Internet Security II

50.520 Systems Security

Paweł Szałachowski

Security Issues

- TLS PKI
  - Transparency
  - Downgrading attacks
    - Key pinning
- Mail security
  - Spam
  - Encryption and Authentication

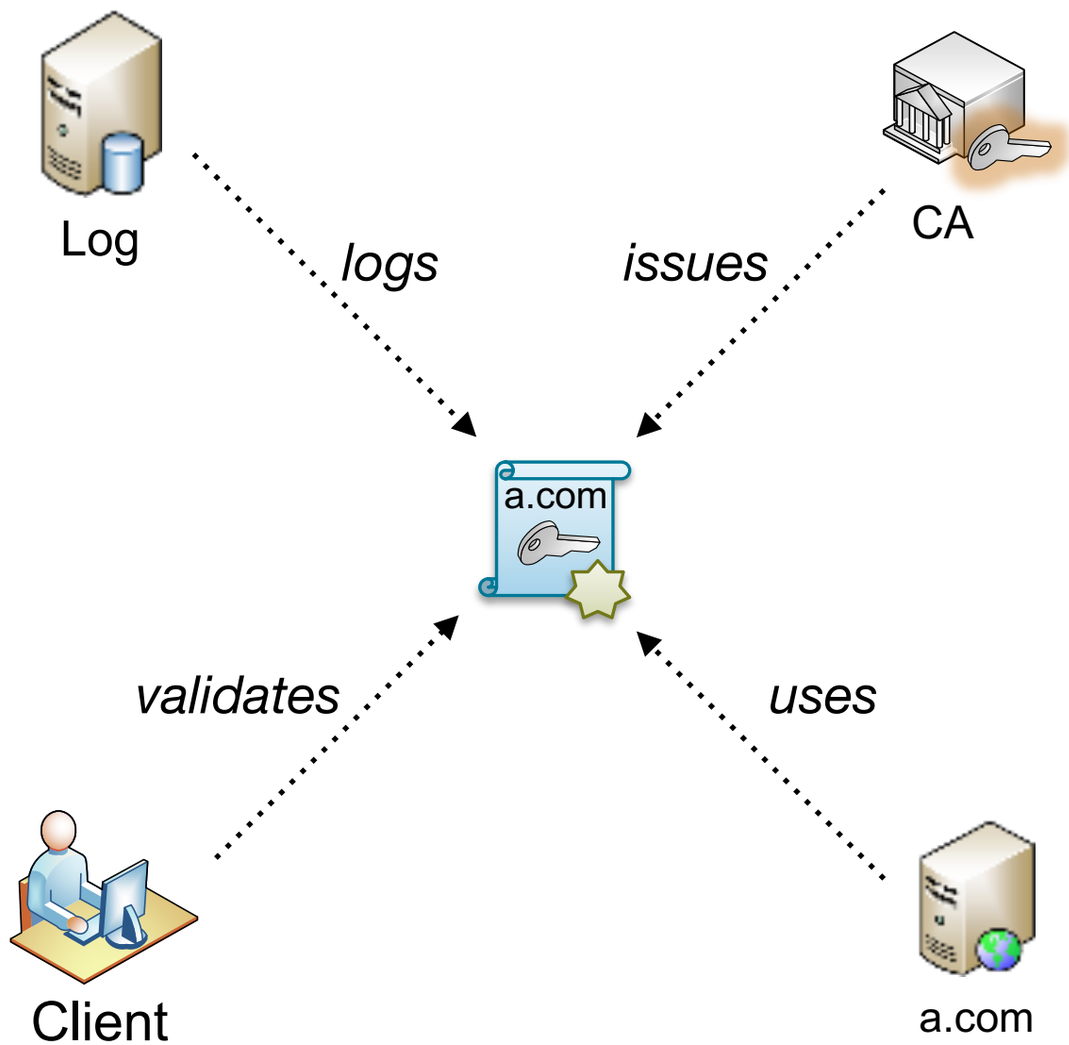
Transparency in TLS PKI

Transparency in PKI

- Targeted attacks
  - Comodo, DigiNotar, …
- How to protect from them?
  - DANE, CAA, …
- Protection vs detection
- How to detect attacks?
  - Malicious certificate can be used only once…
  - Can we detect such an attack?

