CS306: Introduction to IT Security

Fall 2018

Lecture 11: Topics

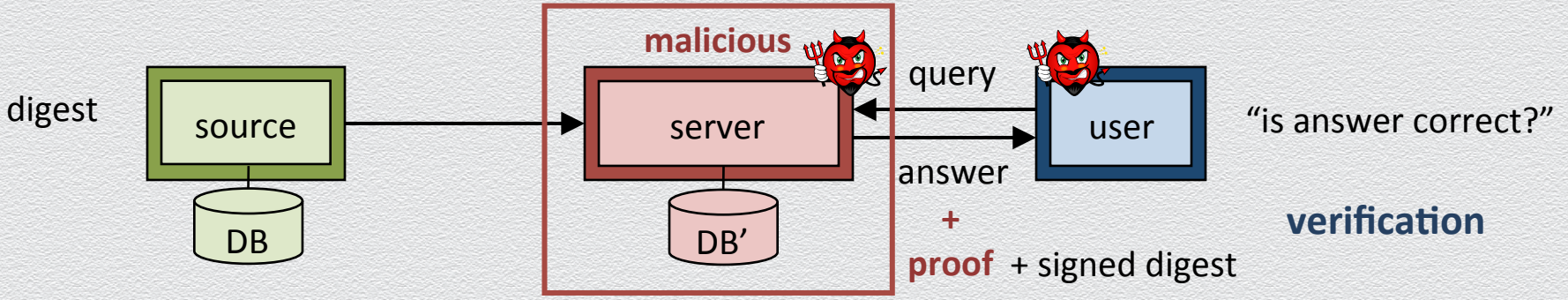
Instructor: **Nikos Triandopoulos**

November 20, 2018

STEVENS
INSTITUTE *of* TECHNOLOGY
THE INNOVATION UNIVERSITY®

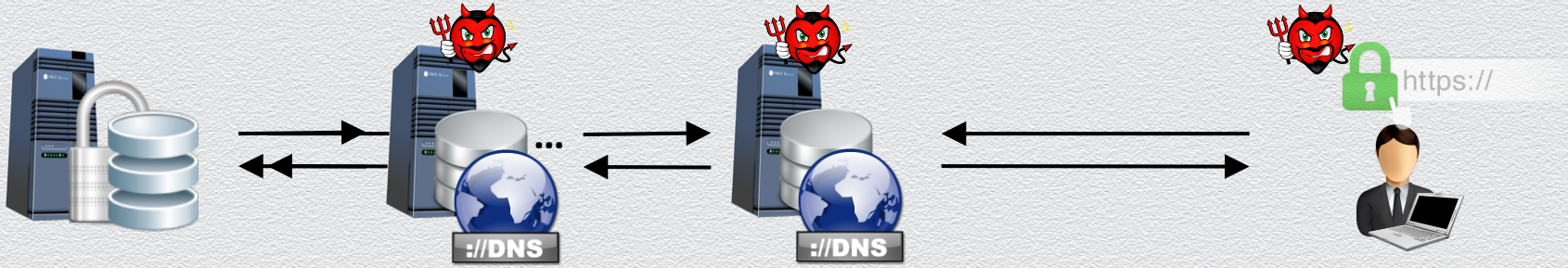
From NSEC3 to NSEC5

A critical asset prone to attacks



integrity

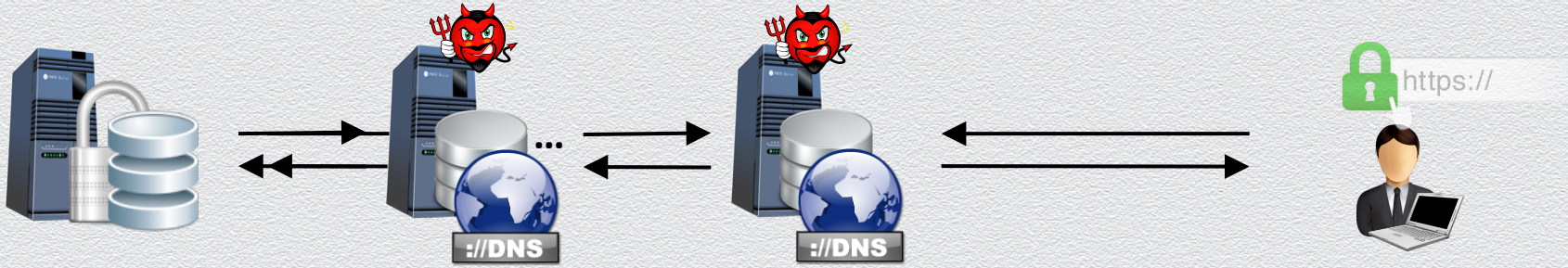
availability / confidentiality



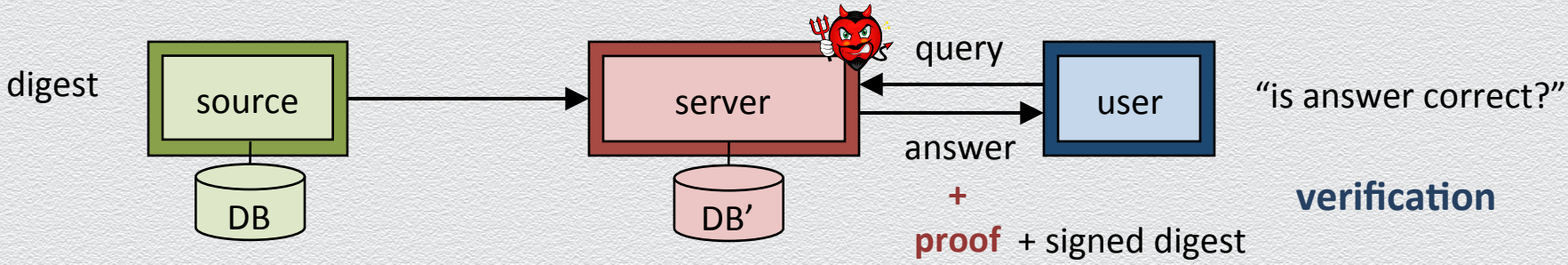
DNSSEC (& NSEC)

Security extension of DNS protocol to protect integrity of DNS data

