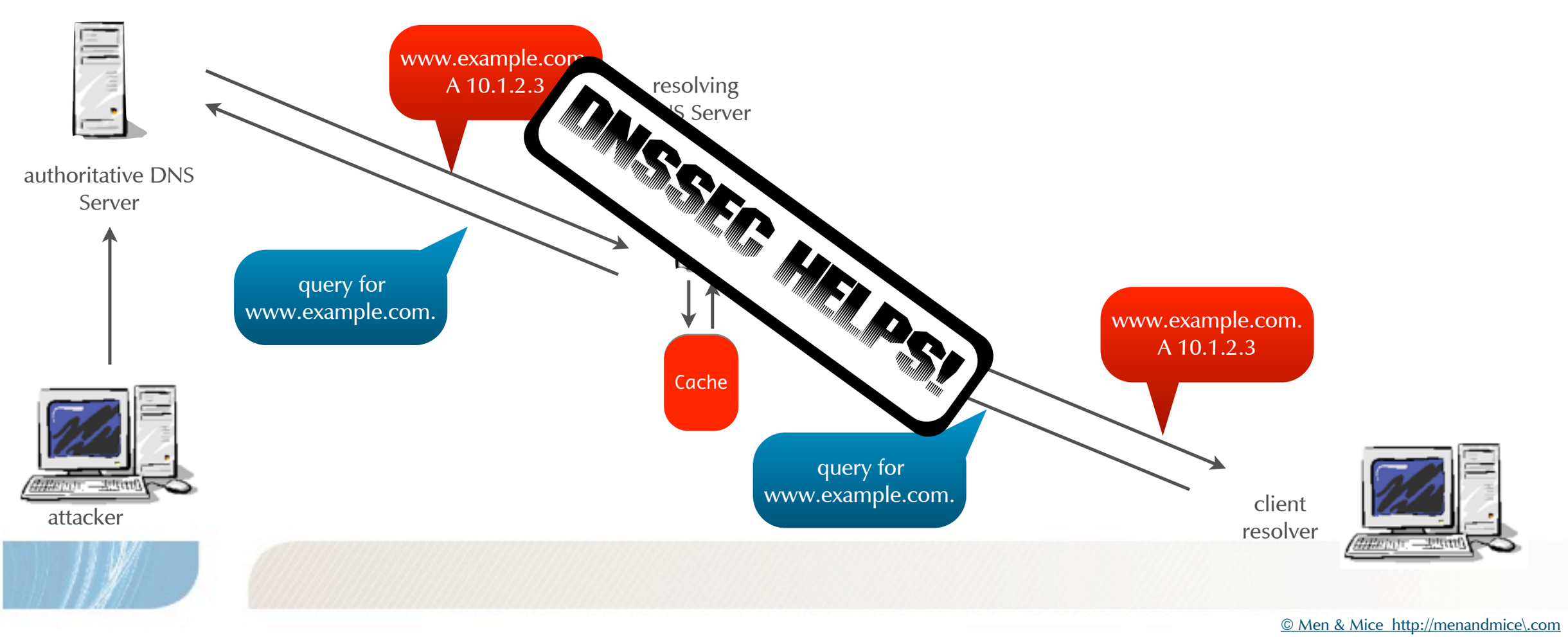
attack on an authoritative DNS Server

- an attacker changes the authoritative data on the DNS Server



attack on an authoritative DNS Server

- an attacker changes the authoritative data on the DNS Server



DNSSEC

"One Key to rule them all,
one Key to find them,
one Key to bring them all
and in the Resolver bind them."
—Modified from Lord of the Rings
Miek Gieben.

MEN&MICE

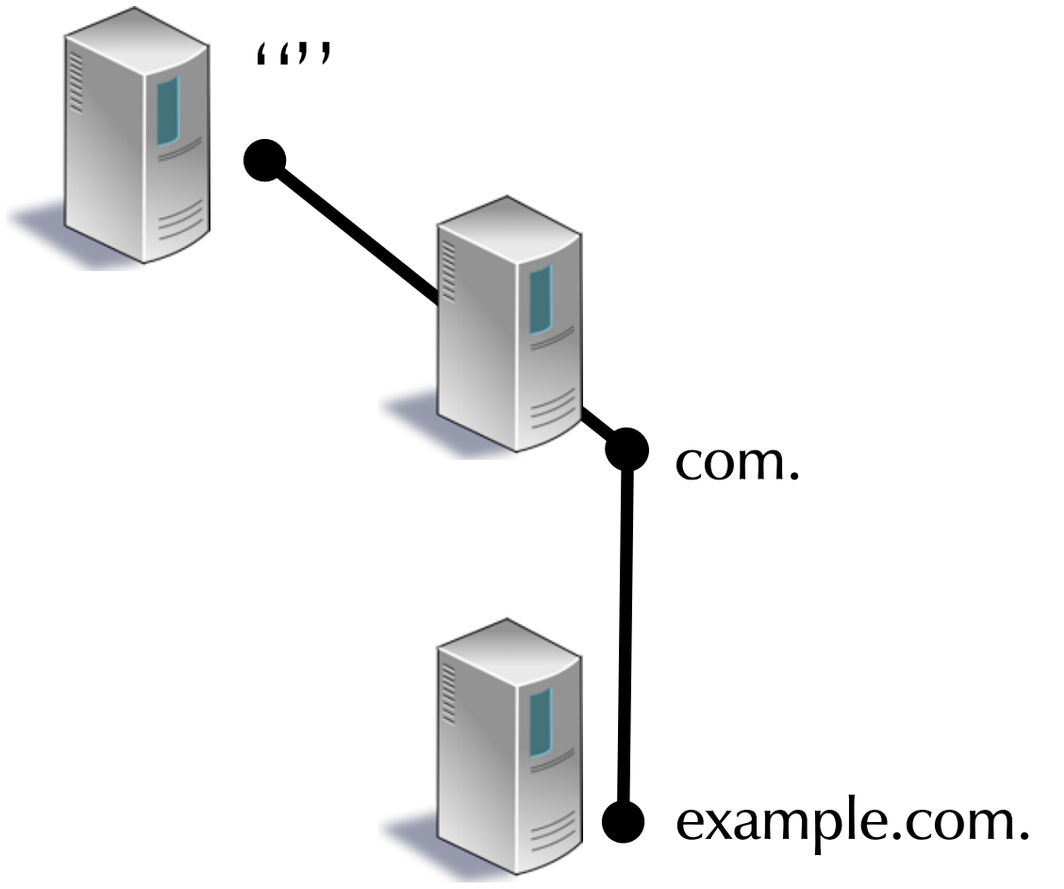
DNSSEC Technical Overview

- Without DNSSEC, in unsigned DNS, there are four types of response:
  - Positive answer
  - Negative answer
    - No data
    - NXDOMAIN (name error)
  - Referral
  - Error

DNSSEC Technical Overview

- DNSSEC must be able to prove authoritative answers
  - Positive answer
  - Negative answer
- DNSSEC also needs a trust mechanism (the chain of trust), which just happens to mirror delegations (the chain of authority)