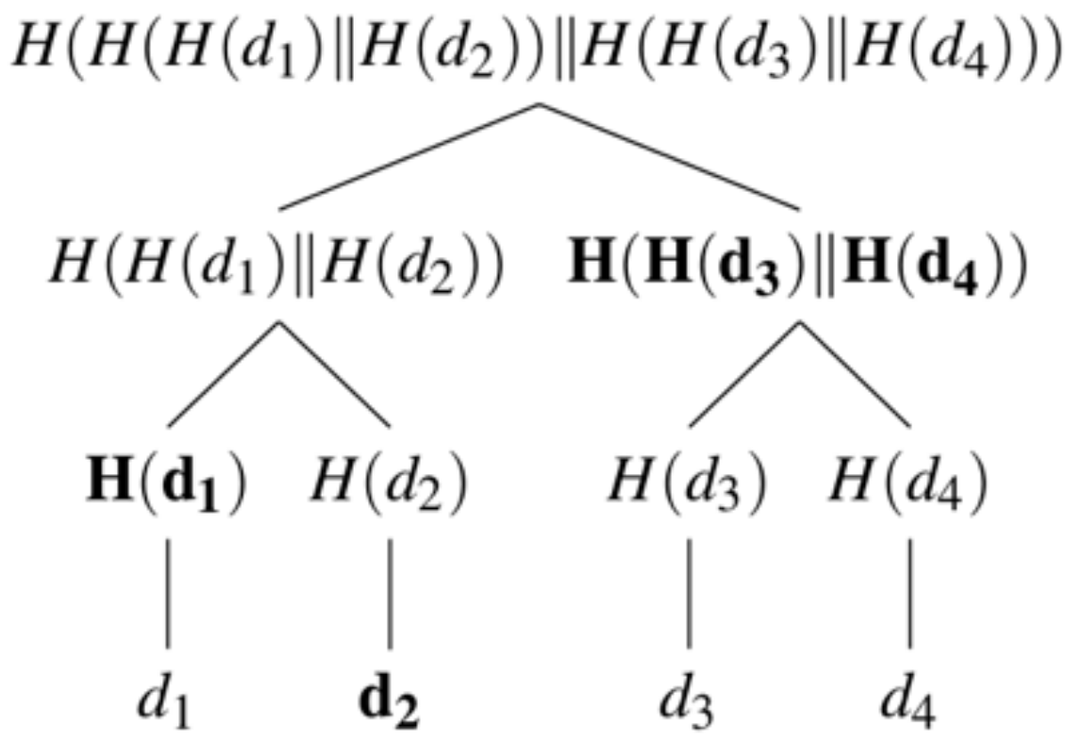
Certificate Transparency (CT)

- Google's proposal to increase transparency in TLS PKI
- New party: Log
  - (semi)trusted
  - Monitors CAs
  - Logs all certificates



Merkle Hash Trees

- Cryptographic hash function H(.)
- Hash tree: every non-leaf node is labelled with the hash of the labels of its child nodes.
- Efficient Proofs
  - Presence
  - Absence (if sorted)
  - Extension (if append-only)



Certificate Log

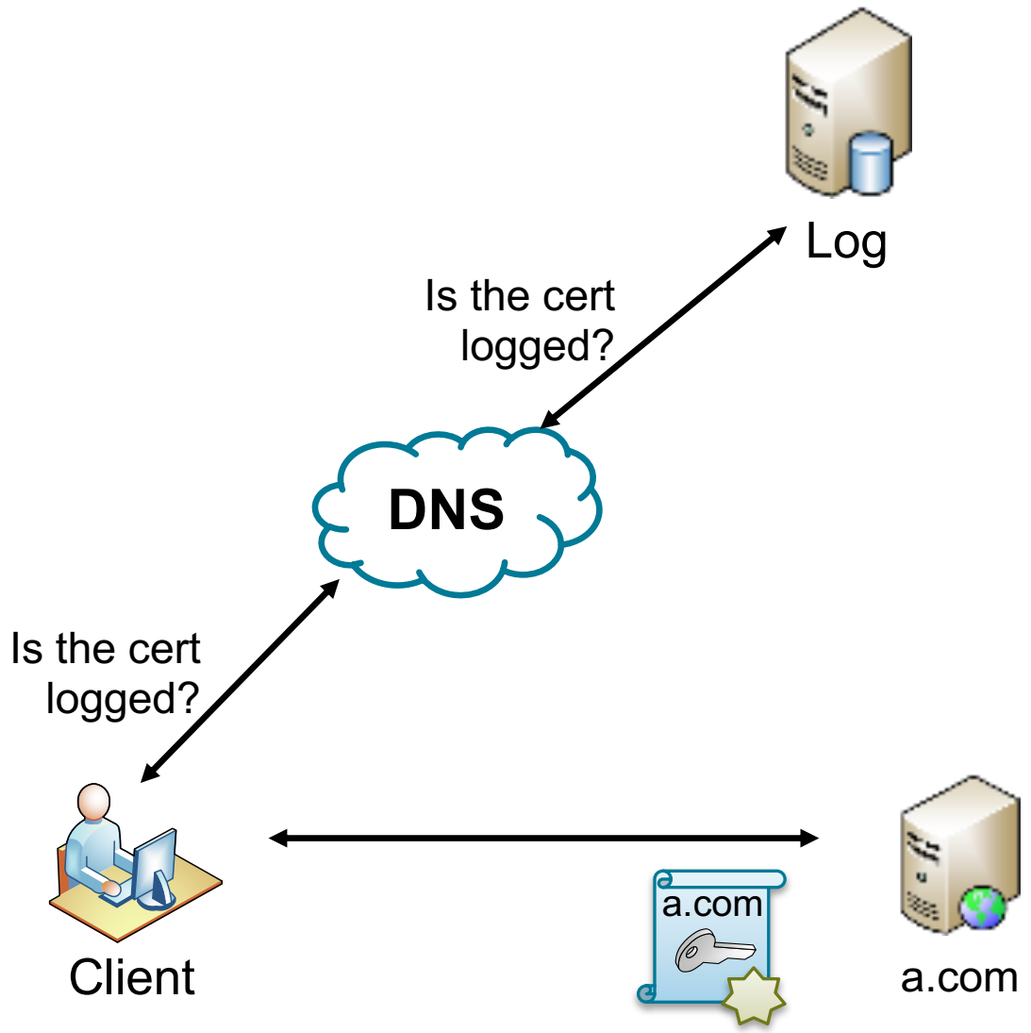
- For every certificate, Log issues a Signed Certificate Timestamp (SCT)
    - Promise to append the certificate to the log
    - Certificates are sent to clients along with the corresponding SCTs
- Every update it appends new certificates to an append-only tree and generates a Signed Tree Head (STH)
    - STH is a root of the tree
    - Anyone can download it
- Anyone can audit that a certificate is logged
- Possible to prove that log is append-only