- correct resolution, origin authentication, authenticated denial of existence
- specifications made by Internet Engineering Task Force (IETF) via RFCs
  - an RFC (request for comments) is a suggested solution under peer review
- challenges: backward-compatible, simplicity, confidentiality, who signs
  - NSEC (next secure record): extension that provides proofs of denial of existence



DNSSEC & NSEC: core idea

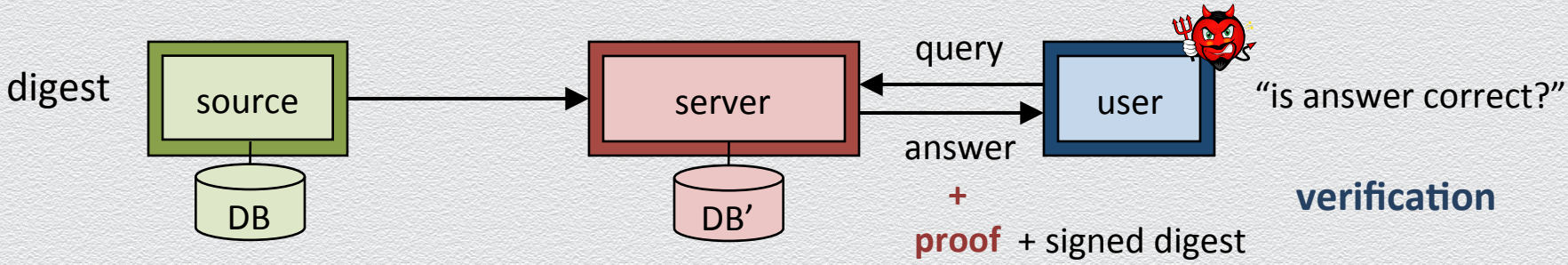


DNSSEC protocol: each DNS entry is pre-signed by primary name server

NSEC protocol:

- domain names are lexicographically ordered and then each pair of neighboring existing domain names is pre-signed by the primary name server
- non-existing names, e.g., aWa2j3netflix.com are proved by providing this pair "containing" missed query name, e.g., <awa.com, awb.com>

From NSEC to NSEC3



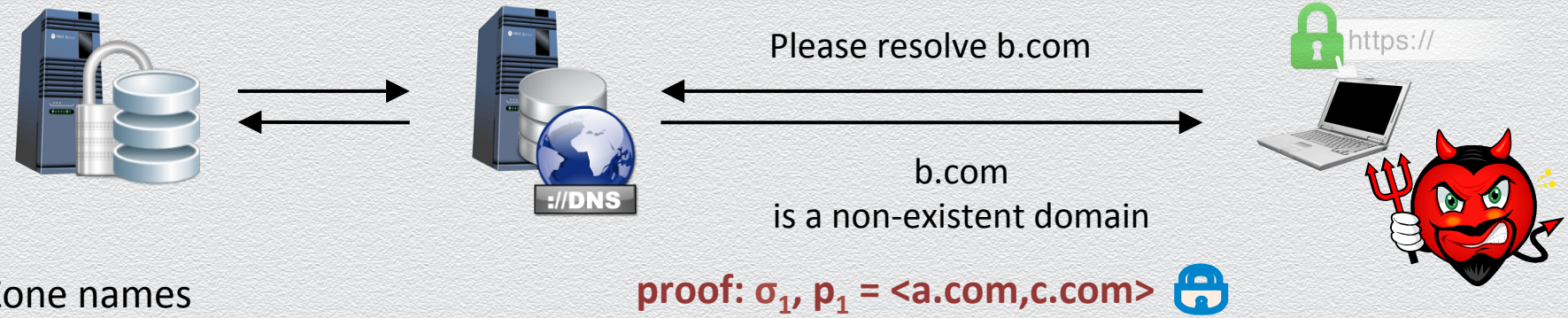
Vulnerability in NSEC protocol

confidentiality

- proofs of non-existing names reveal information about domain names
- an attacker can simply as a "querier" learns information about the network structure of a target organization!

DNS zone enumeration attack

The attacker acts as the user in order to learn the domain names of an entire zone