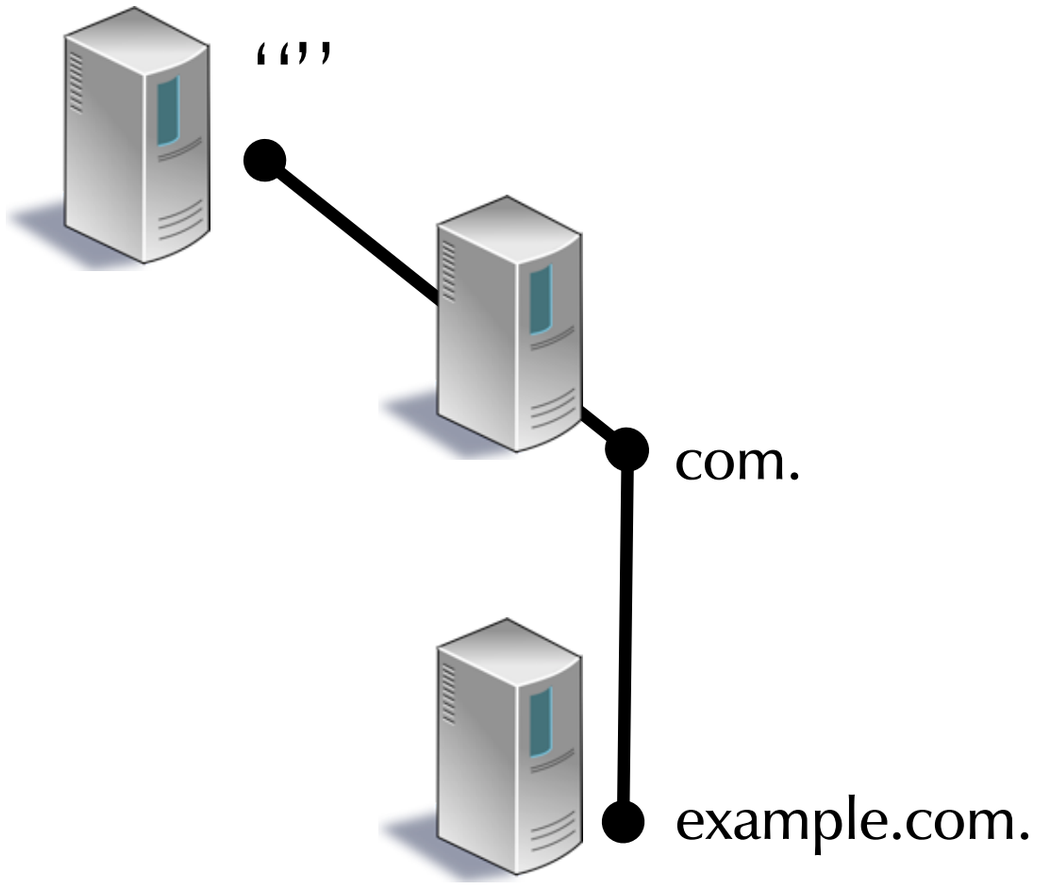
Name Resolution in Pictures



local caching
DNS Server

MEN&MICE

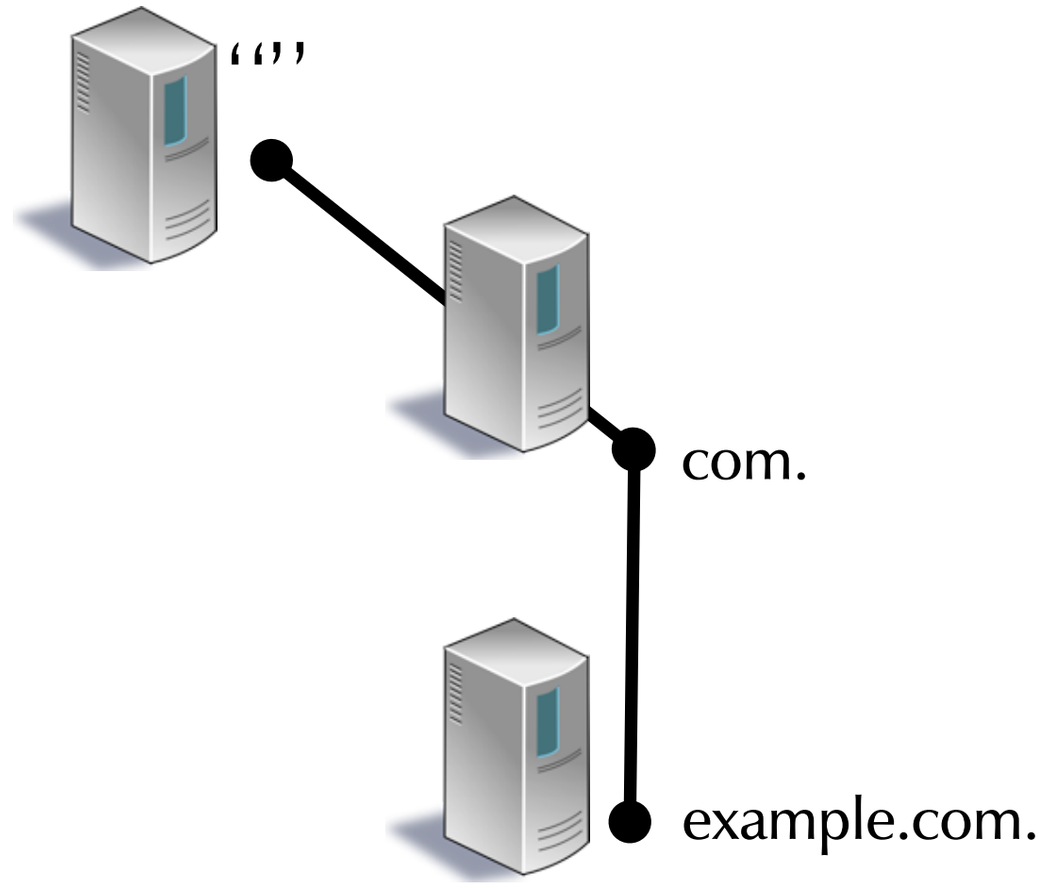
Name Resolution in Pictures



local caching
DNS Server

MEN&MICE

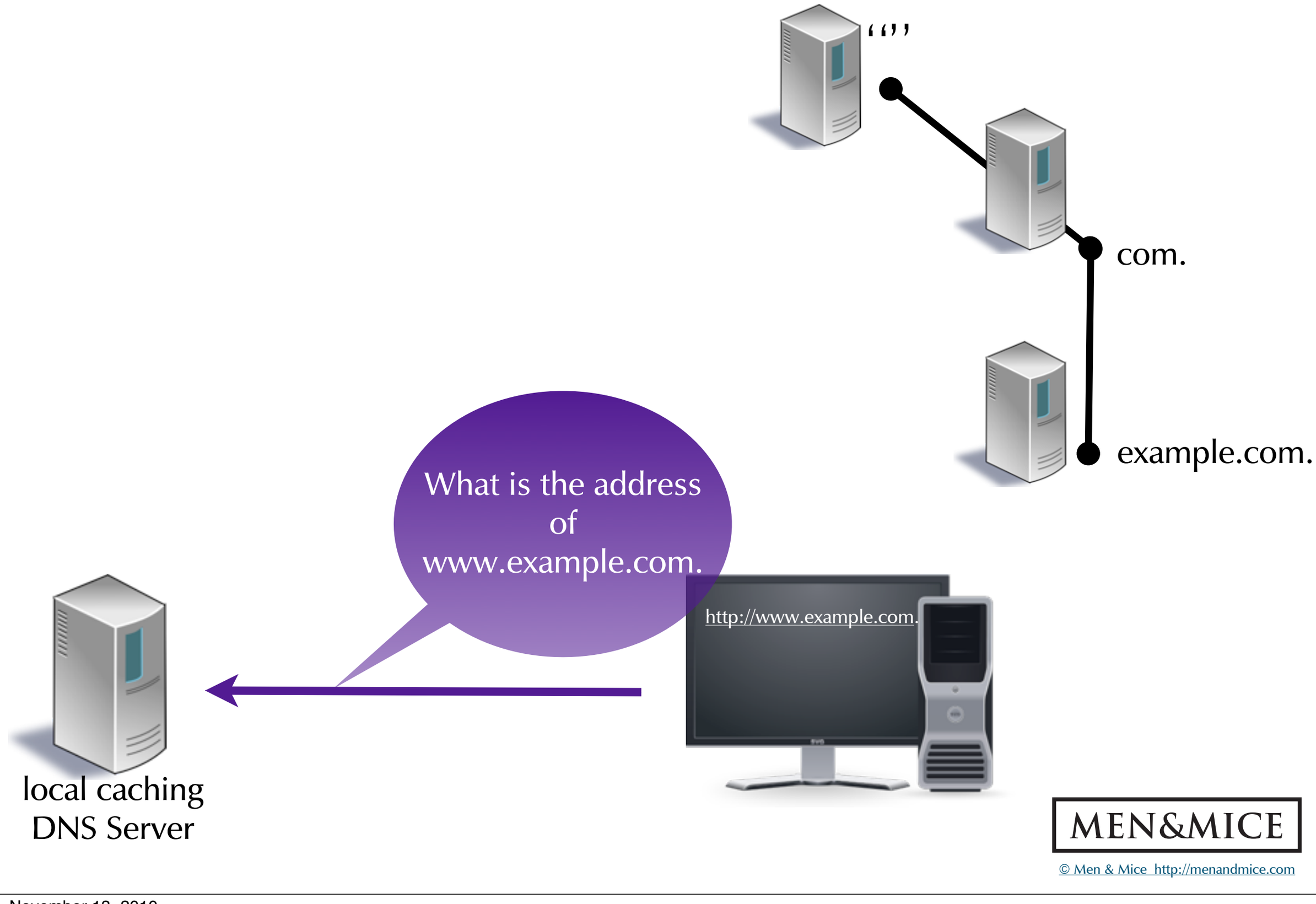
Name Resolution in Pictures



local caching
DNS Server

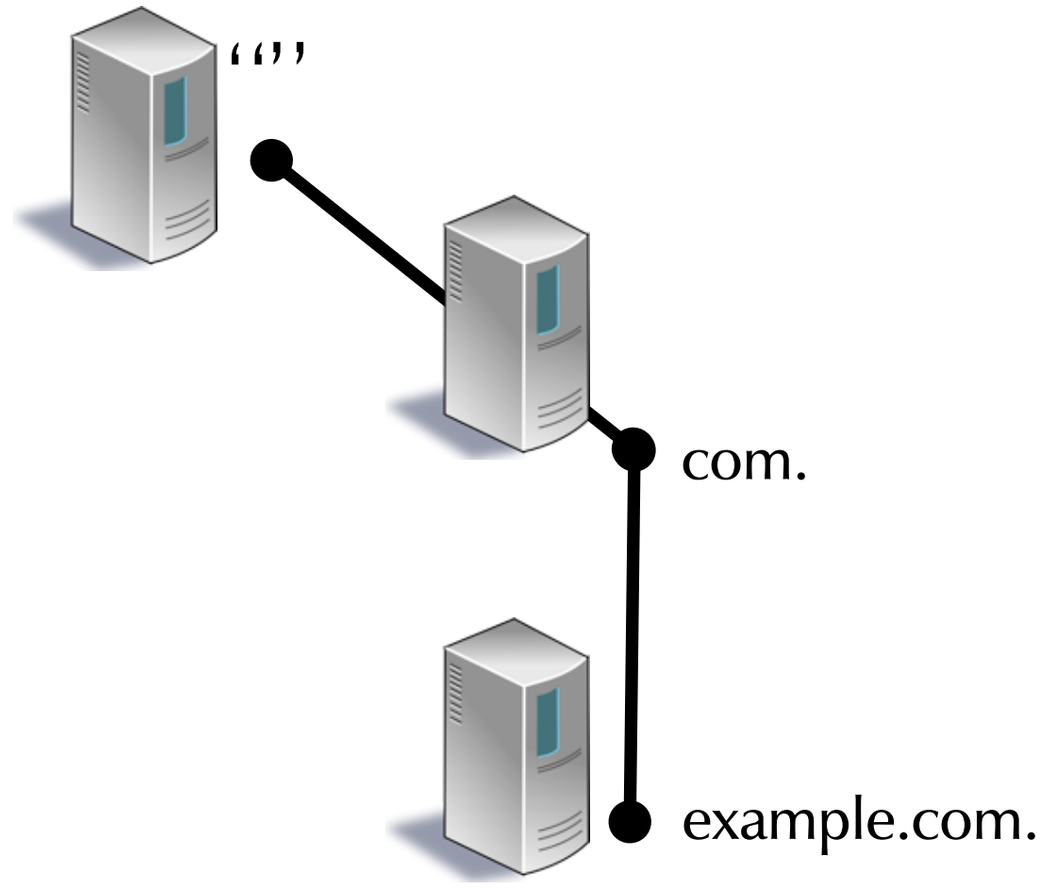
MEN&MICE

Name Resolution in Pictures



MEN&MICE

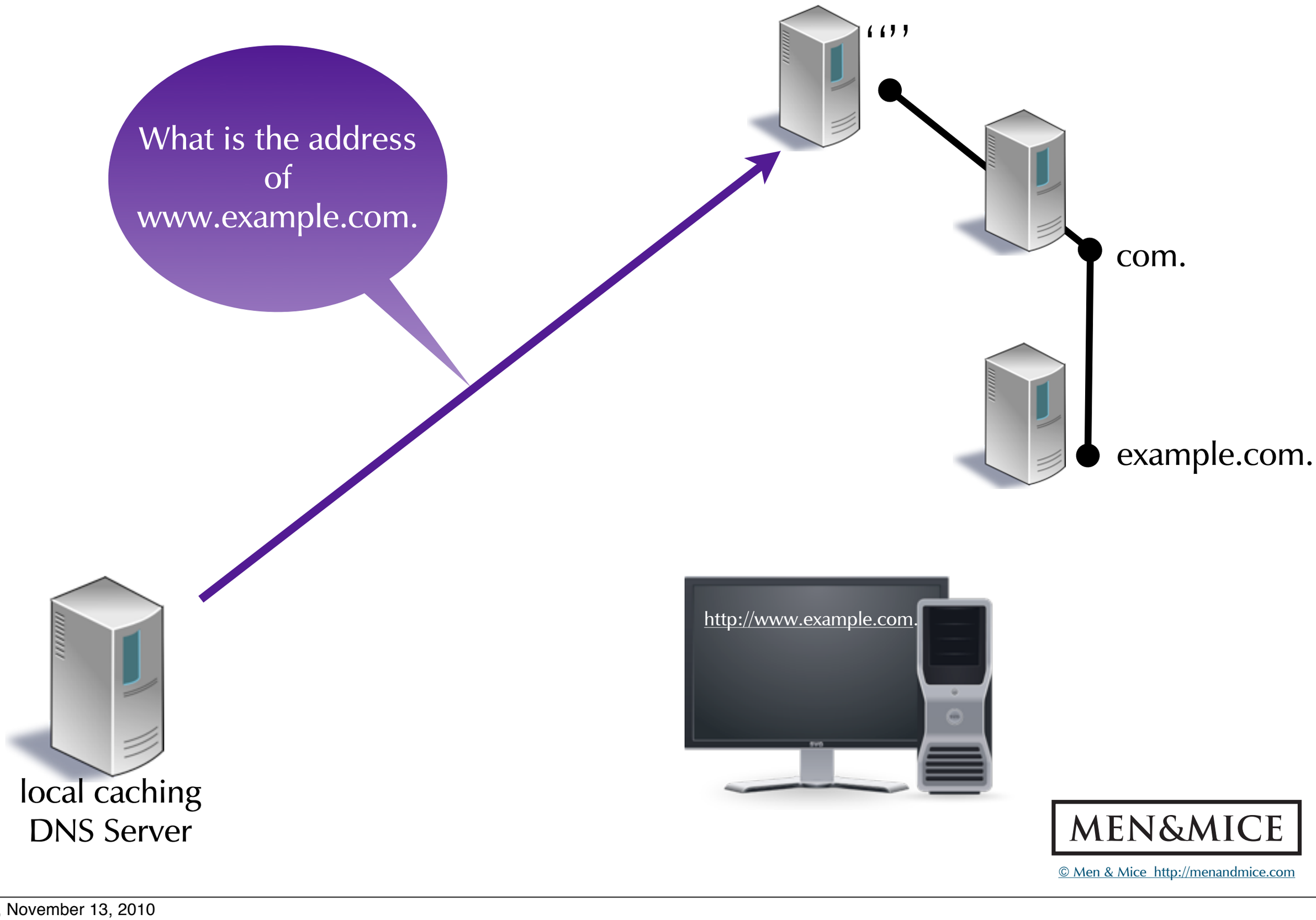
Name Resolution in Pictures



local caching
DNS Server

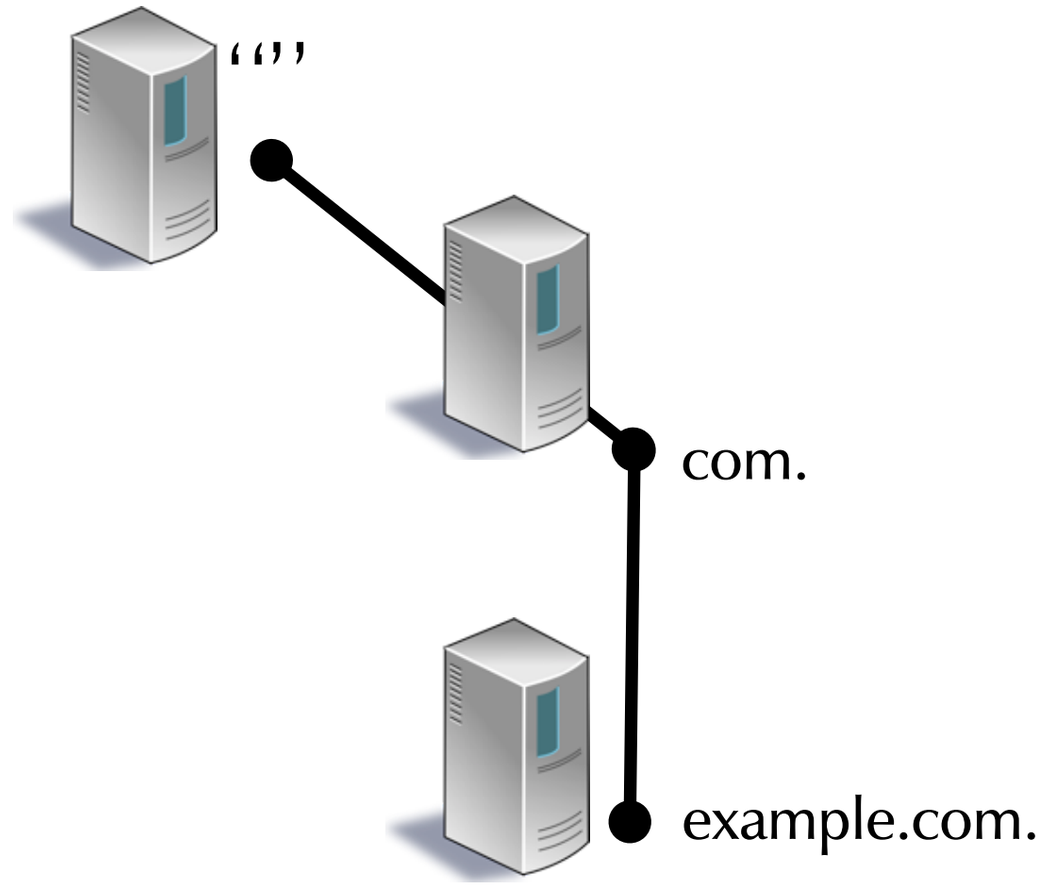
MEN&MICE

Name Resolution in Pictures



MEN&MICE

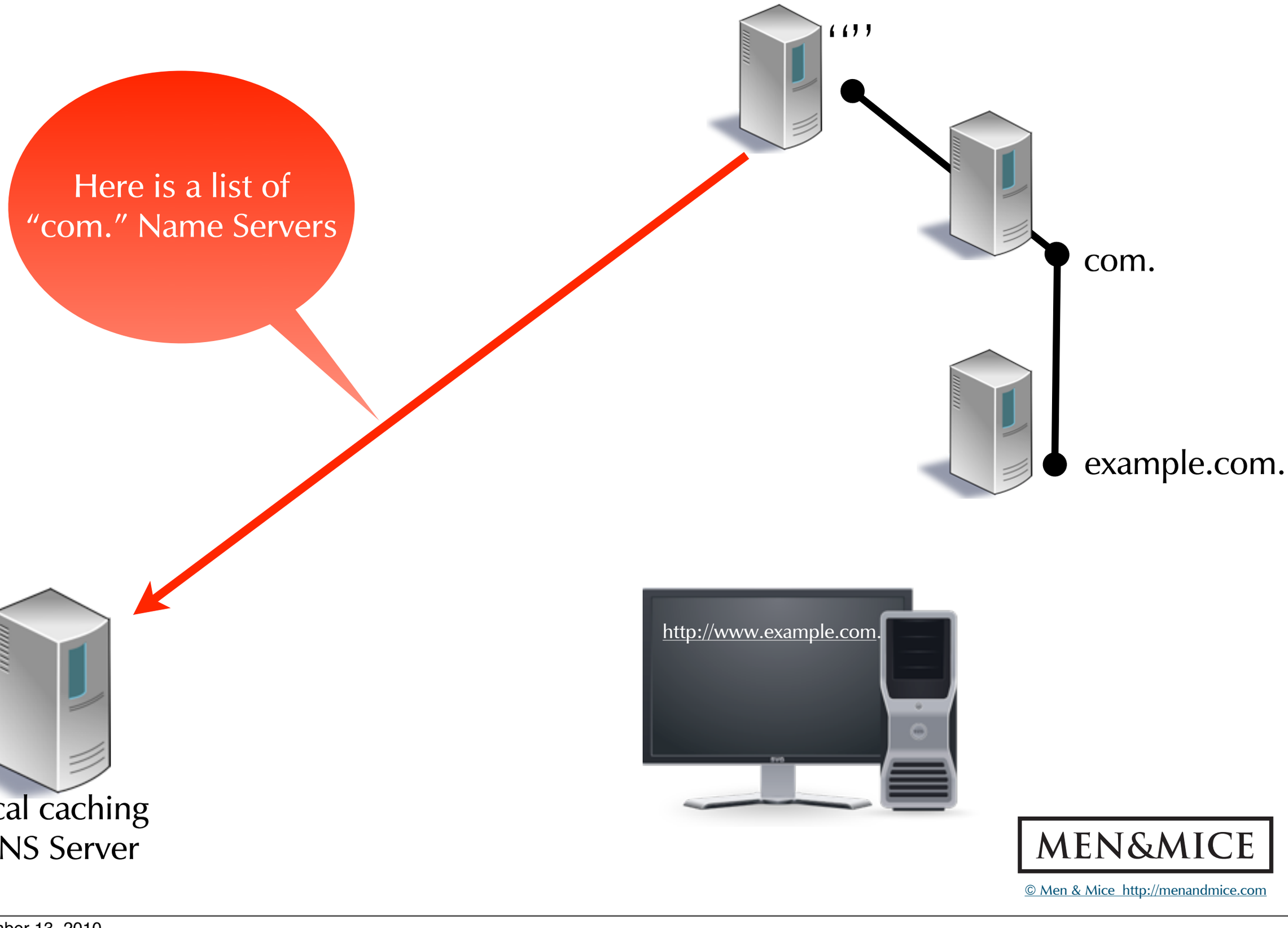
Name Resolution in Pictures



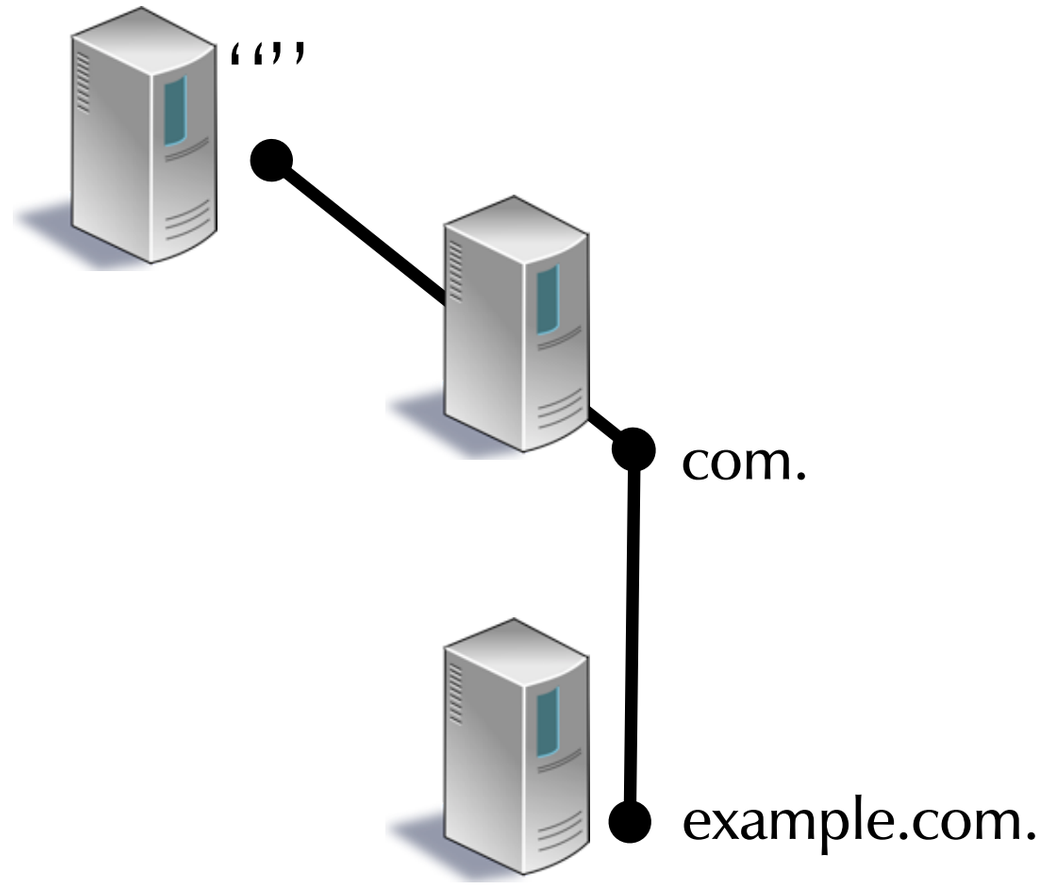
local caching
DNS Server

MEN&MICE

Name Resolution in Pictures



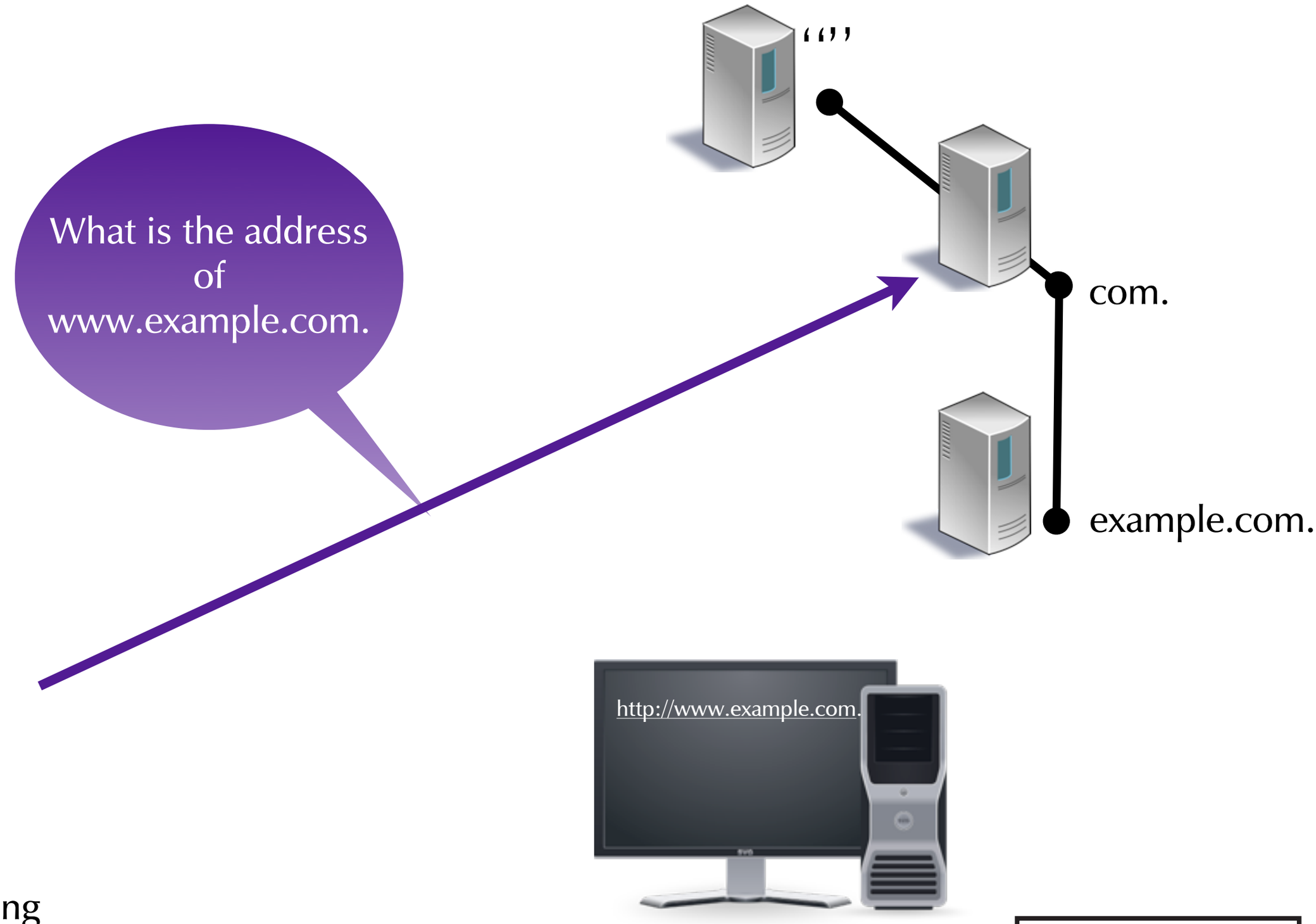
Name Resolution in Pictures



local caching
DNS Server

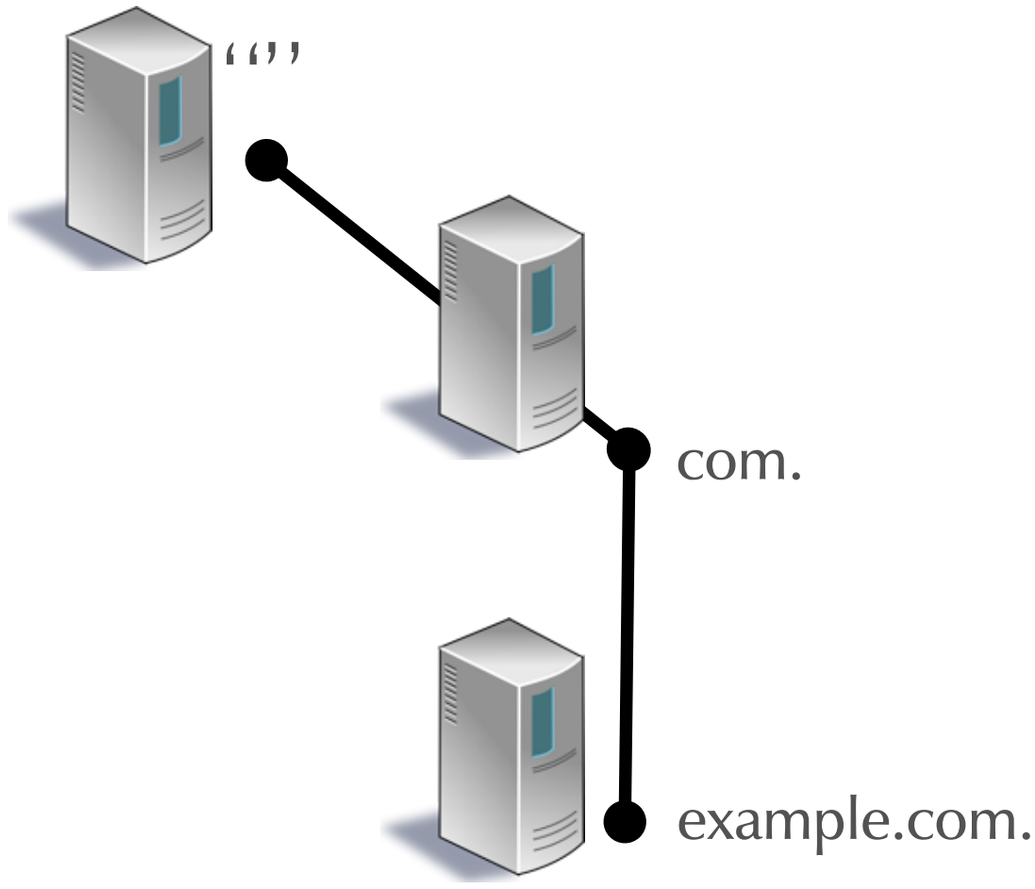
MEN&MICE

Name Resolution in Pictures



MEN&MICE

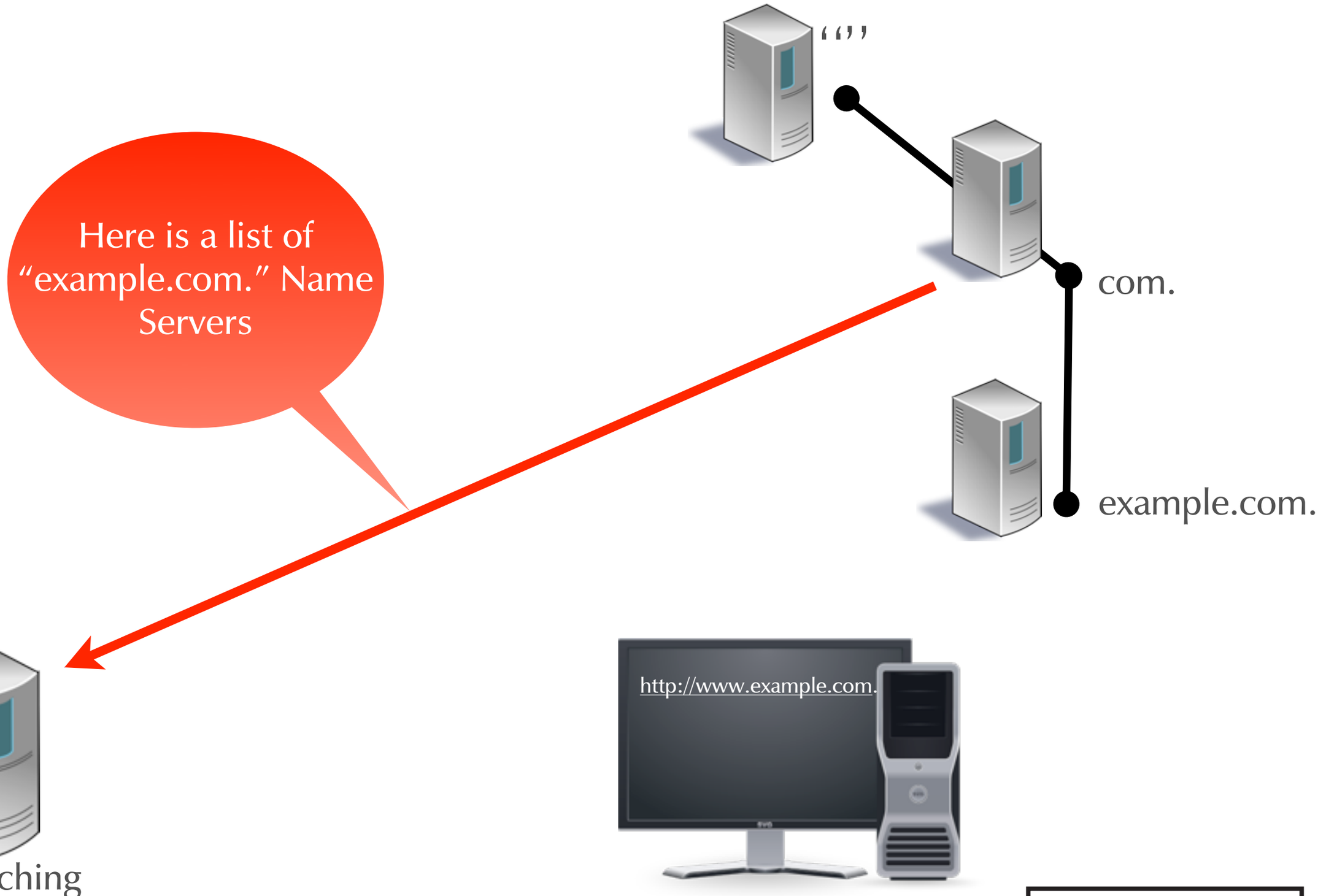
Name Resolution in Pictures



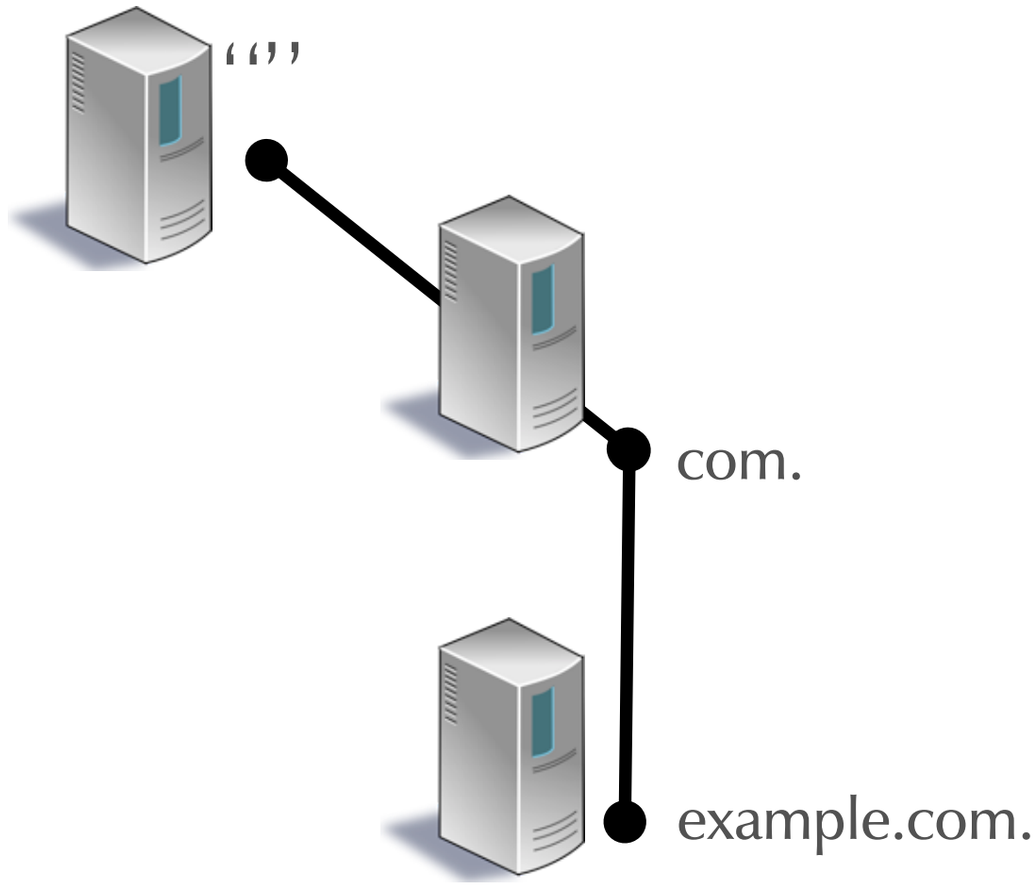
local caching
DNS Server

MEN&MICE

Name Resolution in Pictures



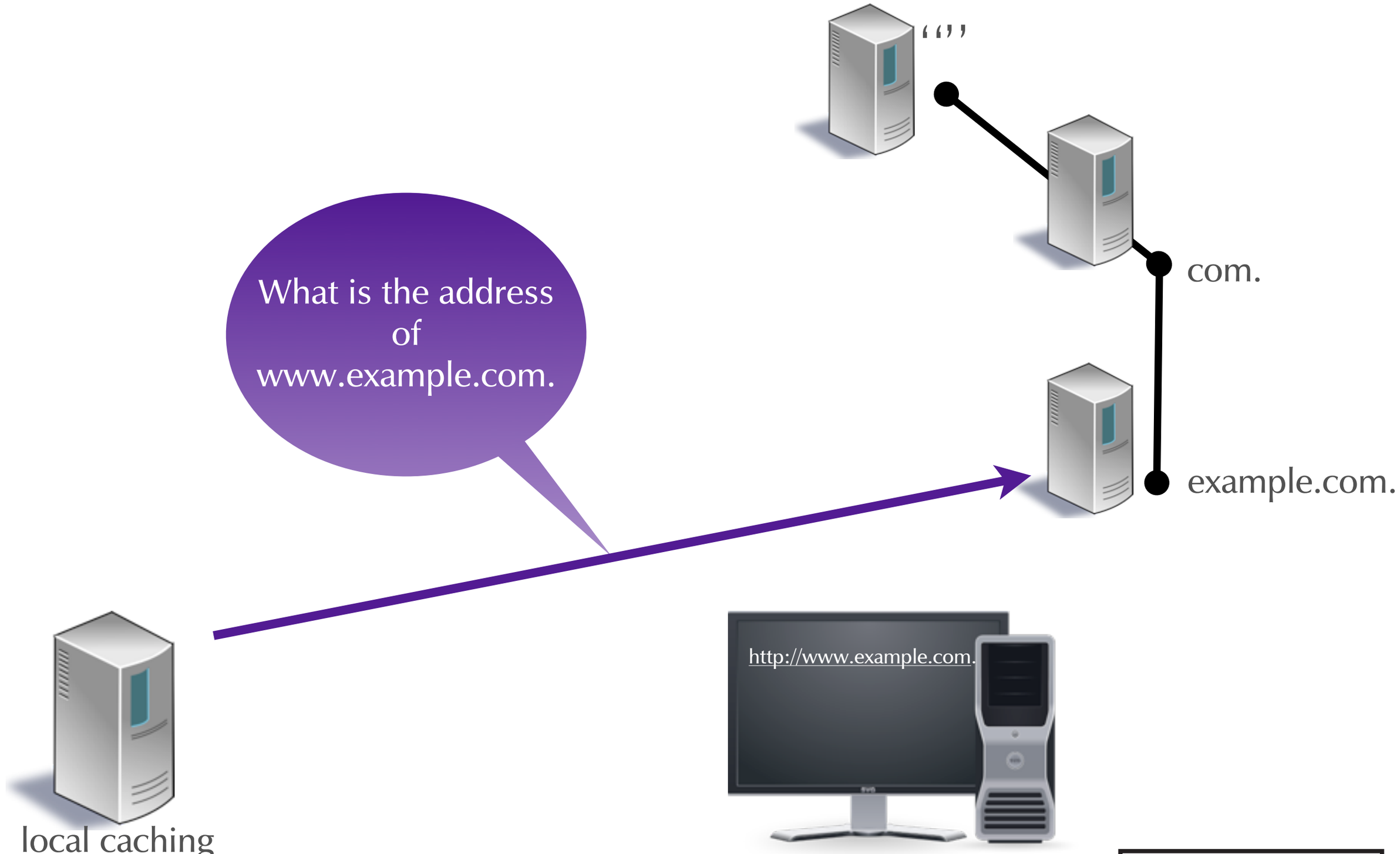
Name Resolution in Pictures



local caching
DNS Server

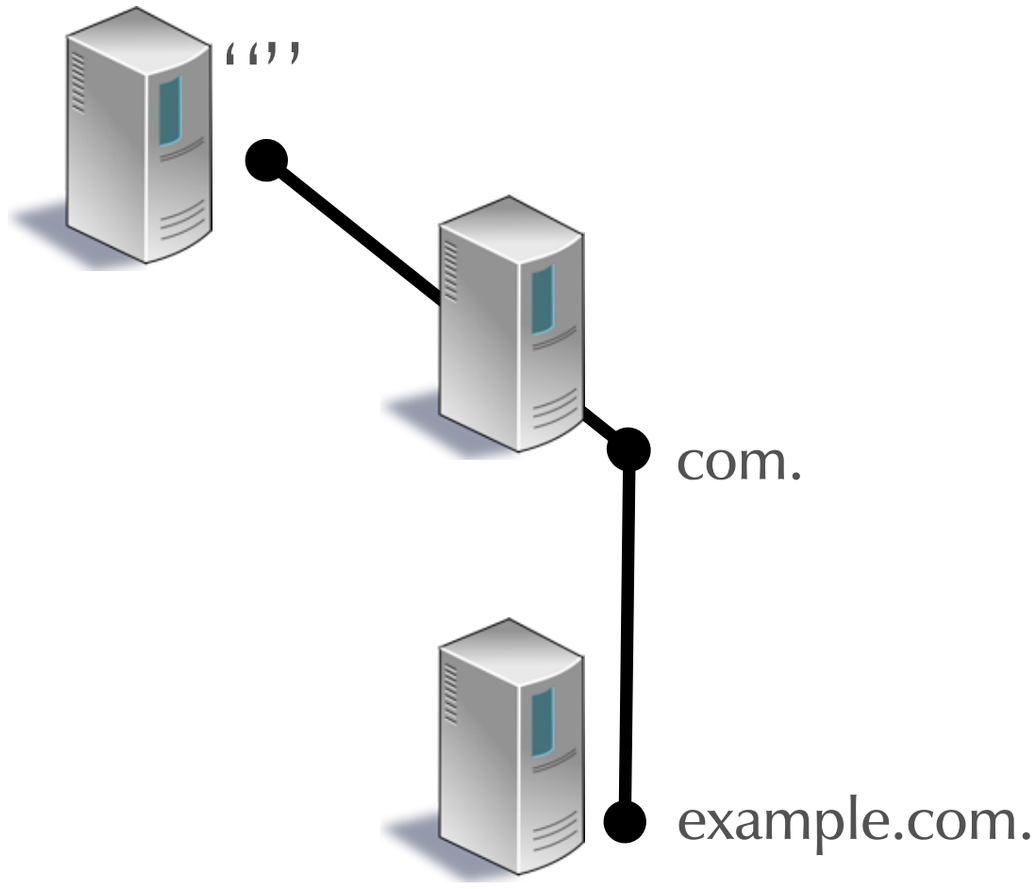
MEN&MICE

Name Resolution in Pictures



MEN&MICE

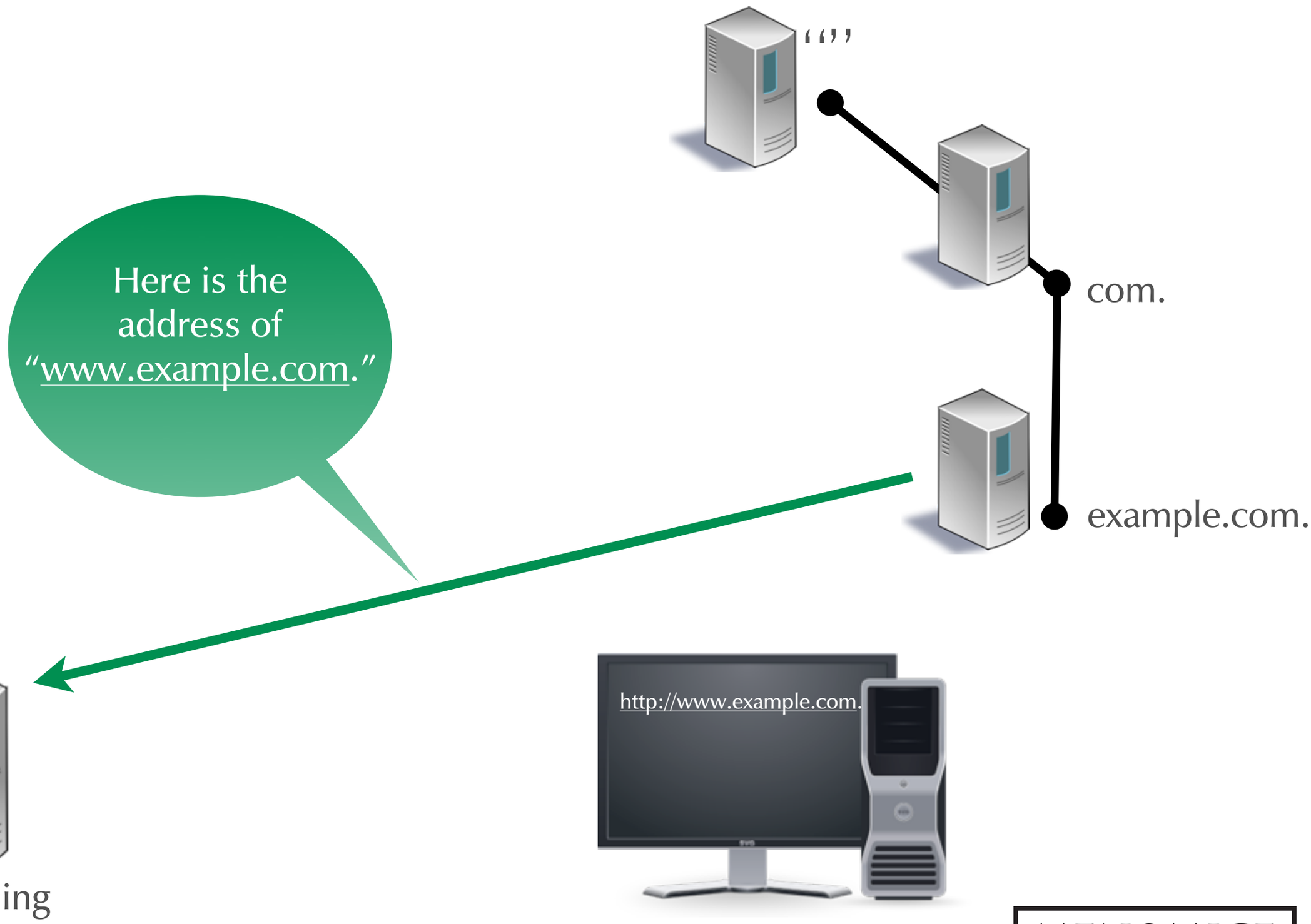
Name Resolution in Pictures



local caching
DNS Server

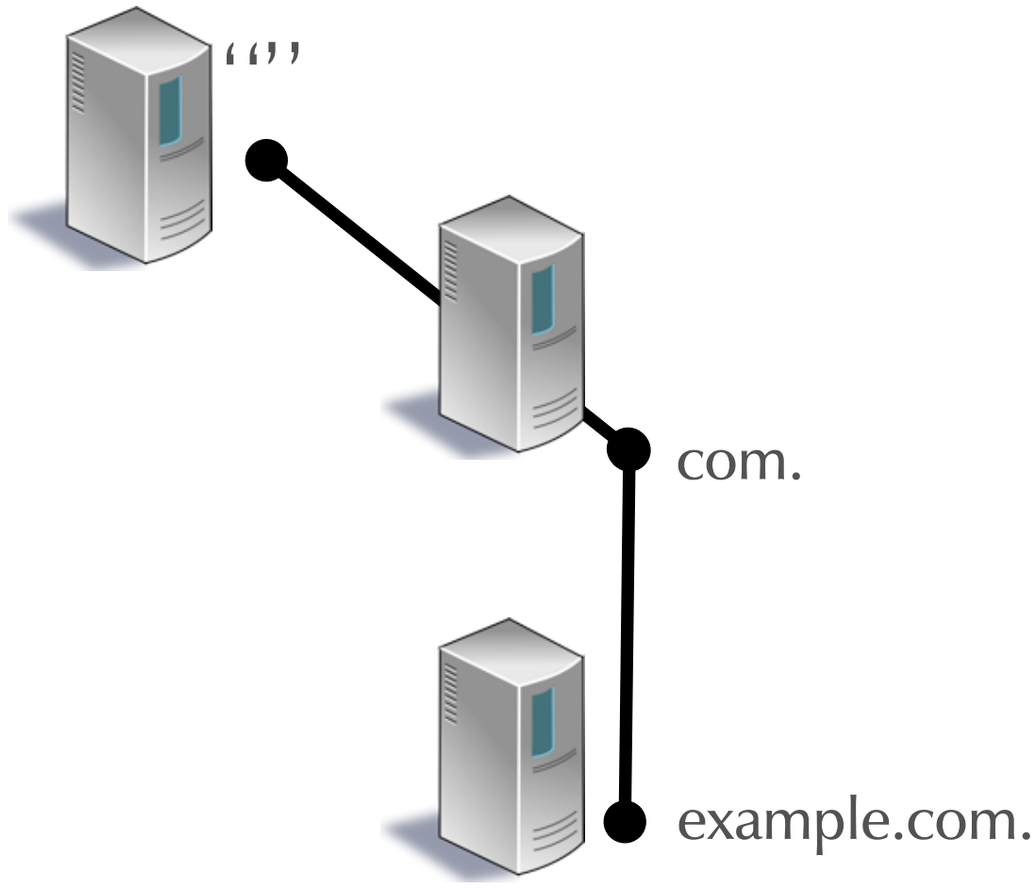
MEN&MICE

Name Resolution in Pictures



MEN&MICE

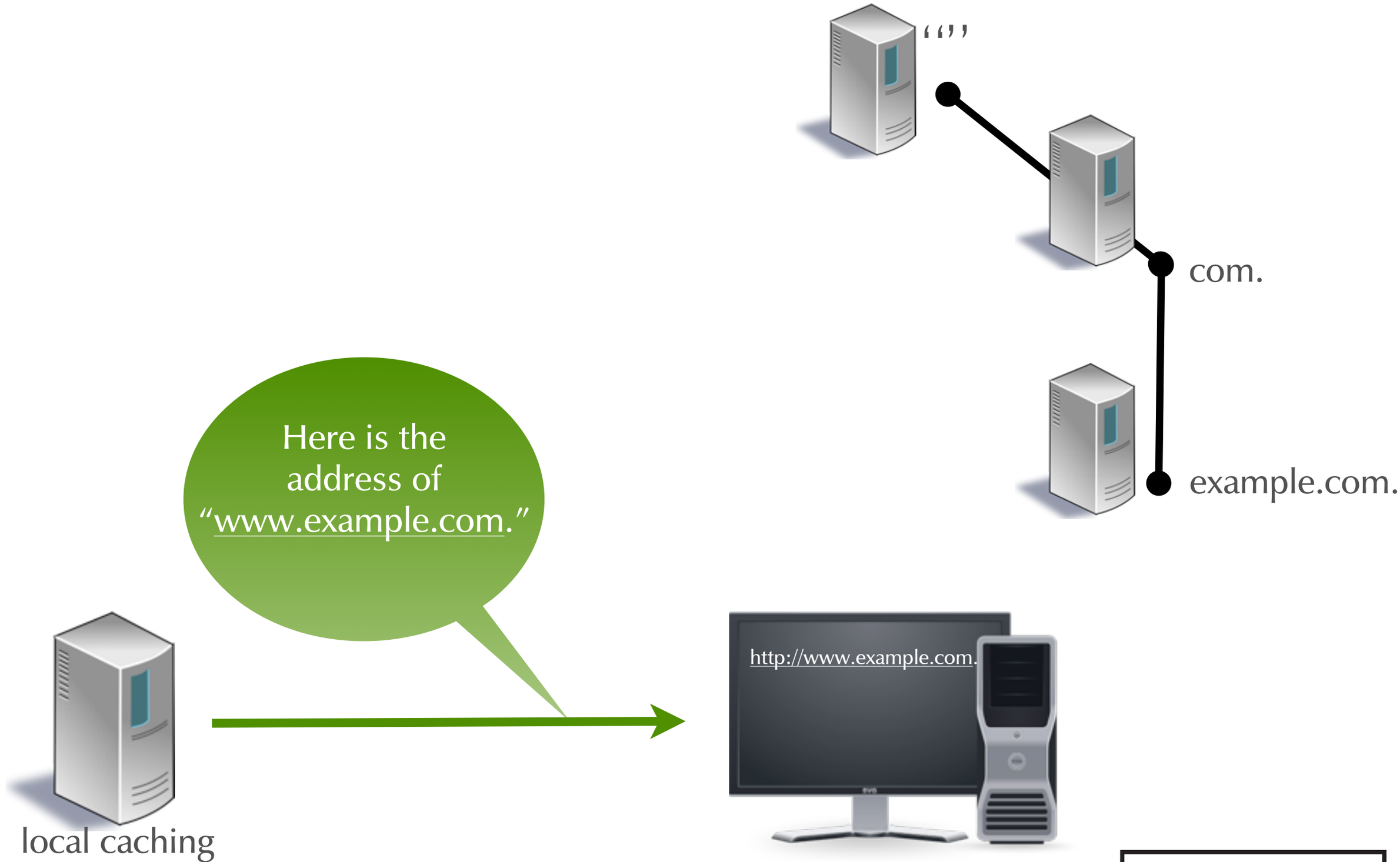
Name Resolution in Pictures



local caching
DNS Server

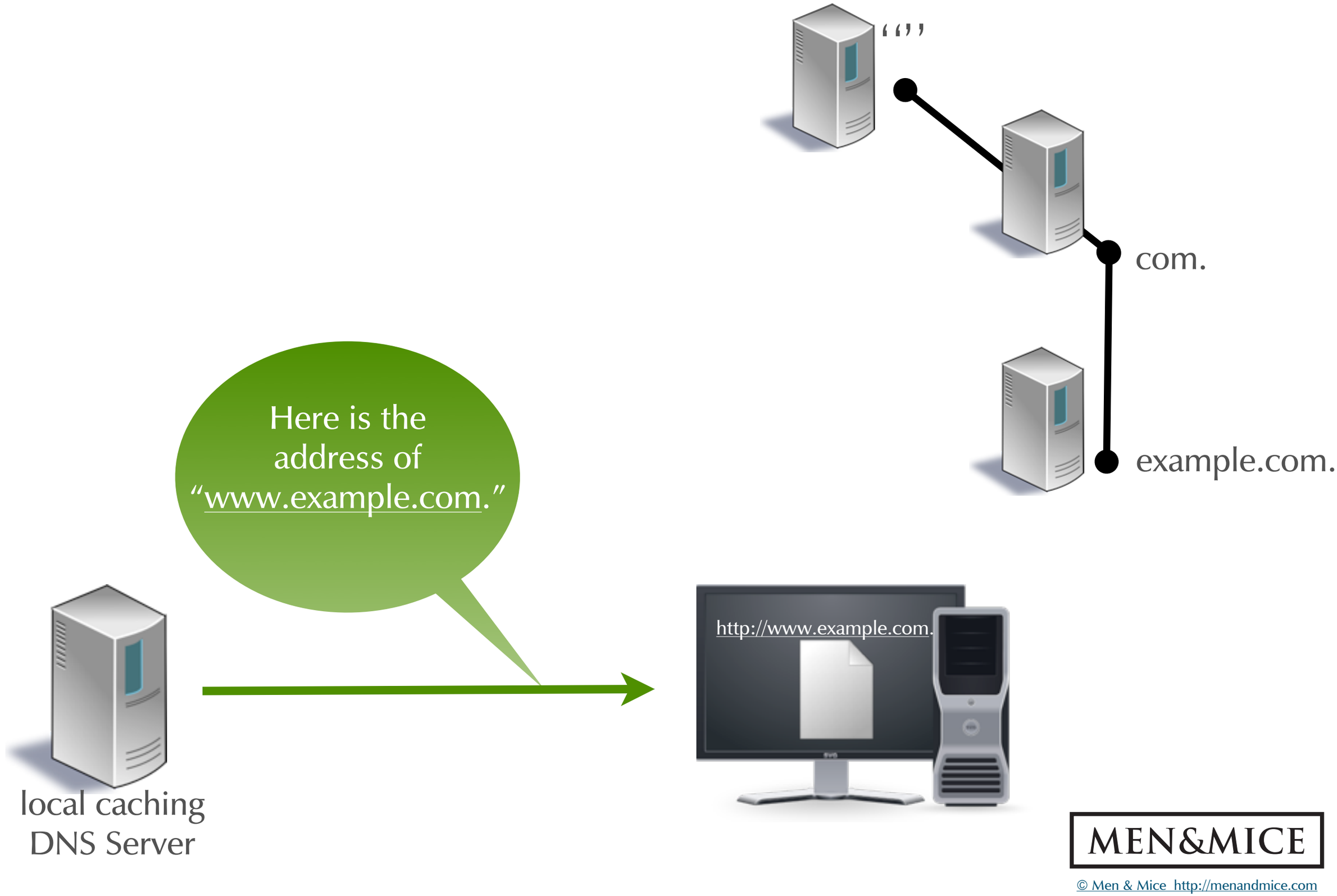
MEN&MICE

Name Resolution in Pictures



MEN&MICE

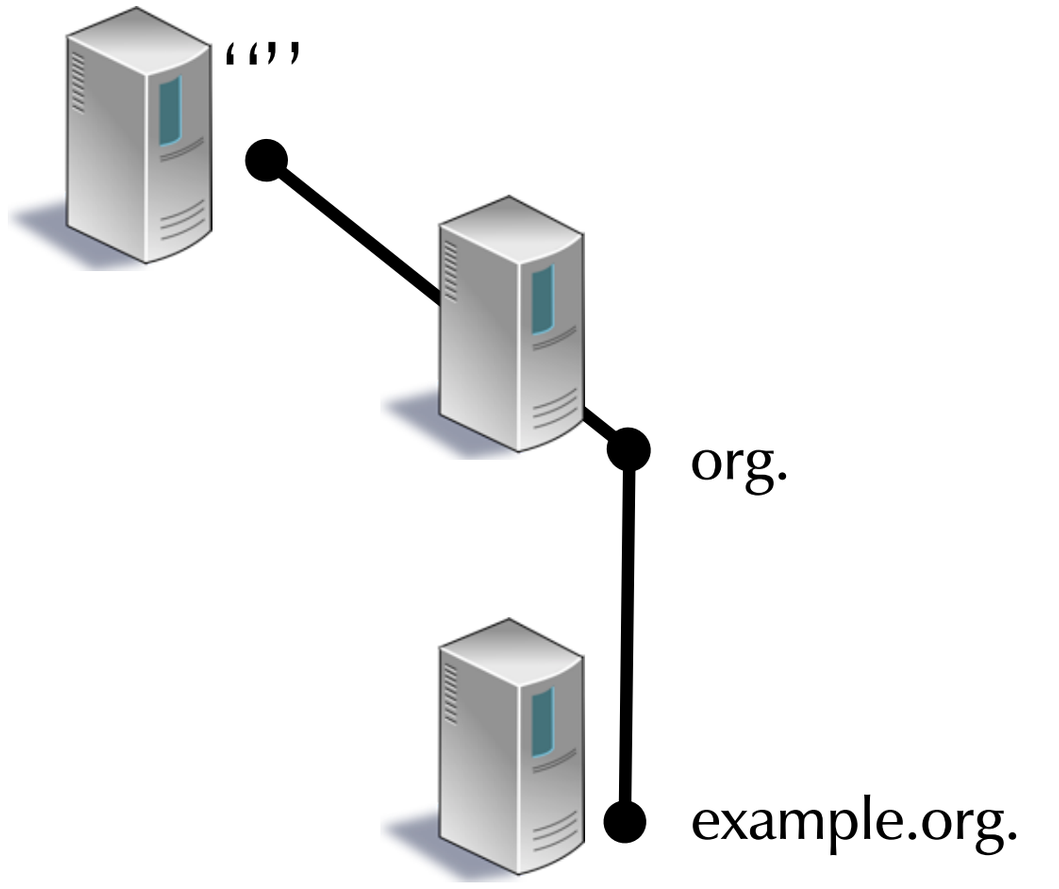
Name Resolution in Pictures



MEN&MICE

DNSSEC Name resolution (simplified)