Is promise met?

- A client gets an SCT (promise)
  - How to make sure that the associated certificate is in the log?
- The client has to talk to the log
  - Can be done after establishing connection (non-blocking)
- Any problems?

Audit log via DNS

- Contact log servers
  - To check whether certificates are indeed logged
    - Use DNS for anonymity
- Under development
- New proposals are welcome



Deployment

- Client accepts certificates with their SCTs
- What if certificates is sent without its SCT?
  - Legacy certificates (issued before CT introduction)
  - …
- CT logs to work effectively have to store all certificates
  - Effectively, CAs are *forced* to submit any new certificate to a CT log

Certificate Transparency

- Issues
  - What if logs are malicious?
    - How to monitor logs effectively?
  - Availability of logs is critical
- Ongoing deployment and standardization
  - Mandatory for EV certificates

Downgrade Attacks

Downgrading Attacks

- How to deploy a new security protocol's extension?
  - Design, implement, deploy
    - Usually, designs are backward-compatible
      - add new headers, implement user-agents, deploy, …
- What if a MITM adversary acts like there is no new security extension deployed
  - Dropping headers, block connections to secure ports, …
  - Backward-compatibility helps with such attacks

HTTP -> HTTPS

- How servers can enforce HTTPS?
  - Clients type domain.com and browsers prepend http://
  - Server-side enforcement
    - Redirect all traffic to https://
    - Is that secure?

HTTP -> HTTPS

- MITM adversary can drop redirections and impersonate server
  - http connection between client-adversary
  - https connection between adversary-server
- How to prevent that?

Strict Transport Security (HSTS)

- Server's side protection against downgrade attacks
- HTTPS header

```
Strict-Transport-Security: max-age=<# of seconds>
```

  - Can include subdomains (`includeSubDomains`)

- TOFU model (i.e., initial request)
- Browser expects that for `<# of seconds>` the domain will be using HTTPS
- Deployed and getting popular
- Adversary with a malicious certificate can block http-only website

HTTP Public Key Pinning (HPKP)

- Websites can pin their (or their CAs) keys
- HTTPS header
  - `Public-Key-Pins` or `Public-Key-Pins-Report-Only`
- TOFU model
- Clients saves policies and will expect certificates identified by the pins
- Risky misconfigurations
  - Often back-up keys are pinned
- Abandoned due to operational issues

```
Public-Key-Pins:
  pin-sha256="cUPcTAZWKaASuYWhhneDttWpY3oBAkE3h2+soZS7sWs=";
  pin-sha256="M8HztCzM3elUxkcjR2S5P4hhyBNf6lHkmjAHKhpGPWE=";
  max-age=5184000; includeSubDomains;
  report-uri="https://www.example.org/hpkp-report"
```