Zone names

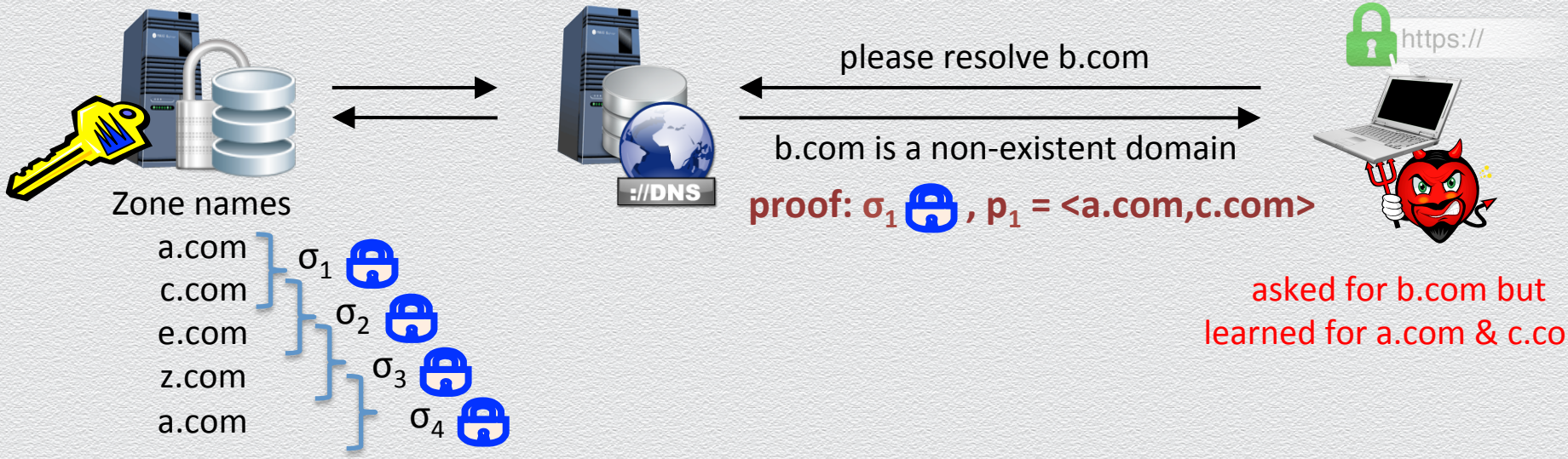
- a.com
- c.com
- e.com
- z.com

proof: σ_1, p_1 = <a.com,c.com>

confidentiality

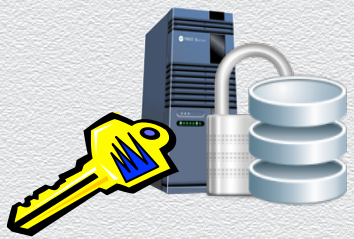
DNS zone enumeration attacks

The attacker acts as the user in order to learn the domain names of an entire zone



- exposes private device names (e.g., IoT devices which can be toehold for other attacks)
- reveal registrant data (that many registries may have legal obligations to protect)

NSEC3



please resolve b.com

b.com is a non-existent domain

proof: σ_3 , p_3 = <dde45,zrit5>

Zone names

a.com
c.com
e.com
z.com

hash h →

a1bb5
23ced
zrit5
dde45

sort →

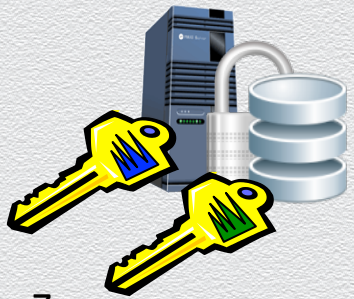
23ced
a1bb5
dde45
zrit5
23ced

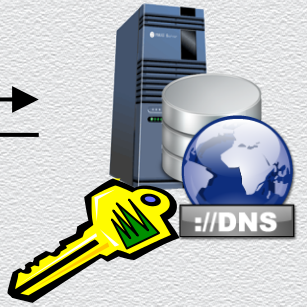
σ_1
σ_2
σ_3
σ_4

asked for b.com but
learned h(a.com) & h(c.com)

h(b.com) = ntwo4
e.g., h is SHA-256

NSEC5





please resolve b.com

b.com is a non-existent domain

proof: σ_3 , p_3 = <dde45,zrit5>

Zone names

a.com
c.com
e.com
z.com

hash h’ →

a1bb5
23ced
zrit5
dde45

sort →

23ced σ_1
a1bb5 σ_2
dde45 σ_3
zrit5 σ_4
23ced

asked for b.com but
learned h’(a.com) & h’(c.com)

h’(b.com) = ntwo4
h: as in NSEC3
f: “message transformation” hash

h’(x) = h(RSA-Sign(, f(x)))

Certificate Transparency

How to secure your website browsing

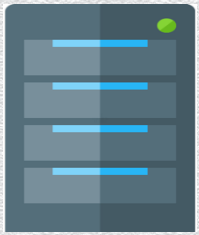
visa.com

Please sign VISA's PK!

1

2

cert = $\mathbf{Sig}_{CA}$(visa.com, PK)



Certificate Authority (CA)

3

PK, cert

4

$\mathbf{Enc}_{PK}$(passwd)

… and how things can fail!

[A] impersonate VISA

attacker's fake visa.com

Please sign VISA's **PK'**!

1

2

cert' = **Sig**$_{CA}$(visa.com, **PK'**)

3

PK', cert'

4

Enc$_{PK'}$(passwd)

[B] compromise CA

Certificate Authority (CA)

VISA might never find out the impersonation attack

Not just an academic problem...