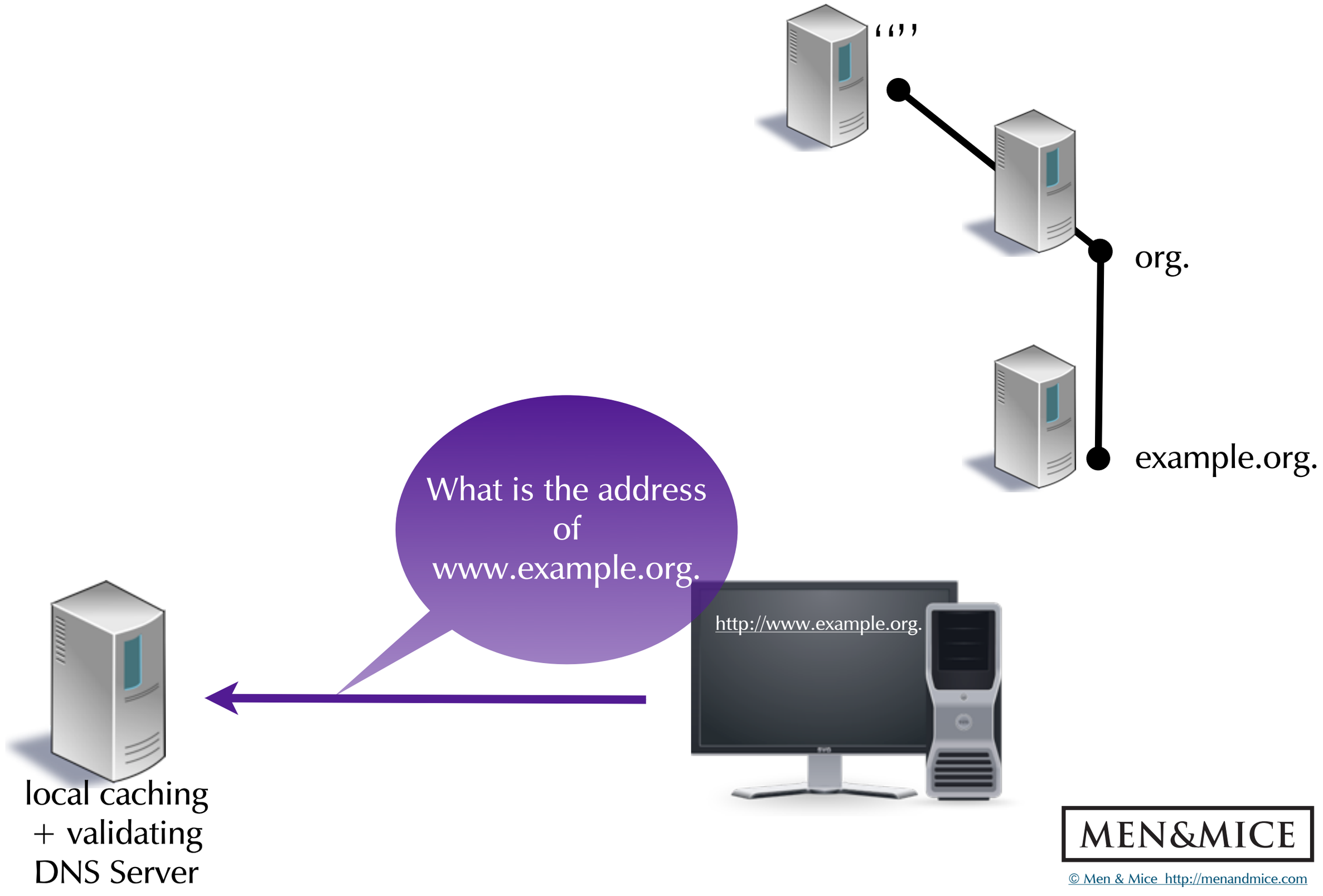
DNSSEC Name Resolution



local caching
+ validating
DNS Server

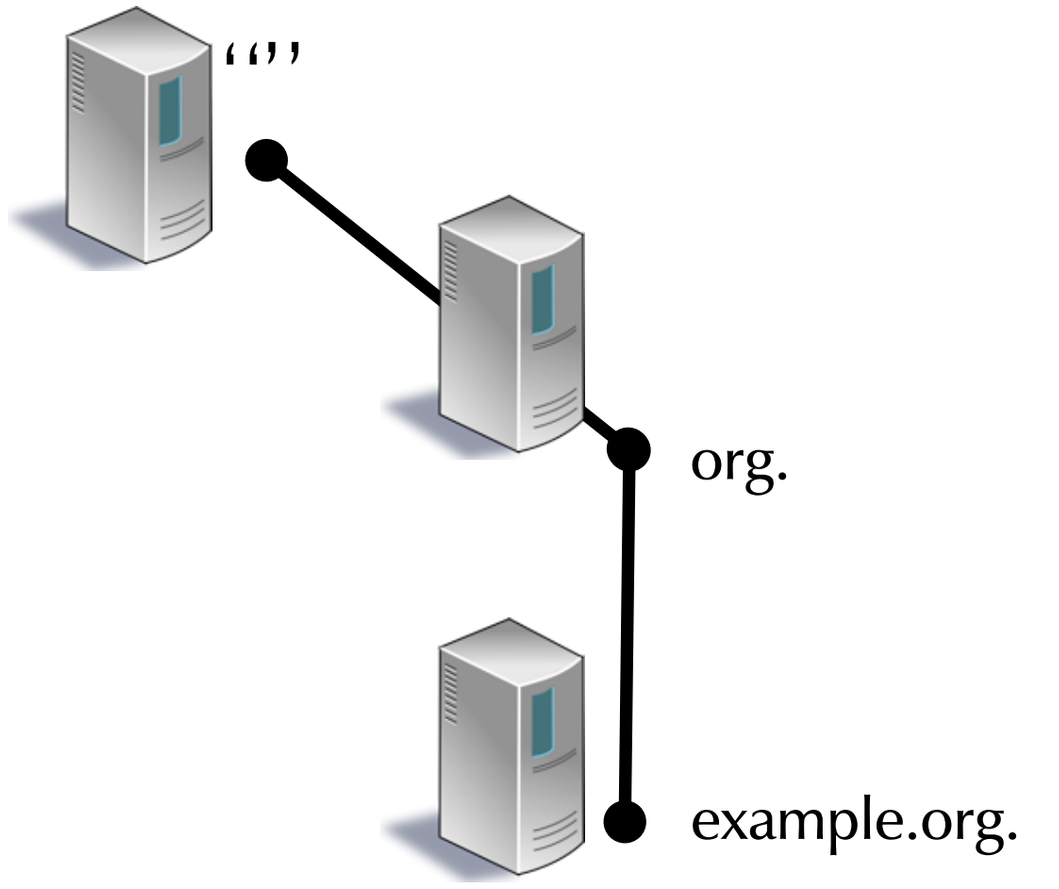
MEN&MICE

DNSSEC Name Resolution



MEN&MICE

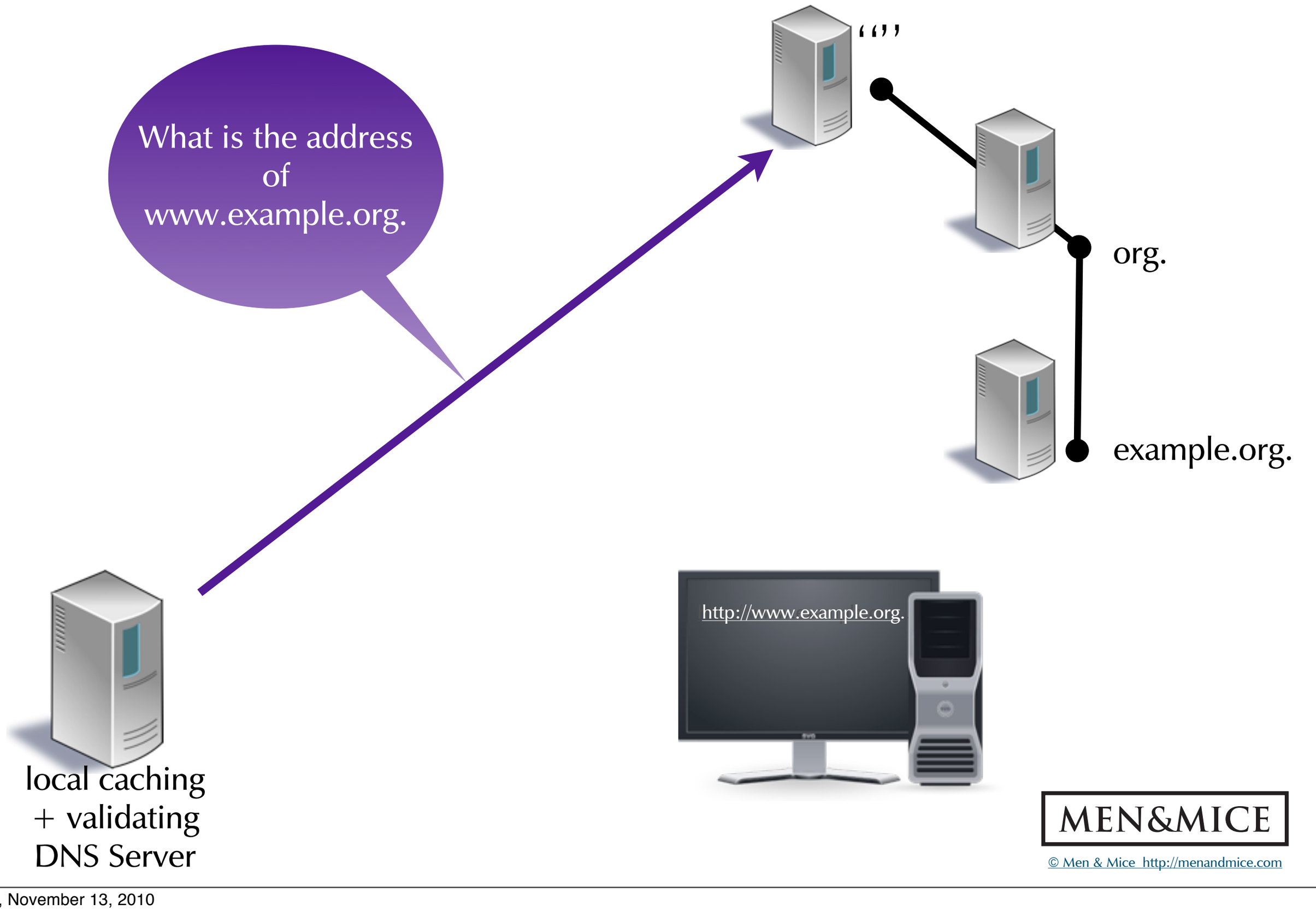
DNSSEC Name Resolution



local caching
+ validating
DNS Server

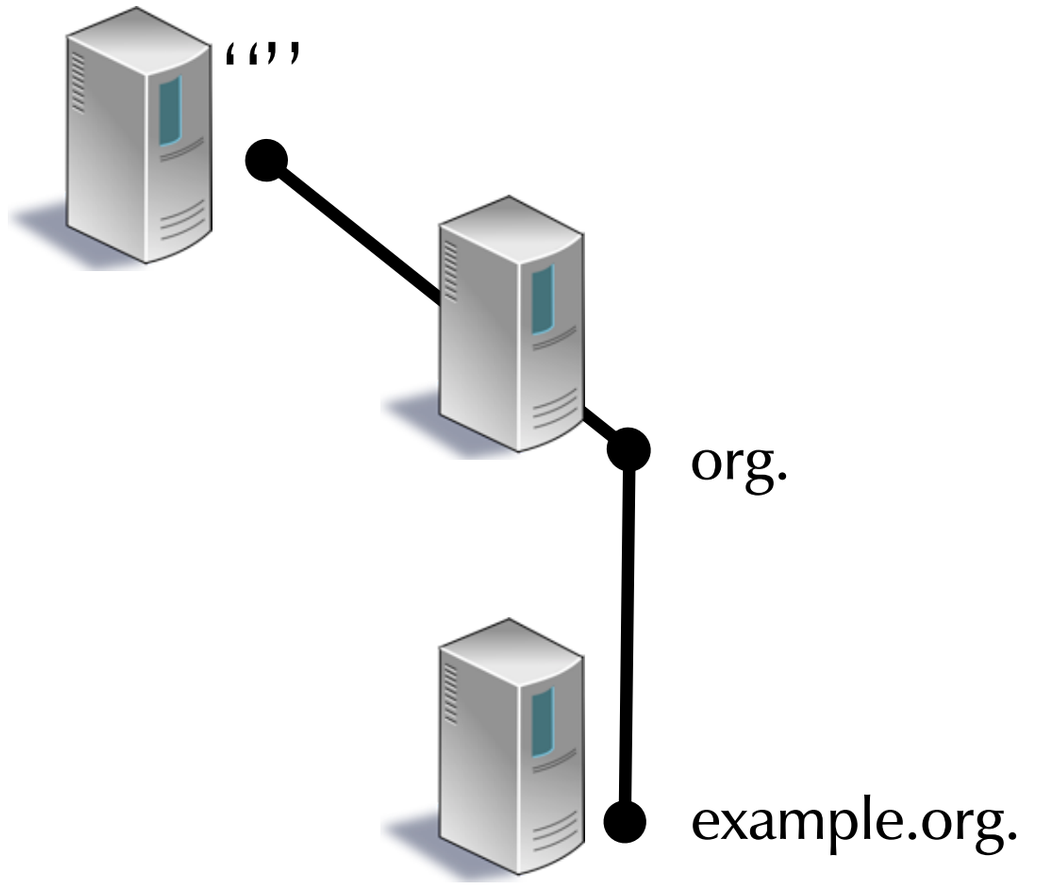
MEN&MICE

DNSSEC Name Resolution



MEN&MICE

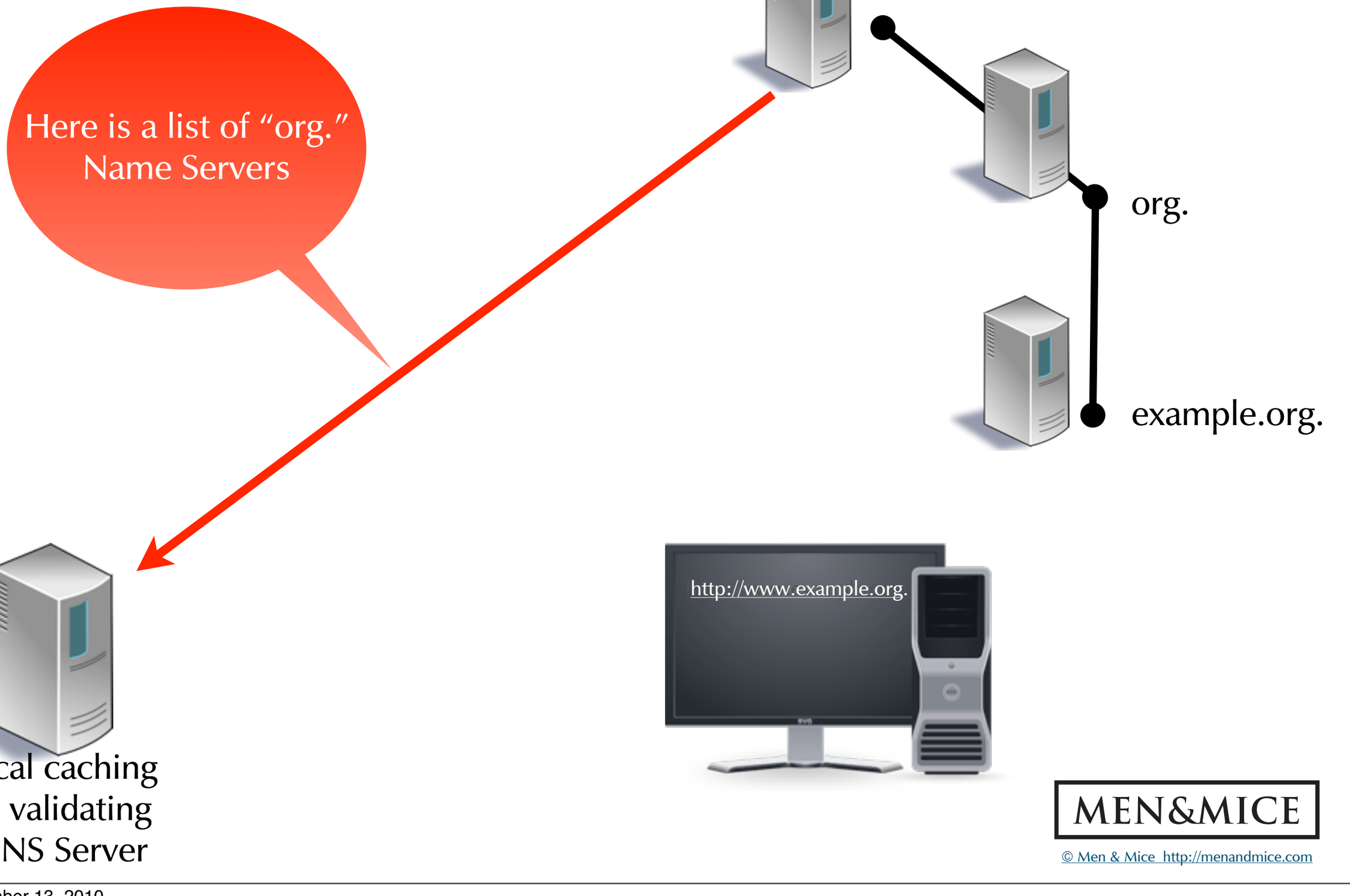
DNSSEC Name Resolution



local caching
+ validating
DNS Server

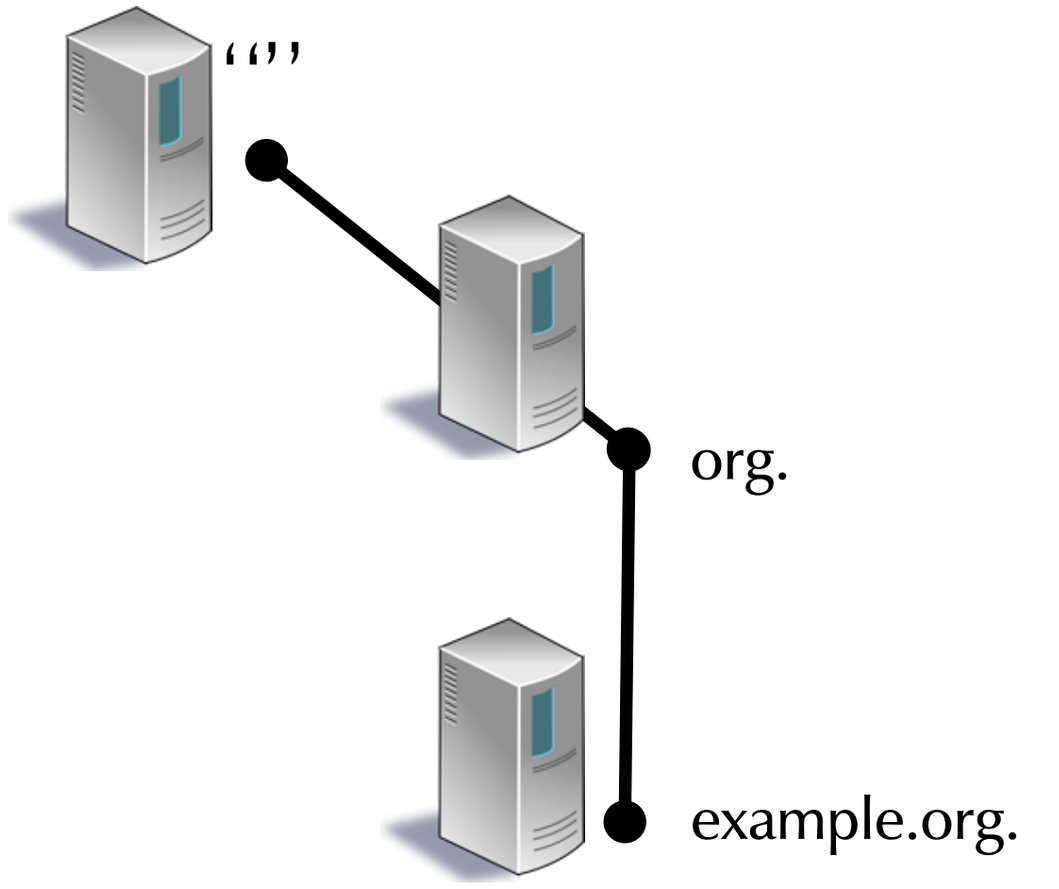
MEN&MICE

DNSSEC Name Resolution



MEN&MICE

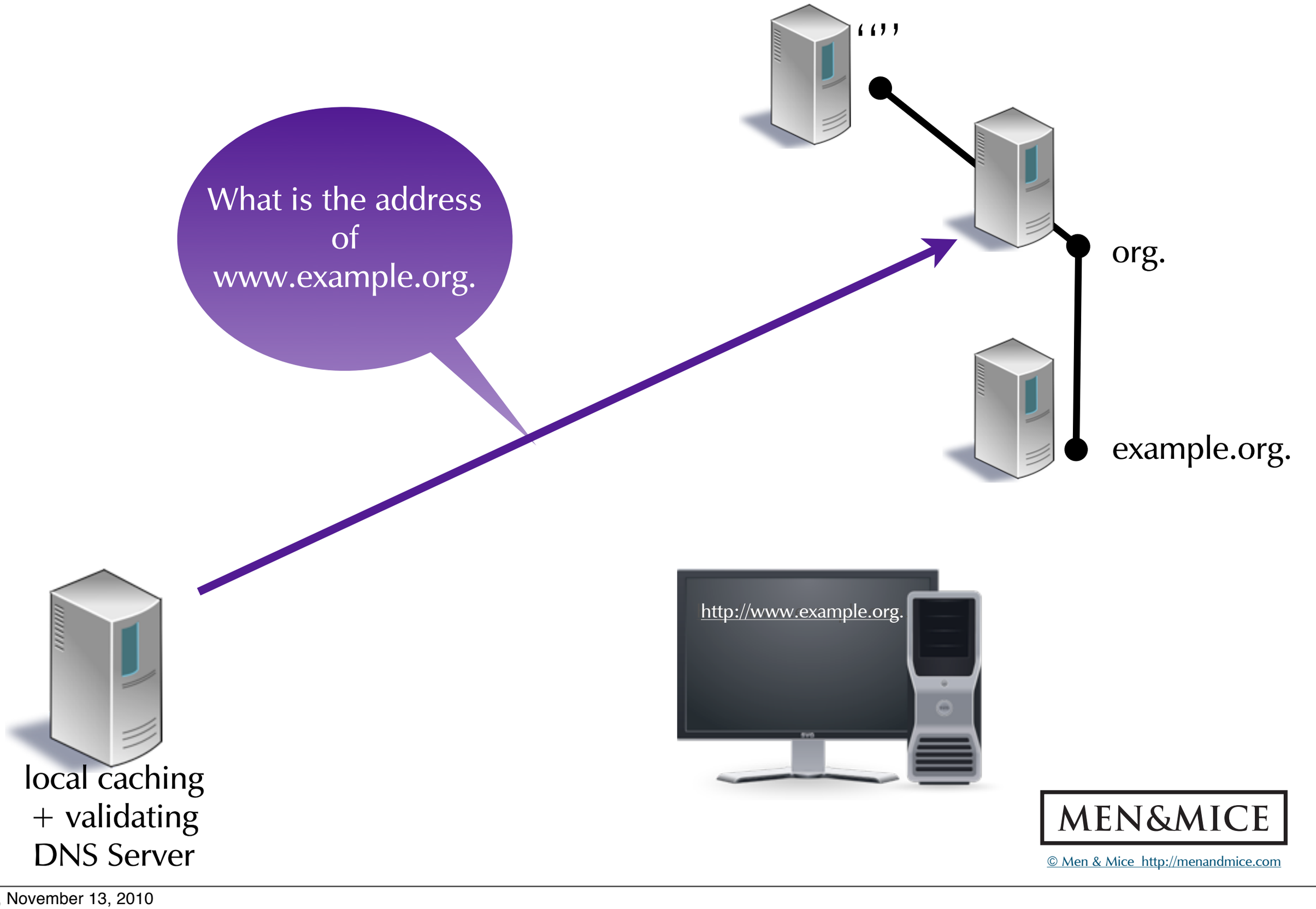
DNSSEC Name Resolution



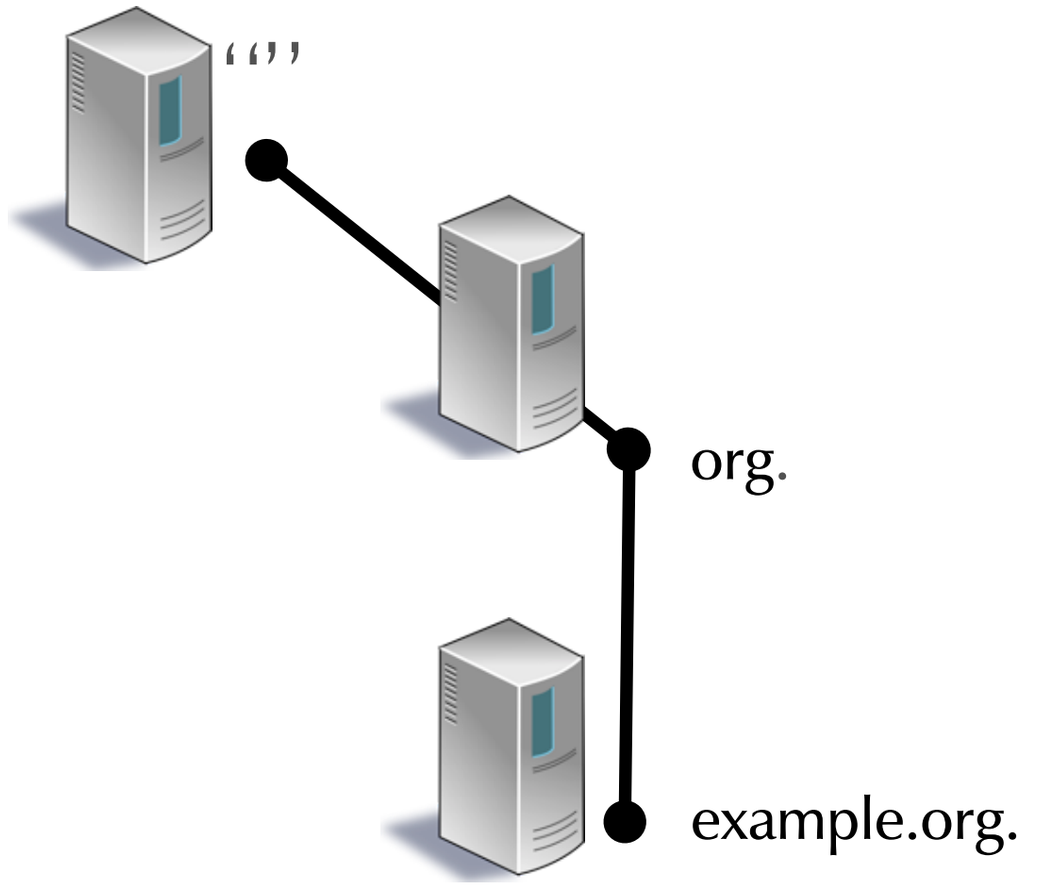
local caching
+ validating
DNS Server

MEN&MICE

DNSSEC Name Resolution



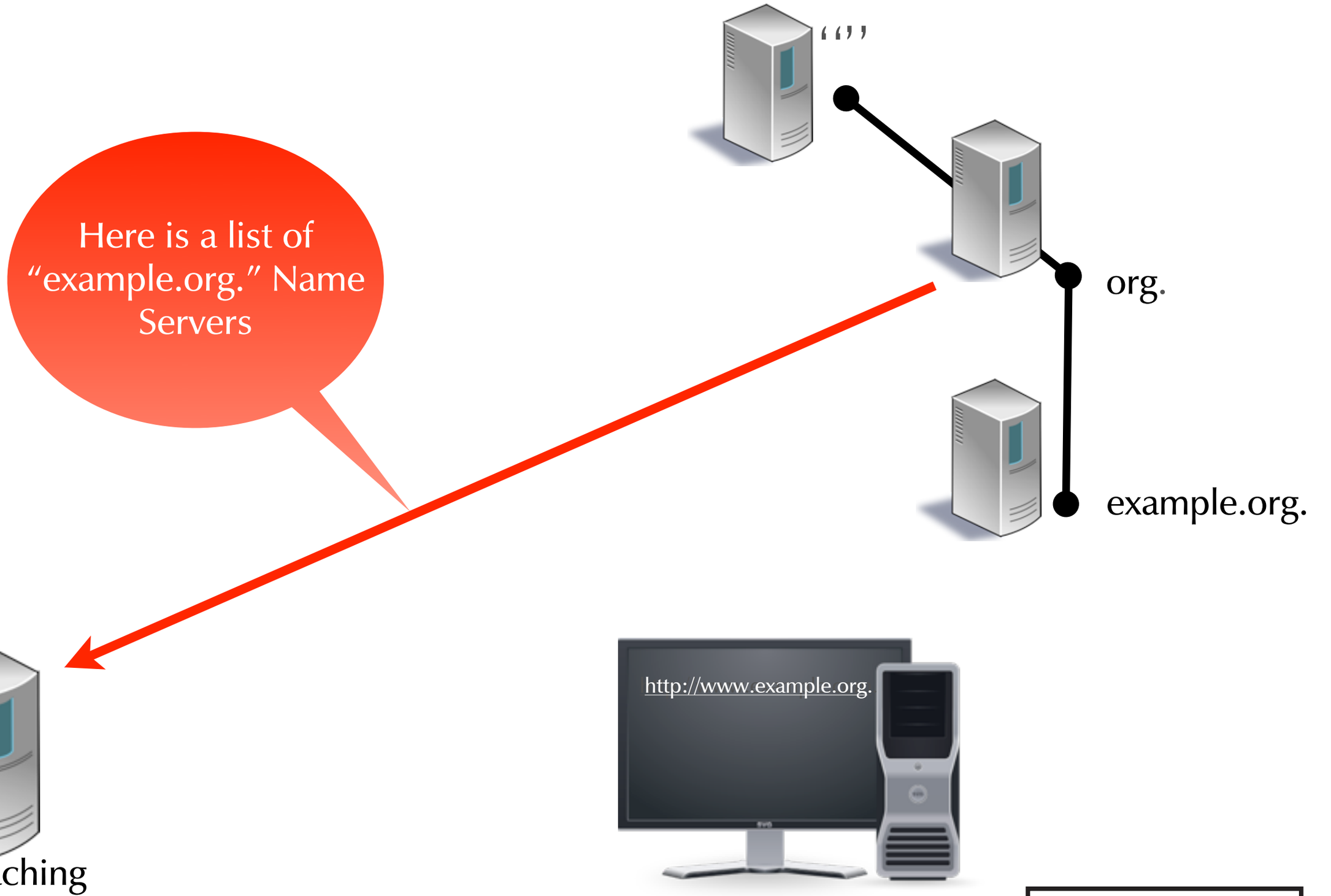
DNSSEC Name Resolution



local caching
+ validating
DNS Server

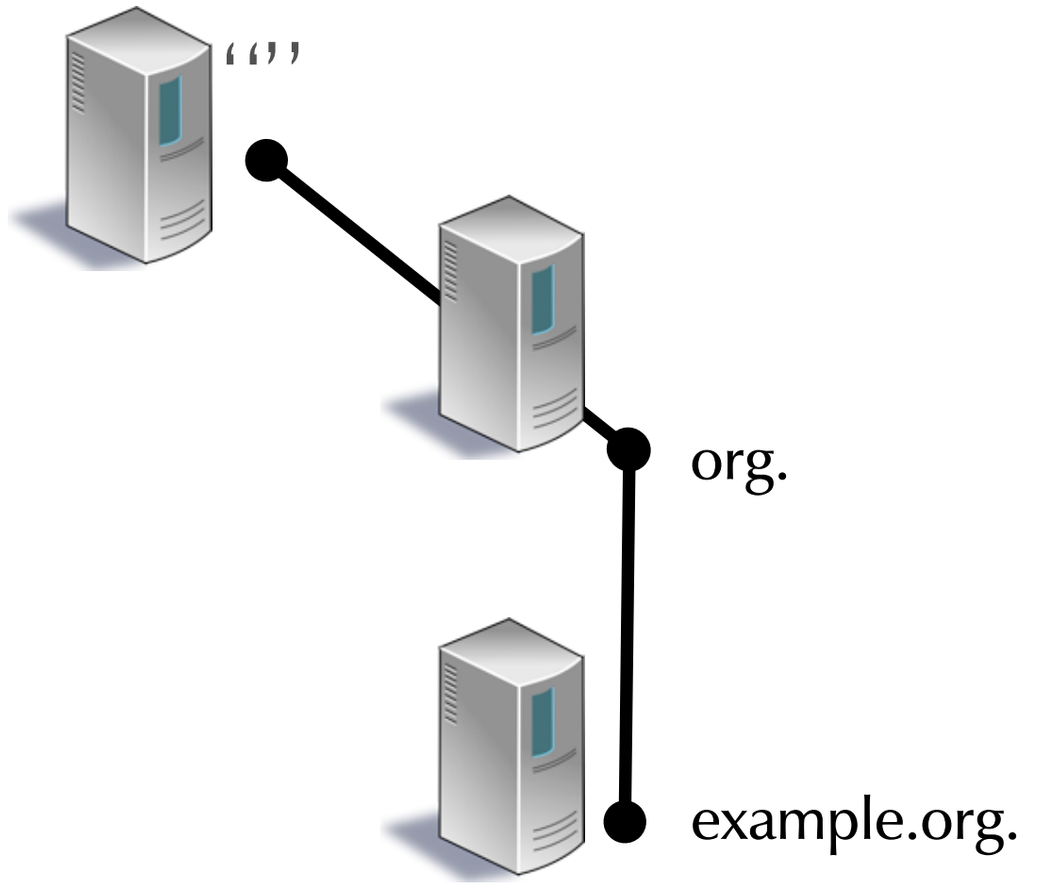
MEN&MICE

DNSSEC Name Resolution



MEN&MICE

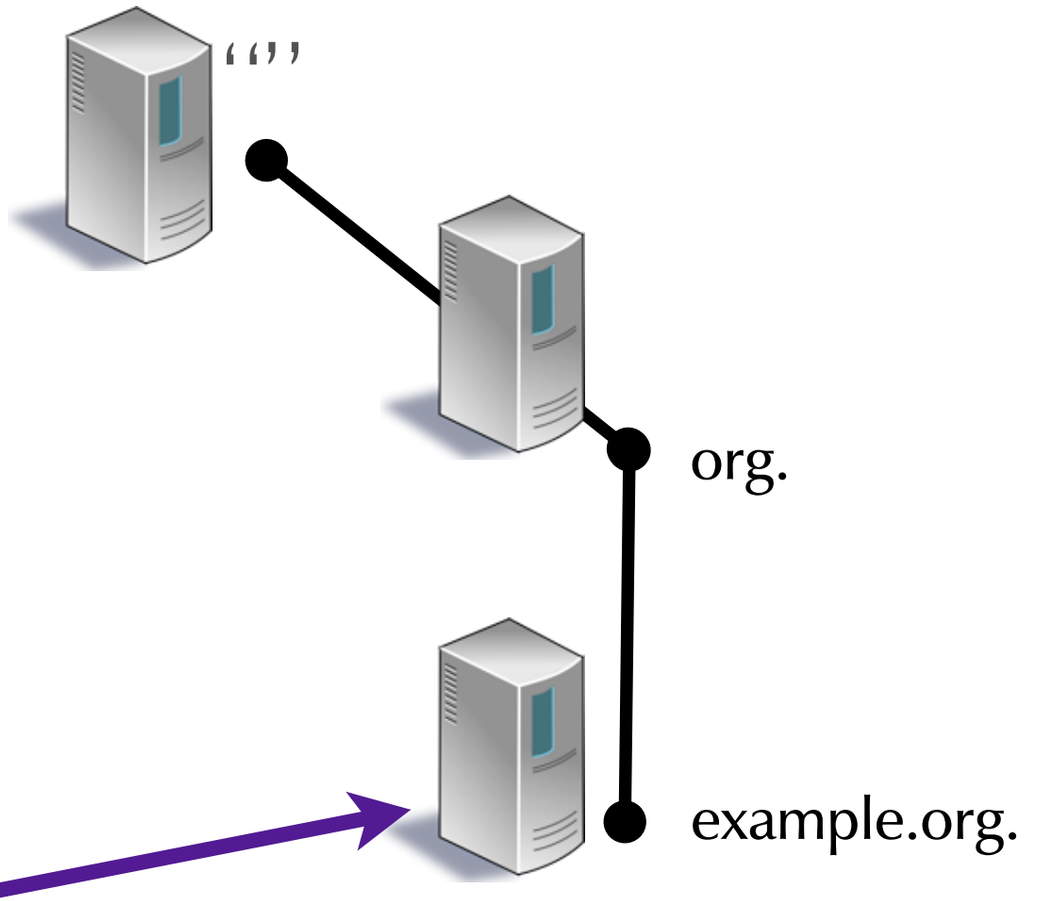
DNSSEC Name Resolution



local caching
+ validating
DNS Server

MEN&MICE

DNSSEC Name Resolution

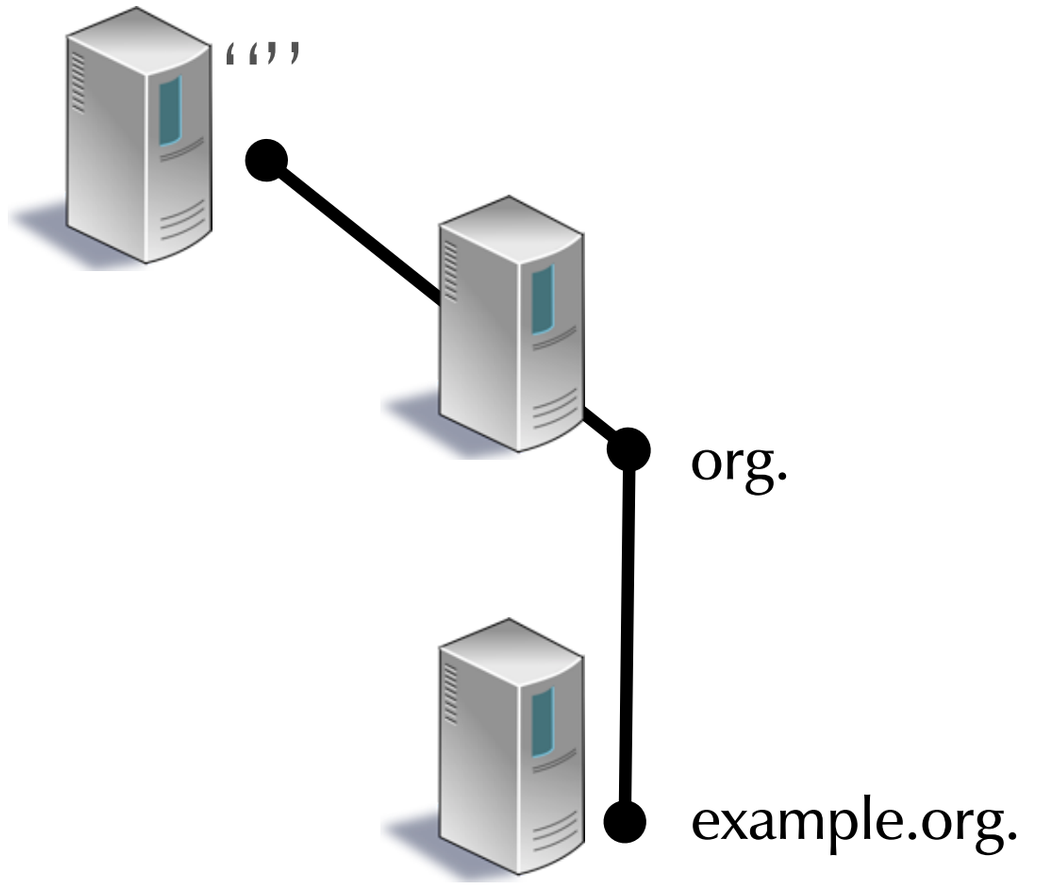


What is the address of www.example.org.

