OFF-PATH ATTACKS AGAINST PKI

Markus Brandt, Tianxiang Dai, Amit Klein, Haya Shulman, Michael Waidner

Member of CRISP Center for Research in Security and Privacy

WHO AM I?

- Security Researcher
  - Technische Universität Darmstadt
  - Fraunhofer-Institut für Sichere Informationstechnologie
- Freelancer
- Teaching IT Security at TU Darmstadt
- Special interest:
  - Network Security
  - Crypto
  - Reverse engineering
  - …

OUTLINE

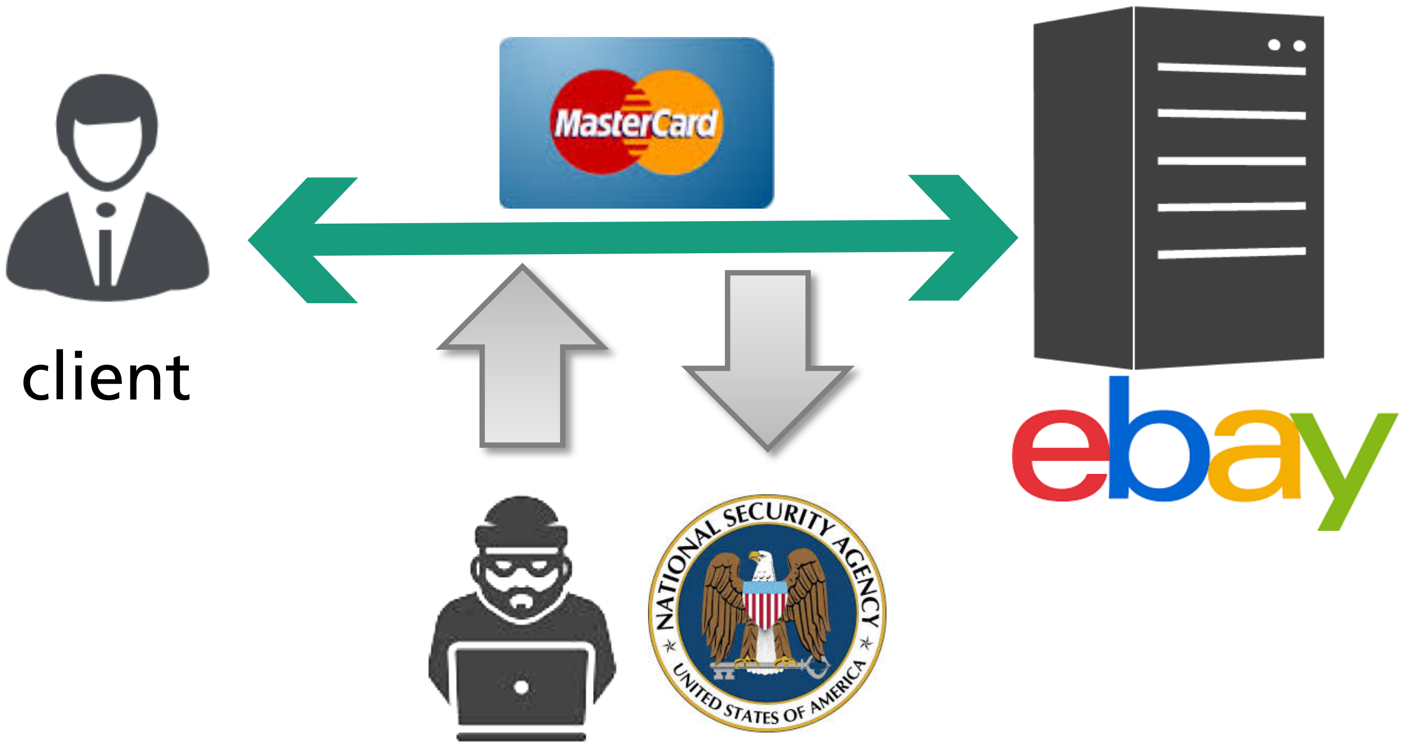
- Public-Key Infrastructures
- Domain Validation
- Off-Path attack against Domain Validation
- Defences
- Conclusion

DEEPSEC

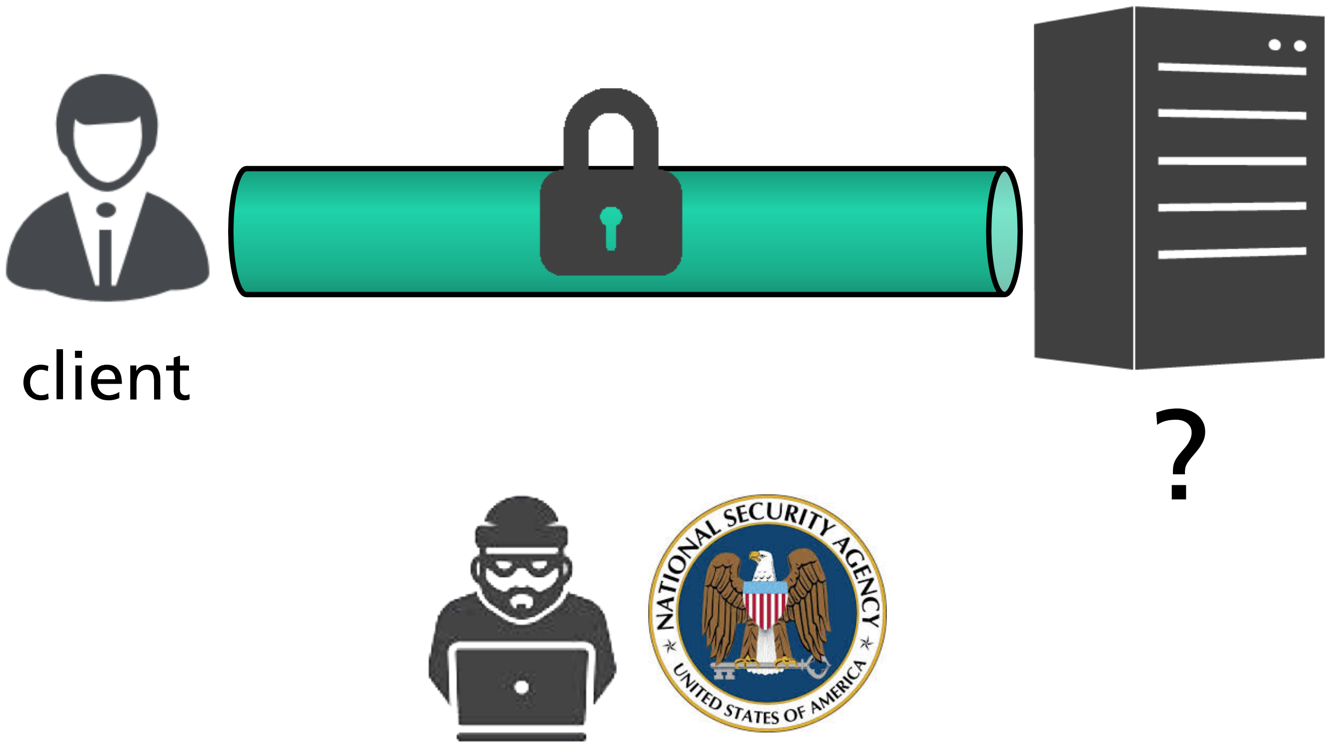
PUBLIC-KEY INFRASTRUCTURES

LET'S START WITH A SIMPLE EXAMPLE

INSECURE WITHOUT ENCRYPTION



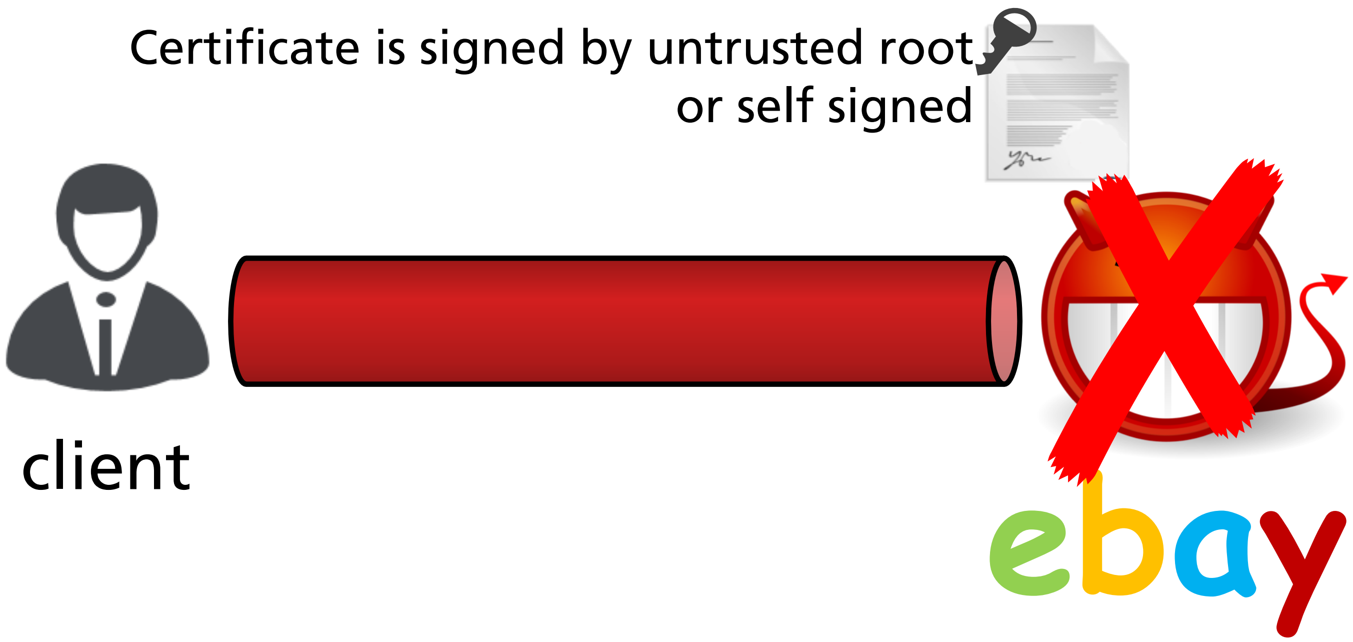
ENCRYPTION PROTECTS THE DATA



DEEPSEC

CERTIFICATES BIND CRYPTOGRAPHIC KEYS TO SUBJECTS

CERTIFICATES BIND CRYPTOGRAPHIC KEYS TO SUBJECTS



DEEPSEC

DOMAIN VALIDATION

MORE THAN 100 ROOT CA IN BROWSERS

Domain Validation (DV)