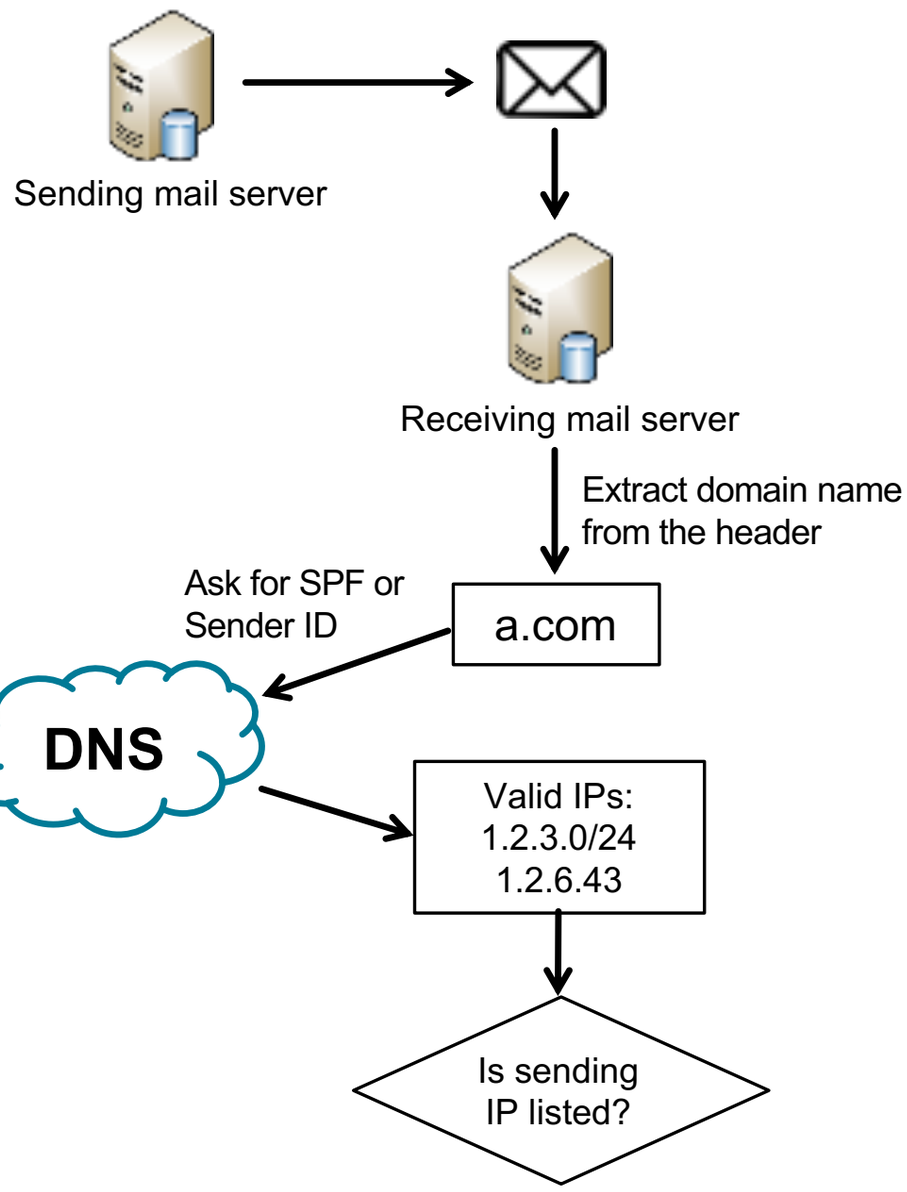
Spam&Email Security

Email

- SMTP protocol
  - No security properties provided
  - No spam prevention
- Mail servers/exchangers
  - Server lookup via DNS (MX records)
- No end-to-end protection

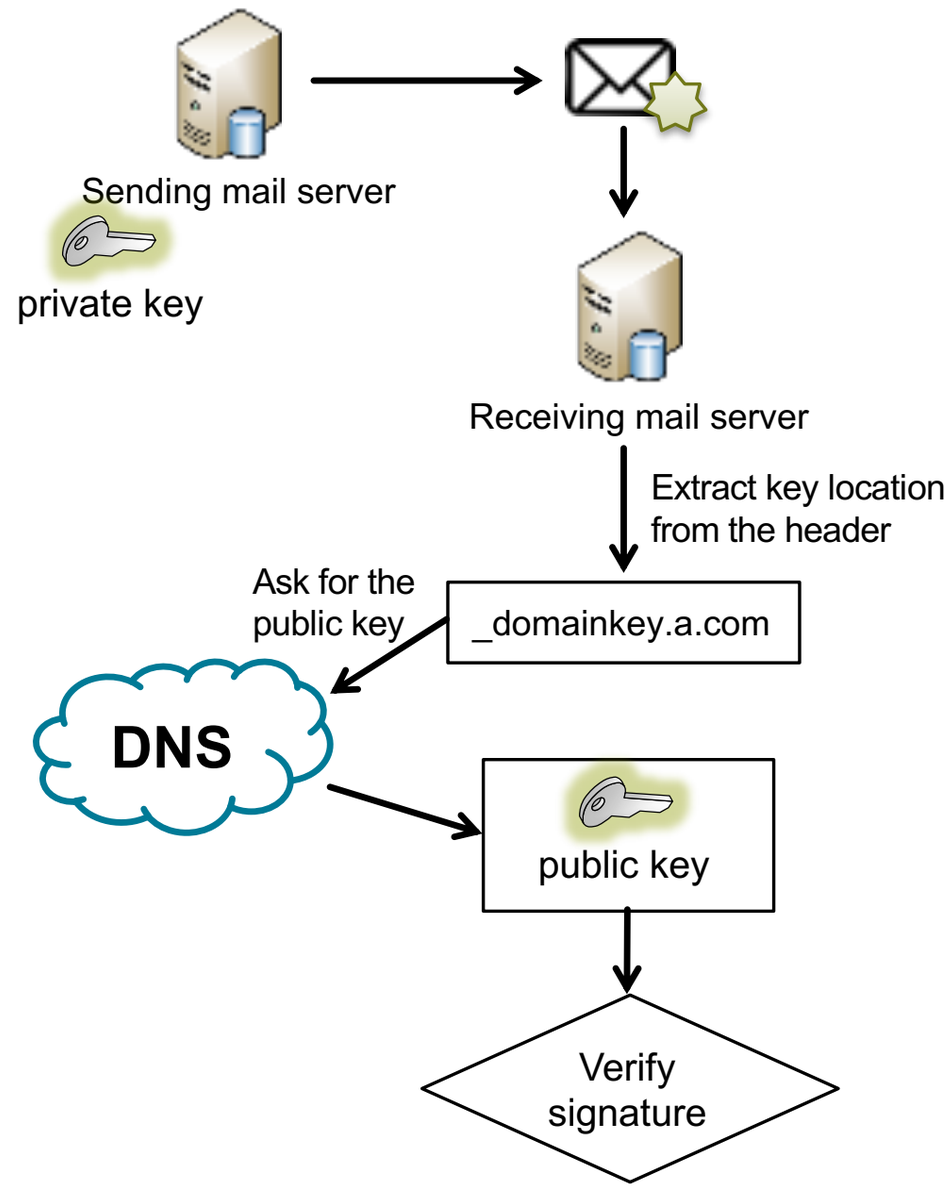
Sender Policy Framework (SPF) and Sender ID