local caching + validating DNS Server

MEN&MICE

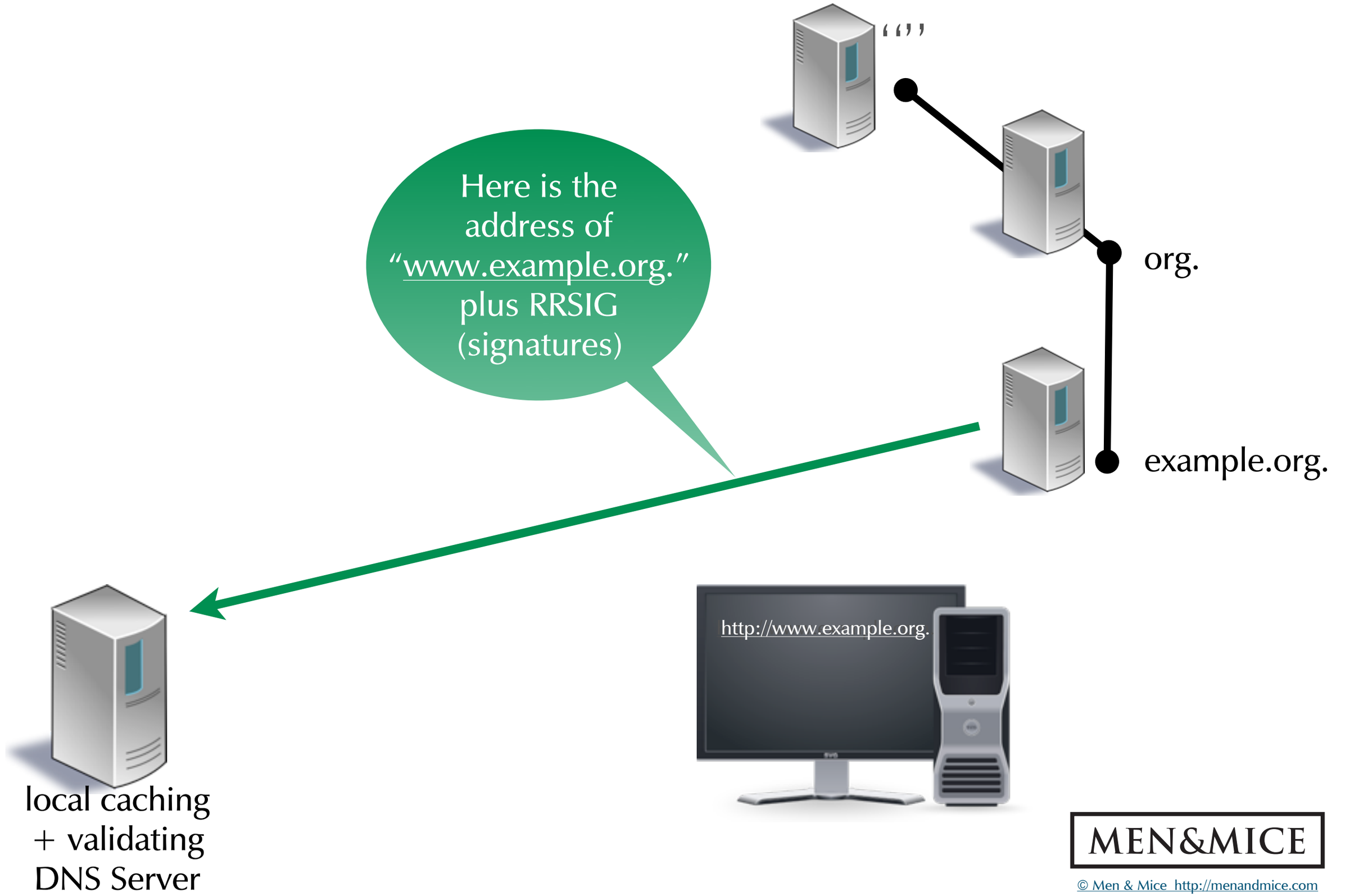
DNSSEC Name Resolution



local caching
+ validating
DNS Server

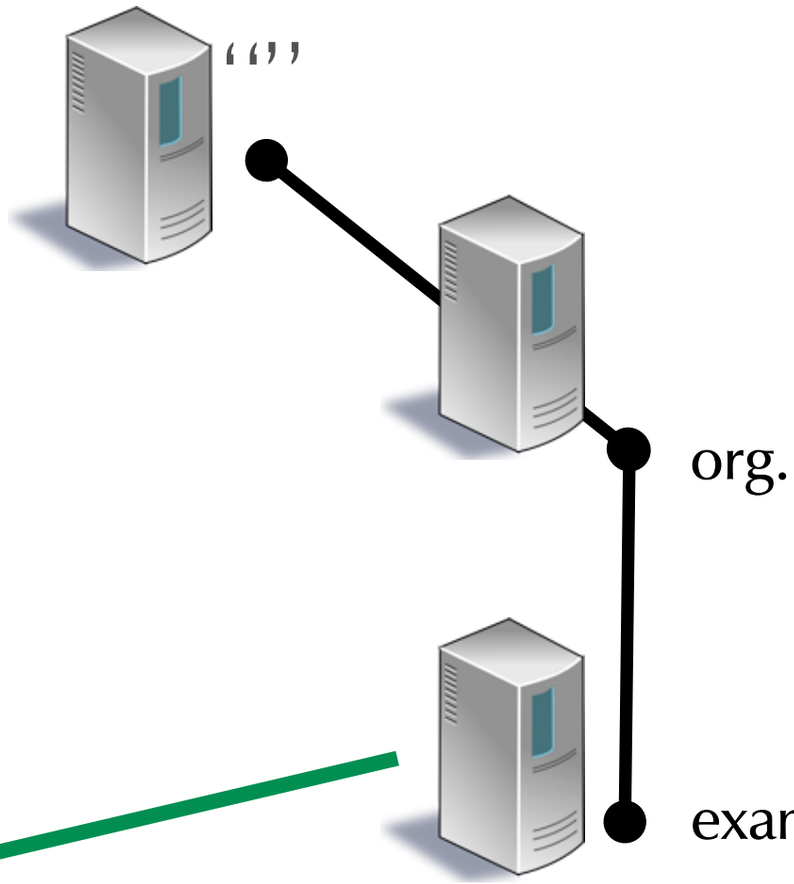
MEN&MICE

DNSSEC Name Resolution



MEN&MICE

DNSSEC Name Resolution



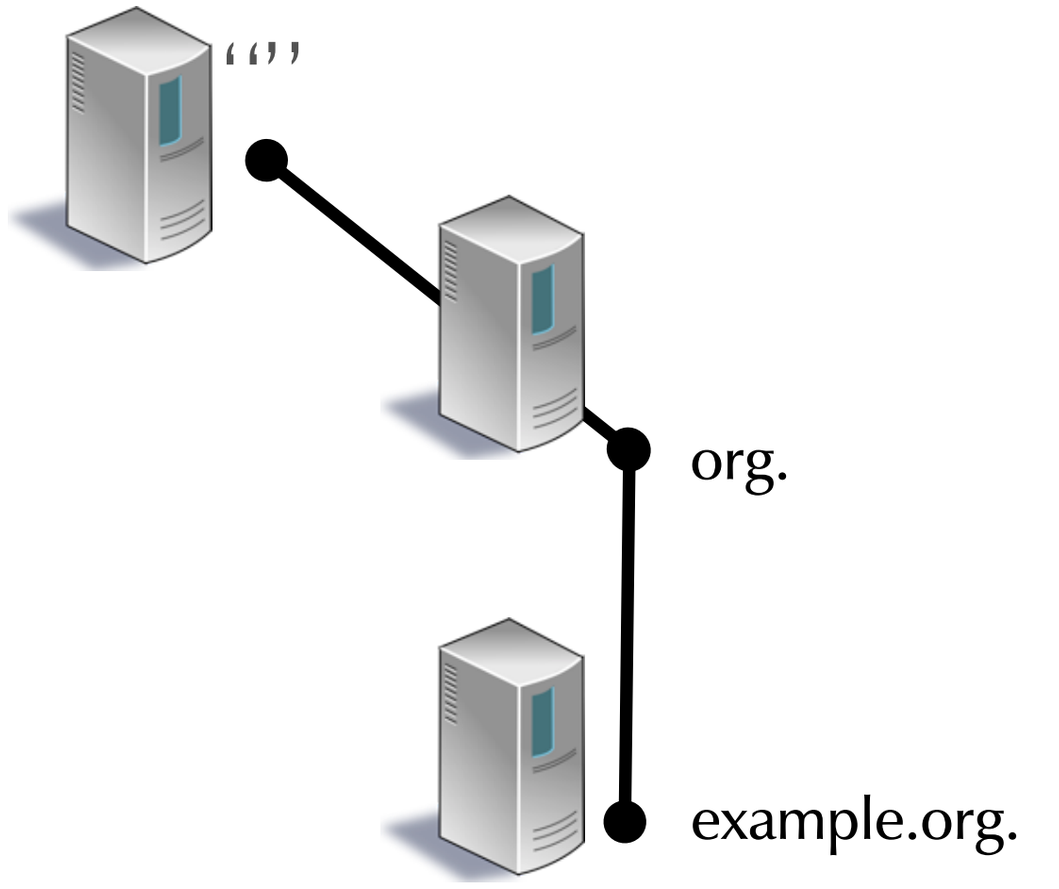
Here is the address of “www.example.org.” plus RRSIG (signatures)

Record	Function
www.example.org.A	IPv4 Address
www.example.org. RRSIG	signature ↑

local caching + validating DNS Server

MEN&MICE

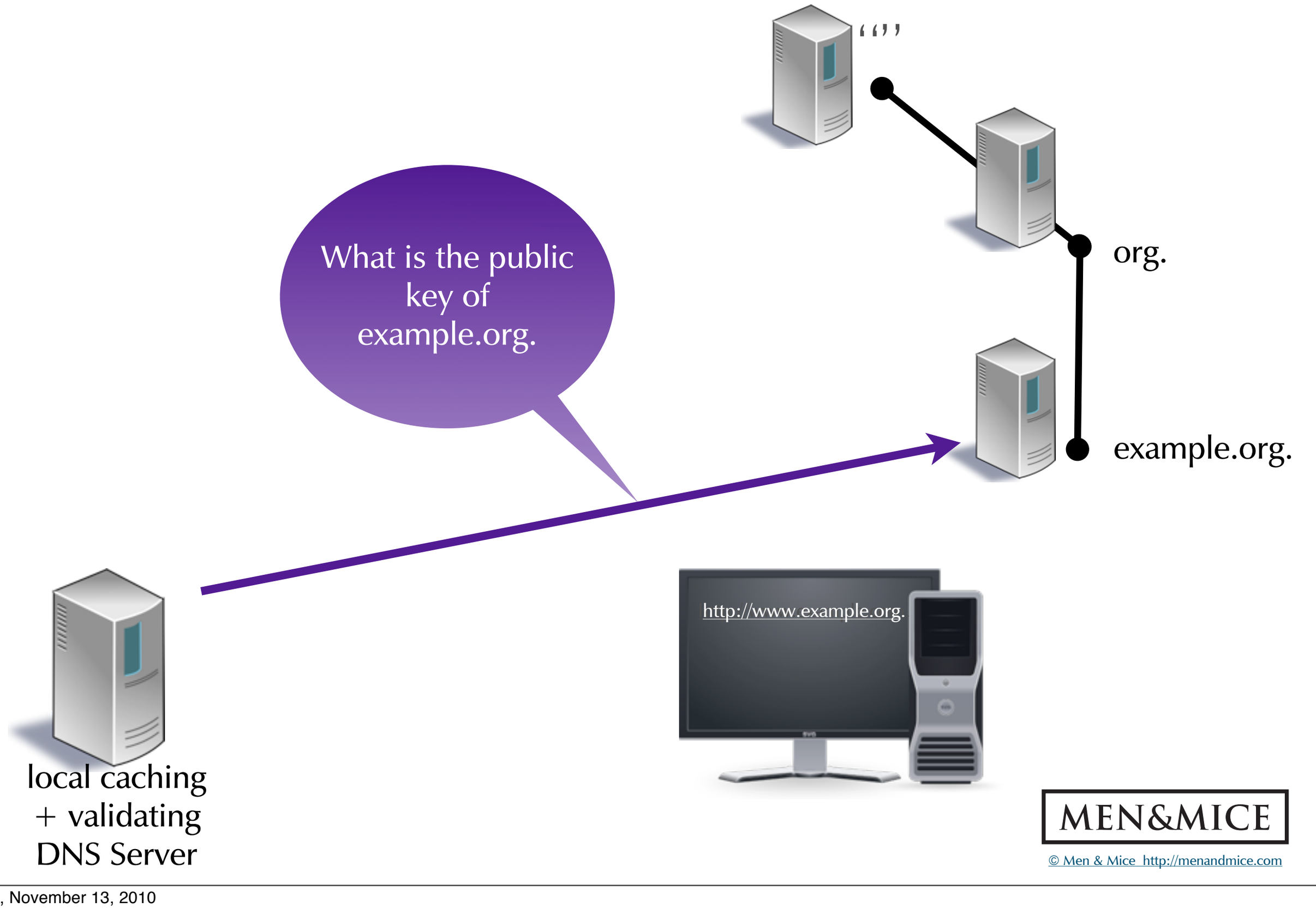
DNSSEC Name Resolution



local caching
+ validating
DNS Server

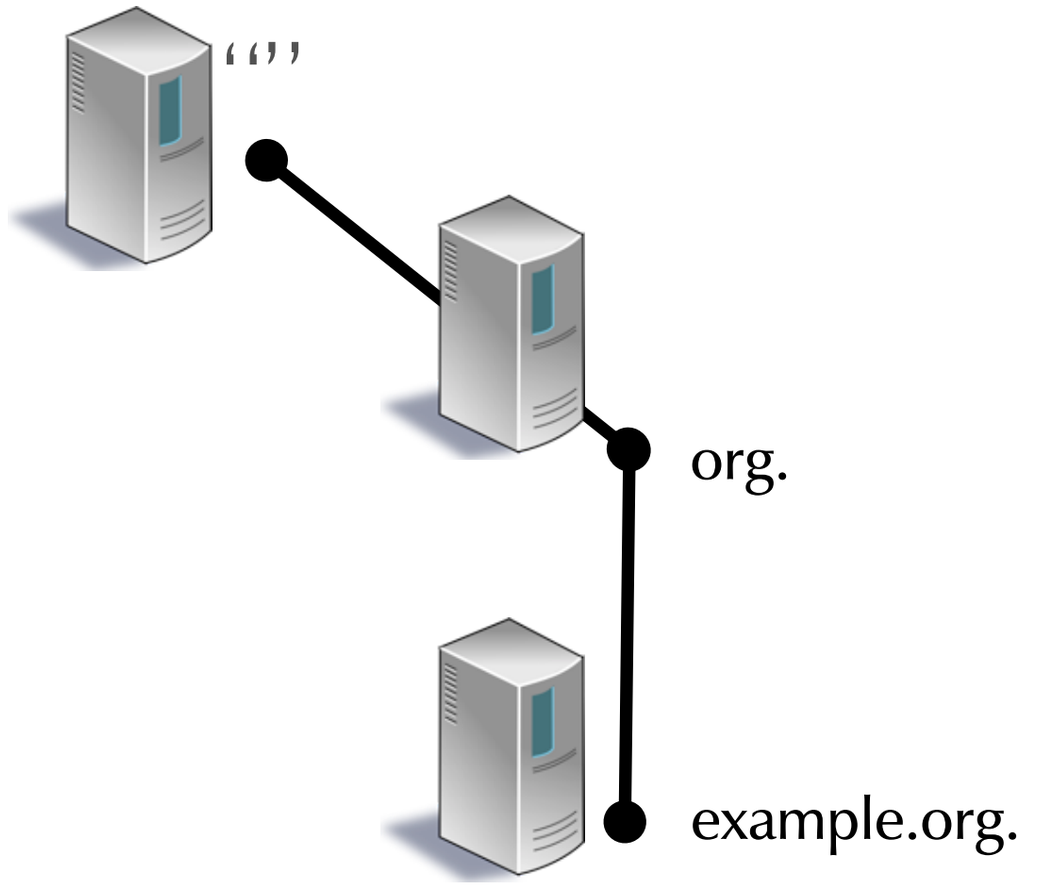
MEN&MICE

DNSSEC Name Resolution



MEN&MICE

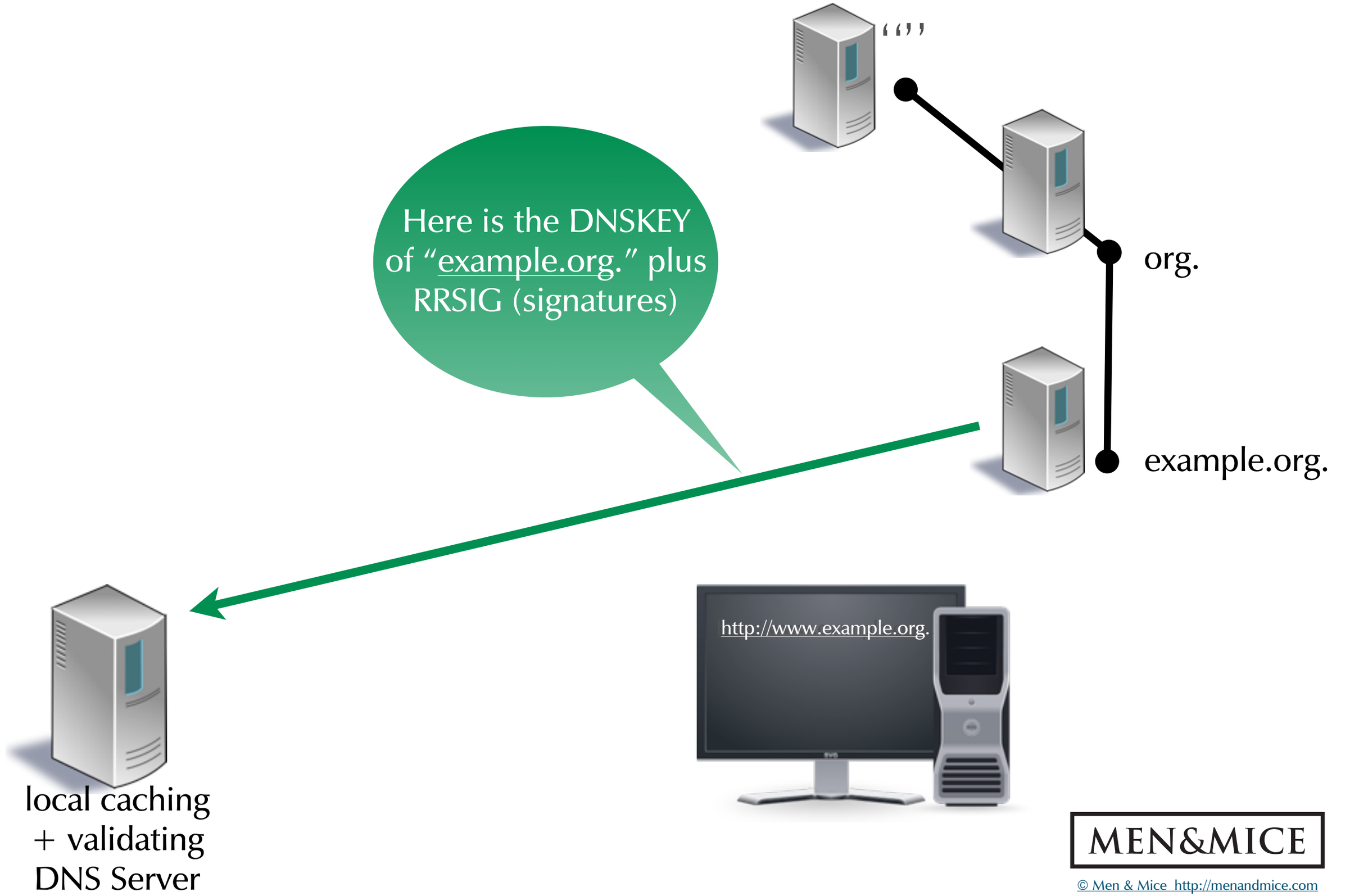
DNSSEC Name Resolution



local caching
+ validating
DNS Server

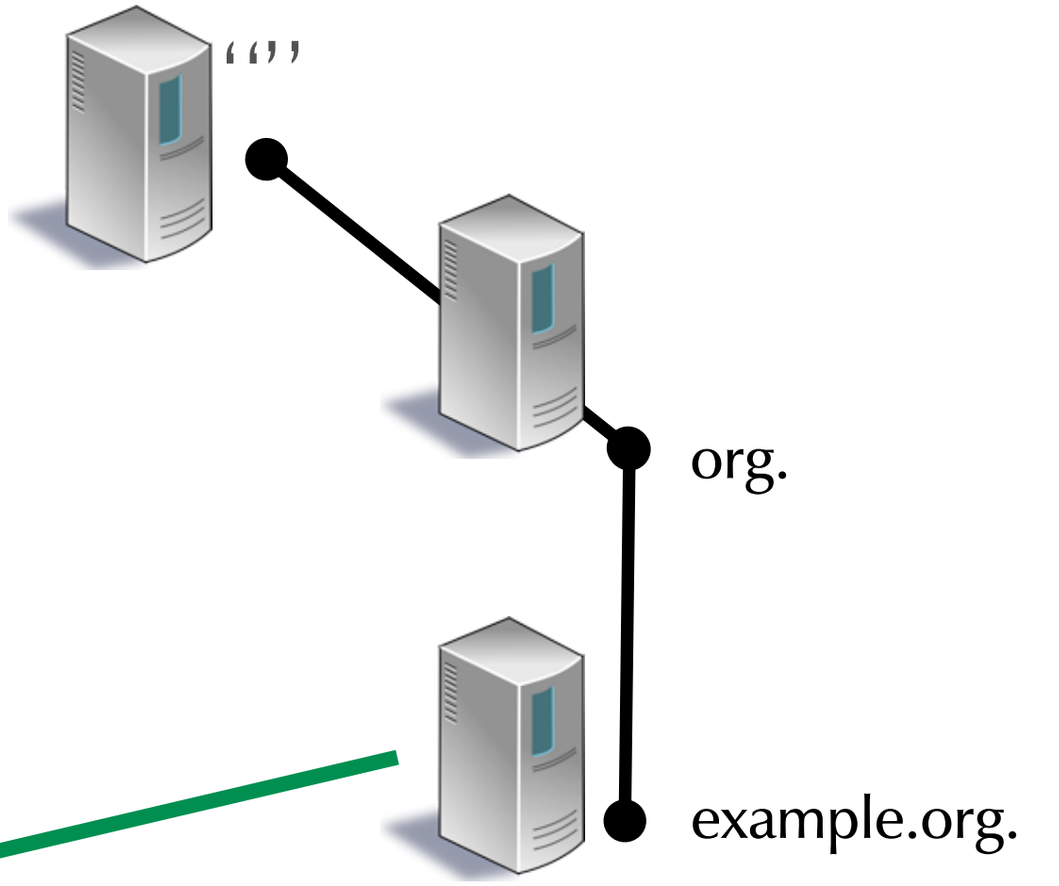
MEN&MICE

DNSSEC Name Resolution



MEN&MICE

DNSSEC Name Resolution



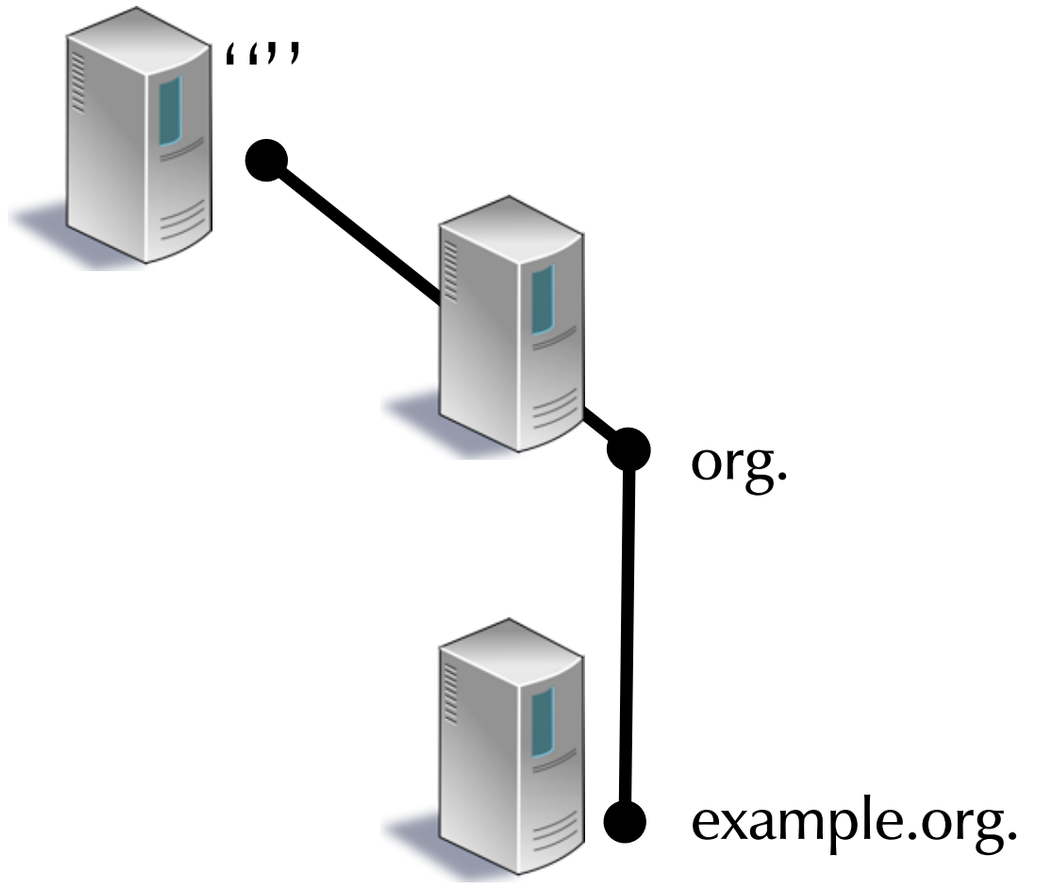
Here is the DNSKEY of “example.org.” plus RRSIG (signatures)

Record	Function
www.example.org.A	IPv4 Address
www.example.org. RRSIG	signature ↑
example.org. DNSKEY	public key
example.org. RRSIG	signature ↑

local caching + validating DNS Server

MEN&MICE

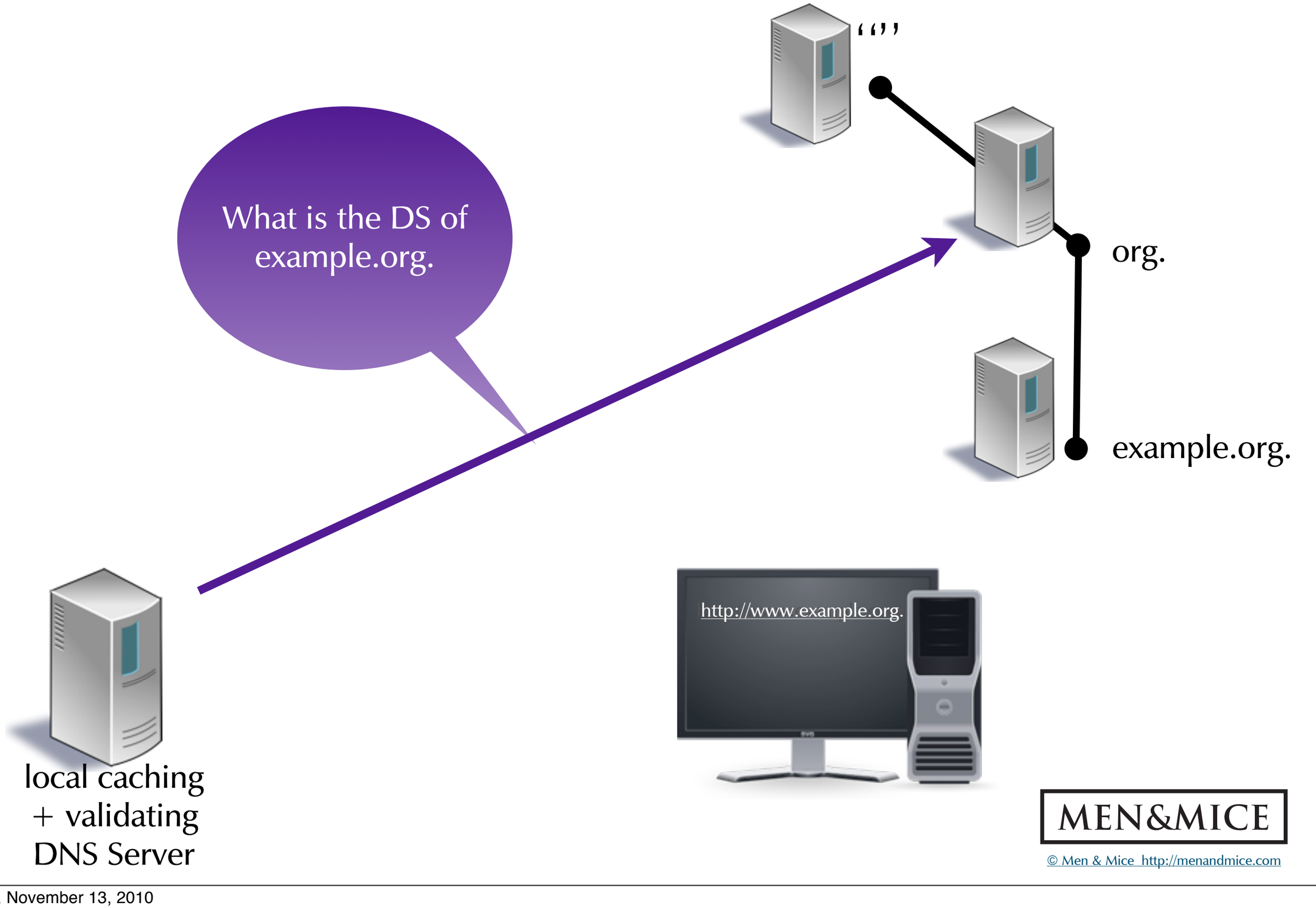
DNSSEC Name Resolution



local caching
+ validating
DNS Server

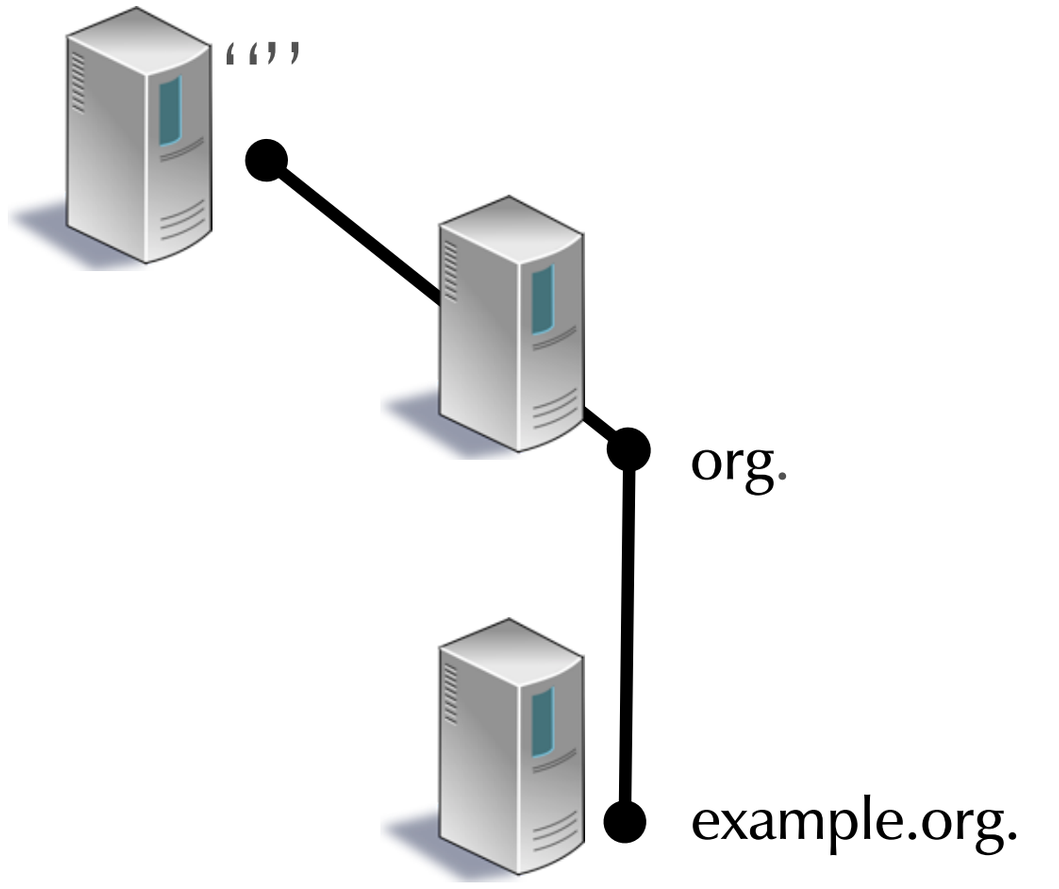
MEN&MICE

DNSSEC Name Resolution



MEN&MICE

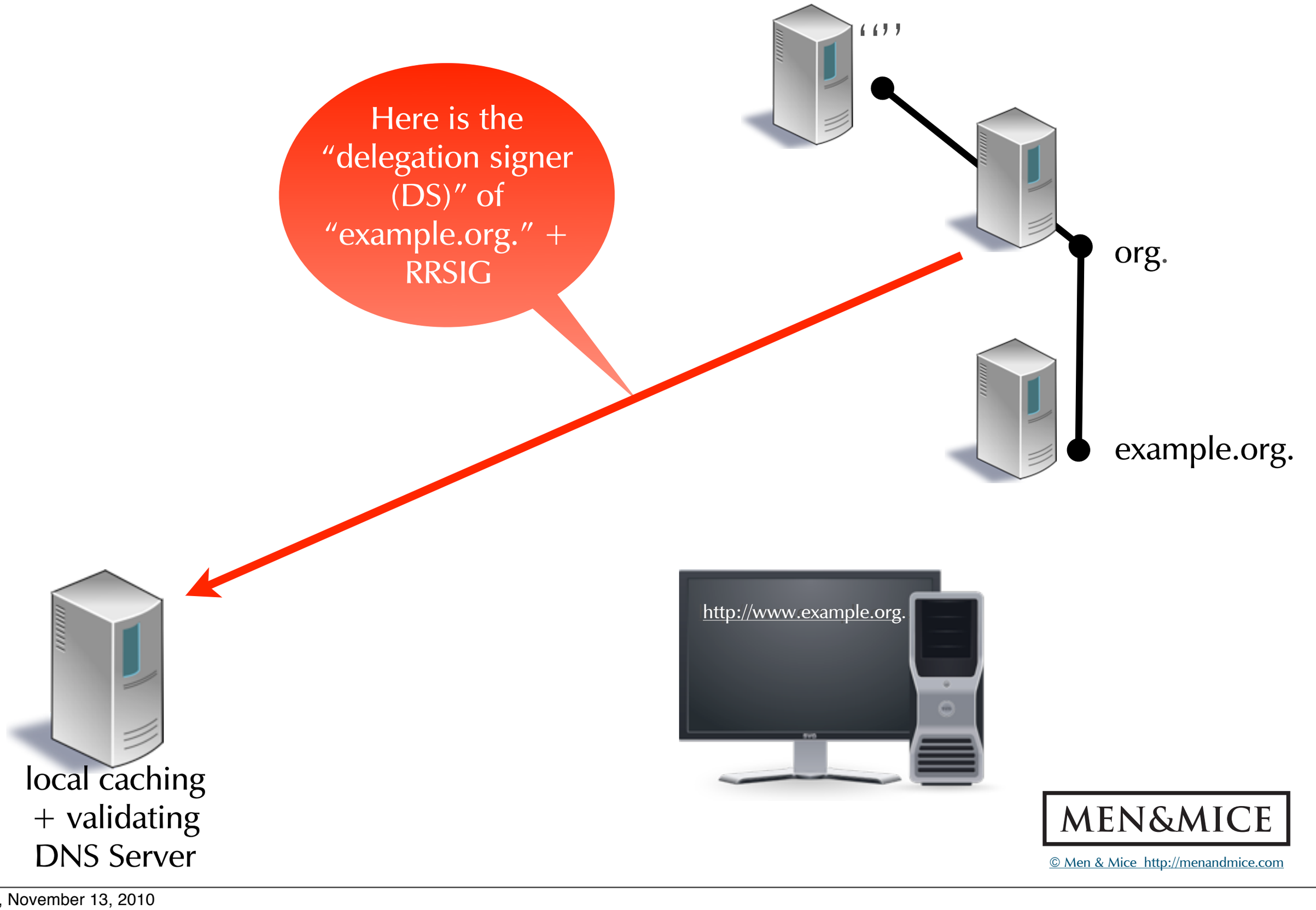
DNSSEC Name Resolution



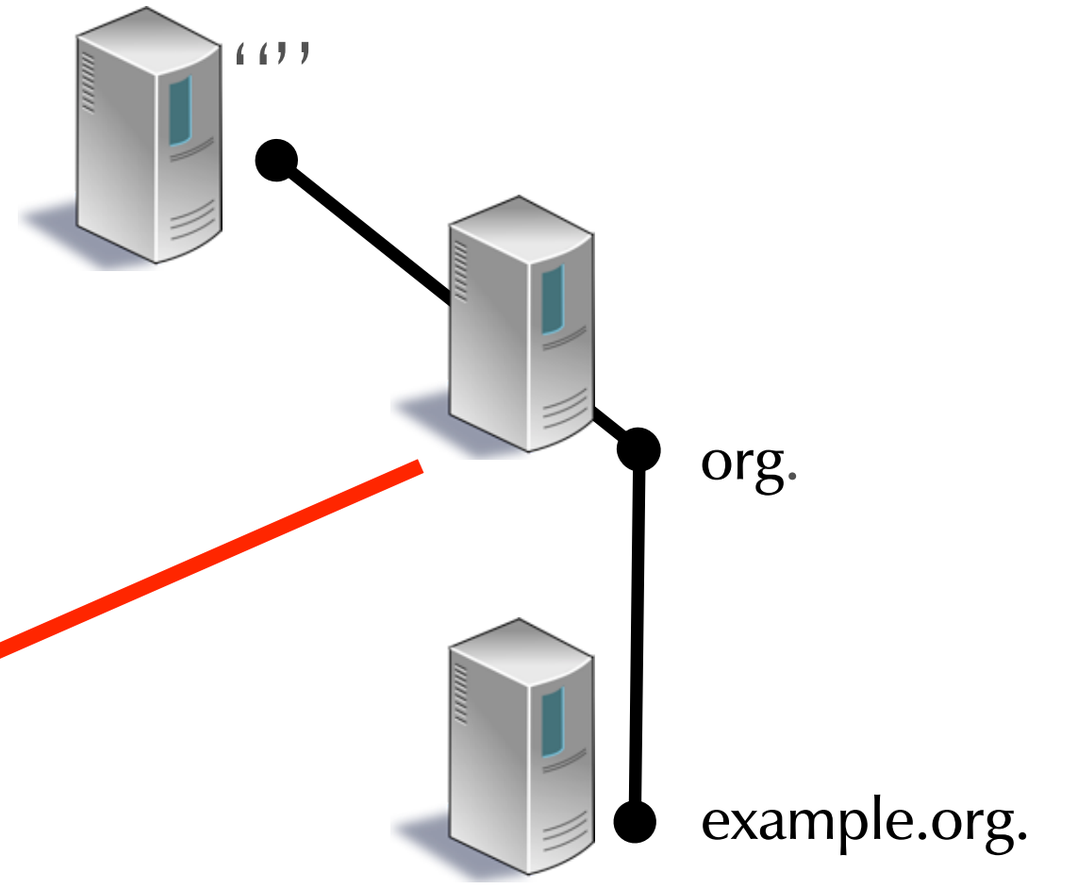
local caching
+ validating
DNS Server

MEN&MICE

DNSSEC Name Resolution



DNSSEC Name Resolution



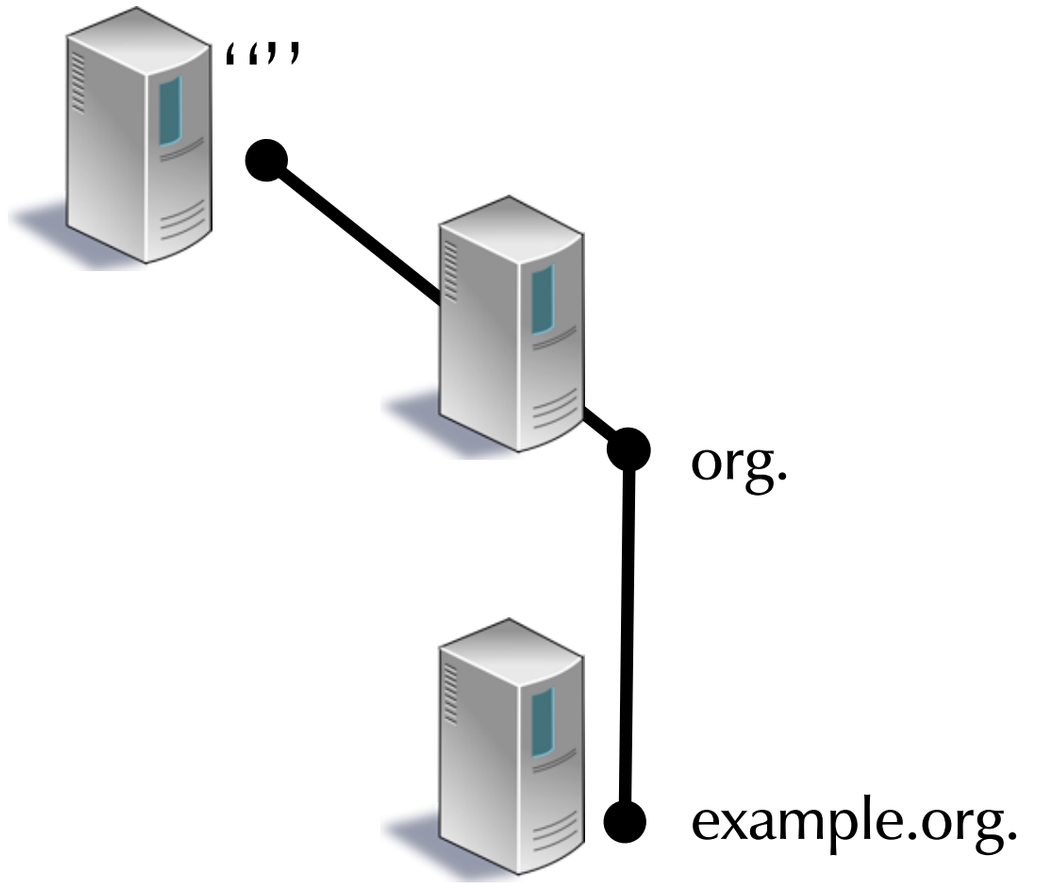
Here is the “delegation signer (DS)” of “example.org.” + RRSIG