Automated
Free / cheap
Immediate

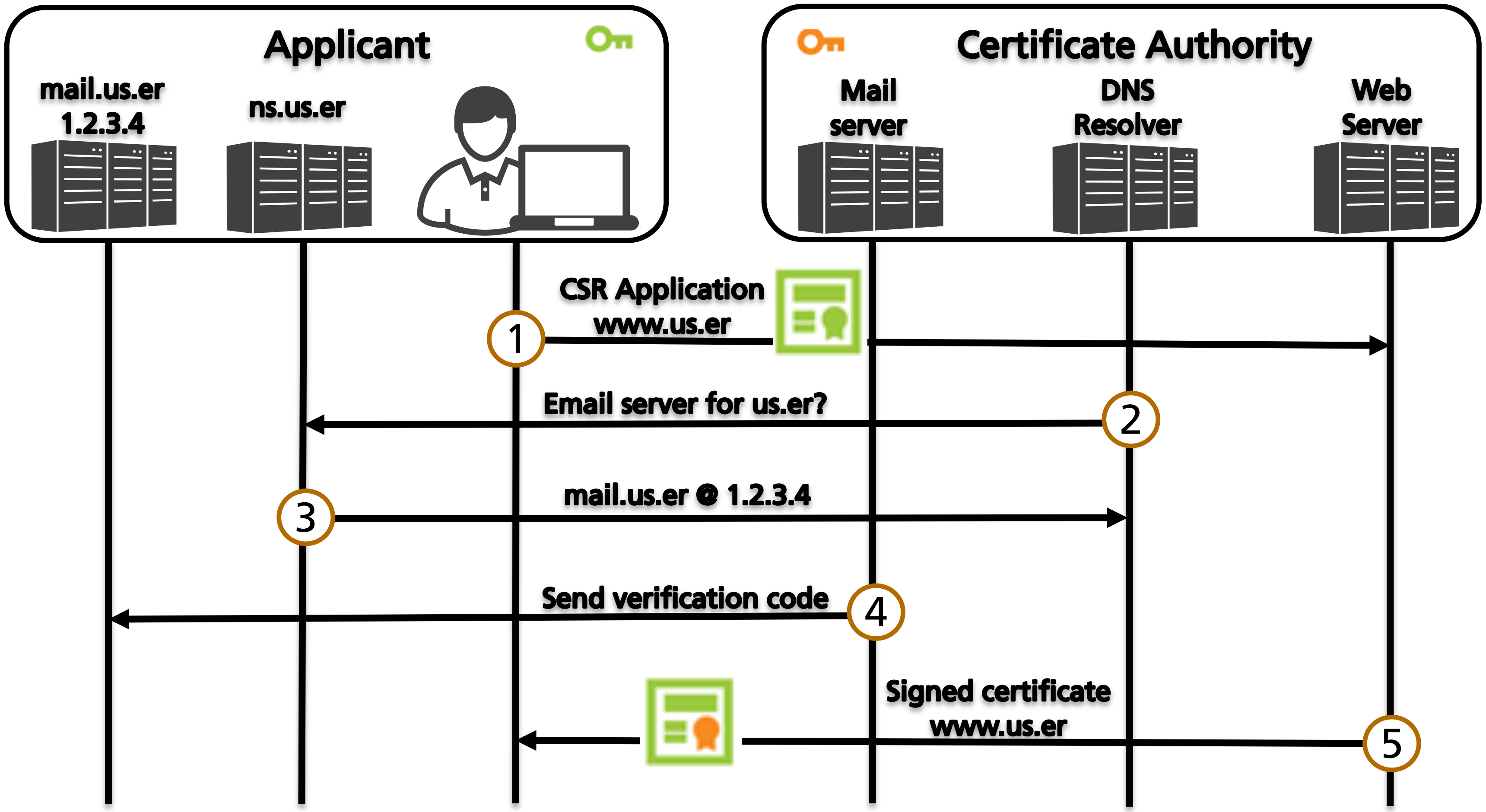
Organisation Validation (OV) | **Extended Validation (EV)**

Manual | Time-consuming | Expensive

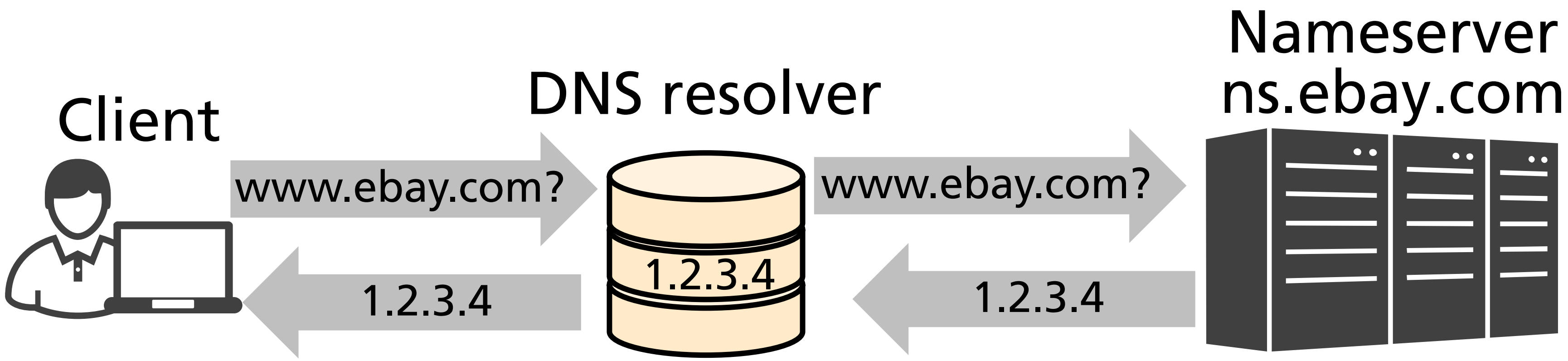
- We tested 17 CAs that perform DV
  - They control over 95% of the certificates market
- Five were vulnerable
- Only one vulnerable CA is sufficient
  - Usually it doesn’t matter which CA signed it