- Anti-spoofing
  - Domains can authorize hosts to originate emails for them
- SPF (RFC7208)
  - Parses the `From` address field
- Sender ID (RFC4405)
  - Parses other fields (presented to users)
- TXT and SPF records are used



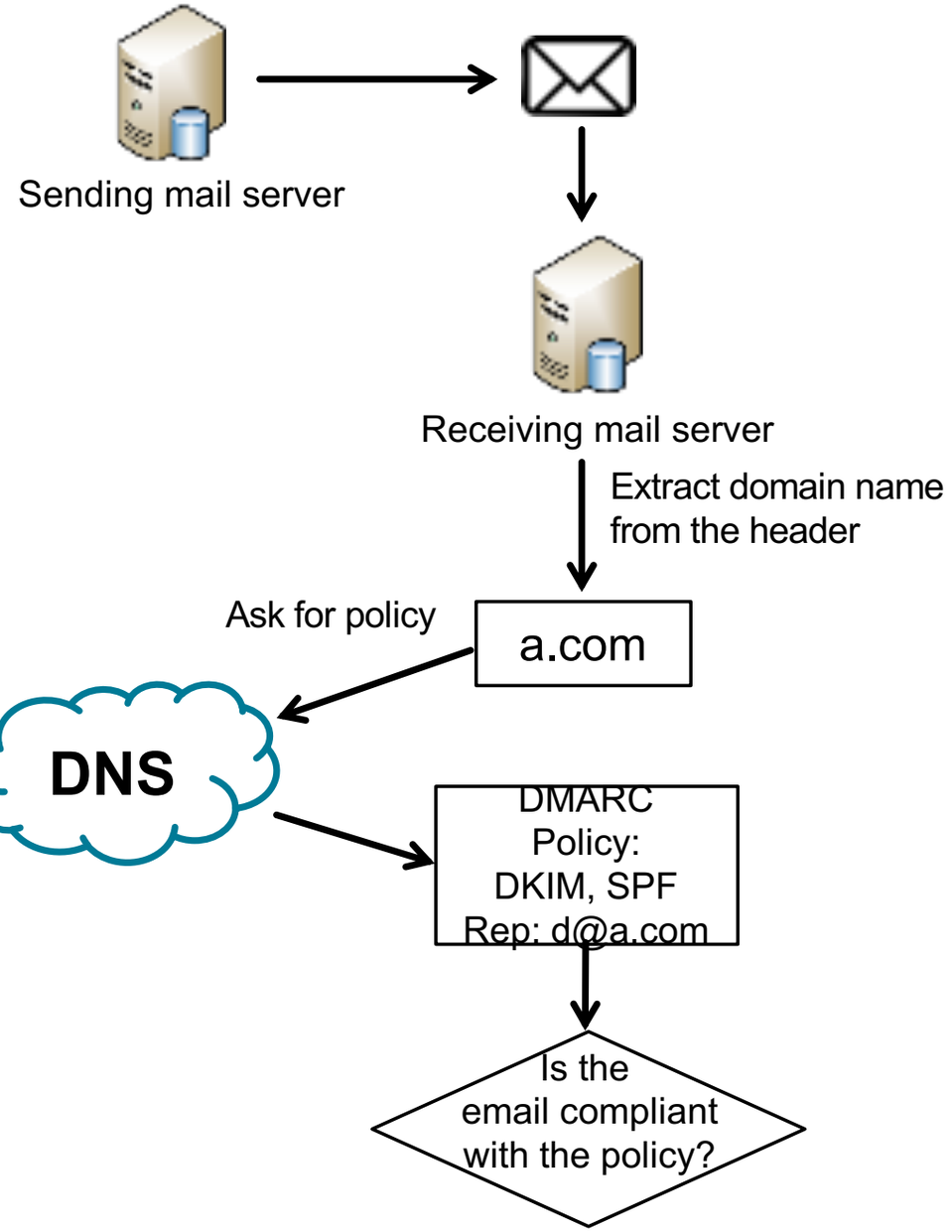
DomainKeys Identified Mail (DKIM)

- RFC6376
- Mail authentication
  - Signing emails in transit (i.e., server-to-server protection)
  - DNS for public key lookup
- DKIM headers can be simply dropped by an adversary
- TXT records are used



Domain-based Message Authentication, Reporting and Conformance (DMARC)

- RFC7489
- Domain-level policies for email management
  - What to do when SPF or DKIM fail (or are not present)?
  - Reporting
- TXT records are used