Record	Function
www.example.org.A	IPv4 Address
www.example.org. RRSIG	signature ↑
example.org. DNSKEY	public key
example.org. RRSIG	signature ↑
example.org. DS	hash of public key
org. RRSIG	signature ↑

local caching + validating DNS Server

MEN&MICE

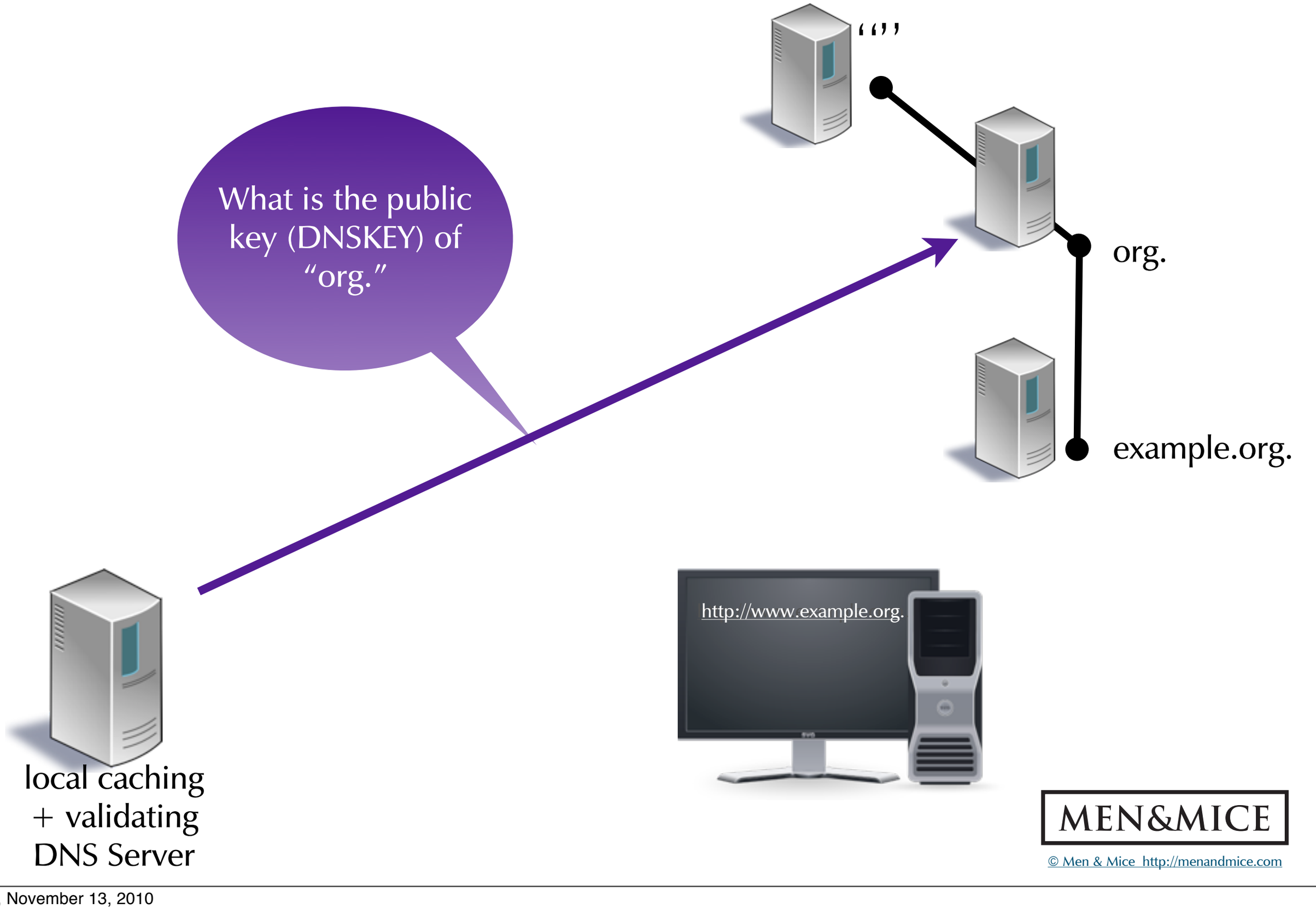
DNSSEC Name Resolution



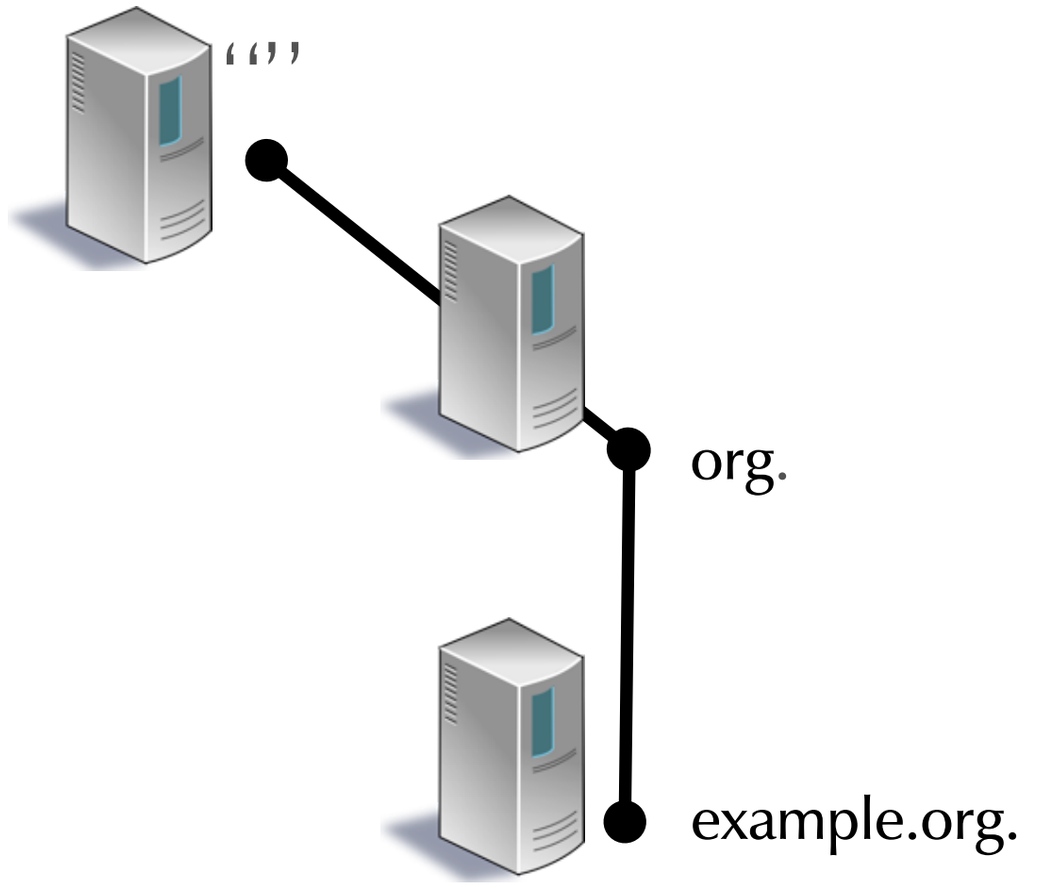
local caching
+ validating
DNS Server

MEN&MICE

DNSSEC Name Resolution



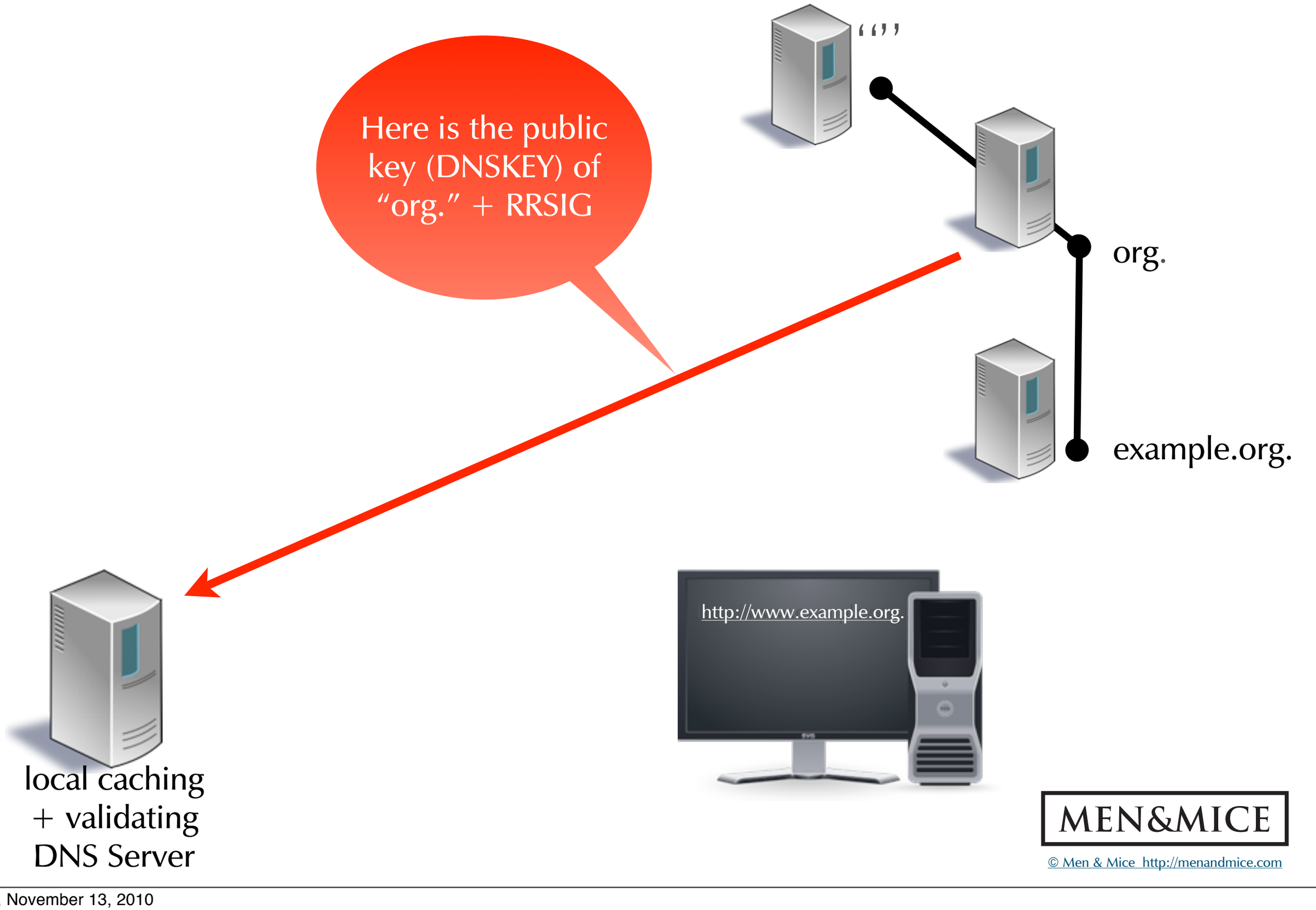
DNSSEC Name Resolution



local caching
+ validating
DNS Server

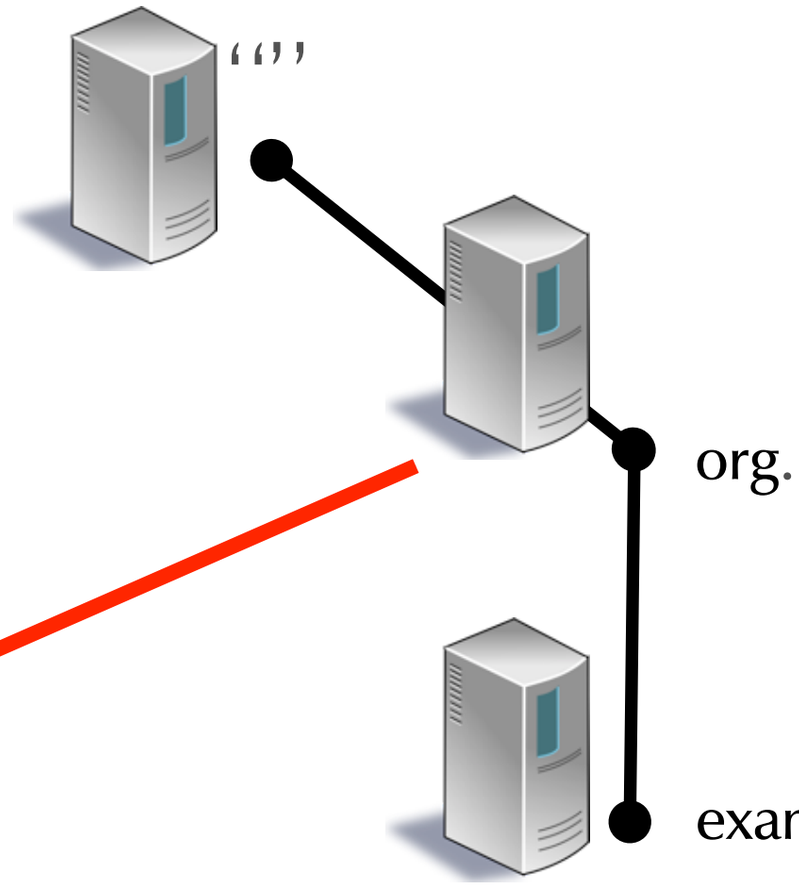
MEN&MICE

DNSSEC Name Resolution



MEN&MICE

DNSSEC Name Resolution



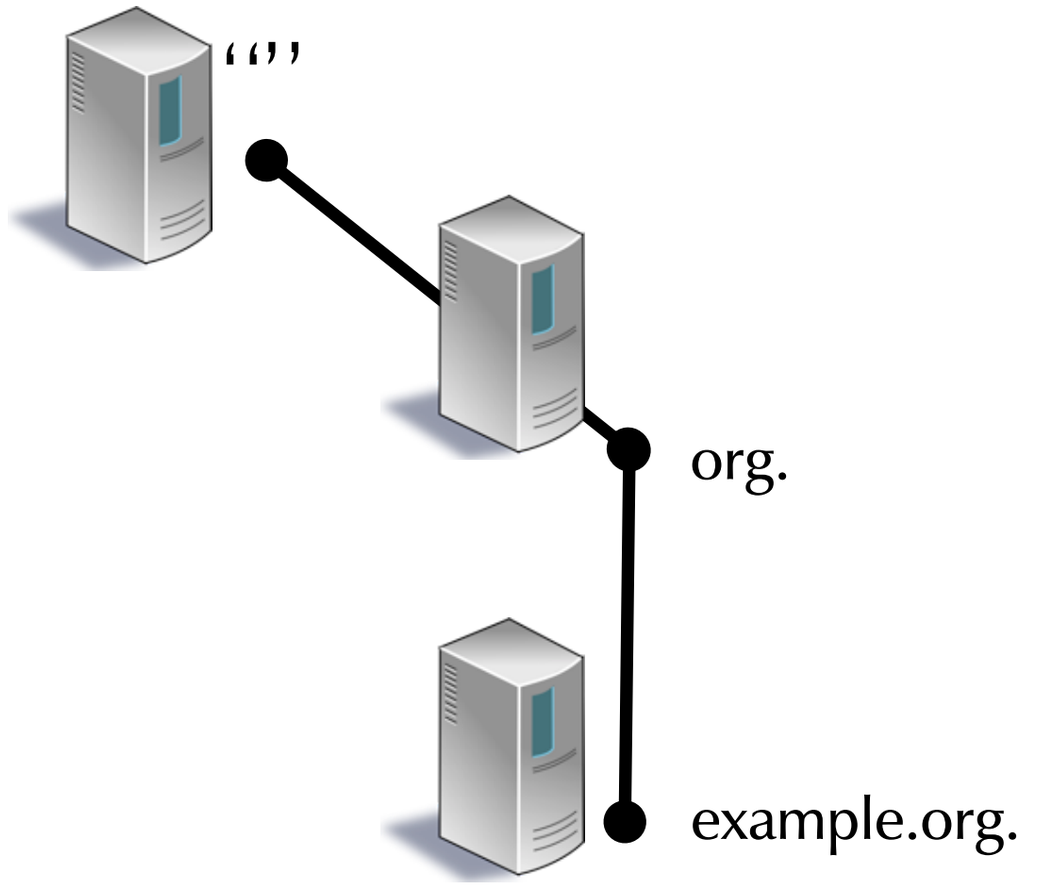
Here is the public key (DNSKEY) of “org.” + RRSIG

Record	Function
www.example.org.A	IPv4 Address
www.example.org. RRSIG	signature ↑
example.org. DNSKEY	public key
example.org. RRSIG	signature ↑
example.org. DS	hash of public key
org. RRSIG	signature ↑
org DNSKEY	public key
org RRSIG	signature ↑

local caching
+ validating
DNS Server

MEN&MICE

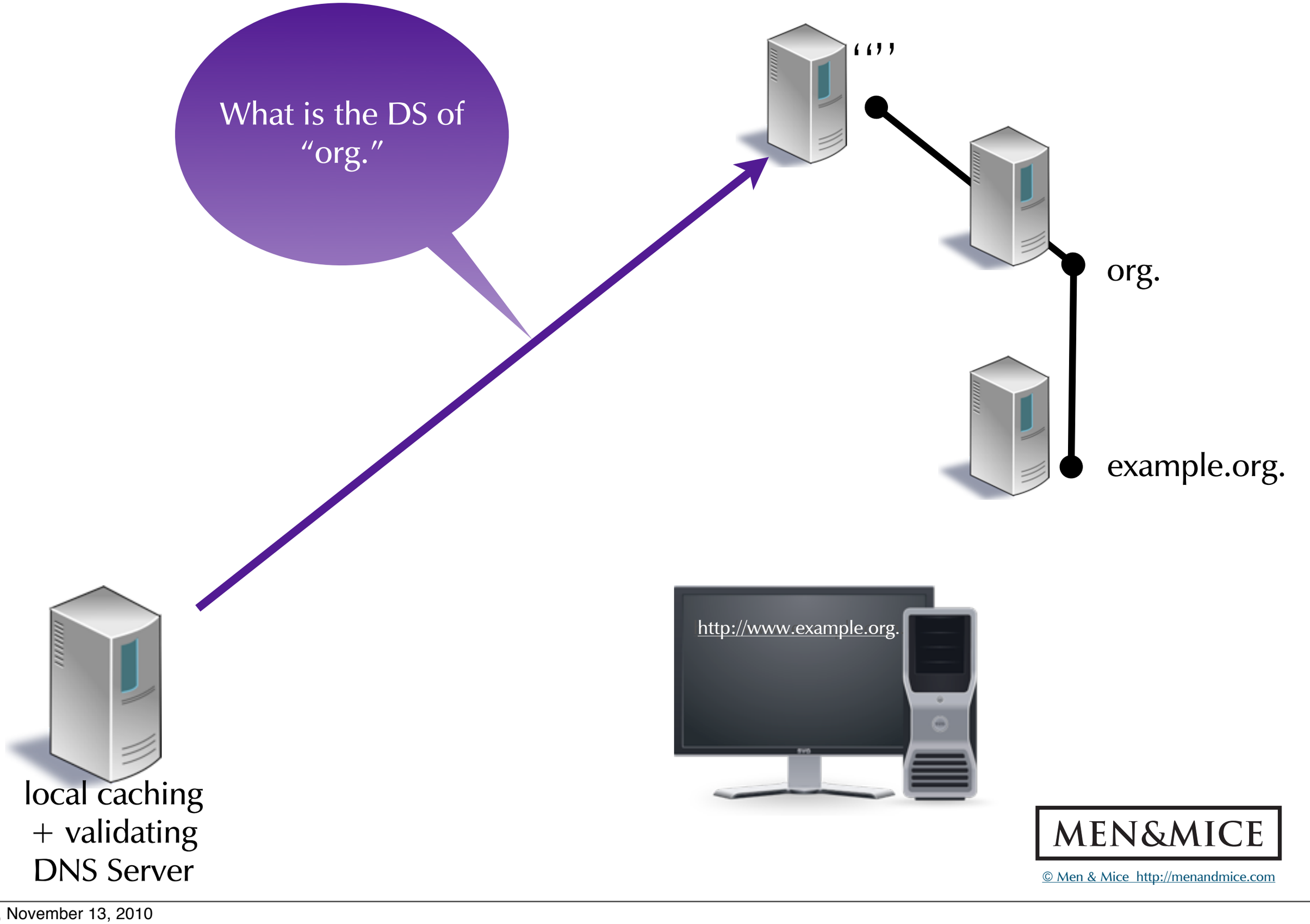
DNSSEC Name Resolution



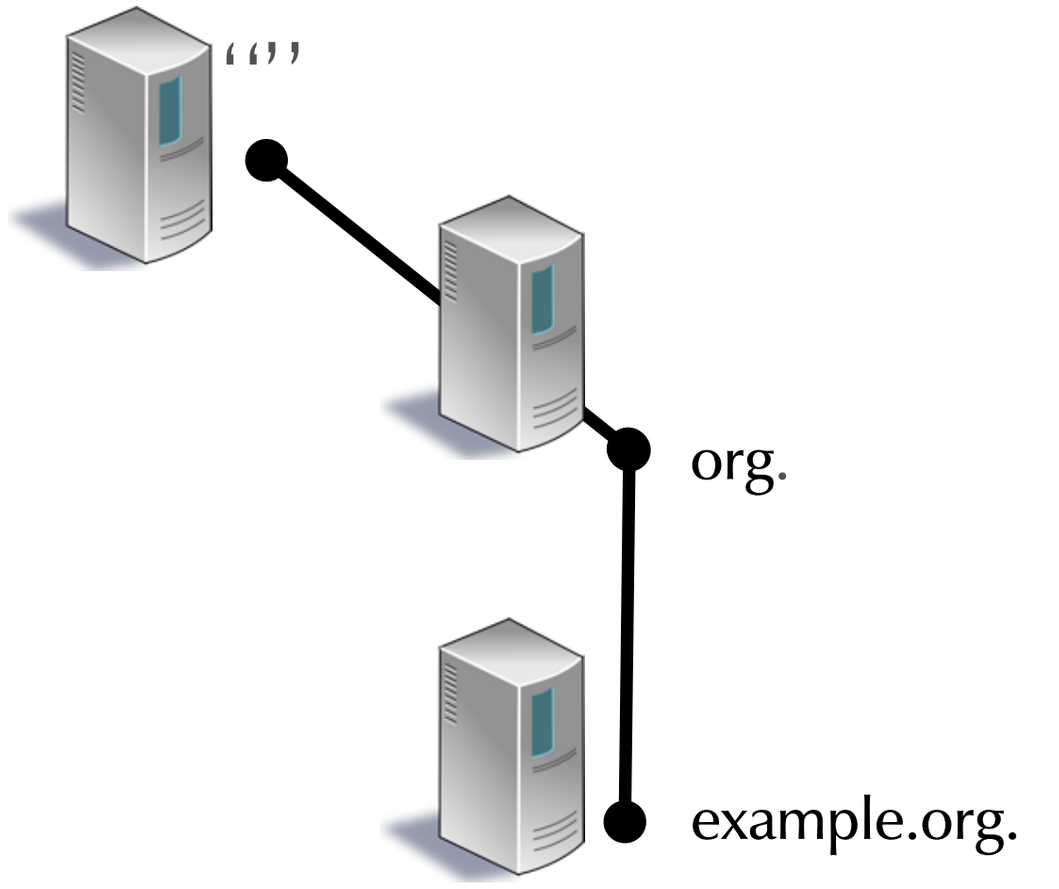
local caching
+ validating
DNS Server

MEN&MICE

DNSSEC Name Resolution



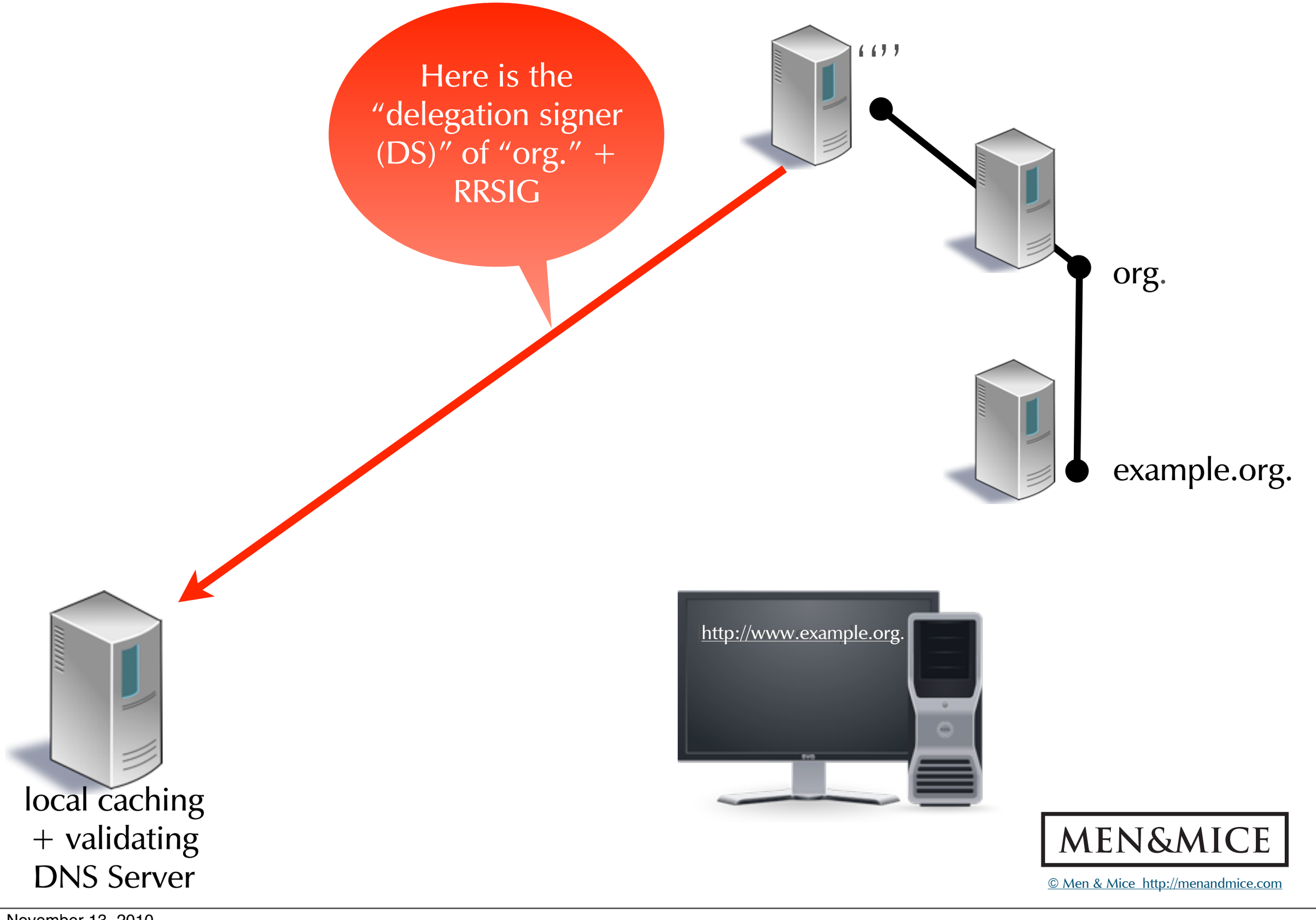
DNSSEC Name Resolution



local caching
+ validating
DNS Server

MEN&MICE

DNSSEC Name Resolution

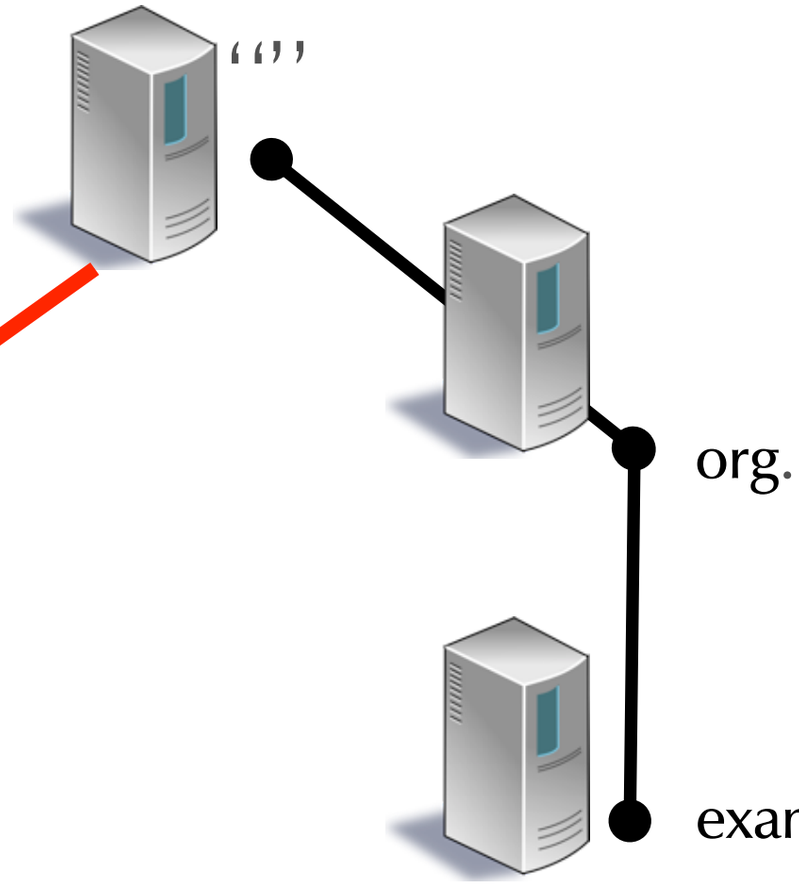


MEN&MICE

DNSSEC Name Resolution

Record	Function
www.example.org.A	IPv4 Address
www.example.org. RRSIG	signature ↑
example.org. DNSKEY	public key
example.org. RRSIG	signature ↑
example.org. DS	hash of public key
org. RRSIG	signature ↑
org DNSKEY	public key
org RRSIG	signature ↑
org DS	hash of public key
. RRSIG	signature ↑