DEEPSEC

CERTIFICATE ISSUANCE WITH DOMAIN VALIDATION



RESOLVING A DOMAIN NAME

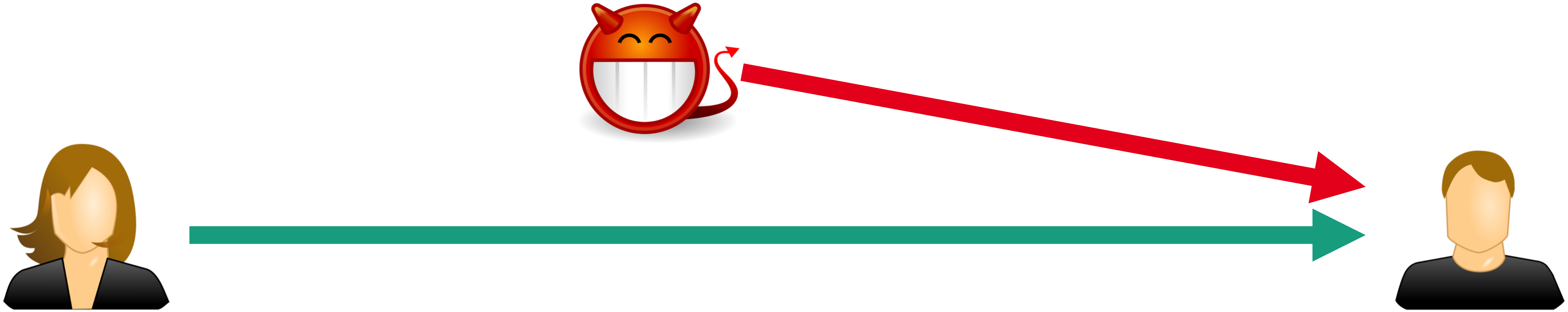


DEEPSEC

REPLYING FROM CACHE

OFF-PATH ATTACK AGAINST DOMAIN VALIDATION

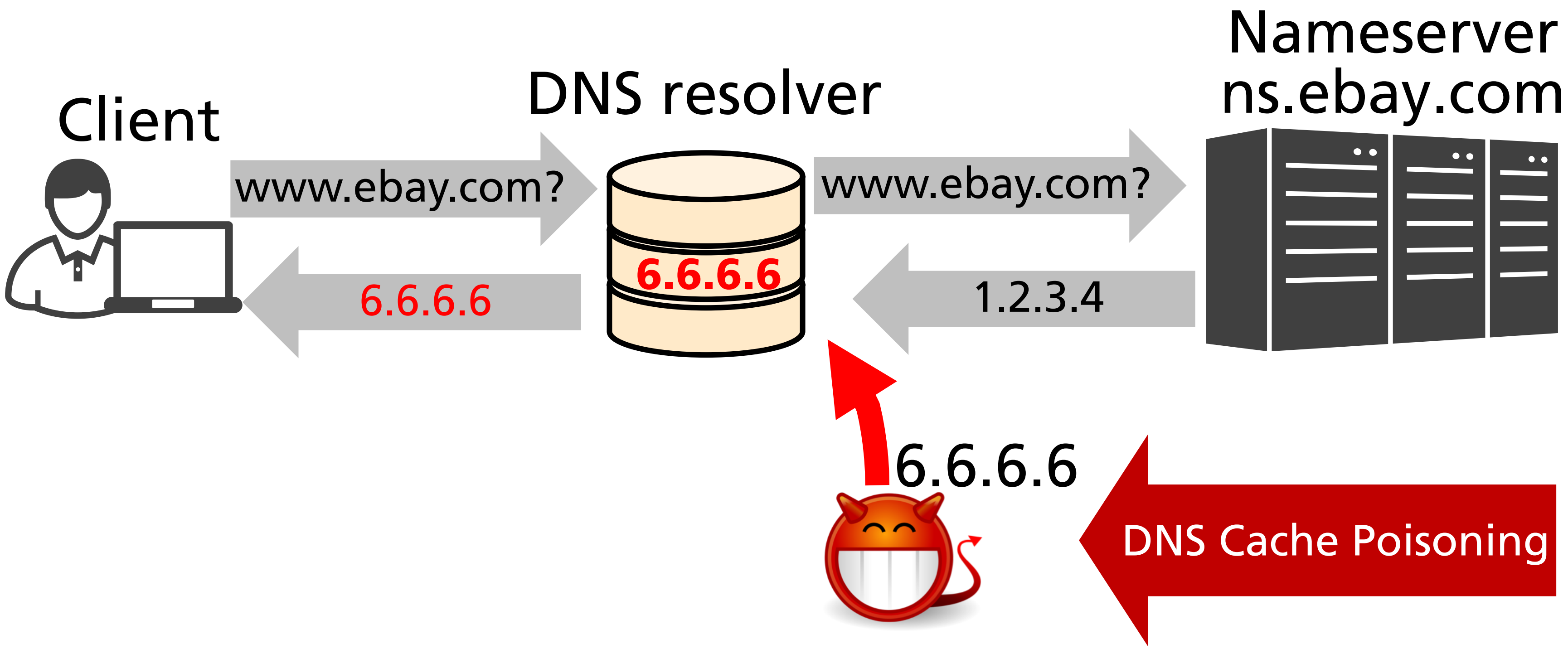
WE ASSUME THE WEAKEST ATTACKER



Off-Path (Spoofing) Attackers can:

- only inject packets
- not eavesdrop
- not modify or delay packets in any way

DNS CACHE POISONING



AGAINST OFF-PATH POISONING: CHALLENGE-RESPONSE

- Send request from random port (16 Bit)
- Select random DNS transaction ID (also 16 Bit)
  - 2^{32} values

→impractical to guess!

DNS PACKET: IP HEADER

<table>
<tr><th>Bit</th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th><th>15</th><th>16</th><th>17</th><th>18</th><th>19</th><th>20</th><th>21</th><th>22</th><th>23</th><th>24</th><th>25</th><th>26</th><th>27</th><th>28</th><th>29</th><th>30</th><th>31</th><th rowspan="6">IP Header</th></tr>
<tr><td>0</td><td colspan="4">v4</td><td colspan="4">IHL</td><td colspan="8">TOS</td><td colspan="16">Total Length</td></tr>
<tr><td>32</td><td colspan="16">IP Identifier</td><td colspan="3">Flags</td><td colspan="13">Fragment Offset</td></tr>
<tr><td>64</td><td colspan="8">Time To Live</td><td colspan="8">Protocol</td><td colspan="16">IP Header Checksum</td></tr>
<tr><td>96</td><td colspan="32">Source IP Address</td></tr>
<tr><td>128</td><td colspan="32">Destination IP Address</td></tr>
</table>

DNS PACKET: UDP HEADER

<table>
<tr><th>Bit</th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th><th>15</th><th>16</th><th>17</th><th>18</th><th>19</th><th>20</th><th>21</th><th>22</th><th>23</th><th>24</th><th>25</th><th>26</th><th>27</th><th>28</th><th>29</th><th>30</th><th>31</th><th></th></tr>
<tr><td>0</td><td colspan="4">v4</td><td colspan="4">IHL</td><td colspan="8">TOS</td><td colspan="16">Total Length</td><td rowspan="5">IP Header</td></tr>
<tr><td>32</td><td colspan="16">IP Identifier</td><td colspan="3">Flags</td><td colspan="13">Fragment Offset</td></tr>
<tr><td>64</td><td colspan="8">Time To Live</td><td colspan="8">Protocol</td><td colspan="16">IP Header Checksum</td></tr>
<tr><td>96</td><td colspan="32">Source IP Address</td></tr>
<tr><td>128</td><td colspan="32">Destination IP Address</td></tr>
<tr><td>160</td><td colspan="16">Source Port</td><td colspan="16">Destination Port</td><td rowspan="2">UDP Header</td></tr>
<tr><td>192</td><td colspan="16">Length</td><td colspan="16">UDP Checksum</td></tr>
</table>

DNS PACKET: DNS HEADER

<table>
<tr><th>Bit</th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th><th>15</th><th>16</th><th>17</th><th>18</th><th>19</th><th>20</th><th>21</th><th>22</th><th>23</th><th>24</th><th>25</th><th>26</th><th>27</th><th>28</th><th>29</th><th>30</th><th>31</th><th></th></tr>
<tr><td>0</td><td colspan="4">v4</td><td colspan="4">IHL</td><td colspan="8">TOS</td><td colspan="16">Total Length</td><td rowspan="6">IP Header</td></tr>
<tr><td>32</td><td colspan="16">IP Identifier</td><td colspan="3">Flags</td><td colspan="13">Fragment Offset</td></tr>
<tr><td>64</td><td colspan="8">Time To Live</td><td colspan="8">Protocol</td><td colspan="16">IP Header Checksum</td></tr>
<tr><td>96</td><td colspan="32">Source IP Address</td></tr>
<tr><td>128</td><td colspan="32">Destination IP Address</td></tr>
<tr><td>160</td><td colspan="16">Source Port</td><td colspan="16">Destination Port</td><td rowspan="2">UDP Header</td></tr>
<tr><td>192</td><td colspan="16">Length</td><td colspan="16">UDP Checksum</td></tr>
<tr><td>224</td><td colspan="16">Transaction Identifier (TXID)</td><td colspan="16">DNS Flags</td><td rowspan="3">DNS Header</td></tr>
<tr><td>256</td><td colspan="16">Question Count</td><td colspan="16">Answer Record Count</td></tr>
<tr><td>288</td><td colspan="16">Authority Record Count</td><td colspan="16">Additional Record Count</td></tr>
</table>

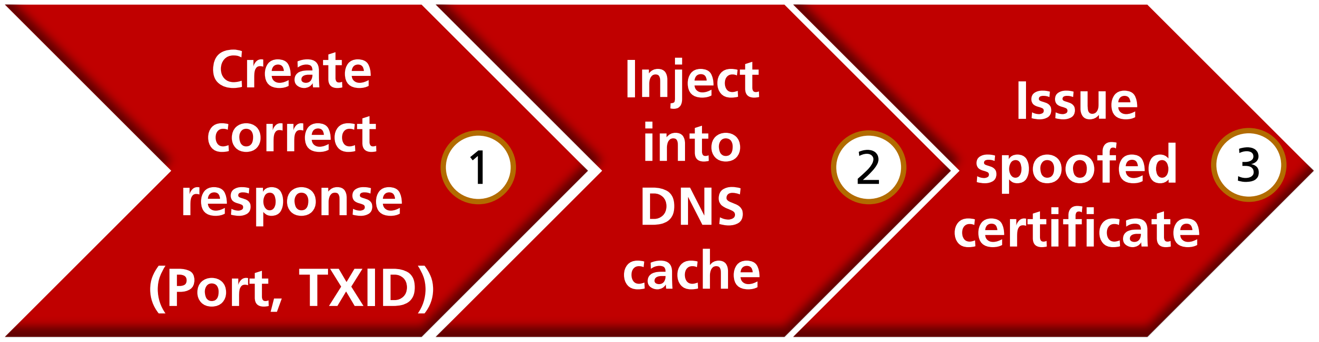
DNS PACKET: DNS PAYLOAD

<table>
<tr><th>Bit</th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th><th>15</th><th>16</th><th>17</th><th>18</th><th>19</th><th>20</th><th>21</th><th>22</th><th>23</th><th>24</th><th>25</th><th>26</th><th>27</th><th>28</th><th>29</th><th>30</th><th>31</th><th></th></tr>
<tr><td>0</td><td colspan="4">v4</td><td colspan="4">IHL</td><td colspan="8">TOS</td><td colspan="16">Total Length</td><td rowspan="5">IP Header</td></tr>
<tr><td>32</td><td colspan="16">IP Identifier</td><td colspan="3">Flags</td><td colspan="13">Fragment Offset</td></tr>
<tr><td>64</td><td colspan="8">Time To Live</td><td colspan="8">Protocol</td><td colspan="16">IP Header Checksum</td></tr>
<tr><td>96</td><td colspan="32">Source IP Address</td></tr>
<tr><td>128</td><td colspan="32">Destination IP Address</td></tr>
<tr><td>160</td><td colspan="16">Source Port</td><td colspan="16">Destination Port</td><td rowspan="2">UDP Header</td></tr>
<tr><td>192</td><td colspan="16">Length</td><td colspan="16">UDP Checksum</td></tr>
<tr><td>224</td><td colspan="16">Transaction Identifier (TXID)</td><td colspan="16">DNS Flags</td><td rowspan="3">DNS Header</td></tr>
<tr><td>256</td><td colspan="16">Question Count</td><td colspan="16">Answer Record Count</td></tr>
<tr><td>288</td><td colspan="16">Authority Record Count</td><td colspan="16">Additional Record Count</td></tr>
<tr><td>…</td><td colspan="32">Question Section</td><td rowspan="4">DNS Payload</td></tr>
<tr><td></td><td colspan="32">Answer Section</td></tr>
<tr><td></td><td colspan="32">Authority Section</td></tr>
<tr><td></td><td colspan="32">Additional Section</td></tr>
</table>