Other solutions

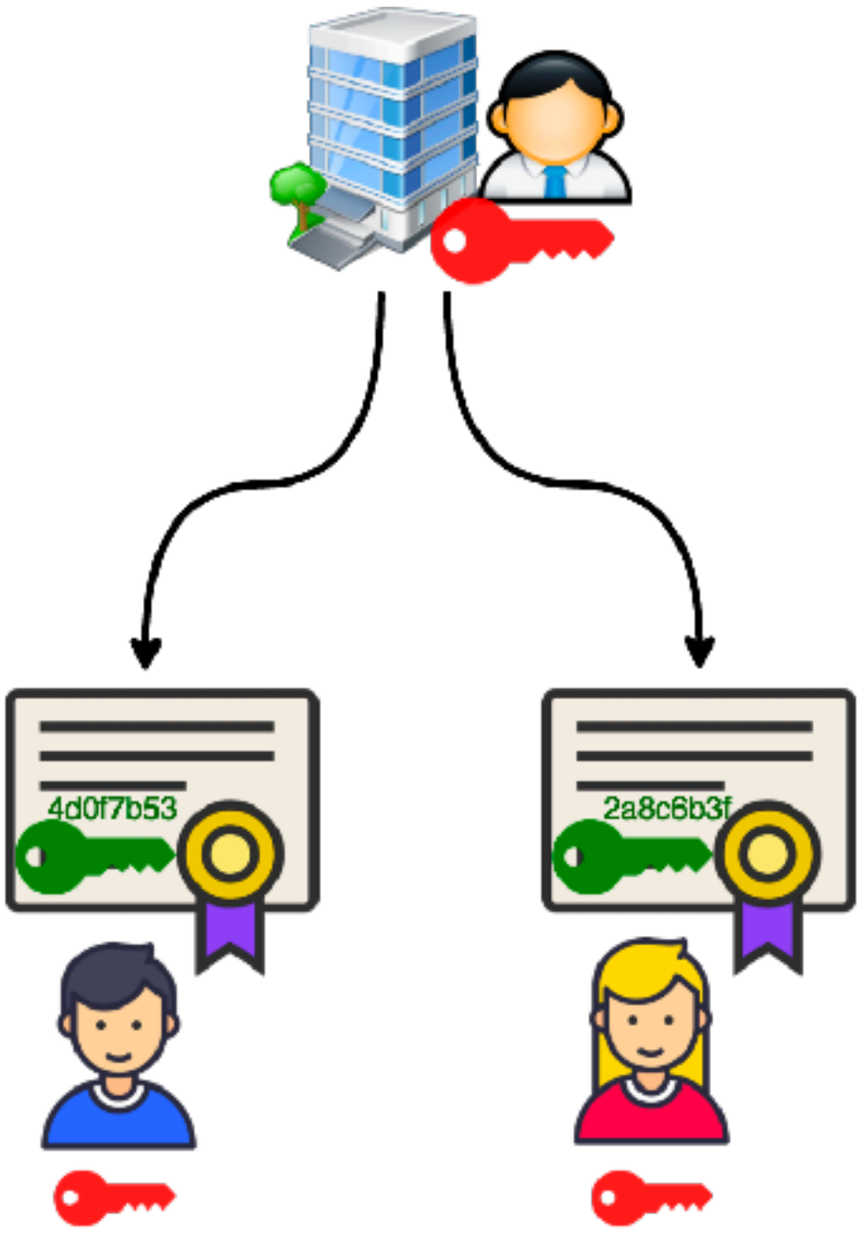
- Filters
- Proof-of-Work (HashCash)
- Detecting botnets (Honeypots)
- …

Pretty Good Privacy (PGP)