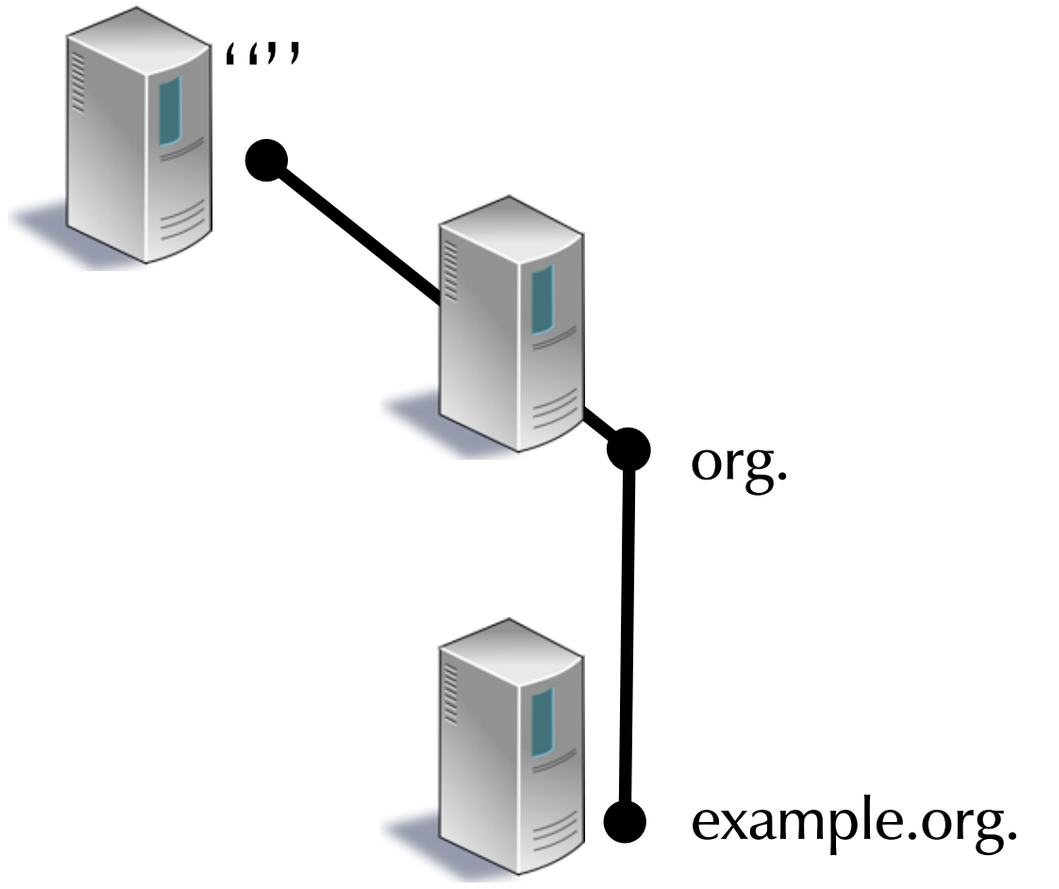
Here is the "delegation signer (DS)" of "org." + RRSIG



local caching
+ validating
DNS Server

MEN&MICE

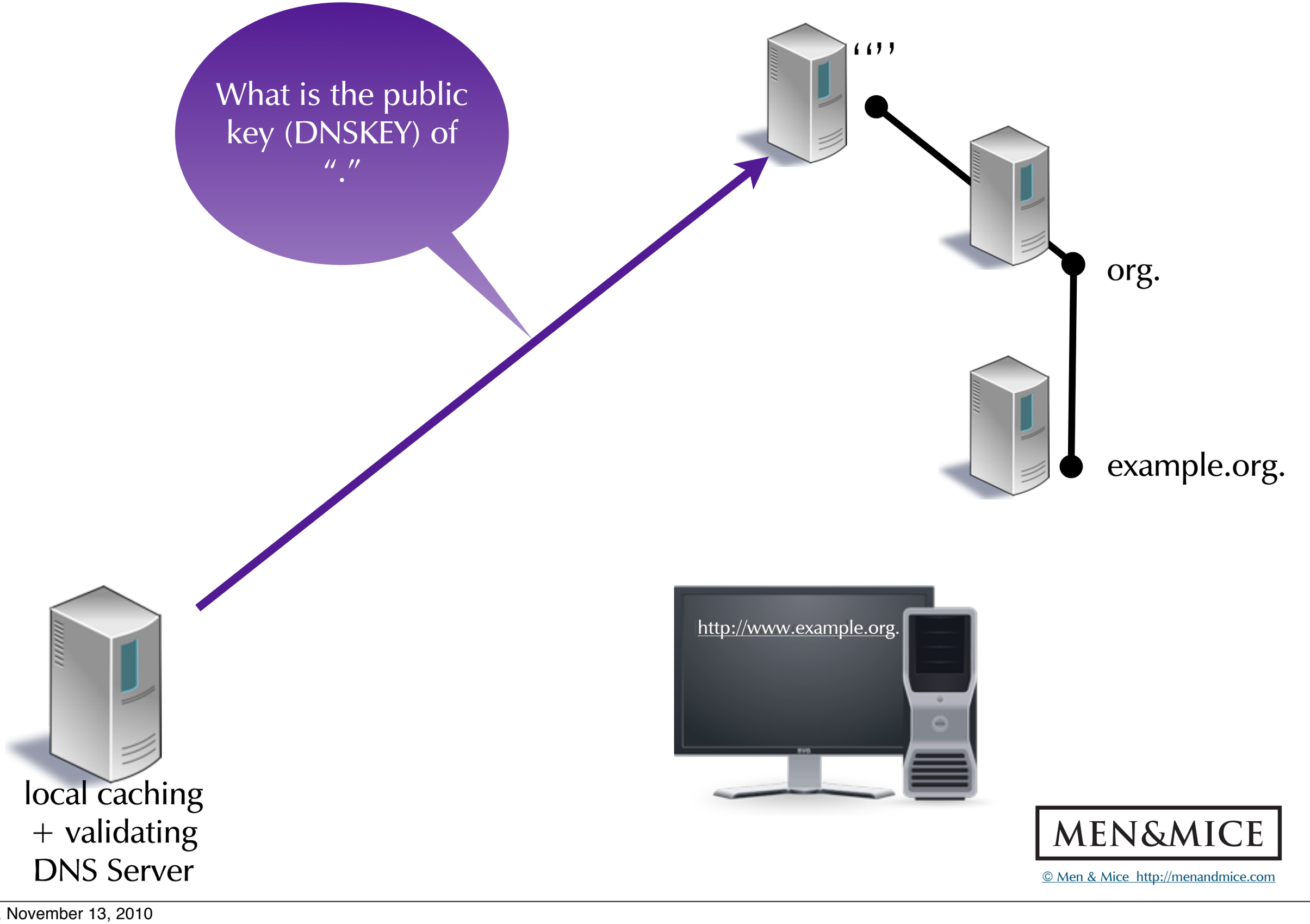
DNSSEC Name Resolution



local caching
+ validating
DNS Server

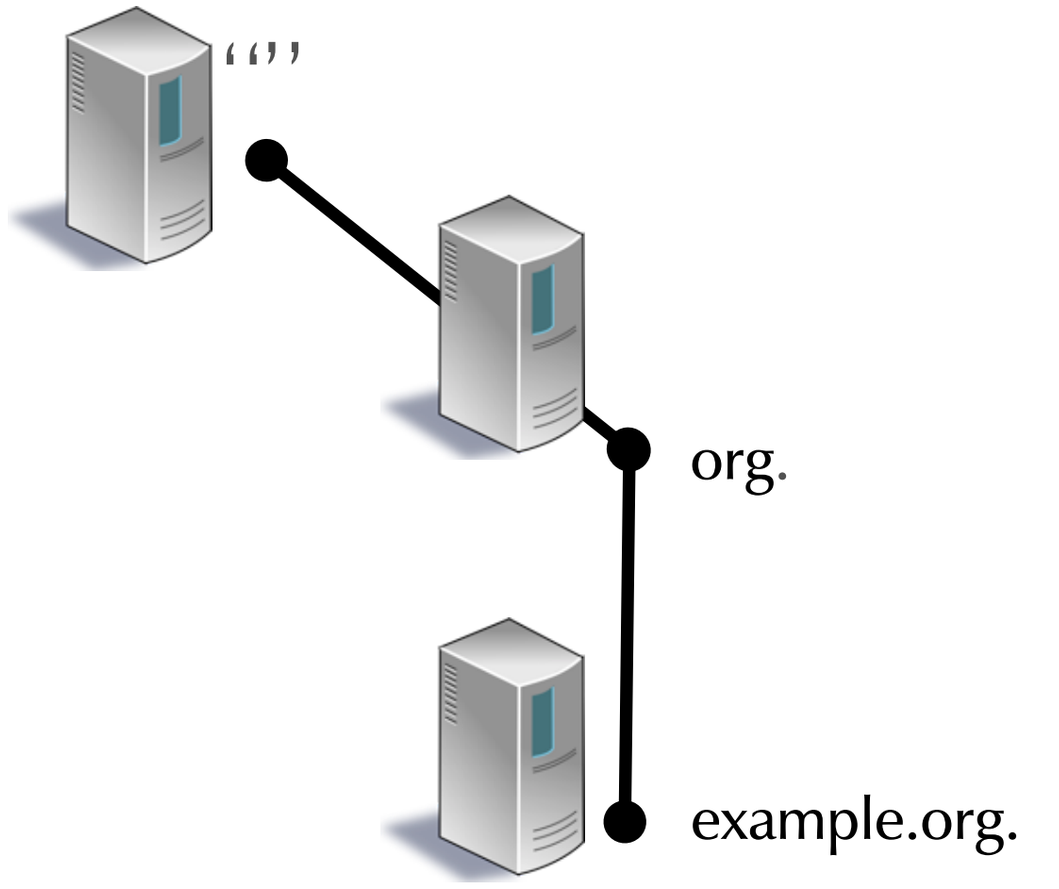
MEN&MICE

DNSSEC Name Resolution



MEN&MICE

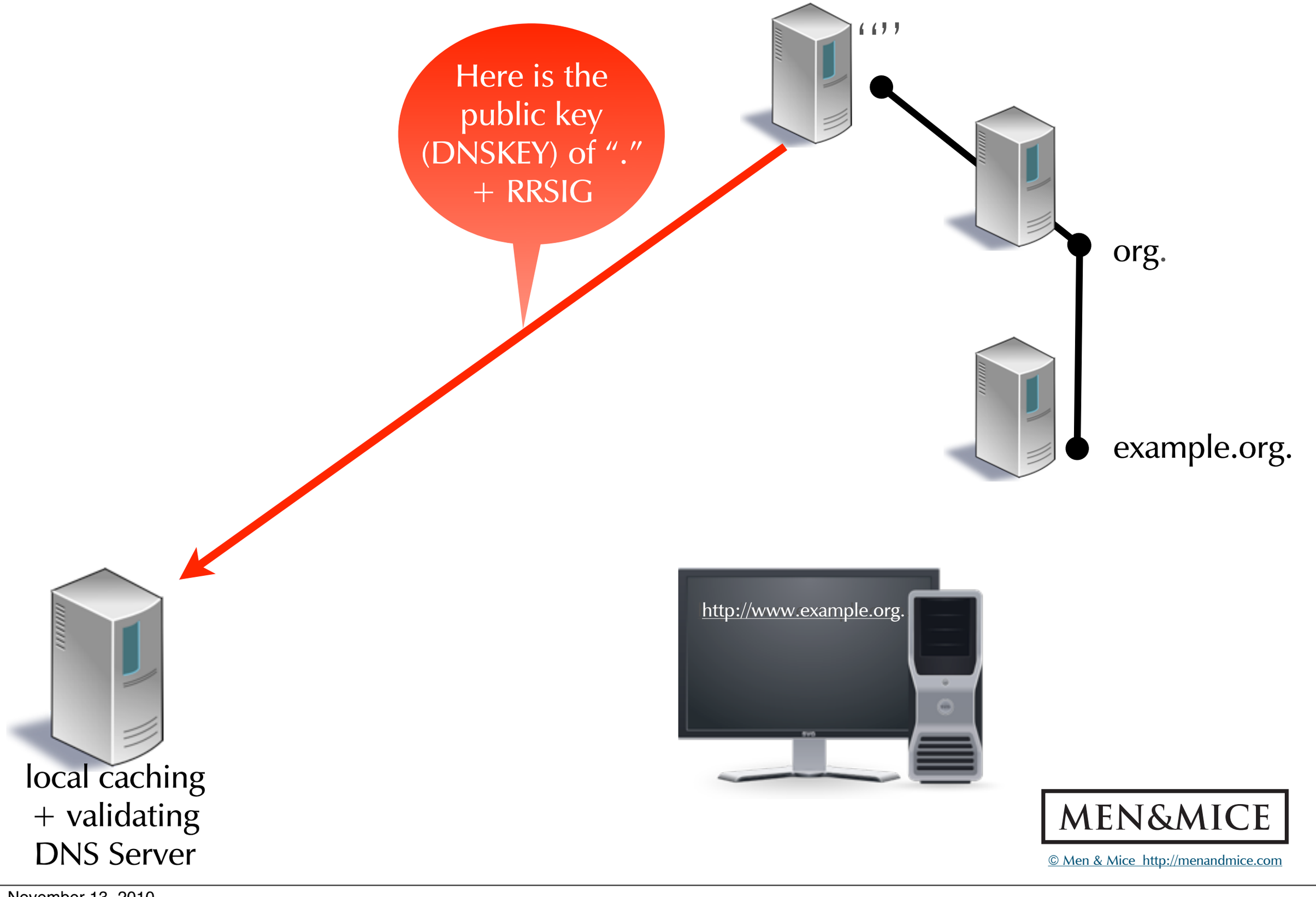
DNSSEC Name Resolution



local caching
+ validating
DNS Server

MEN&MICE

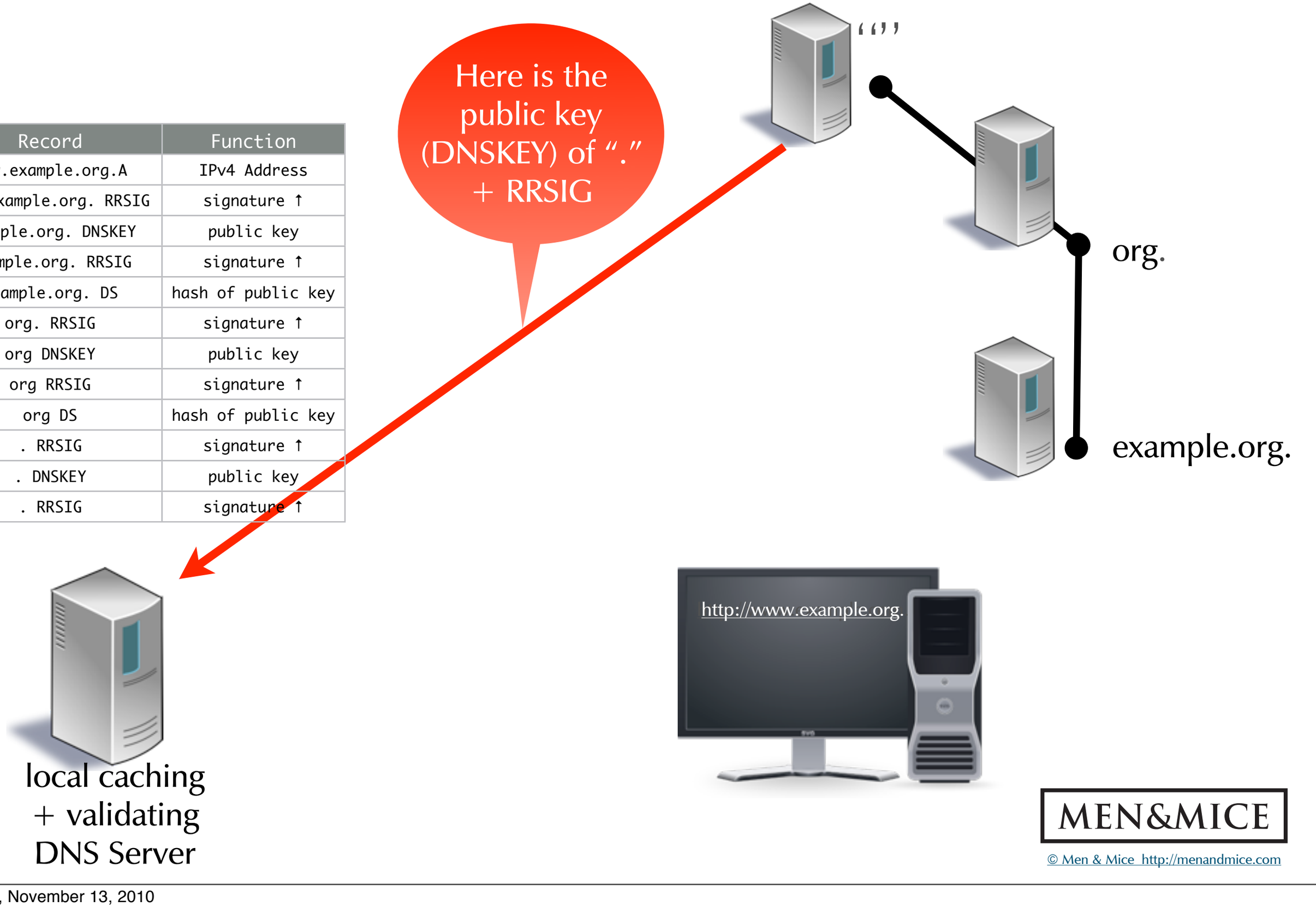
DNSSEC Name Resolution



MEN&MICE

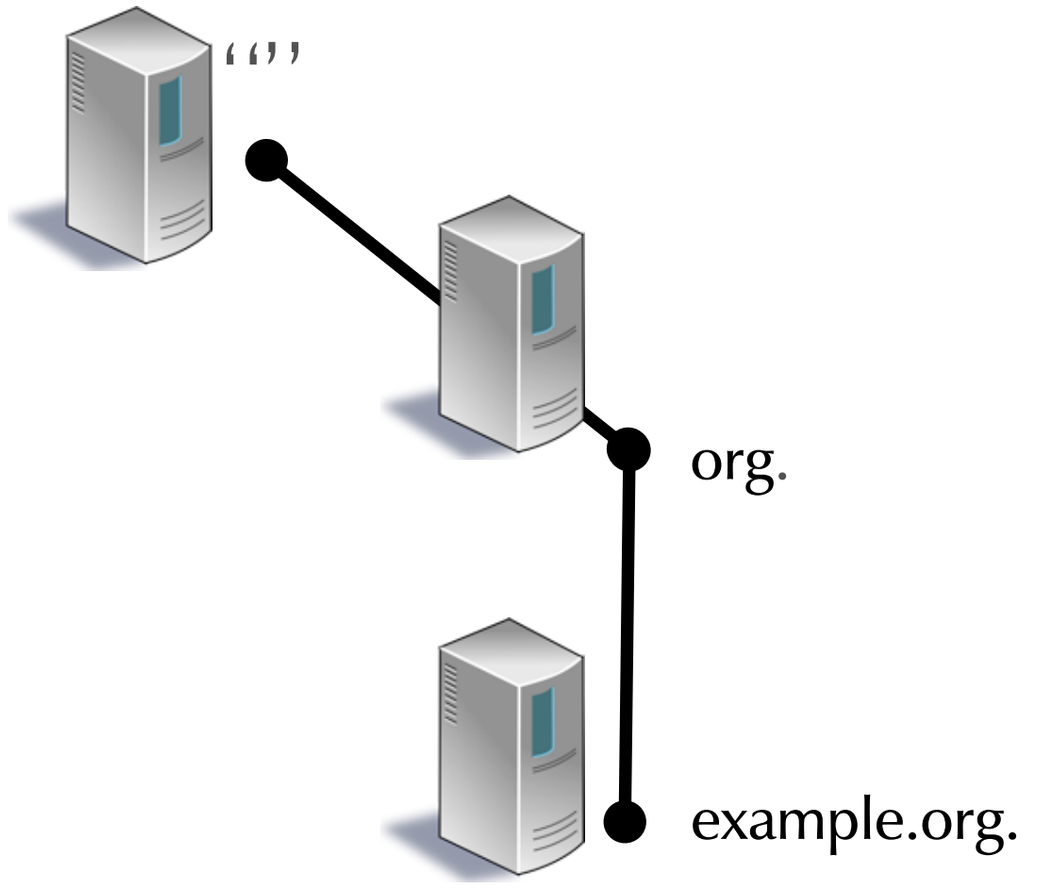
DNSSEC Name Resolution

Record	Function
www.example.org.A	IPv4 Address
www.example.org. RRSIG	signature ↑
example.org. DNSKEY	public key
example.org. RRSIG	signature ↑
example.org. DS	hash of public key
org. RRSIG	signature ↑
org DNSKEY	public key
org RRSIG	signature ↑
org DS	hash of public key
. RRSIG	signature ↑
. DNSKEY	public key
. RRSIG	signature ↑



MEN&MICE

DNSSEC Name Resolution

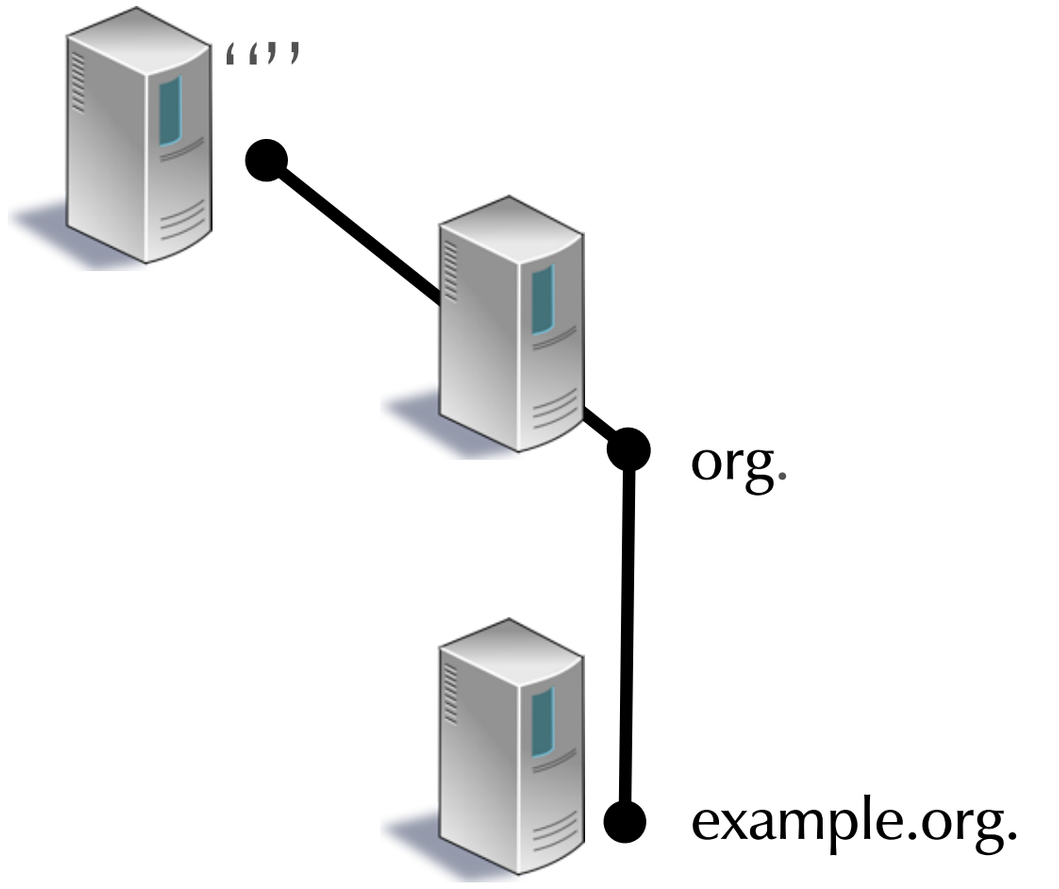


local caching
+ validating
DNS Server

```
Trush Anchor for
“.” (root zone) from
configuration file
```

MEN&MICE

DNSSEC Name Resolution

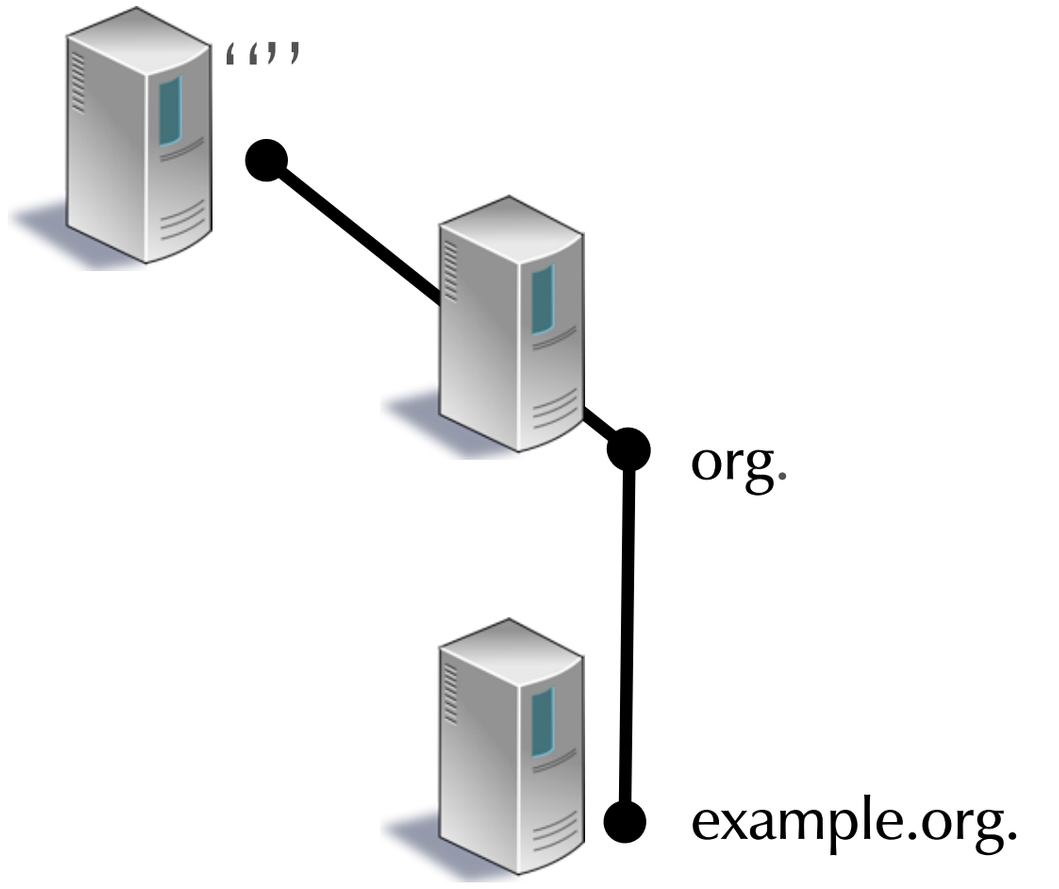


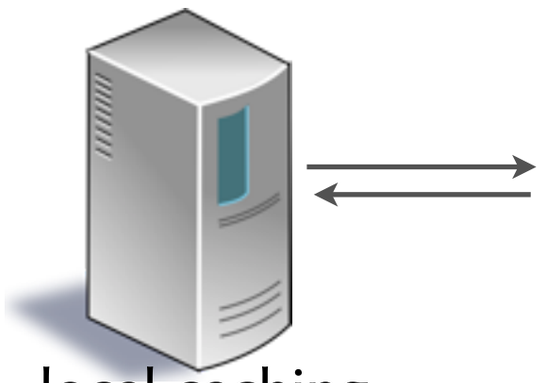
local caching
+ validating
DNS Server

Trush Anchor for
“.” (root zone) from
configuration file

MEN&MICE

DNSSEC Name Resolution





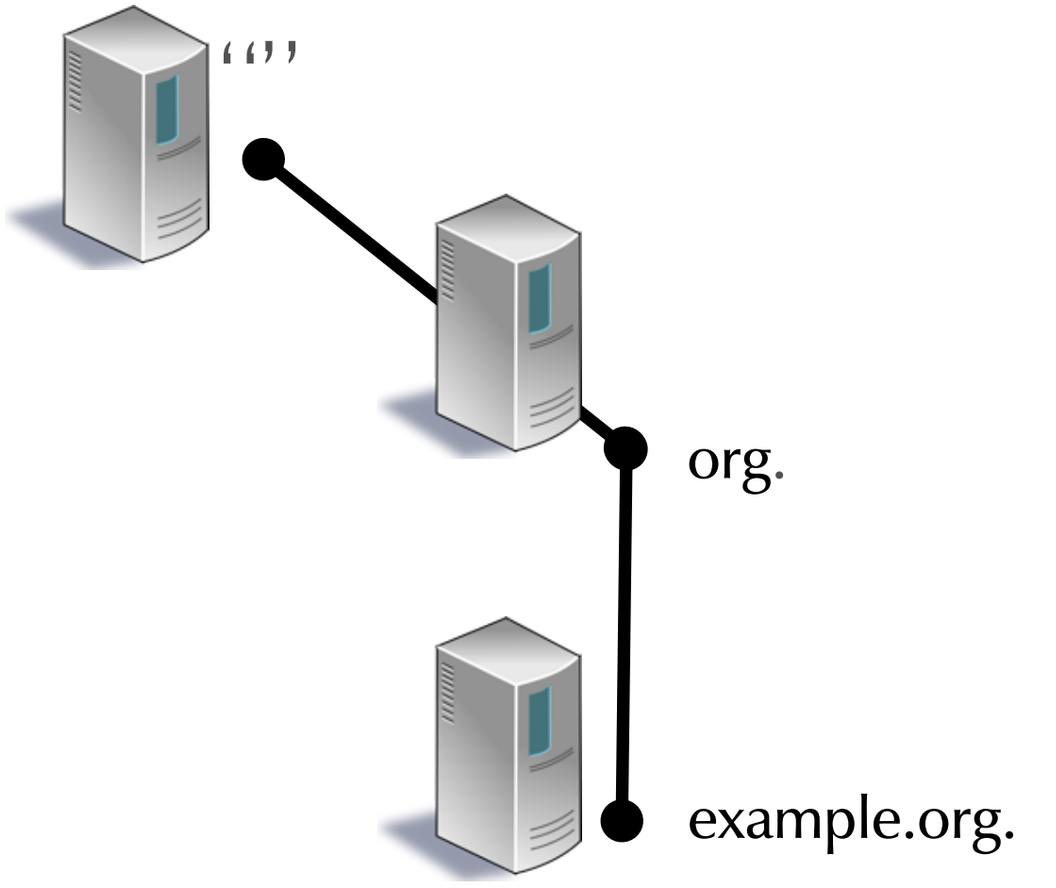
local caching
+ validating
DNS Server

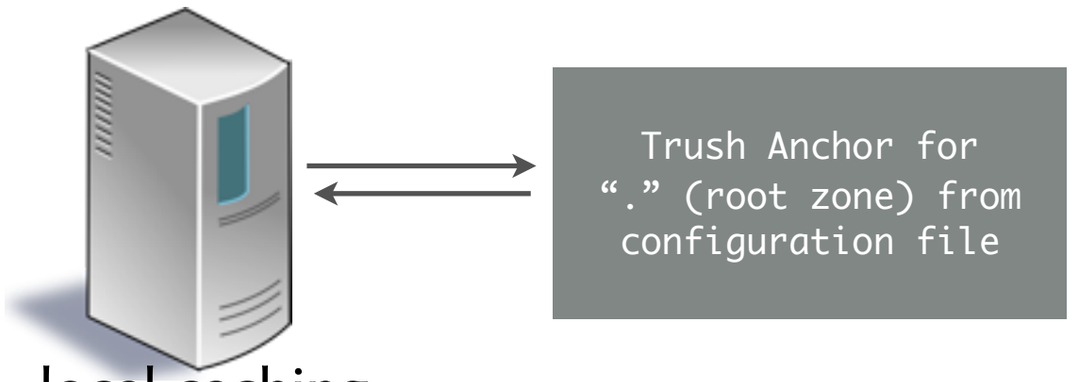
Trush Anchor for
“.” (root zone) from
configuration file

MEN&MICE

DNSSEC Name Resolution

Record	Function
www.example.org.A	IPv4 Address
www.example.org. RRSIG	signature ↑
example.org. DNSKEY	public key
example.org. RRSIG	signature ↑
example.org. DS	hash of public key
org. RRSIG	signature ↑
org DNSKEY	public key
org RRSIG	signature ↑
org DS	hash of public key
. RRSIG	signature ↑
. DNSKEY	public key
. RRSIG	signature ↑
Trust Anchor for “.”	hash of public key



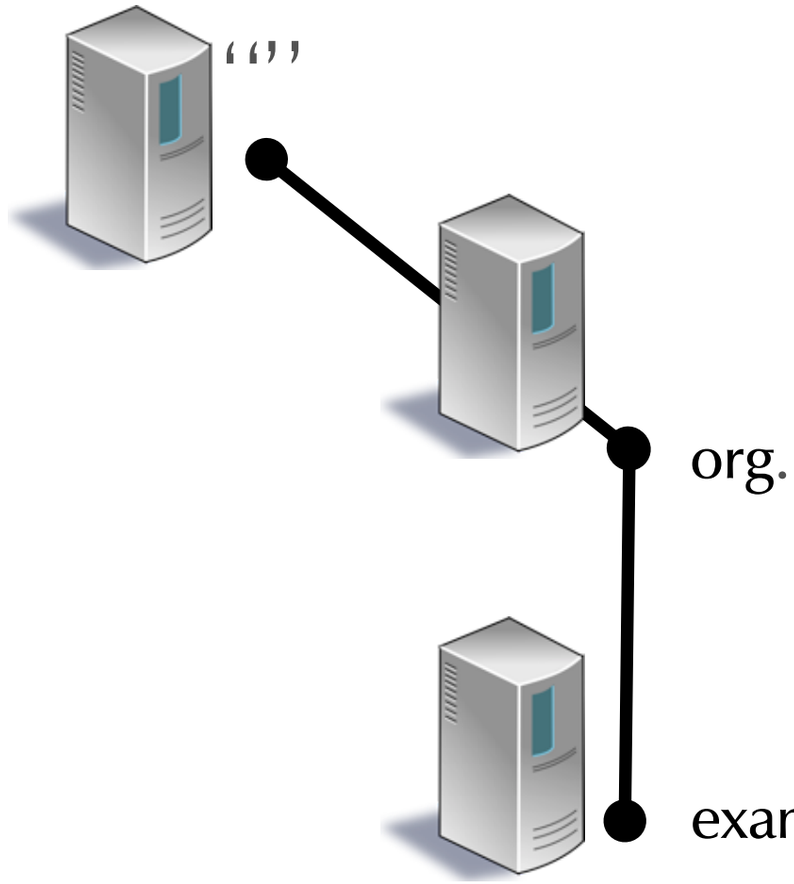


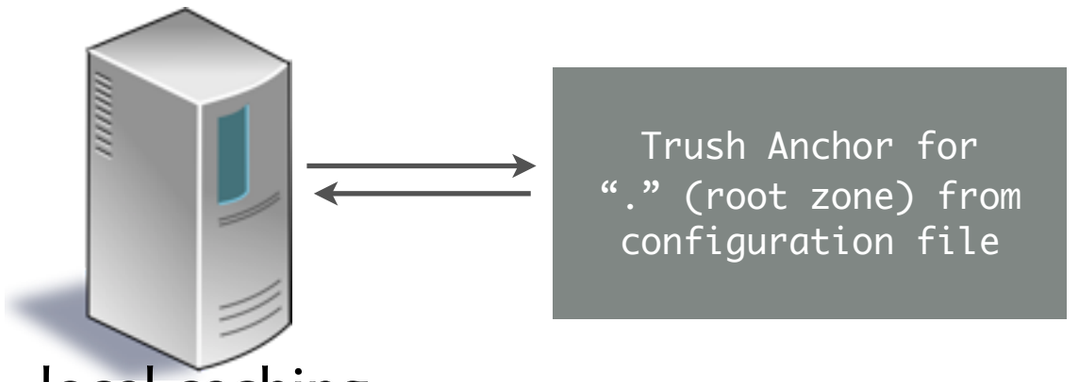
local caching
+ validating
DNS Server

MEN&MICE

DNSSEC Name Resolution

Record	Function
www.example.org.A	IPv4 Address
www.example.org. RRSIG	signature ↑
example.org. DNSKEY	public key
example.org. RRSIG	signature ↑
example.org. DS	hash of public key
org. RRSIG	signature ↑
org DNSKEY	public key
org RRSIG	signature ↑
org DS	hash of public key
. RRSIG	signature ↑
. DNSKEY	public key
. RRSIG	signature ↑
Trust Anchor for “.”	hash of public key

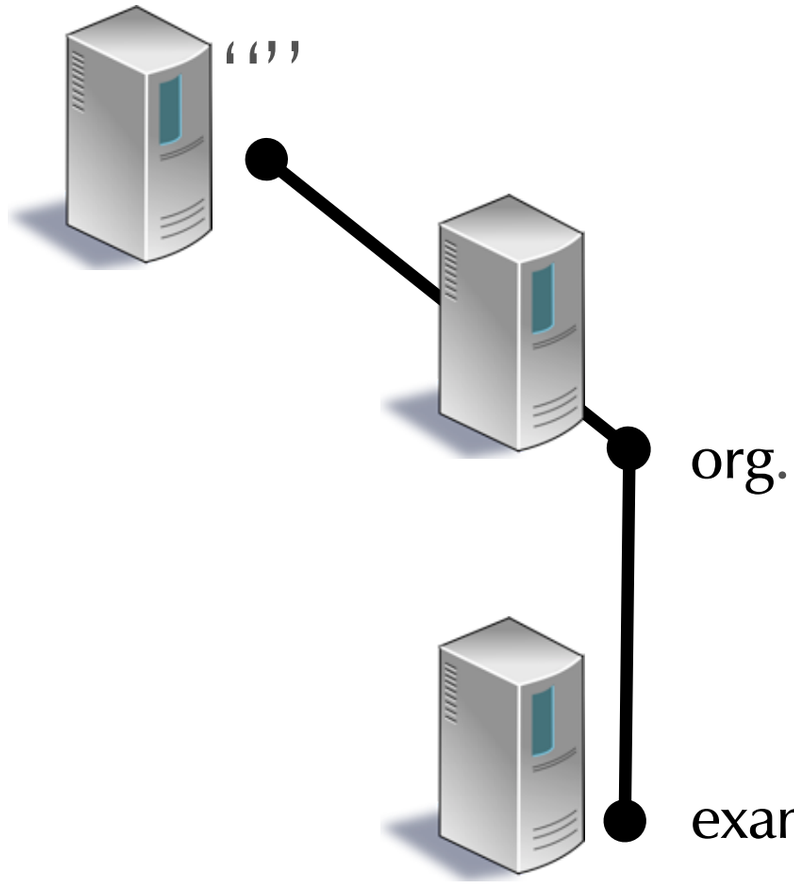


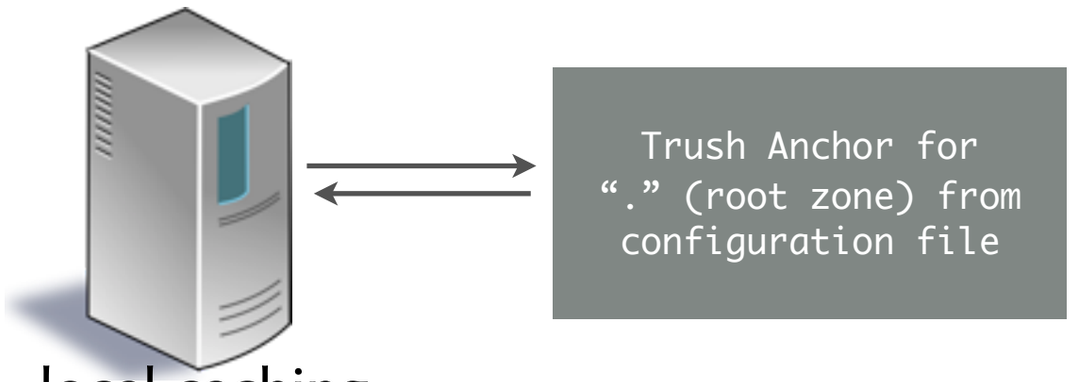


MEN&MICE

DNSSEC Name Resolution

Record	Function
www.example.org.A	IPv4 Address
www.example.org. RRSIG	signature ↑
example.org. DNSKEY	public key
example.org. RRSIG	signature ↑
example.org. DS	hash of public key
org. RRSIG	signature ↑
org DNSKEY	public key
org RRSIG	signature ↑
org DS	hash of public key
. RRSIG	signature ↑
. DNSKEY	public key ✔
. RRSIG	signature ↑ ✔
Trust Anchor for “.”	hash of public key