CHALLENGES

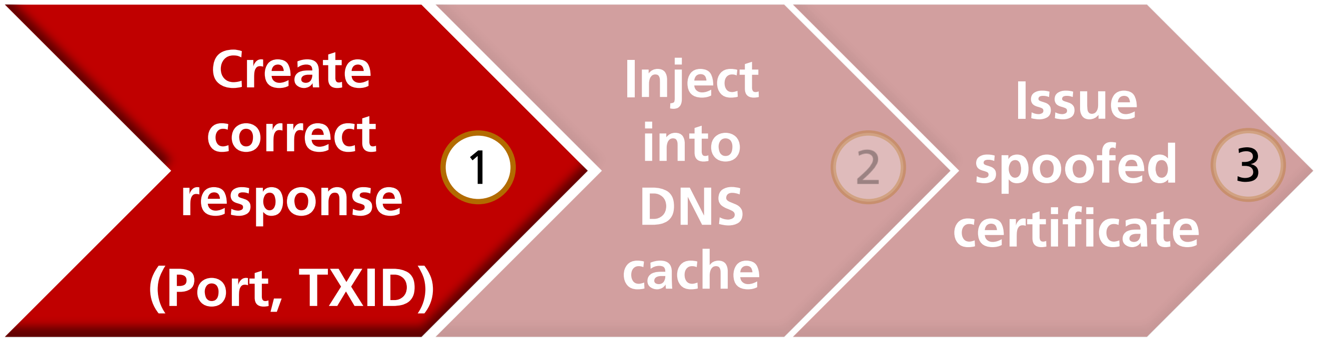
- Modify communication without seeing it and without access to it
- Overwrite cached record with incorrect value
- Exploit DNS cache poisoning to circumvent PKI authentication (and issue certificate)

GOALS

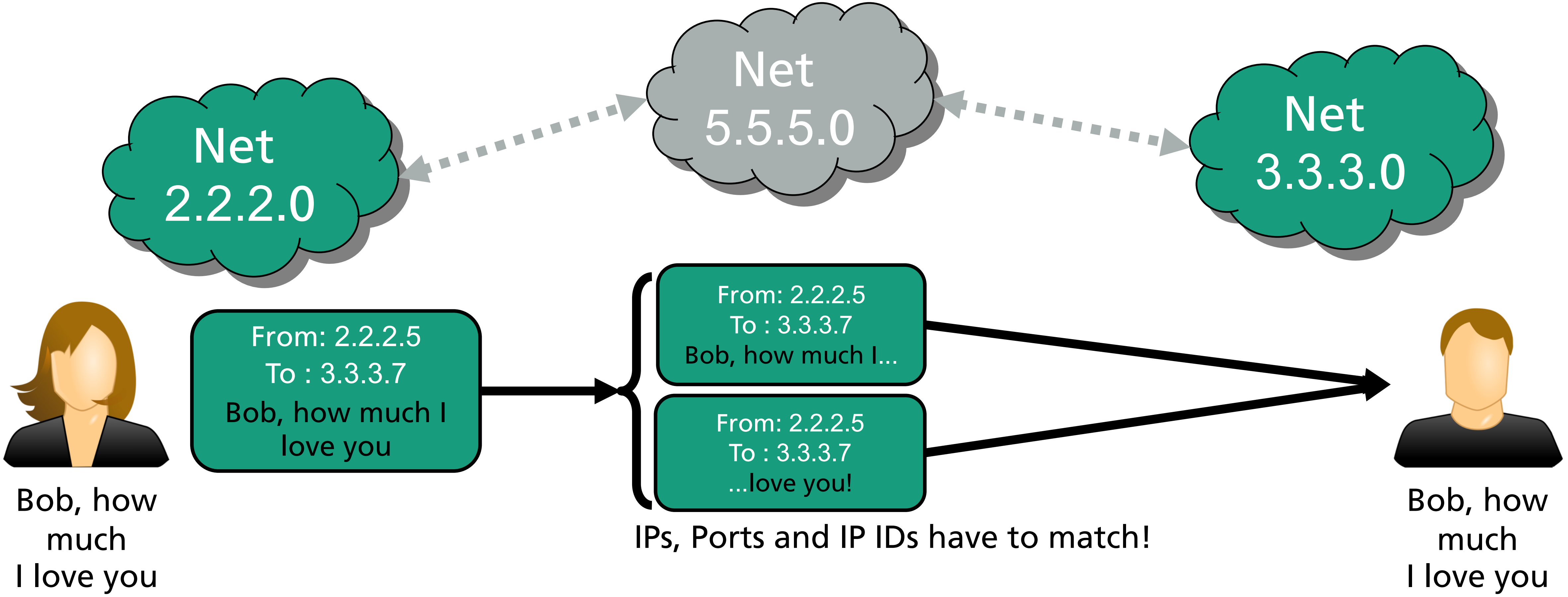


DEEPSEC

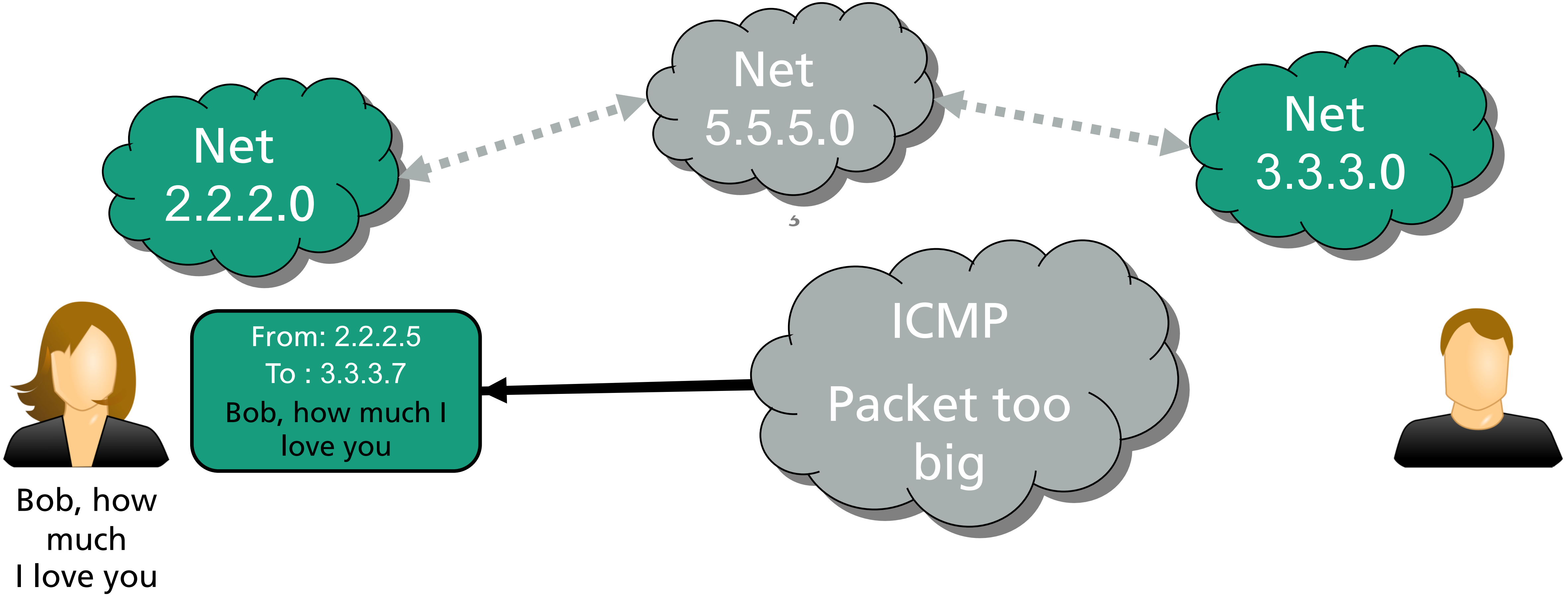
LET'S START WITH STEP 1



LARGE PACKETS CAN GET FRAGMENTED ON PATH



FRAGMENTATION CAN ALSO BE REQUESTED



DEEPSEC

ICMP

FRAGMENTATION NEEDED AND DON'T FRAGMENT WAS SET

<table>
<tr><th>Bit</th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th><th>15</th><th>16</th><th>17</th><th>18</th><th>19</th><th>20</th><th>21</th><th>22</th><th>23</th><th>24</th><th>25</th><th>26</th><th>27</th><th>28</th><th>29</th><th>30</th><th>31</th><th></th></tr>
<tr><td>0</td><td colspan="4">v4</td><td colspan="4">IHL</td><td colspan="8">TOS</td><td colspan="16">Total Length</td><td rowspan="5">IP Header</td></tr>
<tr><td>32</td><td colspan="16">IP Identifier</td><td colspan="3">Flags</td><td colspan="13">Fragment Offset</td></tr>
<tr><td>64</td><td colspan="8">Time To Live</td><td colspan="8">Protocol</td><td colspan="16">IP Header Checksum</td></tr>
<tr><td>96</td><td colspan="32">Source IP Address</td></tr>
<tr><td>128</td><td colspan="32">Destination IP Address</td></tr>
<tr><td>160</td><td colspan="8">Type = 3</td><td colspan="8">Code = 4</td><td colspan="16">ICMP Checksum</td><td rowspan="2">ICMP Header</td></tr>
<tr><td>192</td><td colspan="16">Unused</td><td colspan="16">MTU = 100</td></tr>
<tr><td>224</td><td colspan="4">v4</td><td colspan="4">IHL</td><td colspan="8">TOS</td><td colspan="16">Total Length</td><td rowspan="5">IP Header</td></tr>
<tr><td>256</td><td colspan="16">IP Identifier</td><td colspan="3">Flags</td><td colspan="13">Fragment Offset</td></tr>
<tr><td>288</td><td colspan="8">Time To Live</td><td colspan="8">Protocol</td><td colspan="16">IP Header Checksum</td></tr>
<tr><td>320</td><td colspan="32">Source IP Address</td></tr>
<tr><td>352</td><td colspan="32">Destination IP Address</td></tr>
<tr><td></td><td colspan="32">...</td><td>IP Payload</td></tr>
</table>