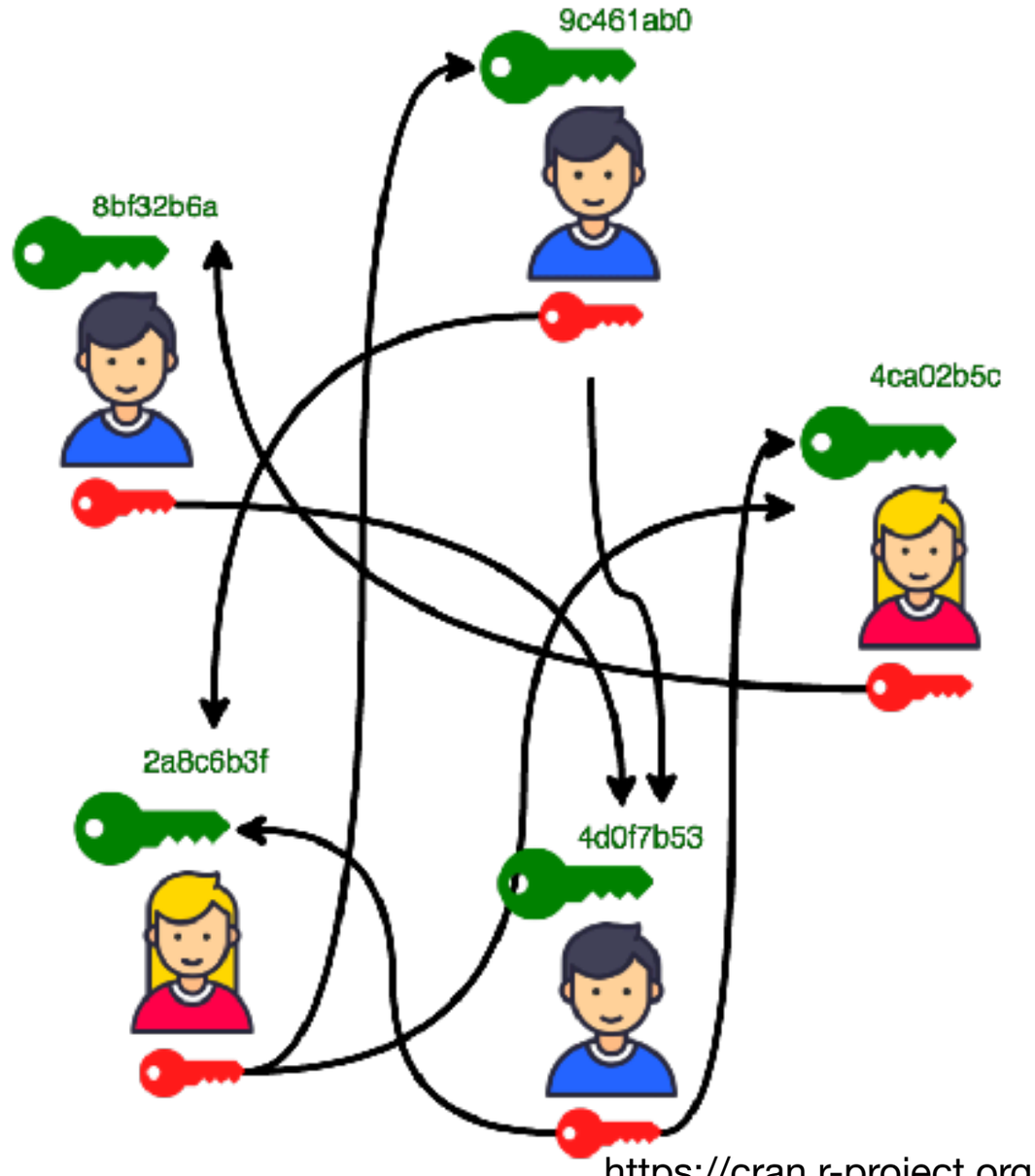
- End-to-end email protection
- Web-of-Trust
    - No-authority trust model
    - Special algorithm to estimate trust
- Key distribution
    - Face2Face
    - Key servers
    - Friends/Emails