Google Security Blog

The latest news and insights from Google on security and safety on the Internet

Gmail account security in Iran

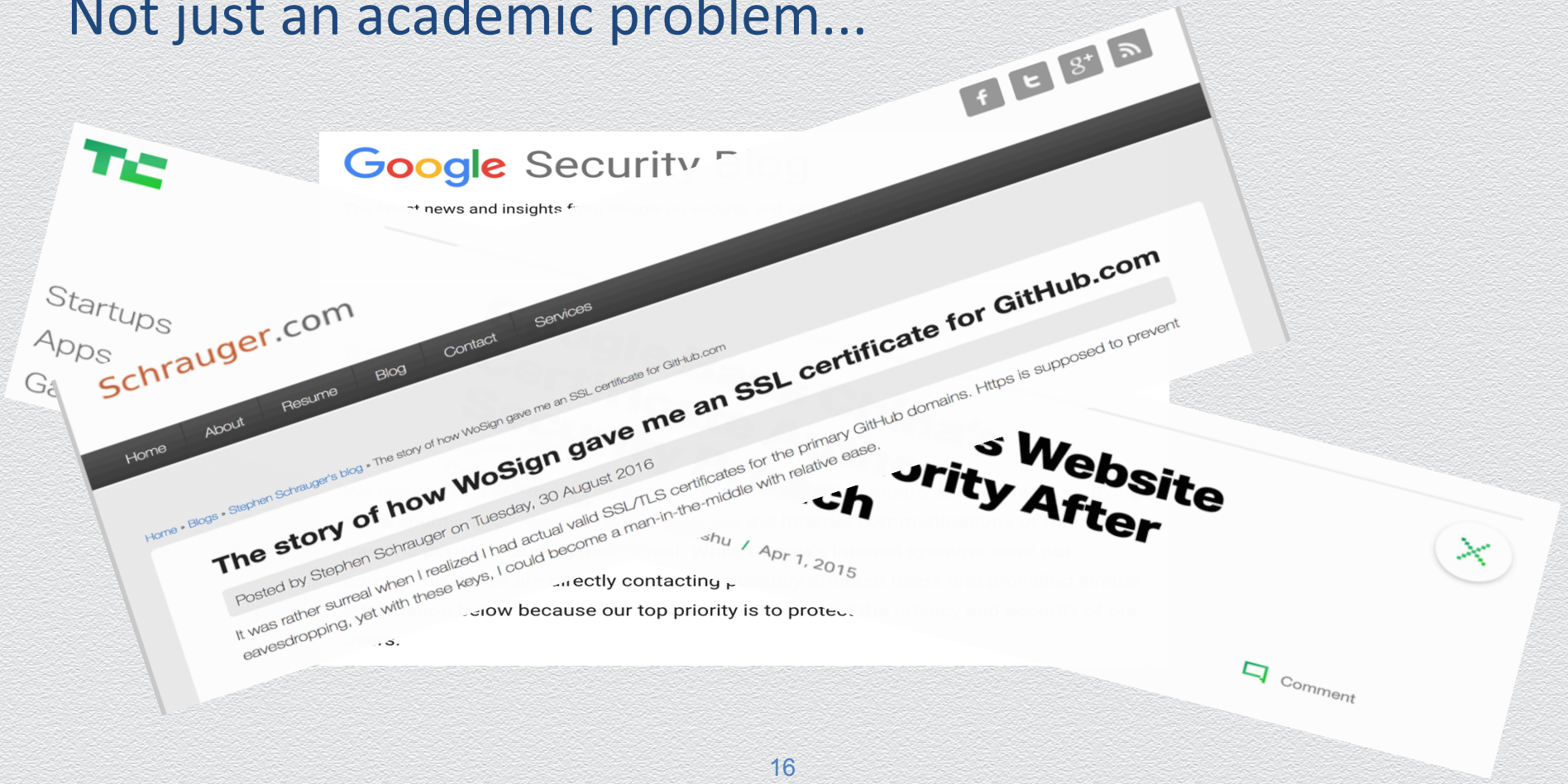
September 8, 2011

Posted by Eric Grosse, VP Security Engineering

We learned last week that the compromise of a Dutch company involved with verifying the authenticity of websites could have put the Internet communications of many Iranians at risk, including their Gmail. While Google's internal systems were not compromised, we are directly contacting possibly affected users and providing similar information below because our top priority is to protect the privacy and security of our users.