DEEPSEC

FORCING FRAGMENTATION

- Among 5K-top Alexa that reduce the MTU
  - 33,4% allow >= 296 bytes
  - 11% allow < 296 bytes
- ICMP Messages can be sent by anyone
- OSes typically do not apply any checks for UDP
  - UDP is stateless.

EXPLOITING FRAGMENTATION AGAINST DNS

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	v4				IHL				TOS								Total Length															
32	IP Identifier																Flags			Fragment Offset												
64	Time To Live								Pr								IP Header Checksum															
96																																
128																																
160	Source Port																Destination Port															
192	Length																UDP Checksum															
224	Transaction Identifier (TXID)																DNS Flags															
256	Question Count																Answer Record Count															
288	Authority Record Count																Additional Record Count															
…	Question Section																															
	Answer Section																															
	Authority Section																															
	Additional Section																															

But what about the IP ID?

Modify only this!

RFC 791

September 1981

Internet Protocol

Overview

Fragmentation

[...]
The identification field is used to distinguish the fragments of one datagram from those of another. The originating protocol module of an internet datagram **sets the identification field to a value that must be unique** for that source-destination pair and protocol for the time the datagram will be active in the internet system. The originating protocol module of a complete datagram sets the more-fragments flag to zero and the fragment offset to zero.
[...]

DEEPSEC

HOW DO MAKE IP IDENTIFIERS UNIQUE?

RANDOM IP IDENTIFIERS

- Very few servers use random IP ID values (<1%)
- Quite complicated
  - Needs enough entropy
  - Has to check for collisions

HOW DO MAKE IP IDENTIFIERS UNIQUE?

Per-Destination IP ID

- <40% of the nameservers use a per-destination incrementing IP ID
- Default in Linux
- Attacks exist [KC14]

[KC14] Jeffrey Knockel and Jedidiah R Crandall. 2014. Counting Packets Sent Between Arbitrary Internet Hosts.. In FOCI.

HOW DO MAKE IP IDENTIFIERS UNIQUE?

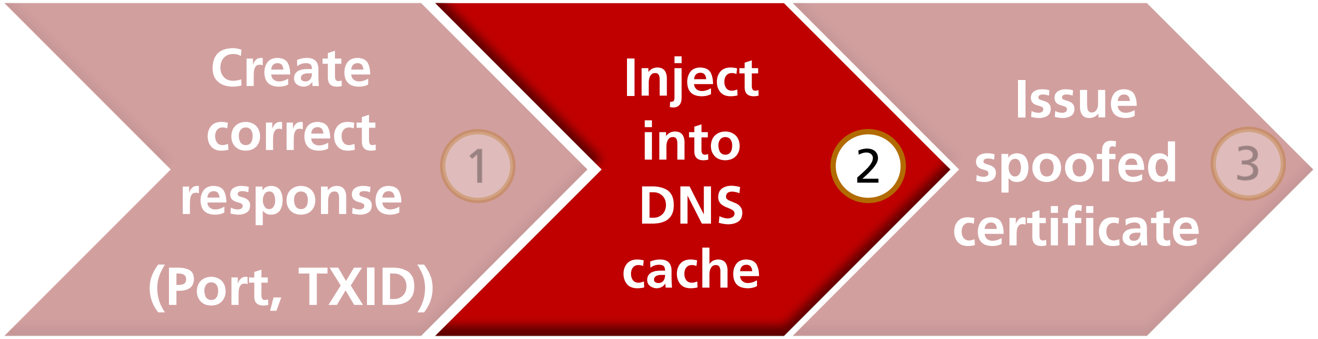
Sequentially Incrementing IP ID

- >60% of 10K-top Alexa domains use sequentially incrementing IP ID values
- Easiest to attack
- Simply estimate incremention

WHAT HAPPENS IF THE 2ND FRAGMENT ARRIVES FIRST?

- Operating systems keep 2nd fragment and wait for 1st fragment
  - Windows keeps 100 fragments
  - Linux keeps 64 fragments
  - Older Linux kernels allow for thousands of fragments
  - Can be set via *ip_frag_max_dist*

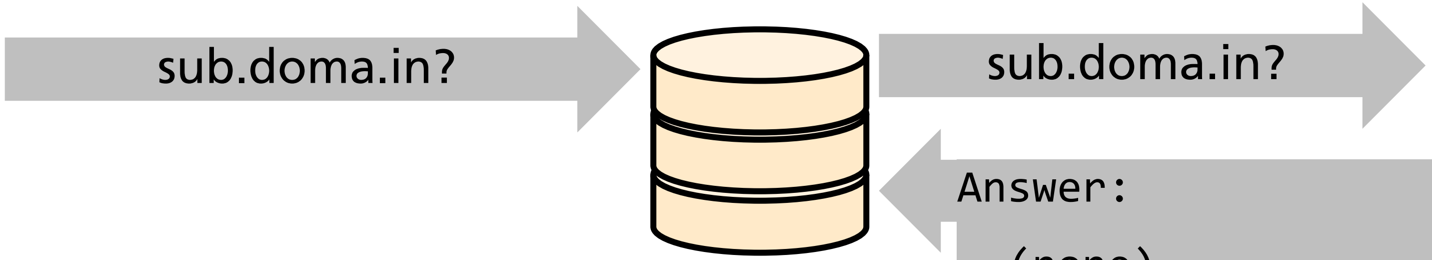
NOW WE WANT TO INSERT OUR ENTRY INTO THE CACHE



DEEPSEC

ADVANCED CACHE POISONING

DNS resolver



ns.doma.in
2.2.2.2

Cache:

Domain	IP
doma.in	1.1.1.1
ns.doma.in	2.2.2.2
…	…

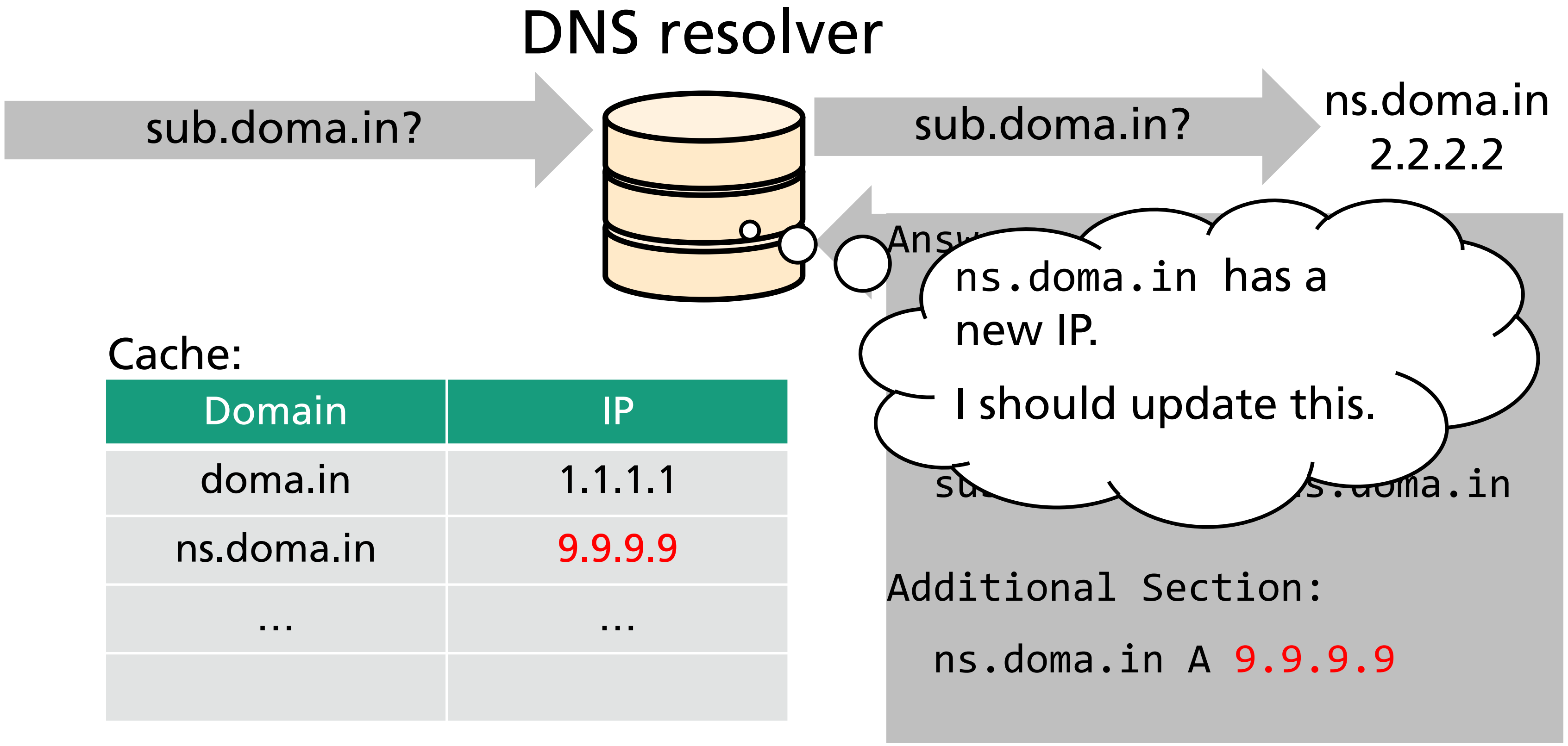
```
Answer:
  (none)

Authoritative Server:
  sub.doma.in NS ns.doma.in

Additional Section:
  ns.doma.in A 9.9.9.9
```