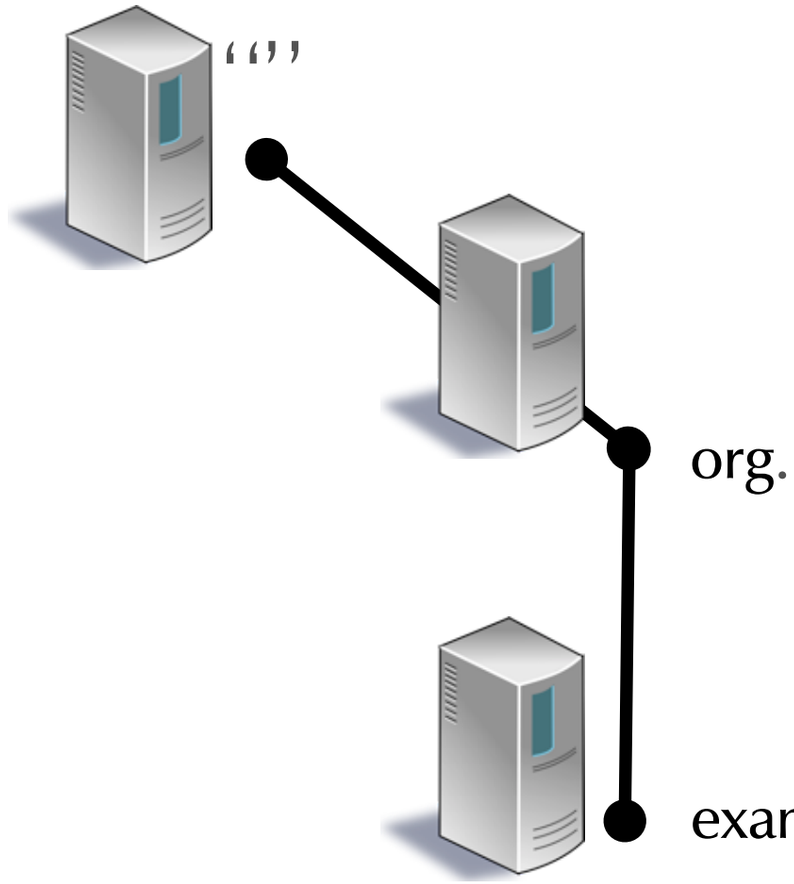


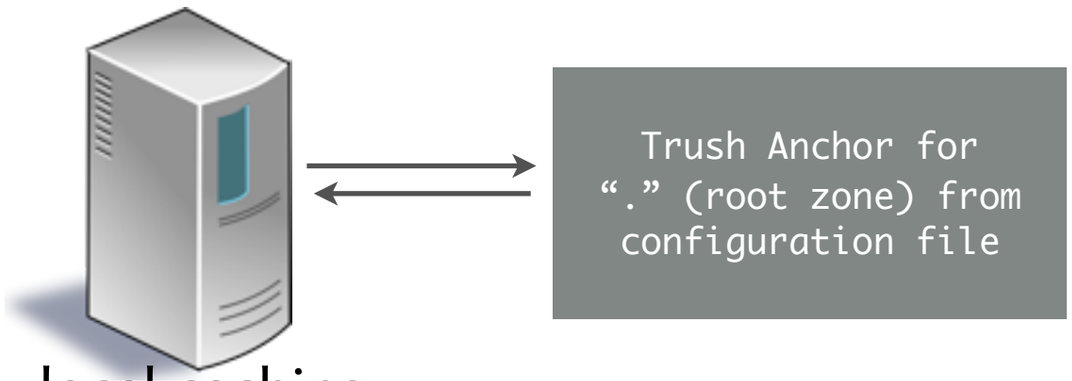
local caching
+ validating
DNS Server

MEN&MICE

DNSSEC Name Resolution

Record	Function
www.example.org.A	IPv4 Address
www.example.org. RRSIG	signature ↑
example.org. DNSKEY	public key
example.org. RRSIG	signature ↑
example.org. DS	hash of public key
org. RRSIG	signature ↑
org DNSKEY	public key
org RRSIG	signature ↑
org DS	hash of public key
. RRSIG	signature ↑ ✔
. DNSKEY	public key ✔
. RRSIG	signature ↑ ✔
Trust Anchor for “.”	hash of public key



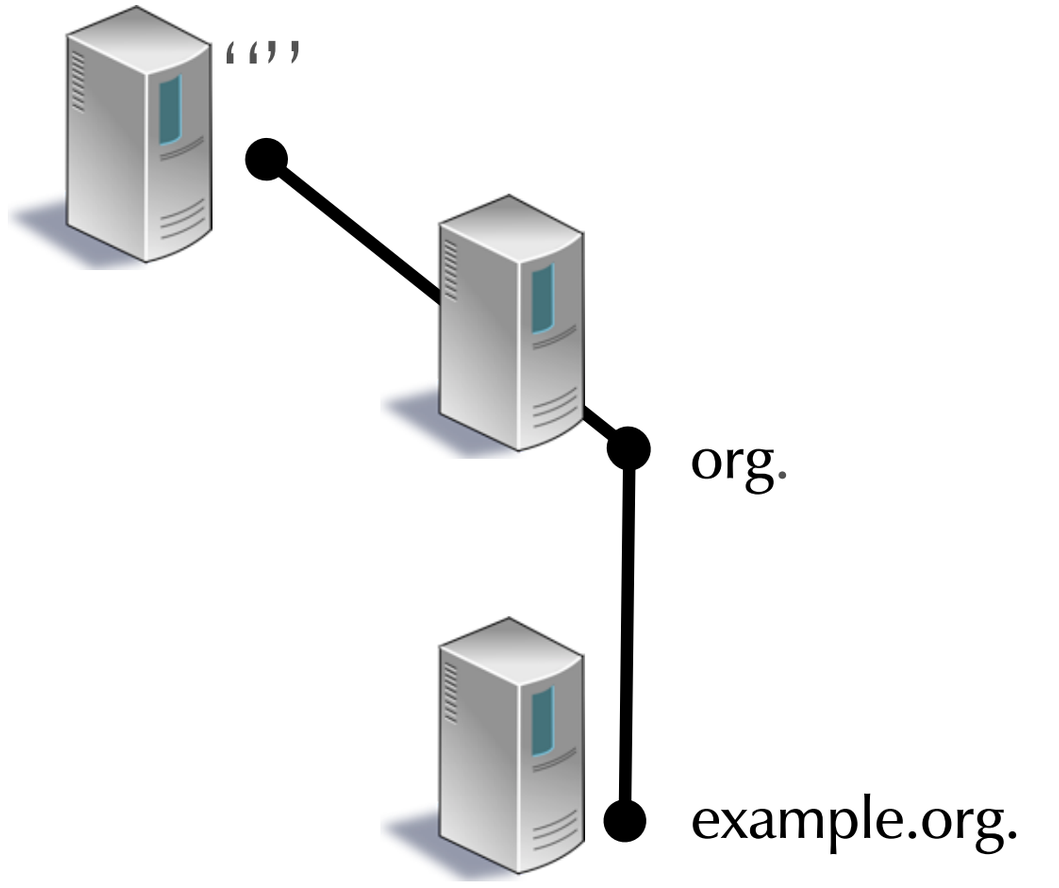


local caching
+ validating
DNS Server

MEN&MICE

DNSSEC Name Resolution

Record	Function	
www.example.org.A	IPv4 Address	
www.example.org. RRSIG	signature ↑	
example.org. DNSKEY	public key	
example.org. RRSIG	signature ↑	
example.org. DS	hash of public key	
org. RRSIG	signature ↑	
org DNSKEY	public key	
org RRSIG	signature ↑	
org DS	hash of public key	✔
. RRSIG	signature ↑	✔
. DNSKEY	public key	✔
. RRSIG	signature ↑	✔
Trust Anchor for “.”	hash of public key	



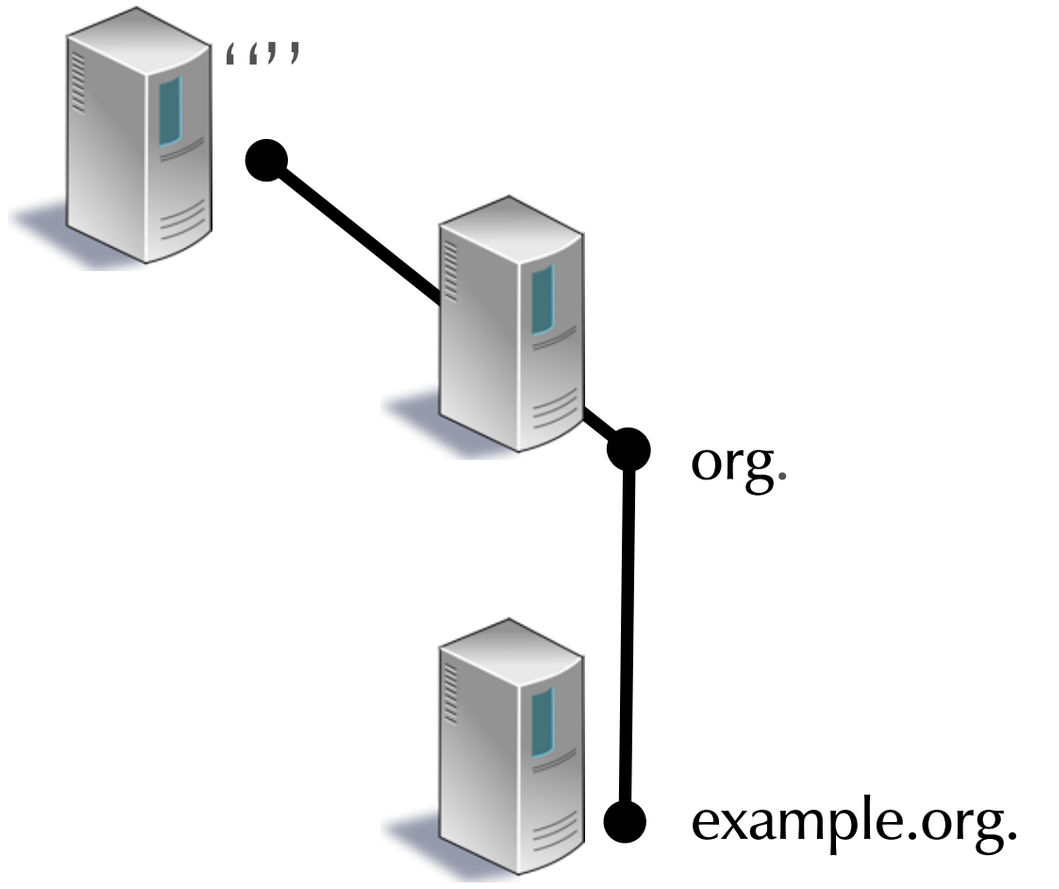
Trush Anchor for “.” (root zone) from configuration file

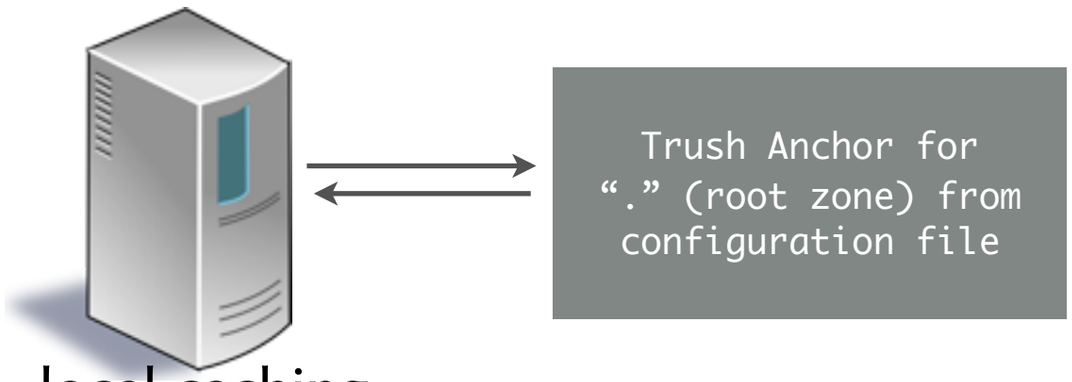
local caching + validating DNS Server

MEN&MICE

DNSSEC Name Resolution

Record	Function	
www.example.org.A	IPv4 Address	
www.example.org. RRSIG	signature ↑	
example.org. DNSKEY	public key	
example.org. RRSIG	signature ↑	
example.org. DS	hash of public key	
org. RRSIG	signature ↑	
org DNSKEY	public key	
org RRSIG	signature ↑	✔
org DS	hash of public key	✔
. RRSIG	signature ↑	✔
. DNSKEY	public key	✔
. RRSIG	signature ↑	✔
Trust Anchor for “.”	hash of public key	✔

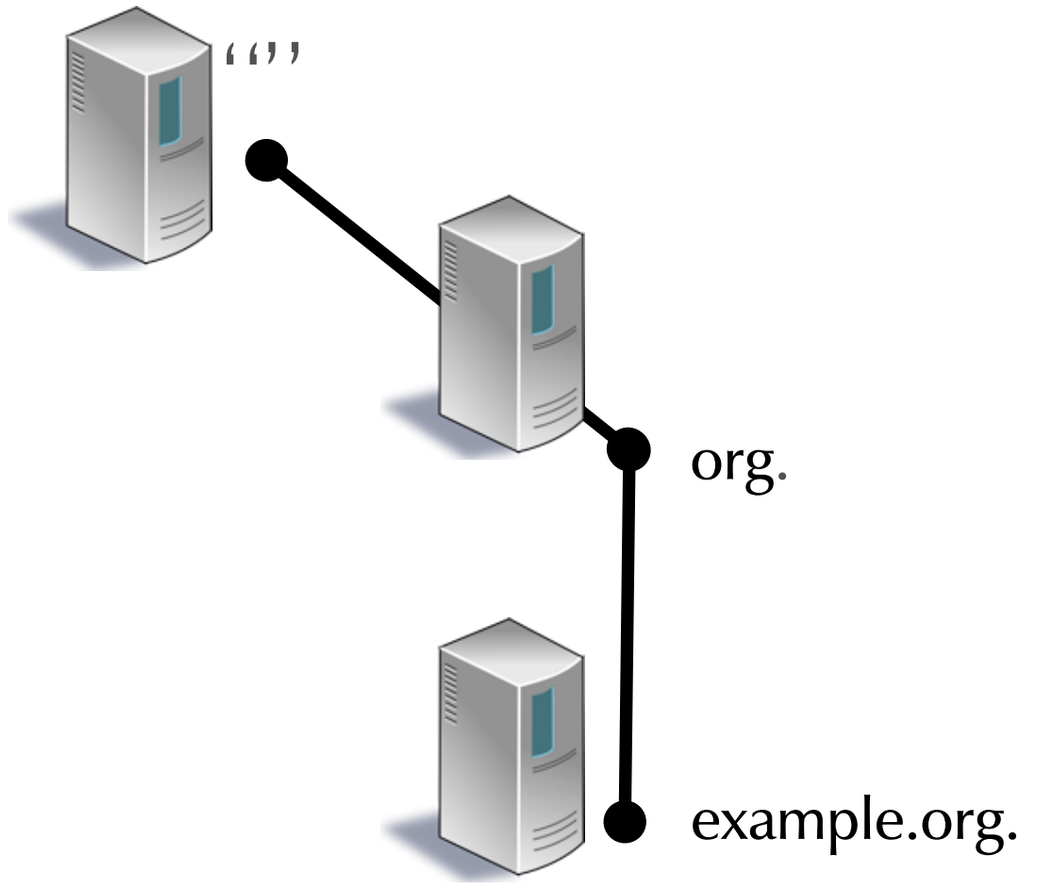


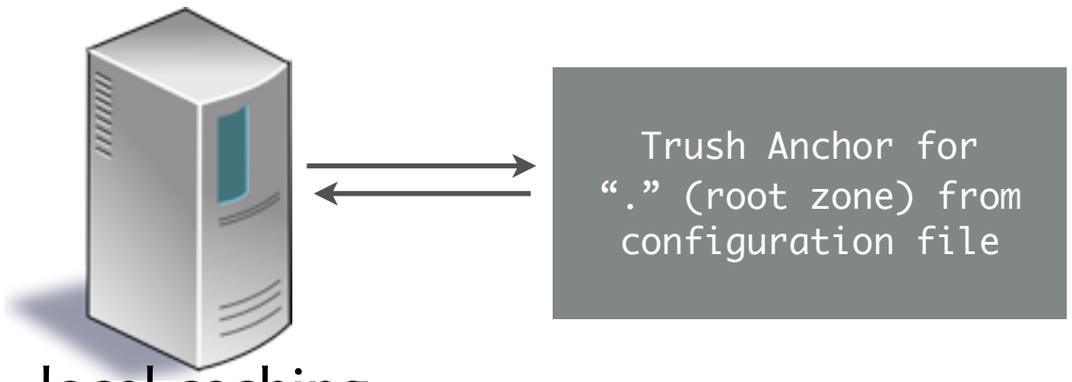


MEN&MICE

DNSSEC Name Resolution

Record	Function	
www.example.org.A	IPv4 Address	
www.example.org. RRSIG	signature ↑	
example.org. DNSKEY	public key	
example.org. RRSIG	signature ↑	
example.org. DS	hash of public key	
org. RRSIG	signature ↑	
org DNSKEY	public key	✔
org RRSIG	signature ↑	✔
org DS	hash of public key	✔
. RRSIG	signature ↑	✔
. DNSKEY	public key	✔
. RRSIG	signature ↑	✔
Trust Anchor for “.”	hash of public key	✔



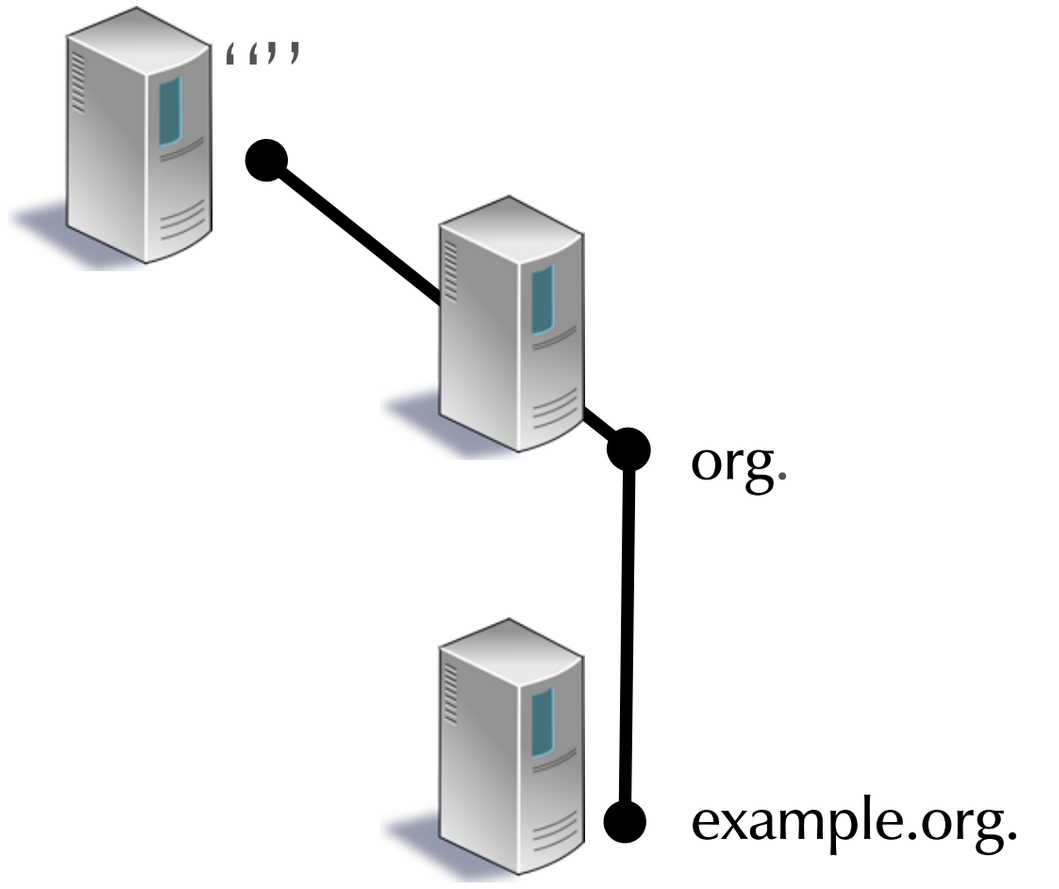


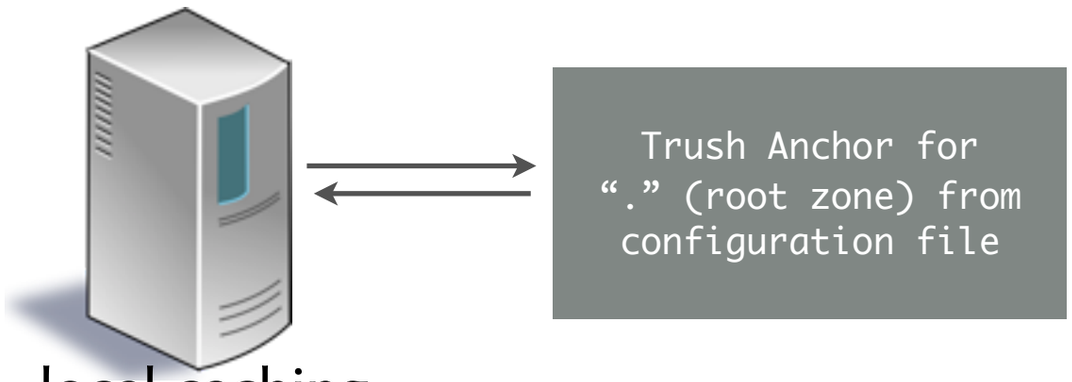
local caching
+ validating
DNS Server

MEN&MICE

DNSSEC Name Resolution

Record	Function	
www.example.org.A	IPv4 Address	
www.example.org. RRSIG	signature ↑	
example.org. DNSKEY	public key	
example.org. RRSIG	signature ↑	
example.org. DS	hash of public key	
org. RRSIG	signature ↑	✔
org DNSKEY	public key	✔
org RRSIG	signature ↑	✔
org DS	hash of public key	✔
. RRSIG	signature ↑	✔
. DNSKEY	public key	✔
. RRSIG	signature ↑	✔
Trust Anchor for “.”	hash of public key	

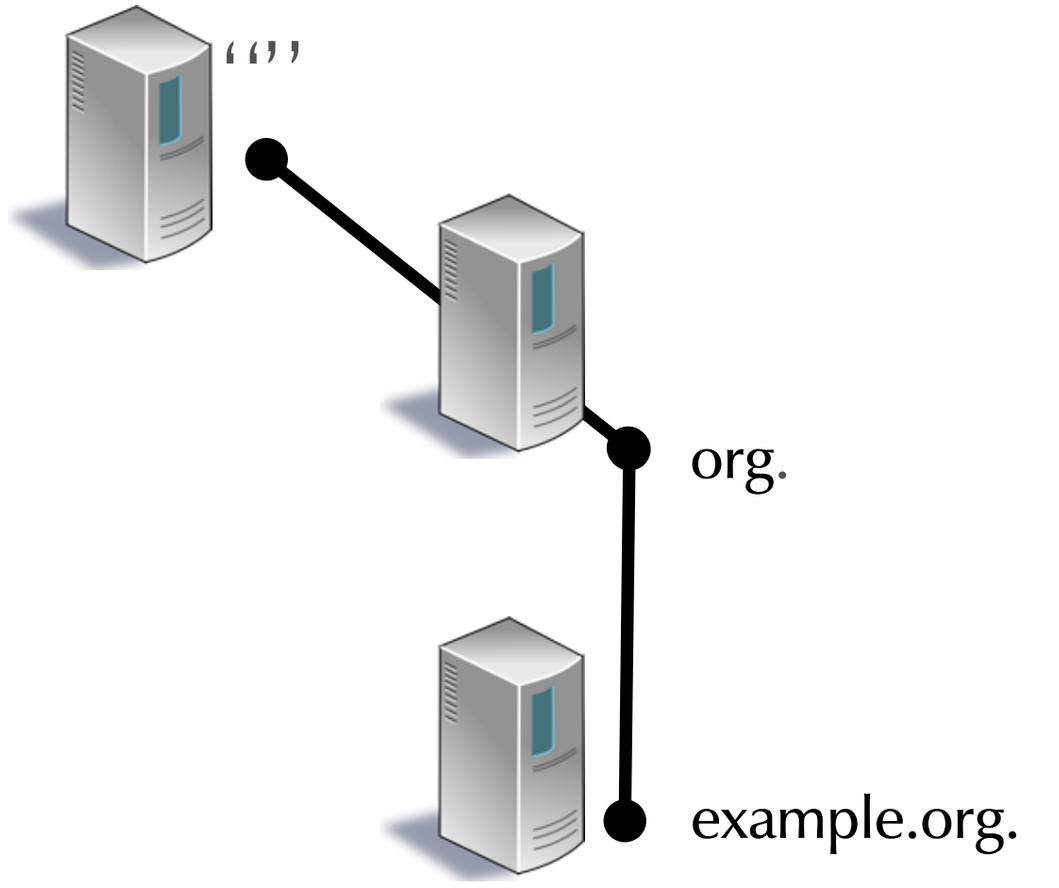


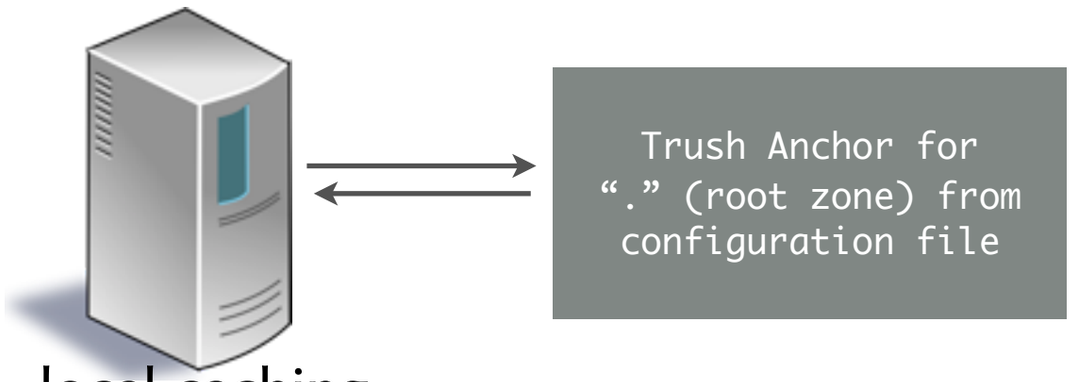


MEN&MICE

DNSSEC Name Resolution

Record	Function	
www.example.org.A	IPv4 Address	
www.example.org. RRSIG	signature ↑	
example.org. DNSKEY	public key	
example.org. RRSIG	signature ↑	
example.org. DS	hash of public key	✔
org. RRSIG	signature ↑	✔
org DNSKEY	public key	✔
org RRSIG	signature ↑	✔
org DS	hash of public key	✔
. RRSIG	signature ↑	✔
. DNSKEY	public key	✔
. RRSIG	signature ↑	✔
Trust Anchor for “.”	hash of public key	



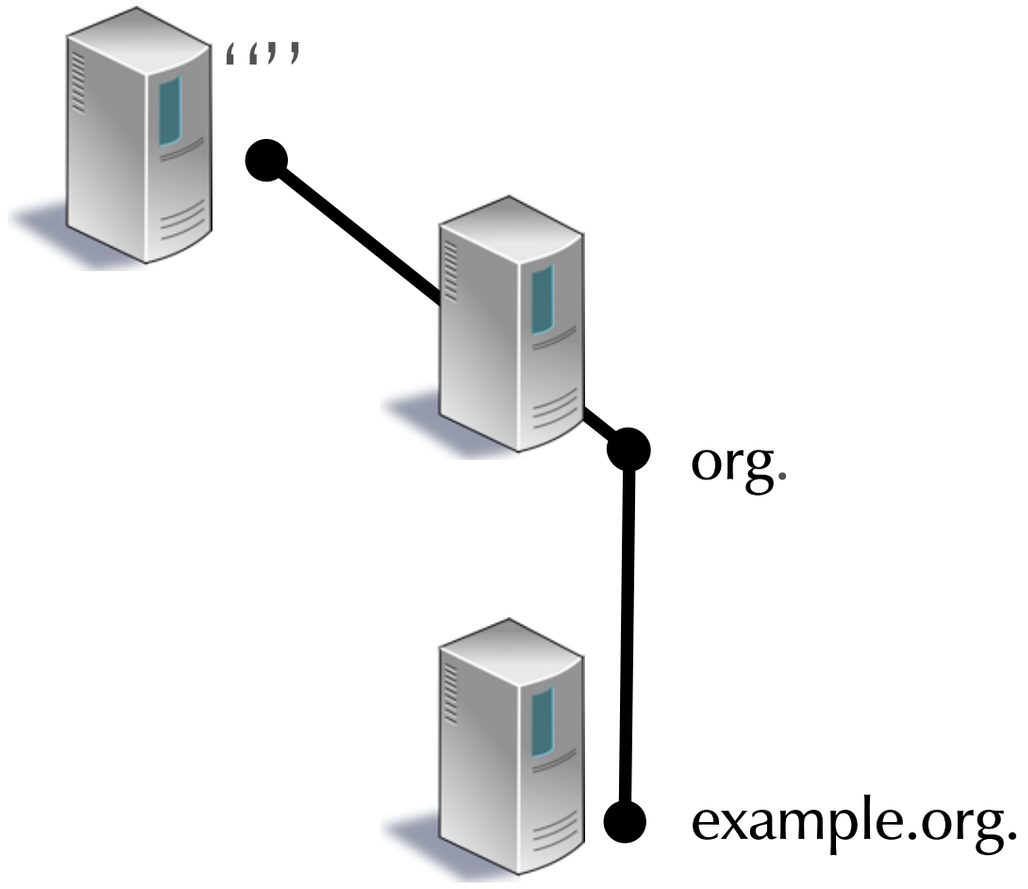


local caching
+ validating
DNS Server

MEN&MICE

DNSSEC Name Resolution

Record	Function	
www.example.org.A	IPv4 Address	
www.example.org. RRSIG	signature ↑	
example.org. DNSKEY	public key	
example.org. RRSIG	signature ↑	✔
example.org. DS	hash of public key	✔
org. RRSIG	signature ↑	✔
org DNSKEY	public key	✔
org RRSIG	signature ↑	✔
org DS	hash of public key	✔
. RRSIG	signature ↑	✔
. DNSKEY	public key	✔
. RRSIG	signature ↑	✔
Trust Anchor for “.”	hash of public key	



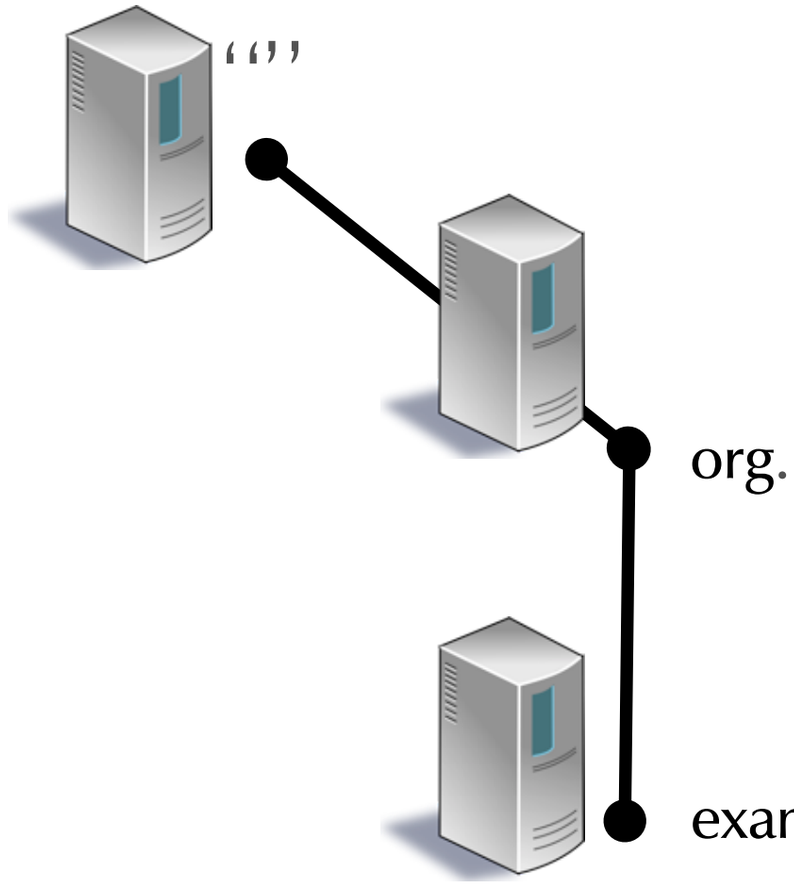
Trush Anchor for “.” (root zone) from configuration file

local caching + validating DNS Server

MEN&MICE

DNSSEC Name Resolution

Record	Function	
www.example.org.A	IPv4 Address	
www.example.org. RRSIG	signature ↑	✔
example.org. DNSKEY	public key	✔
example.org. RRSIG	signature ↑	✔
example.org. DS	hash of public key	✔
org. RRSIG	signature ↑	✔
org DNSKEY	public key	✔
org RRSIG	signature ↑	✔
org DS	hash of public key	✔
. RRSIG	signature ↑	✔
. DNSKEY	public key	✔
. RRSIG	signature ↑	✔
Trust Anchor for “.”	hash of public key	



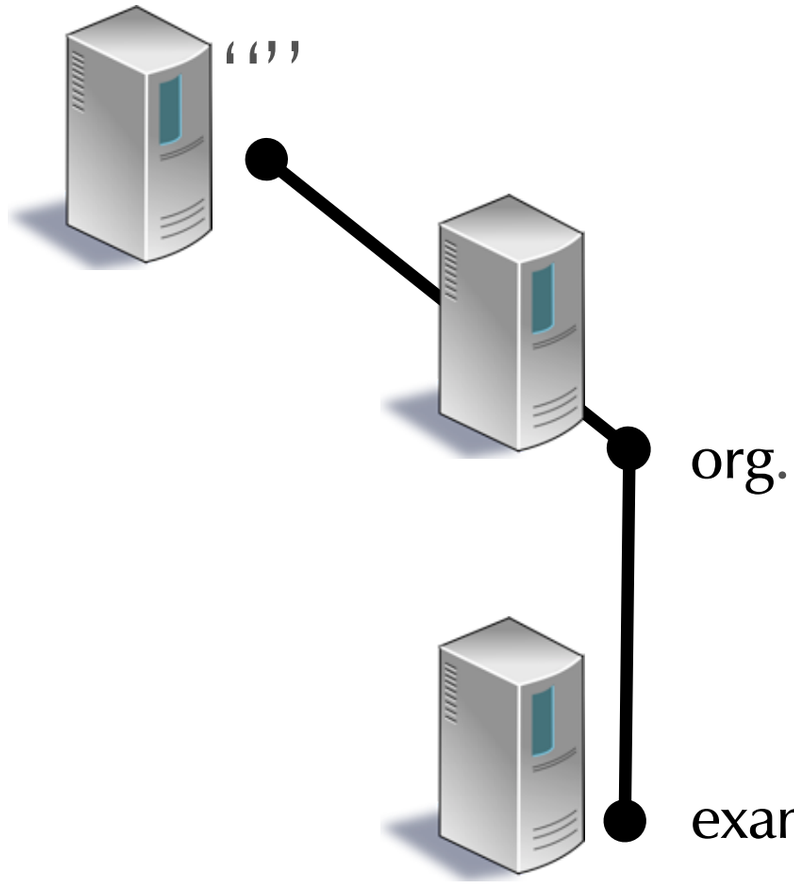
Trush Anchor for “.” (root zone) from configuration file

local caching + validating DNS Server

MEN&MICE

DNSSEC Name Resolution

Record	Function	
www.example.org.A	IPv4 Address	
www.example.org. RRSIG	signature ↑	✔
example.org. DNSKEY	public key	✔
example.org. RRSIG	signature ↑	✔
example.org. DS	hash of public key	✔
org. RRSIG	signature ↑	✔
org DNSKEY	public key	✔
org RRSIG	signature ↑	✔
org DS	hash of public key	✔
. RRSIG	signature ↑	✔
. DNSKEY	public key	✔
. RRSIG	signature ↑	✔
Trust Anchor for “.”	hash of public key	



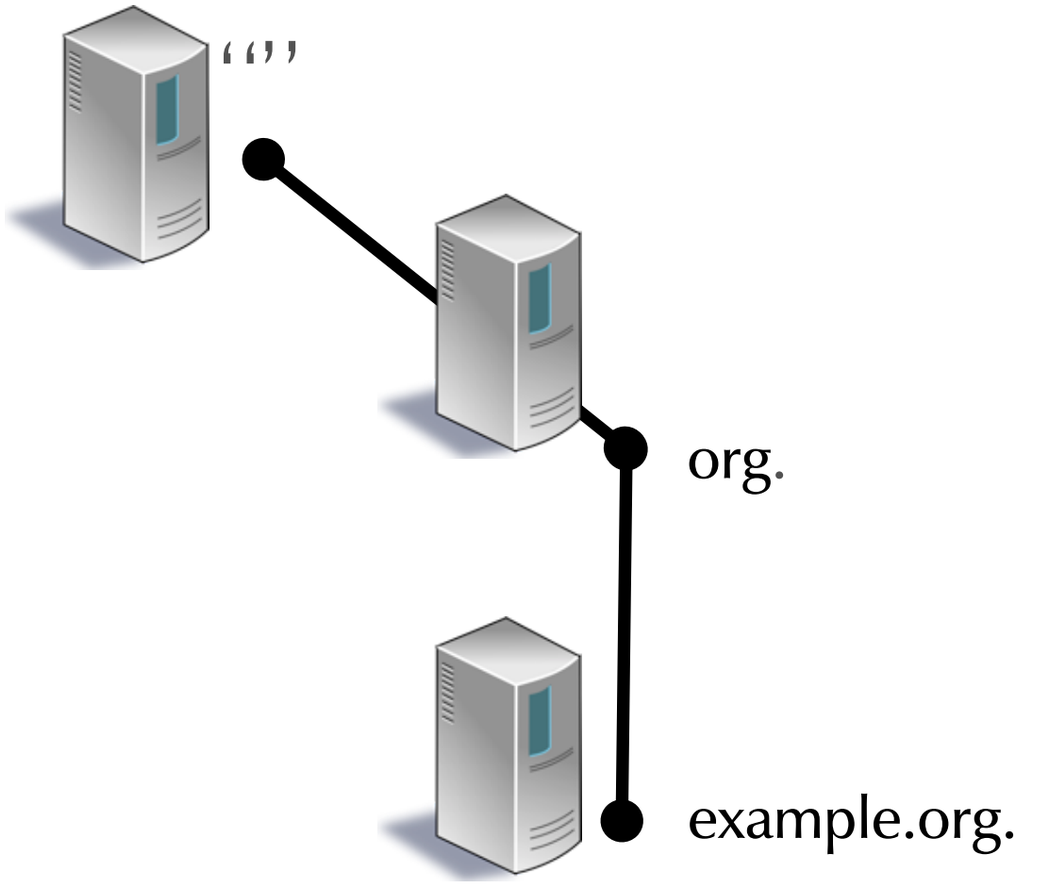
Trush Anchor for “.” (root zone) from configuration file