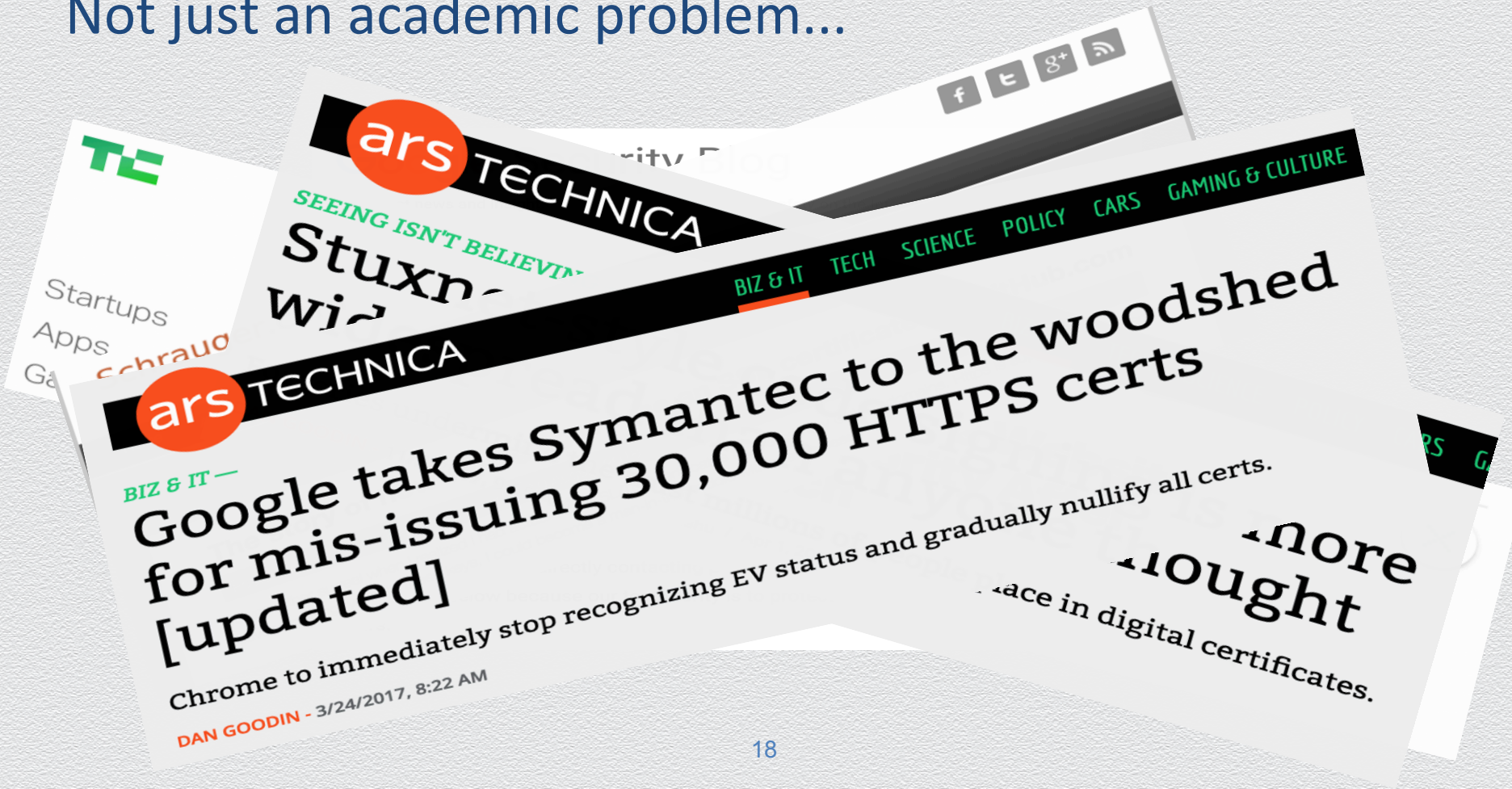
Not just an academic problem...