PGP Web of Trust

CERTIFICATE AUTHORITY

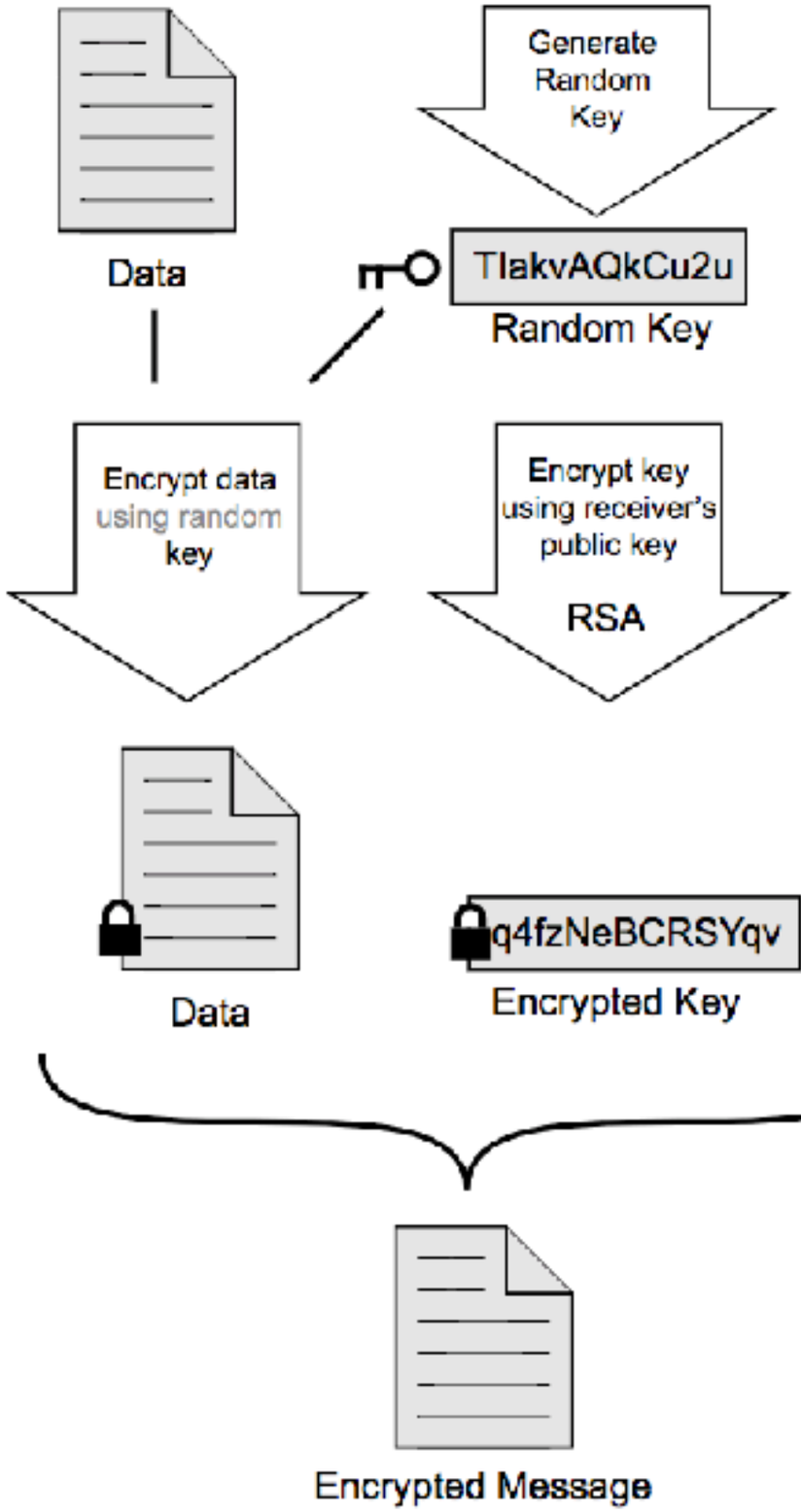


PGP / WEB OF TRUST

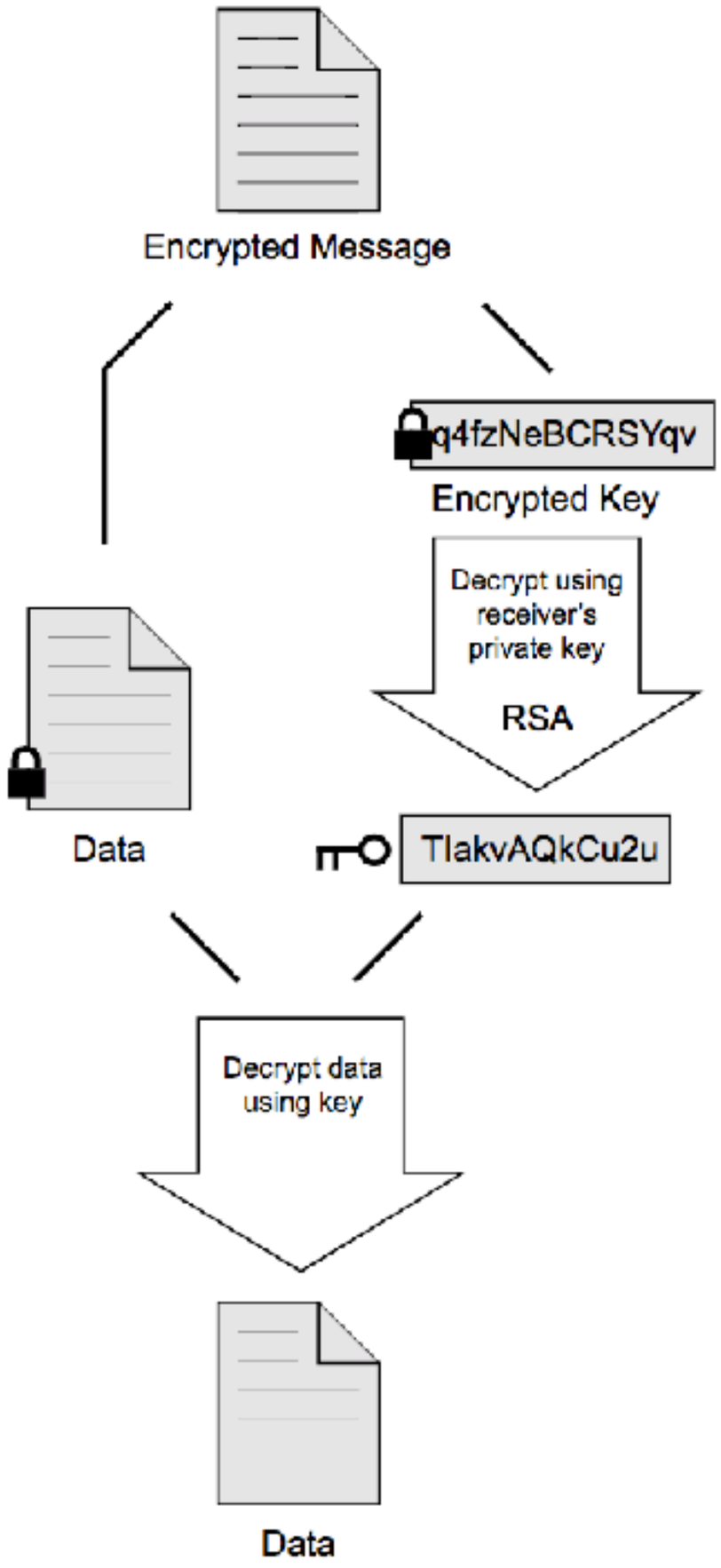


https://cran.r-project.org/

Encrypt



Decrypt



PGP

- Not widely deployed
- Many mail servers modify emails
  - Modify headers, add banners, change content, …
- Usability issues
  - Complex
  - Web-of-trust is hard to use

Questions?