DEEPSEC

ADVANCED CACHE POISONING



Cache:

Domain	IP
doma.in	1.1.1.1
ns.doma.in	9.9.9.9
…	…

THIS WAS JUST ONE (VERY SIMPLE) EXAMPLE

Test Name	DNS Fields	Values in DNS Fields	Overrides Cached	Auth Ref.	Direct	Defence
1. NS0	Q An Au Ad	A? two.test-ns0.TAIL two.test-ns0.TAIL A a.b.c.d test-ns0.TAIL NS ns2.test-ns0.TAIL ns2.test-ns0.TAIL A ATTACKER	test-ns0.TAIL NS ns.test-ns0.TAIL	No	No	(A)/(B)
2. NS0-auth	Q An Au Ad	A? two.test-ns0-auth.TAIL two.test-ns0-auth.TAIL A a.b.c.d test-ns0-auth.TAIL NS ns2.test-ns0-auth.TAIL ns2.test-ns0-auth.TAIL A ATTACKER	test-ns0-auth.TAIL NS ns.test-ns0-auth.TAIL	Yes	No	(A)/(B)
3. NS	Q An Au Ad	A? ns2.test-ns.TAIL ns2.test-ns.TAIL A ATTACKER test-ns.TAIL NS ns2.test-ns.TAIL —	test-ns.TAIL NS ns.test-ns.TAIL	No	No	(A)
4. NS-auth	Q An Au Ad	A? ns2.test-ns-auth.TAIL ns2.test-ns-auth.TAIL A ATTACKER test-ns-auth.TAIL NS ns2.test-ns-auth.TAIL —	test-ns-auth.TAIL NS ns.test-ns-auth.TAIL	Yes	No	(A)
5. NS2	Q An Au Ad	A? two.test-ns2.TAIL two.test-ns2.TAIL A a.b.c.d test-ns2.TAIL NS ns2.magic-ns2.TAIL —	test-ns2.TAIL NS ns.test-ns2.TAIL	No	No	(A)
6. NS2-auth	Q An Au Ad	A? two.test-ns2-auth.TAIL two.test-ns2-auth.TAIL A a.b.c.d test-ns2-auth.TAIL NS ns2.magic-ns2-auth.TAIL —	test-ns2-auth.TAIL NS ns.test-ns2-auth.TAIL	Yes	No	(A)
7. b4	Q An Au Ad	A? two.test-b4.TAIL — sub.test-b4.TAIL NS ns.test-b4.TAIL ns.test-b4.TAIL A ATTACKER	ns.test-b4.TAIL A a.b.c.d	N/A	No	(B)
8. u1-auth	Q An Au Ad	A? two.test-u1-auth.TAIL — test-u1-auth.TAIL NS ns2.test-u1-auth.TAIL ns2.test-u1-auth.TAIL A ATTACKER	test-u1-auth.TAIL NS ns.test-u1-auth.TAIL	Yes	No	(A)/(B)
9. u3-2	Q An Au Ad	A? two.test-u3-2.TAIL two.test-u3-2.TAIL A a.b.c.d test-u3-2.TAIL NS ns.test-u3-2.TAIL ns.test-u3-2.TAIL A ATTACKER	ns.test-u3-2.TAIL A a.b.c.d	N/A	No	(B)
10. u3-3	Q An Au Ad	A? two.test-u3-3.TAIL — test-u3-3.TAIL NS ns.test-u3-3.TAIL ns.test-u3-3.TAIL A ATTACKER	ns.test-u3-3.TAIL A a.b.c.d	N/A	No	(B)
11. u3-4	Q An Au Ad	A? two.sub.test-u3-4.TAIL — sub.test-u3-4.TAIL NS ns.test-u3-4.TAIL ns.test-u3-4.TAIL A ATTACKER	ns.test-u3-4.TAIL A a.b.c.d	N/A	No	(B)
12. w-dname	Q An Au Ad	A? two.test-w-dname.TAIL test-w-dname.TAIL DNAME magic-w-dname.TAIL — —	(all).test-w-dname.TAIL All types	N/A	No	no DNAME from cache
13. w7	Q An Au/Ad	A? two.test-w7.TAIL two.test-w7.TAIL CNAME ns.test-w7.TAIL; ns.test-w7.TAIL A ATTACKER —	ns.test-w7.TAIL A a.b.c.d	N/A	No	[break] CNAME chain
14. w8	Q An Au/Ad	A? ns.sub.test-w8.TAIL sub.test-w8.TAIL DNAME test-w8.TAIL; ns.test-w8.TAIL A ATTACKER —	ns.test-w8.TAIL A a.b.c.d	N/A	No	[break] DNAME chain
15. dname	Q An Au Ad	A? zwei.test-dname.TAIL test-dname.TAIL DNAME magic-dname.TAIL — —	(all).test-dname.TAIL ALL types	N/A	Yes	no DNAME from cache
16. ak1	Q An Au/Ad	A? zwei1.test-ak1.TAIL zwei1.test-ak1.TAIL CNAME one1.test-ak1.TAIL; one1.test-ak1.TAIL CNAME one1.magic-ak1.TAIL —	one1.test-ak1.TAIL ALL TYPES	N/A	Yes	[break] CNAME chain
17. w11	Q An Au Ad	A? zwei.one1.test-w11.TAIL — one1.test-w11.TAIL NS ns2.magic-w11.TAIL —	one1.test-w11.TAIL NS ANY	N/A	Yes	(A)
18. w11bis	Q An Au Ad	A? zwei.one1.test-w11bis.TAIL — one1.test-w11bis.TAIL NS ns2.test-w11bis.TAIL ns2.test-w11bis.TAIL A ATTACKER	one1.test-w11bis.TAIL NS ANY	N/A	Yes	(A)/(B)

"Internet-wide study of DNS cache injections" [KSW17] examines 18 different techniques

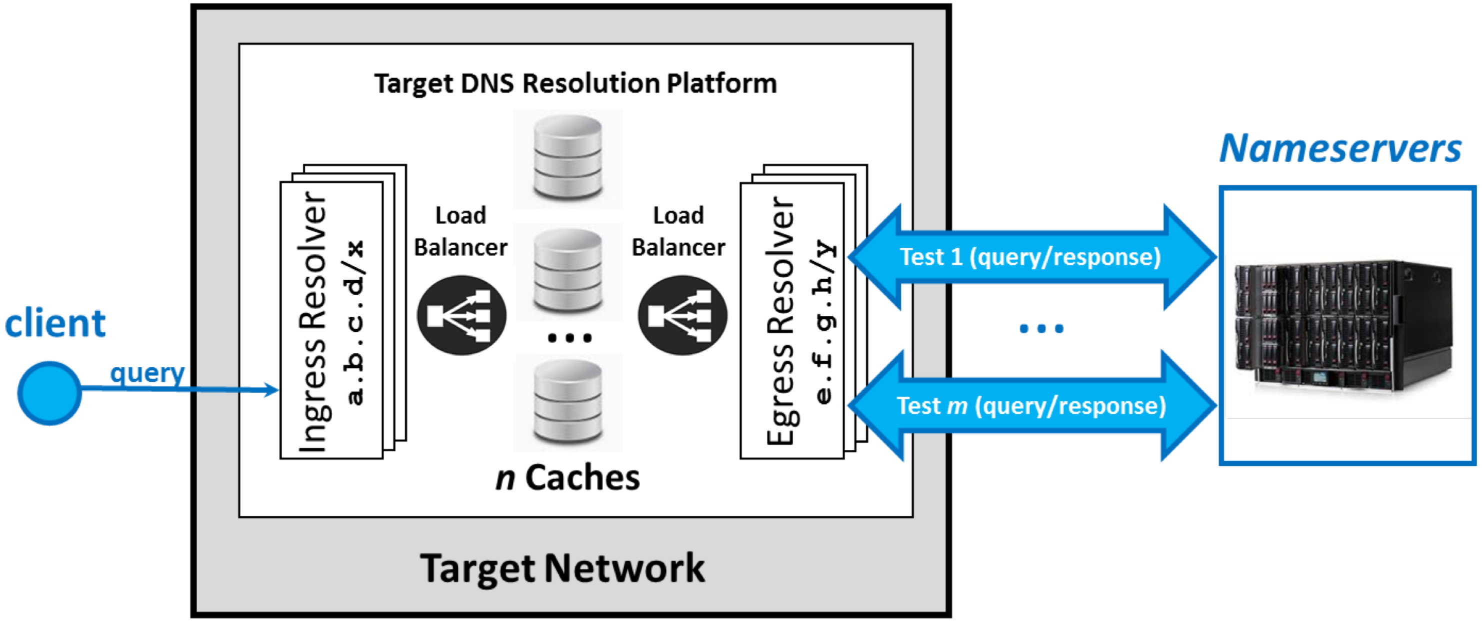
There may be many others yet to be discovered