Not just an academic problem...



Not just an academic problem...

Not just an academic problem...



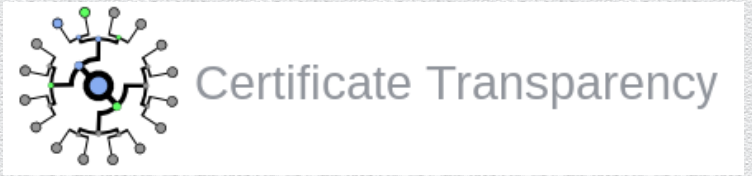
How to deal with rogue certs or compromised CAs?

Impossible to prevent impersonation, but can we **detect** it *after* it happens?

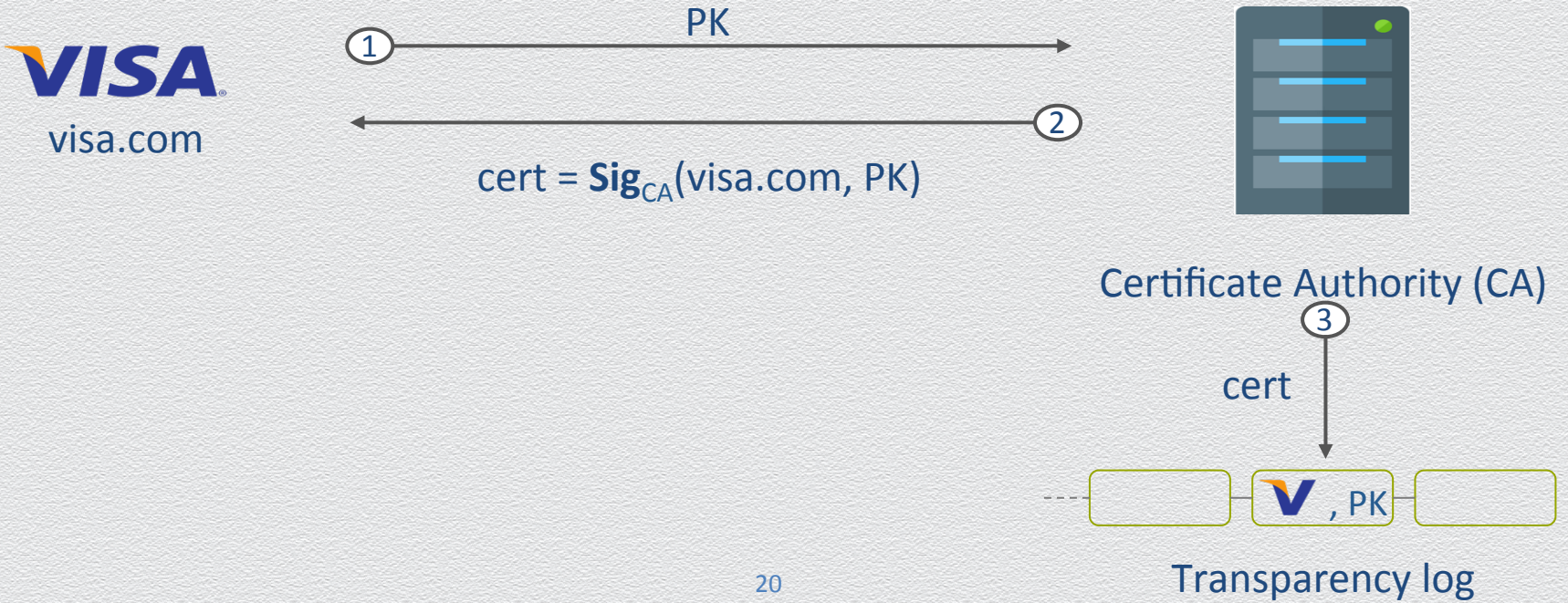
Why should we?