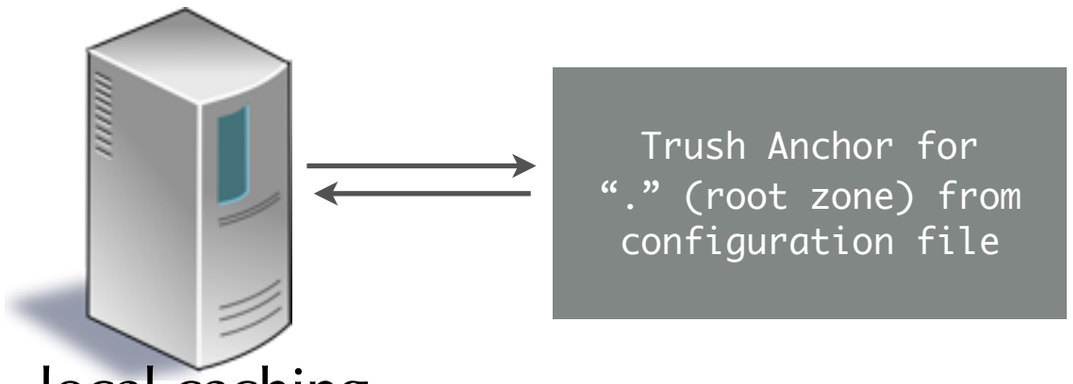
local caching + validating DNS Server

MEN&MICE

DNSSEC Name Resolution

Record	Function	
www.example.org.A	IPv4 Address	✔
www.example.org. RRSIG	signature ↑	✔
example.org. DNSKEY	public key	✔
example.org. RRSIG	signature ↑	✔
example.org. DS	hash of public key	✔
org. RRSIG	signature ↑	✔
org DNSKEY	public key	✔
org RRSIG	signature ↑	✔
org DS	hash of public key	✔
. RRSIG	signature ↑	✔
. DNSKEY	public key	✔
. RRSIG	signature ↑	✔
Trust Anchor for “.”	hash of public key	

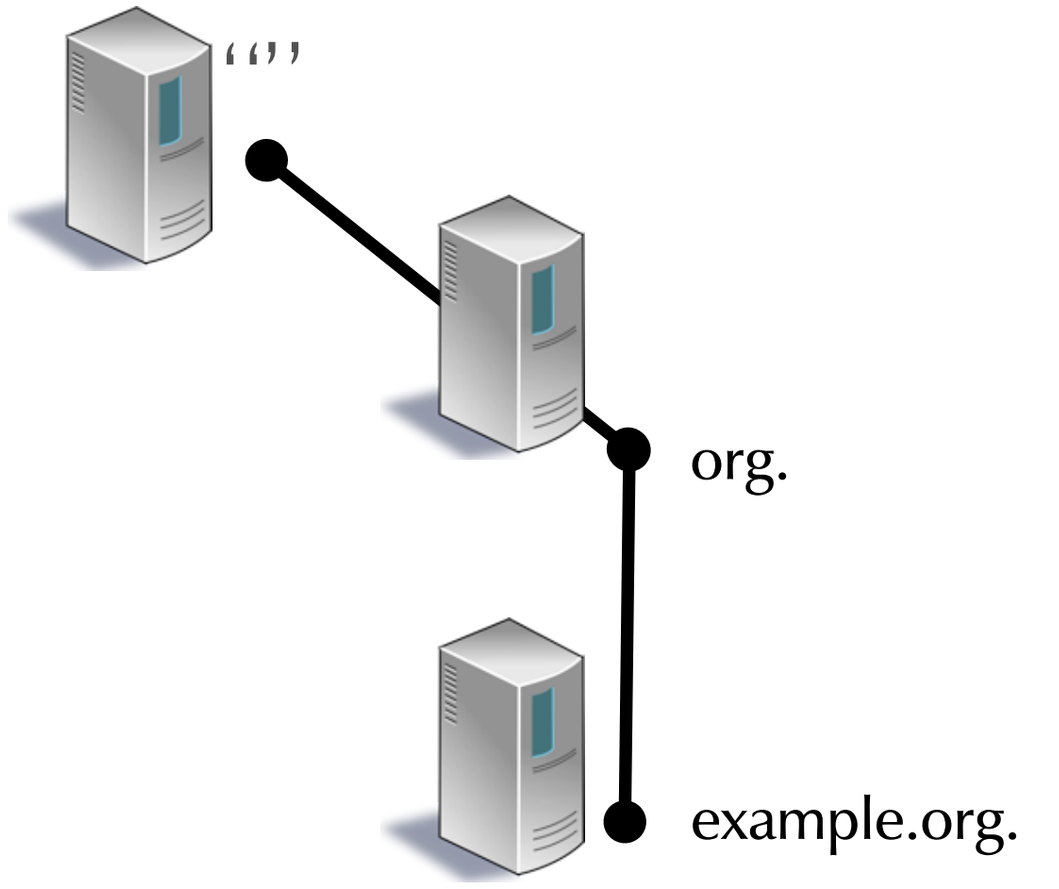




local caching
+ validating
DNS Server

MEN&MICE

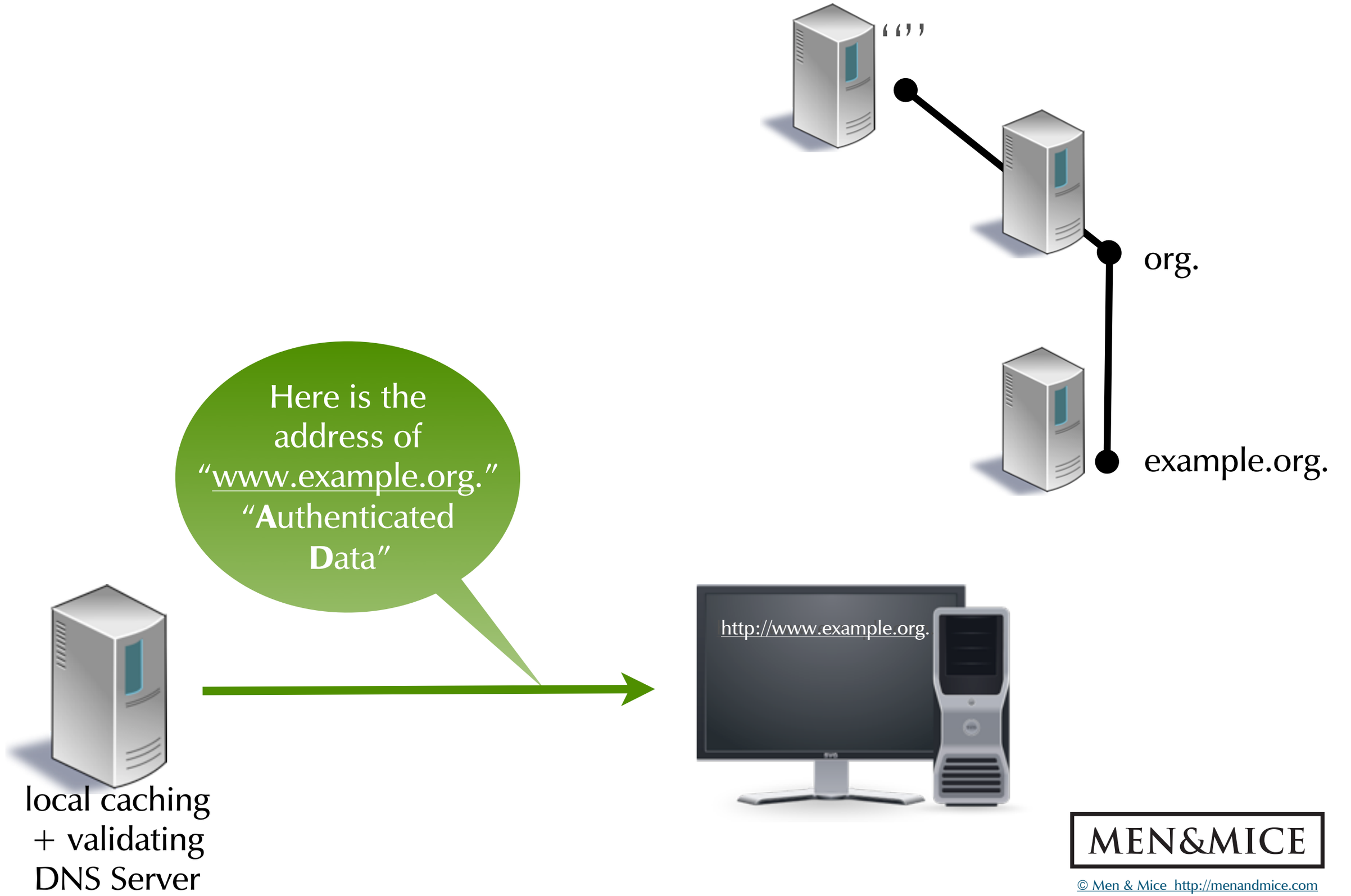
DNSSEC Name Resolution



local caching
+ validating
DNS Server

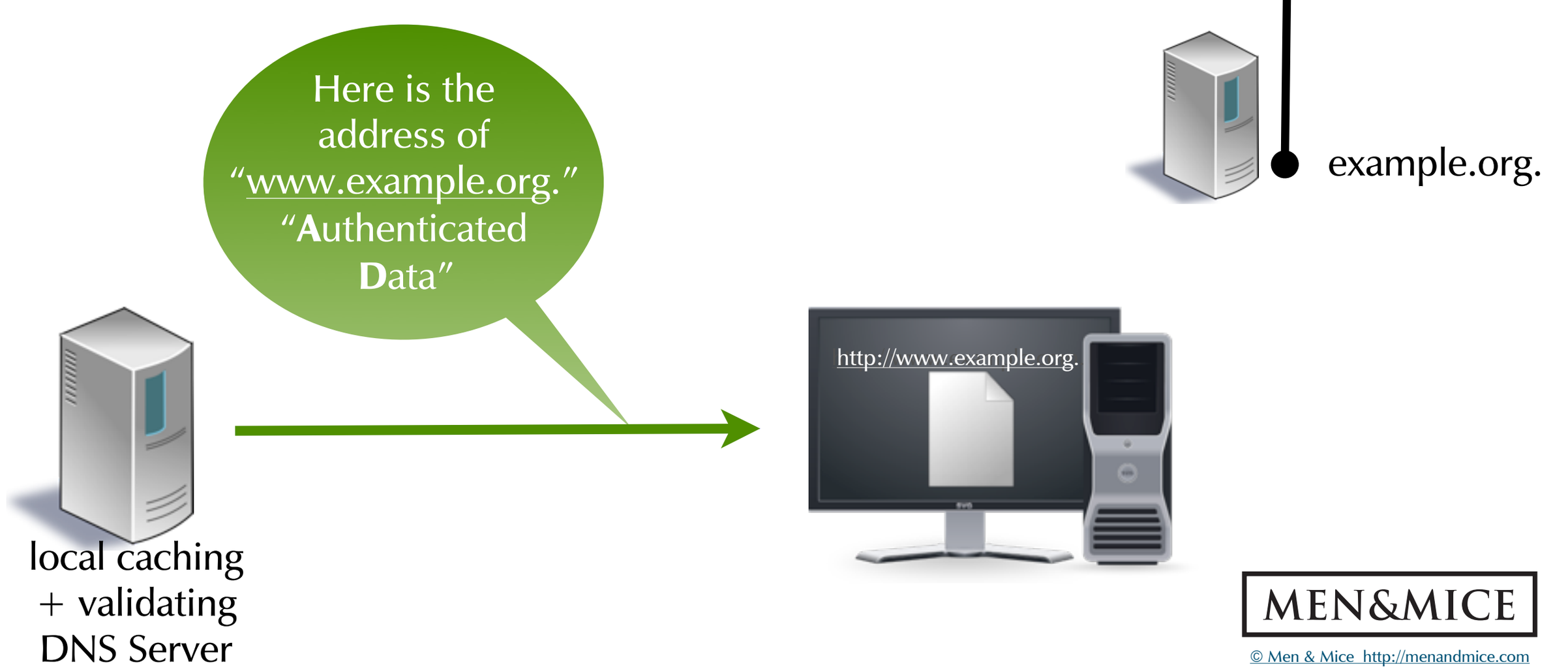
MEN&MICE

DNSSEC Name Resolution



MEN&MICE

DNSSEC Name Resolution



MEN&MICE

The Chain of Trust

- This establishes a chain of trust from the root (or other signed starting point) to any signed zone, each chain link validating the next
  - The root zone's DNSKEY record is widely known
- Or
  - The start of a chain of trust can be verified by a locally configured public key (or hash of a Public Key), an anchor key creating an island of trust
  - A parent zone's DS record is used to identify a child-zone's public key (the DNSKEY record)
  - A childs DS record in the parent zone is signed by the private key of the parent zone
  - The child-zone's DNSKEY record is self-signed by its private zone-signing key (or maybe not)

Implementation of the Chain of Trust

- the root zone has been signed in July 2010
- some top-level zones are already signed today
- for Domains under non-singed parent zones, a validator (resolving DNS Server) can start DNSSEC validation lower down in the namespace tree

Validating DNSSEC in the Internet

MEN&MICE

Internet DNSSEC

- The Internet ROOT-DNS Zone has been DNSSEC signed on 15 July 2010

- For the first time the Internet DNSSEC chain-of-trust is complete

  - it can be used to verify the integrity of the ROOT-Zone DNSSEC data as well as delegated DNSSEC signed zones

DNSSEC in DNS Messages

00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Identification (ID)																QR	Opcode				AA	TC	RD	RA	Z	AD	CD	RCode			
Total Number of Question Resource Records																Total Number of Answer Resource Records															
Total Number of Authority Resource Records																Total Number of Additional Resource Records															
Question Resource Records																															
Answer Resource Records																															
Authority Resource Records																															
Additional Resource Records																															

DNSSEC in DNS Messages

00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Identification (ID)																QR	Opcode				AA	TC	RD	RA	Z	AD	CD	RCode			
Total Number of Question Resource Records																															
Total Number of Authority Resource Records																															
Question Resource Records																															
Answer Resource Records																															
Authority Resource Records																															
Additional Resource Records																															

AD = Authenticated Data

DNSSEC in DNS Messages

00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Identification (ID)																QR	Opcode				AA	TC	RD	RA	Z	AD	CD	RCode			
Total Number of Question Resource Records																															
Total Number of Authority Resource Records																															
Question Resource Records																															
Answer Resource Records																															
Authority Resource Records																															
Additional Resource Records																															

AD = Authenticated Data

CD = Checking disabled

DNSSEC in DNS Messages

00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Identification (ID)																QR	Opcode				AA	TC	RD	RA	Z	AD	CD	RCode			
Total Number of Question Resource Records																															
Total Number of Authority Resource Records																															
Question Resource Records																															
Answer Resource Records																															
Authority Resource Records																															
Additional Resource Records																															

AD = Authenticated Data

CD = Checking disabled

EDNS:
EDNS: version: 0,
flags: do;
udp: 4096

DNSSEC in DNS Messages

- DO Flag in EDNS pseudo record: **D**NSSEC **O**K

- this client can handle DNSSEC records

- in addition, each client signaling “DNSSEC OK” also signals that it can handle UDP DNS responses larger 512 byte

DNSSEC in DNS Messages

- AD Flag:
  - a validating resolver signaling to the client
    - that it has successfully validated the DNSSEC data
    - invalid DNSSEC data will not be send to a downstream resolver (client), instead the resolver will send a SERVFAIL error condition

DNSSEC in DNS Messages

- CD Flag:
  - an Application can signal to the resolving DNS Server that it will validate the DNSSEC information
    - the resolving DNS Server does not need to validate itself, but is free to do so

```
 dig ripe.net +dnssec
; <<>> DiG 9.7.1-P2 <<>> ripe.net +dnssec
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 62183
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 5, ADDITIONAL: 5

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;ripe.net.              IN    A

;; ANSWER SECTION:
ripe.net.         172800      IN    A     193.0.6.139
ripe.net.         172800      IN    RRSIG A 5 2 172800 20101108100147 20101009090147 42006 ripe.net. Jzyeu9MUjNbk[...]5eY=

;; AUTHORITY SECTION:
ripe.net.         172800      IN    NS    sns-pb.isc.org.
ripe.net.         172800      IN    NS    sunic.sunet.se.
ripe.net.         172800      IN    NS    ns-pri.ripe.net.
ripe.net.         172800      IN    NS    ns3.nic.fr.
ripe.net.         172800      IN    RRSIG NS 5 2 172800 20101108100147 20101009090147 42006 ripe.net. I7+d5+U3683o[...]r4U=

;; ADDITIONAL SECTION:
ns-pri.ripe.net.  172800      IN    A     193.0.0.195
ns-pri.ripe.net.  172800      IN    AAAA  2001:610:240:0:53::3
ns-pri.ripe.net.  172800      IN    RRSIG A 5 3 172800 20101108100147 20101009090147 42006 ripe.net. VVZ[...]jwg=
ns-pri.ripe.net.  172800      IN    RRSIG AAAA 5 3 172800 20101108100147 20101009090147 42006 ripe.net. UP/t1m[...]k3k=

;; Query time: 454 msec
;; SERVER: 192.0.2.10#53(192.0.2.10)
;; WHEN: Sat Oct  9 22:39:45 2010
;; MSG SIZE  rcvd: 870
```

```
 dig ripe.net +dnssec
; <<>> DiG 9.7.1-P2 <<>> ripe.net +dnssec
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 62103
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 5, ADDITIONAL: 5

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;ripe.net.                      IN      A

;; ANSWER SECTION:
ripe.net.               172800  IN      A       193.0.6.139
ripe.net.               172800  IN      RRSIG A 5 2 172800 20101108100147 20101009090147 42006 ripe.net. Jzyeu9MUjNbk[...]5eY=

;; AUTHORITY SECTION:
ripe.net.               172800  IN      NS      sns-pb.isc.org.
ripe.net.               172800  IN      NS      sunic.sunet.se.
ripe.net.               172800  IN      NS      ns-pri.ripe.net.
ripe.net.               172800  IN      NS      ns3.nic.fr.
ripe.net.               172800  IN      RRSIG NS 5 2 172800 20101108100147 20101009090147 42006 ripe.net. I7+d5+U3683o[...]r4U=

;; ADDITIONAL SECTION:
ns-pri.ripe.net.        172800  IN      A       193.0.0.195
ns-pri.ripe.net.        172800  IN      AAAA    2001:610:240:0:53::3
ns-pri.ripe.net.        172800  IN      RRSIG A 5 3 172800 20101108100147 20101009090147 42006 ripe.net. VVZ[...]jwg=
ns-pri.ripe.net.        172800  IN      RRSIG AAAA 5 3 172800 20101108100147 20101009090147 42006 ripe.net. UP/t1m[...]k3k=

;; Query time: 454 msec
;; SERVER: 192.0.2.10#53(192.0.2.10)
;; WHEN: Sat Oct  9 22:39:45 2010
;; MSG SIZE  rcvd: 870
```

AD flag: secure answer

EDNS0 information including the DO flag

MEN&MICE

Internet DNSSEC

- Requirement
  - a DNSSEC validating resolver that supports SHA256
    - Unbound 1.4.0 and up
    - BIND 9.6.2 and up
- Windows DNS on Windows 2008R2 currently only supports SHA1 and cannot be used to validate the Internet ROOT.

Unbound
a secure "caching only" DNS Server

MEN&MICE

Unbound

- prototype developed 2006-2007

- the Unbound DNS Server is since maintained by NLNetLabs

EP.NET

nominet

kirei

- BSD license

Unbound

- designed with DNSSEC and IPv6 in mind
  - good scalability on multicore machines
- good security
  - does not trust a single DNS source

Design

- Worker threads access shared hashtable cache
  - Cache LRU, memory use can be configured
- Modular design, state machines work on query
- Python interface for (experimental) extensions

Download Unbound

- the current version is 1.4.7 (Nov 8th 2010)
  - dependencies: libexpat, openssl (1.0x)
  - optional: libevent, libev
- Download:

```
wget http://unbound.net/downloads/unbound-1.4.7.tar.gz
```

Build

- build Unbound:

```
tar xvfz unbound-1.4.7.tar.gz
cd unbound-1.4.7
./configure
make
sudo make install
```

Build

- build Unbound with openssl 0.9.y:

```
tar xvfz unbound-1.4.7.tar.gz
cd unbound-1.4.7
./configure --disable-gost
make
sudo make install
```

Flavors

- Unbound can be compiled into different “flavors”
  - Event loop:
    - build in event library: fast but can only scale for small-medium size traffic
    - libevent or libev (`--with-libevent=/path/to/library`)
  - Threading
    - multithreaded, shared caches
    - forked, dedicated caches per process (`--without-pthreads`)
- more information available at:
  `http://otrs.menandmice.com/otrs/public.pl?Action=PublicFAQ&CategoryID=21&ItemID=47`

Configure

- configure unbound-setup:

```
# unbound-control-setup
generating unbound_server.key
Generating RSA private key, 1536 bit long modulus
.....................................................++++
..............++++
e is 65537 (0x10001)
generating unbound_control.key
Generating RSA private key, 1536 bit long modulus
.++++
.++++
e is 65537 (0x10001)
create unbound_server.pem (self signed certificate)
create unbound_control.pem (signed client certificate)
Signature ok
subject=/CN=unbound-control
Getting CA Private Key
Setup success. Certificates created. Enable in unbound.conf file to use
```

Configure Unbound

- enable unbound-setup:

```
# vi /usr/local/etc/unbound/unbound.conf
[...]
remote-control:
        # Enable remote control with unbound-control(8) here.
        # set up the keys and certificates with unbound-control-setup.
        control-enable: yes

[...]
```

Configure Unbound

- enable auto-trust-anchor updates:

```
# vi /usr/local/etc/unbound/unbound.conf
[...]
# If you want to perform DNSSEC validation, run unbound-anchor before
# you start unbound (i.e. in the system boot scripts).  And enable:
auto-trust-anchor-file: "/usr/local/etc/unbound/root.key"
[...]
```

Configure Unbound

- enable DNSSEC lookaside validation (DLV):

```
# wget -O /usr/local/etc/unbound/dlv.isc.org \
        http://ftp.isc.org/www/dlv/dlv.isc.org.key

# vi /usr/local/etc/unbound/unbound.conf
[...]
# File with DLV trusted keys. Same format as trust-anchor-file.
# There can be only one DLV configured, it is trusted from root down.
# Download http://ftp.isc.org/www/dlv/dlv.isc.org.key
dlv-anchor-file: "dlv.isc.org.key"
[...]
```

Configure Unbound

- enable the german DNSSEC testbed (http://www.denic.de/domains/dnssec/status/resolver-konfiguration):

```
# vi /usr/local/etc/unbound/unbound.conf
[...]
trust-anchor: "de.  DNSKEY  257 3 8
AwEAAZ1FqQED8QBrk3Jk4q96lggh4uiwlbdbZ0posfIgcaJJqfTNBfEhn6PEPqqRP73libD55vujfYzKMN0fVd3
4wrdOpSTpMbw+oqQpJyecfGVYH1fnqws23n5QE03/7SN98O8Cm+HBpB66JurTHWD3f4es8IUoumb/SXY44qb
+oqWfmM3wS8aQVA5d2gHpKrRIPlDHA/
MB3FHGL64VpfV8KJ76kp1RBthR7Y0qalTskOouVeCOEa7gUiIljt1kTf64HFGsRi11klpCHBjtTiTg7MFN25nAS
uhbyTmWlRxPyg79BK7EDQ+tAe09NYkS1P7tOe8ola9IpQHTWO6ttTmSnyE="
[...]
stub-zone:
       name: "de"
      stub-addr: 81.91.161.228  # auth-fra.dnssec.denic.de
      stub-addr: 87.233.175.25  # auth-ams.dnssec.denic.de
      stub-prime: no
```

Configure Unbound

- fetch the public key (DNSKEY) for the german DNSSEC testbed:

```
# dig @81.91.161.228 de dnskey | grep 257

de.          86400   IN  DNSKEY 257 3 8
AwEAAZ1FqQED8QBrk3Jk4q96lggh4uiwlbdbZ0posfIgcaJJqfTNBfEh
n6PEPqqRP73libD55vujfYzKMN0fVd34wrdOpSTpMbw+oqQpJyecfGVY
H1fnqws23n5QE03/7SN98O8Cm+HBpB66JurTHWD3f4es8IUoumb/SXY4 4qb
+oqWfmM3wS8aQVA5d2gHpKrRIPlDHA/MB3FHGL64VpfV8KJ76kp1R
BthR7Y0qalTskOouVeCOEa7gUiIljt1kTf64HFGsRi11klpCHBjtTiTg
7MFN25nASuhbyTmWlRxPyg79BK7EDQ+tAe09NYkS1P7tOe8ola9IpQHT WO6ttTmSnyE=
```

Configure Unbound

- fetch the Internet root trust anchor:

```
# unbound-anchor -v -a /usr/local/etc/unbound/root.key
```

Configure

- create a SYSV startup script:

```
# vi /etc/init.d/unbound
# provide or update the root anchor (if necessary)
unbound-anchor -a "/usr/local/etc/unbound/root.key"
# start validating resolver
# the unbound.conf contains:
#   auto-trust-anchor-file: "/usr/local/etc/unbound/root.key"
unbound -c /usr/local/etc/unbound/unbound.conf

# chmod +x /etc/init.d/unbound
```

Start

- start Unbound:

```
# /etc/init.d/unbound start
```

DNSSEC validation with “dig”

- you can now use “dig” to check if the DNS data resolved is being validated:

```
% dig @127.0.0.1 +dnssec www.ripe.net
; <<>> DiG 9.7.2-P2 <<>> +dnssec www.ripe.net
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 49265
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 5, ADDITIONAL: 5

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;www.ripe.net.                  IN    A

;; ANSWER SECTION:
www.ripe.net.       87489 IN    A     193.0.6.139
www.ripe.net.       87489 IN    RRSIG A 5 3 172800 20100928005807 20100828235807 24225 ripe.net.
ZPvv2TB20Xid1F6EJxmfJJ6+09w4TDoKevr6CxiU7d/voubnLXNZblxw iiFepVw+P1LFVbw40WhRp10K77yjgZHqdbfOTbrV75iW4ED7O/xS/
```

DNSSEC validation with “dig”

- you can now use “dig” to check if the DNS data resolved is being validated:

```
% dig @127.0.0.1 +dnssec www
; <<>> DiG 9.7.2-P2 <<>> +dr
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 49265
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 5, ADDITIONAL: 5

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;www.ripe.net.                  IN    A

;; ANSWER SECTION:
www.ripe.net.       87489 IN    A     193.0.6.139
www.ripe.net.       87489 IN    RRSIG A 5 3 172800 20100928005807 20100828235807 24225 ripe.net.
ZPvv2TB20Xid1F6EJxmfJJ6+09w4TDoKevr6CxiU7d/voubnLXNZblxw iiFepVw+P1LFVbw40WhRp10K77yjgZHqdbfOTbrV75iW4ED7O/xS/
```

AD=authenticated data

DNSSEC validation with "dig"

- you can now use "dig" to check if the DNS data resolved is being validated:

```
% dig @127.0.0.1 +dnssec www
; <<>> DiG 9.7.2-P2 <<>> +dn
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 49265
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 5, ADDITIONAL: 5

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;www.ripe.net.                  IN    A

;; ANSWER SECTION:
www.ripe.net.       87489 IN    A     193.0.6.139
www.ripe.net.       87489 IN    RRSIG A 5 3 172800 20100928005807 20100828235807 24225 ripe.net.
ZPvv2TB20Xid1F6EJxmfJJ6+09w4TDoKevr6CxiU7d/voubnLXNZblxw iiFepVw+P1LFVbw40WhRp10K77yjgZHqdbfOTbrV75iW4ED7O/xS/
```

AD=authenticated data

RRSIG=record signature

DHCP client

- configure your DHCP client to put your machines loopback address in the “resolv.conf” (Example from Ubuntu):

```
# vi /etc/dhcp3/dhclient.conf
[...]
prepend domain-name-servers 127.0.0.1;
[...]
```

DNSSEC validation in Firefox

- Install the Firefox DNSSEC Add-On (`http://www.dnssec-validator.cz/`)

- and then go to

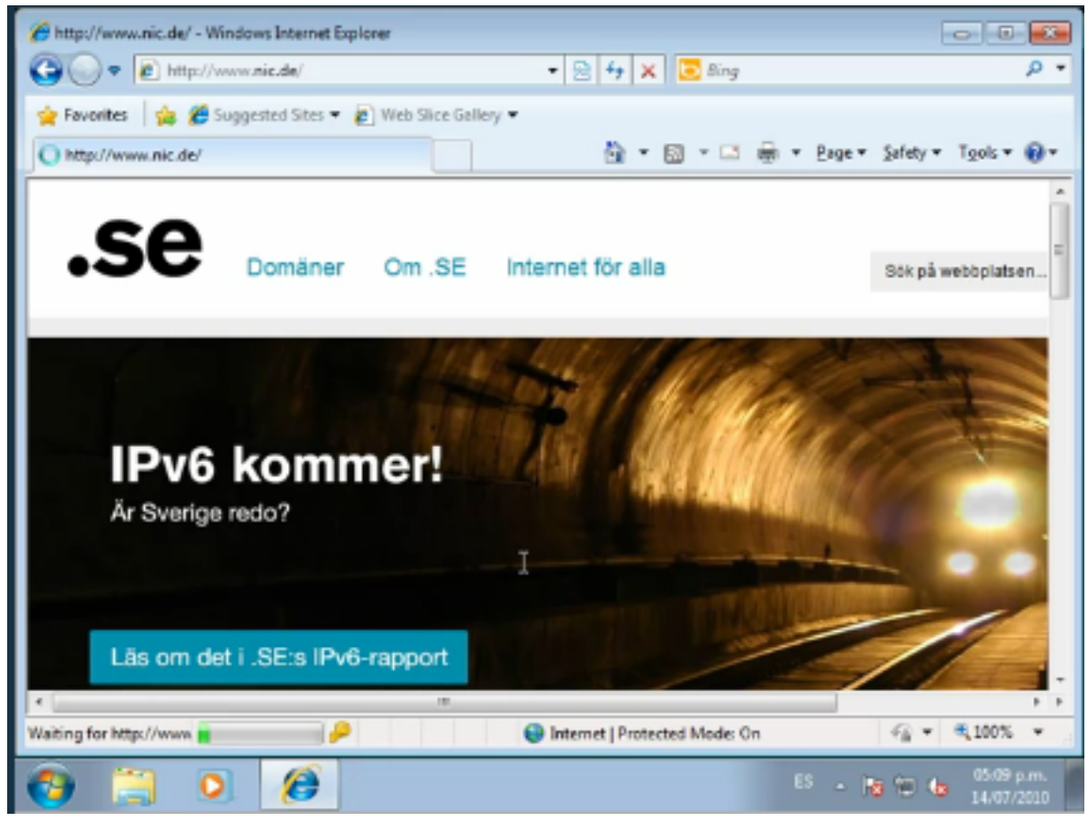
  `http://www.root-dnssec.org`

  or `http://www.ripe.net`

  and you should see a nice green key icon in the URL bar telling you that this DNS information was DNSSEC validated.

DNSSEC validation in Internet Explorer

- ITESM (Instituto Tecnológico y de Estudios Superiores de Monterrey) and Mexico NIC are providing a DNSSEC plugin tool for the Microsoft Internet Explorer



- http://cs.mty.itesm.mx/dnssecmx/

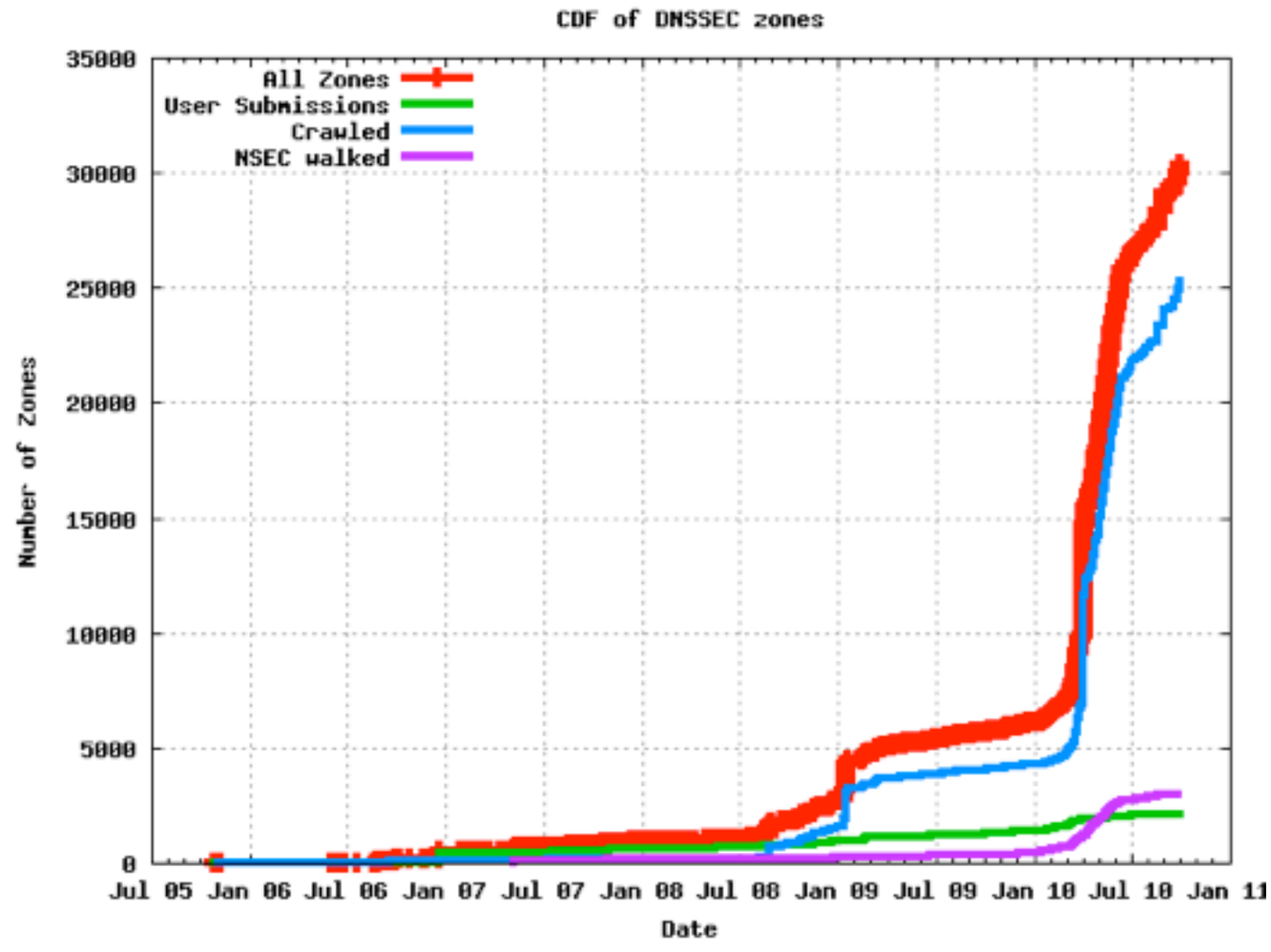
DNS Security Extensions

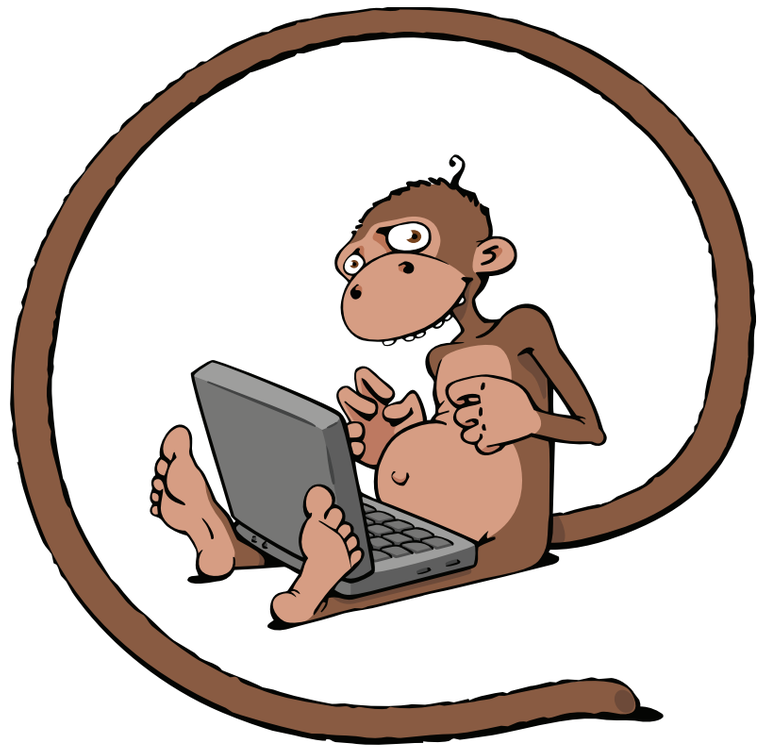
- DNSSEC deployment (http://www.xelerance.com/dnssec/)

http://en.wikipedia.org/wiki/List_of_Internet_top-level_domains

DNS Security Extensions

- DNSSEC growth http://secspider.cs.ucla.edu/images/growth.png





Thank You!

Contact:
carsten@strotmann.de

MEN&MICE