[KSW17] A. Klein, H. Shulman and M. Waidner, "Internet-wide study of DNS cache injections," *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, Atlanta, GA, 2017

DEEPSEC

DNS RESOLVERS APPLY DIFFERENT POLICIES

Deciding which records to cache and which to overwrite



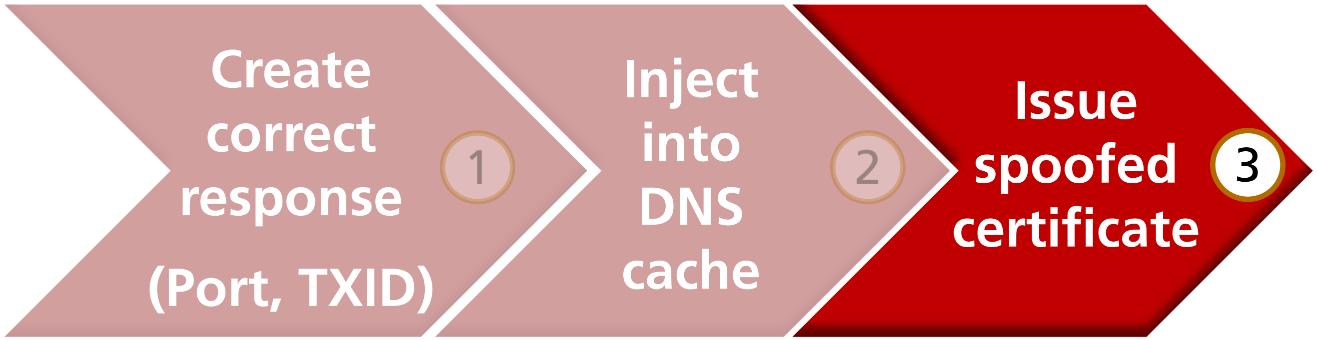
DIFFERENT DNS SERVER ARE VULNERABLE TO DIFFERENT PAYLOADS

	Name	BIND 9.10.2-P2	BIND 9.4.1	Unbound 1.5.4	MaraDNS 3.2.07 Deadwood	PowerDNS 3.7.3	MS DNS 6.1 Win Server'08 R2 6.1.7601)	MS DNS 6.2 Win Server'12 6.2.9200	MS DNS 6.3 Win Server'12 R2 6.3.9600	Google Public DNS	Open DNS	BIND 9.10.2-P2 w/DNSSEC	Nominum Vantio CacheServe v5	Nominum Vantio CacheServe v7
1	ns0	no	yes	yes	no	yes	yes	yes	yes	no	yes	no	no	no
2	ns0-auth	no	yes	yes	no	yes	no	no	no	no	no	no	no	no
3	ns	yes	yes	yes	no	yes	yes	yes	yes	no	yes	no	no	no
4	ns-auth	yes	yes	yes	no	yes	no	no	no	no	no	no	no	no
5	ns2	yes	yes	yes	no	yes	yes	yes	yes	no	yes	no	no	no
6	ns2-auth	yes	yes	yes	no	yes	no	no	no	no	no	no	no	no
7	b4	no	no	no	no	yes	yes	yes	yes	no	yes	no	no	no
8	u1-auth	no	no	yes	no	yes	no	no	no	no	no	no	no	no
9	u3-2	no	no	yes	no	yes	yes	yes	yes	no	yes	no	no	no
10	u3-3	no	no	yes	no	yes	yes	yes	yes	no	no	no	no	no
11	u3-4	yes	yes	yes	no	yes	yes	yes	yes	no	yes	yes	no	no
12	w-dname	yes	yes	no	no	no	no	yes	yes	no	no	yes	no	no
13	w7	no	no	no	no	yes	yes	yes	yes	no	yes	no	no	no
14	w8	yes	yes	no	no	yes	yes	yes	yes	no	yes	no	no	no
15	dname	yes	yes	no	no	no	no	yes	yes	no	no	yes	no	no
16	ak1	yes	yes	no	no	yes	yes	yes	yes	yes	yes	no	no	no
17	w11	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes
18	w11bis	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes

SO WE CAN TRY TO ATTACK OUR VICTIMS BY

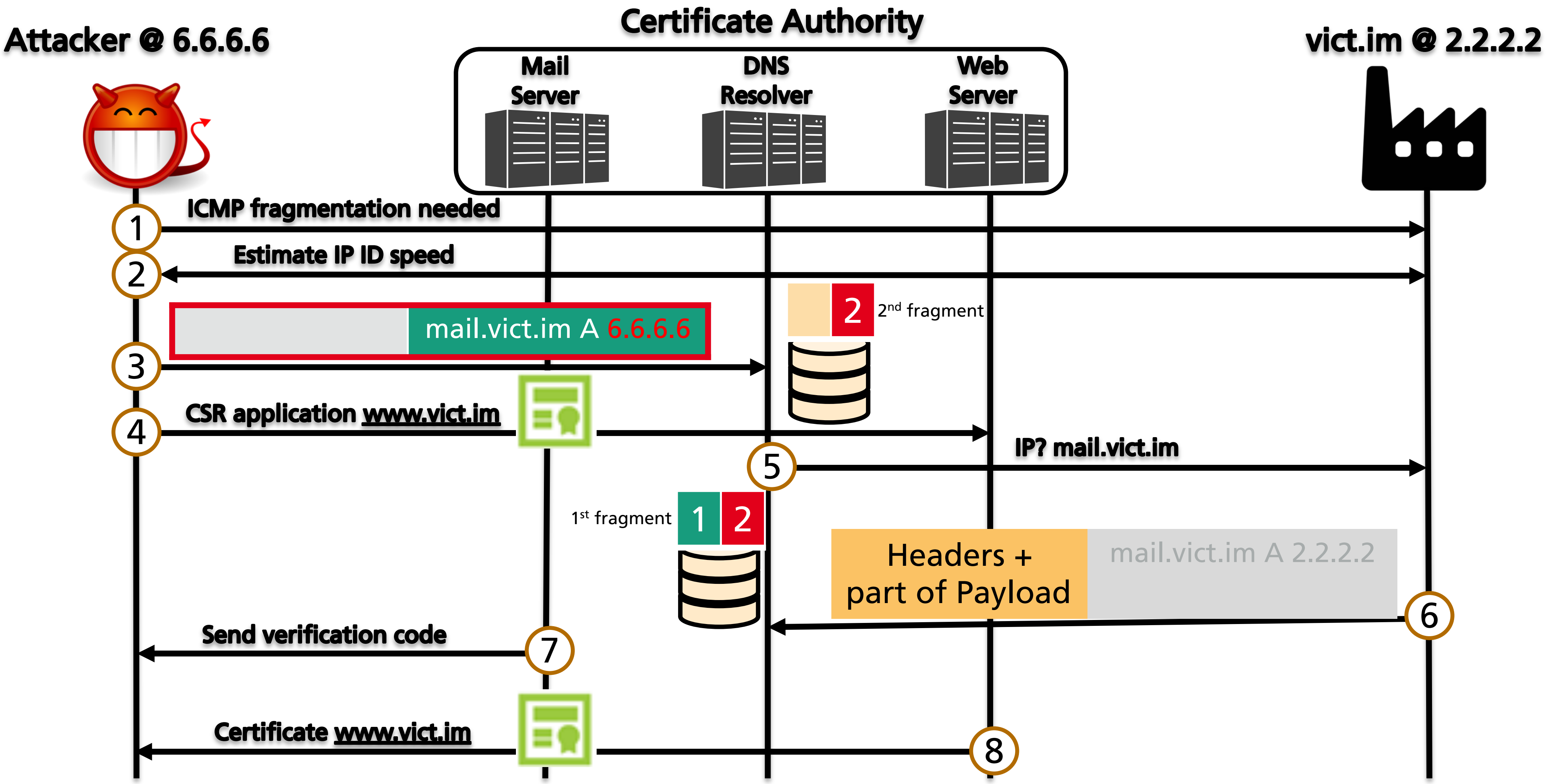
- Fingerprinting DNS server
- Selecting payload
- Poisoning DNS cache with payload
  - e.g. using Fragmentation

AND THE FINAL STEP

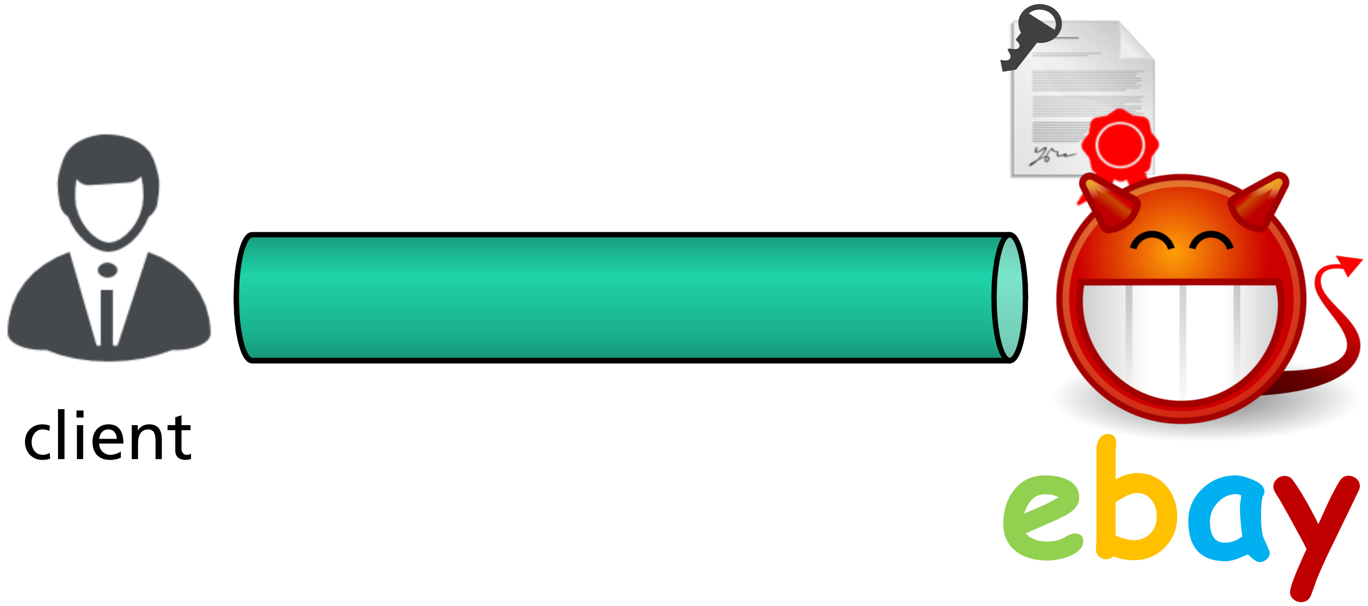


DEEPSEC

PUTTING IT ALL TOGETHER



IMPERSONATION SUCCESSFUL!