- punish bad CAs
- victims learn they were impersonated
- deters attacks

Google proposed & deployed **a solution (CT)**



Certificate Transparency (CT) to the rescue



Certificate Transparency: Key idea

Consequence: Fake PKs must be pushed in the log

Certificate Authority (CA)

visa.com

4

PK, cert

5

$\mathbf{Enc}_{\mathbf{PK}}$(passwd)

3

cert'

Requirement: Accept cert only if it's in the log!

Transparency log

Certificate Transparency: Security properties

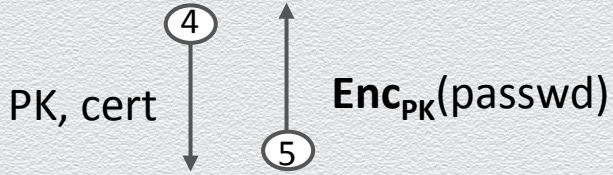
CT "specs"

Transparency:
Once cert in CT
(1) it cannot be deleted &
(2) it can be efficiently discovered

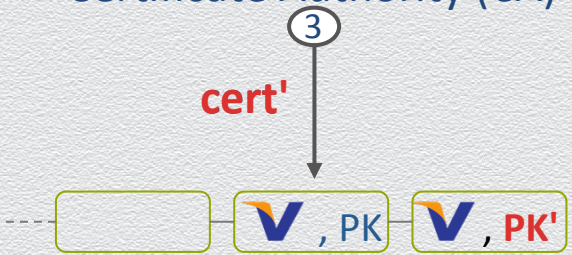
Non-equivocation:
Everybody "sees" the same log!

Consequence: Fake PK for VISA is discovered by VISA in the log

visa.com

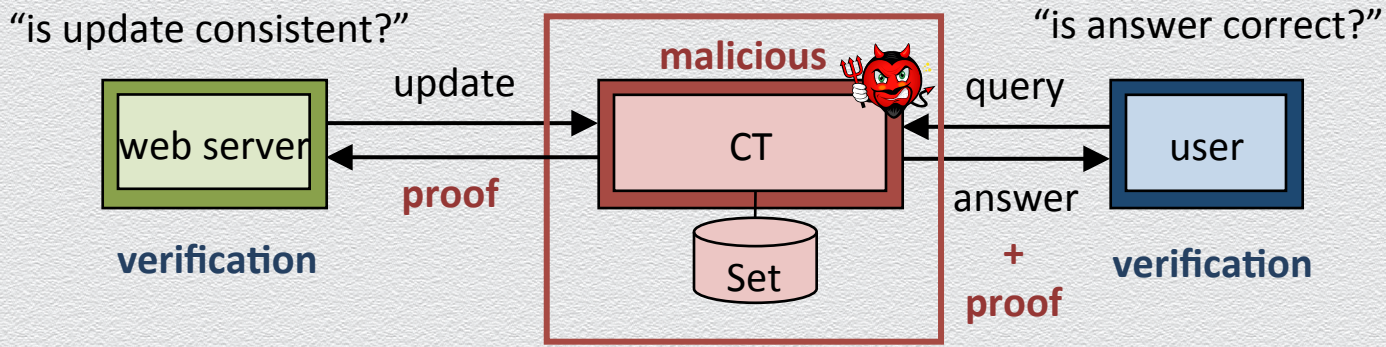


Certificate Authority (CA)



Transparency log

Certificate Transparency: Related to DB-as-a-service model



Computational proofs

- subject to set commitments or **digests**, CT provides two type of proofs
  - look-up proof – returned answer is correct
  - append-only proofs – digests are updated consistently w.r.t. set inclusion

Certificate Transparency: Example