Our certificate is signed by a trusted CA.

DEFENCES

SHORT TERM DEFENCES

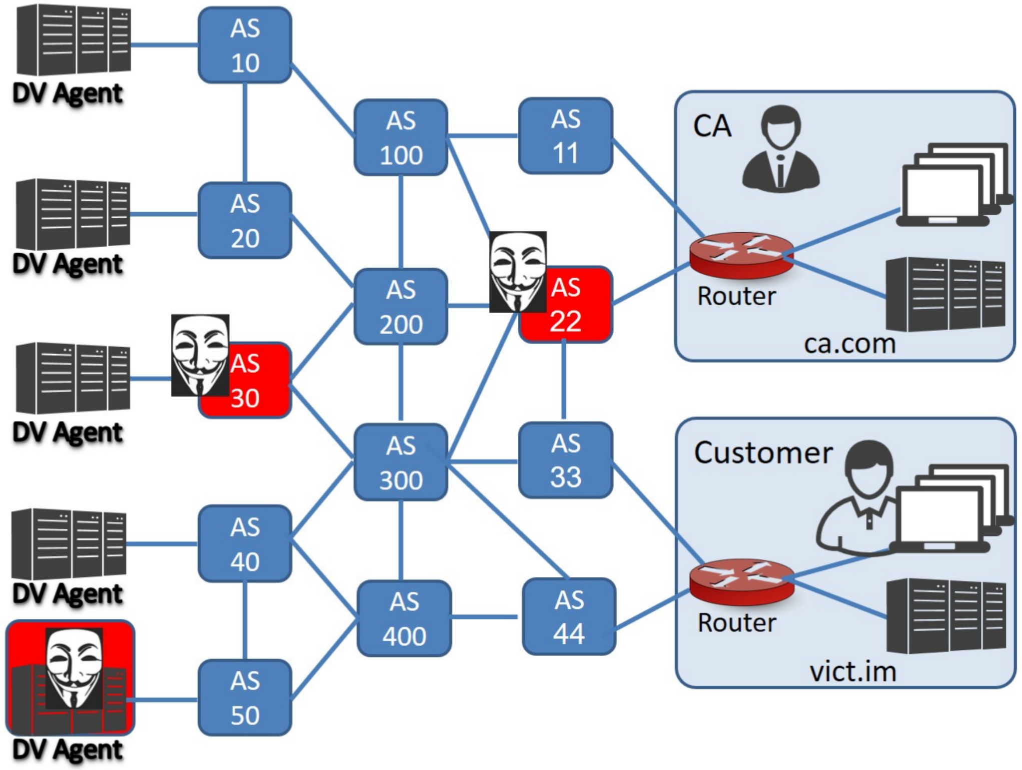
- Disable caching
  - Makes the attack hard but not impossible
- Disable IP fragmentation
  - Will disconnect some networks
- Force DNS over TCP
  - Off-path TCP injections attacks do exist
  - Offers no security against MITM attackers

LONG TERM DEFENCES

- DNS over HTTPS/TLS
  - Securing PKI with PKI?
- DNSSec
  - If fully deployed (proposed in mid-90s)
- Domain Validation++

DEEPSEC

DOMAIN VALIDATION++

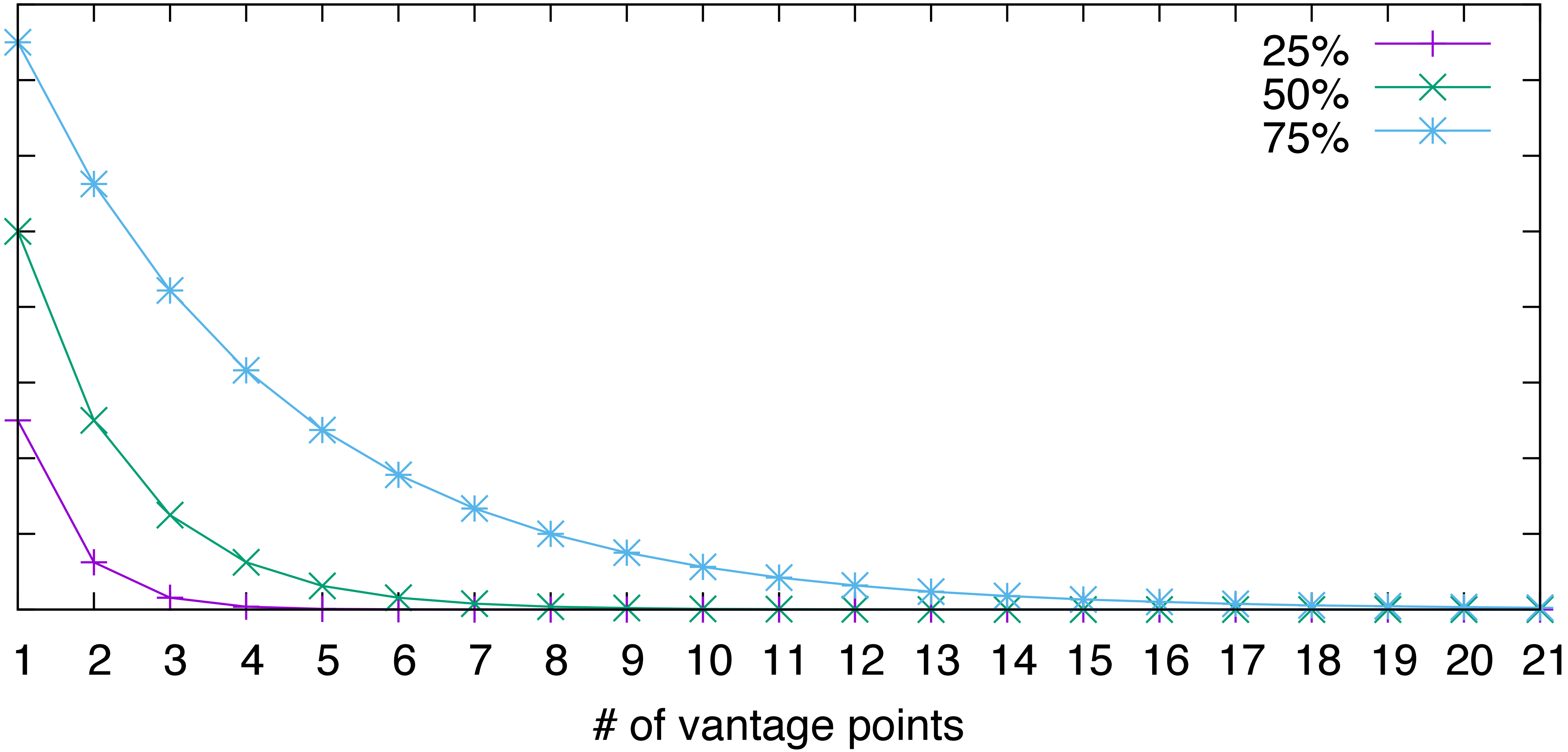


- Drop-in replacement for conventional Domain Validation
- Validation performed from multiple vantage points
- Secures DV even against global MITM attackers
  - even they cannot be everywhere
- Each vantage point has a local resolver
  - Hardened config / Caching disabled
- Uses orchestrator that evaluates voting of DV agents each performing the DNS part
- Communicate via HTTPS (fixed certificates)
- Validation succeeds if majority returns the same response

➢ For more details, visit pki.cad.sit.fraunhofer.de

DEEPSEC

SIMULATION OF ATTACKER'S SUCCESS



ADVICES

- Disable caching for DV resolvers
- Adopt DV++
- Harden DNS resolvers
- Limit fragmentation to reasonable values (e.g. MTU >= 1280)
- **Deploy DNSSec**

DNSSEC DEPLOYMENT IS CHALLENGING…

- 1/3 signed-domains cannot be validated
- 35% domains signed with shared keys
- 90% domains signed with weak keys (≤1024 bits)
- 70% signed domains do not refresh keys

CONCLUSION

CONCLUSION

- Deployment of security in the Internet is challenging
  - Similar problems in many systems

- How to make local enhancement of security work
  - Understand the landscape
  - See beyond the horizon
  - Security at partial adoption
  - Give incentives to adapt new technologies

THANK YOU FOR YOUR ATTENTION

QUESTIONS?

HTTPS://PKI.CAD.SIT.FRAUNHOFER.DE