Current challenges for the OpenPGP keyserver network

Is there a way forward?

Gunnar Wolf

DebConf 2022 • Prizren, Kosovo • 2022.07.19

`https://people.debian.org/~gwolf/dc22/openpgp.pdf`

Warning — What to expect from this talk

- Research project
  - You see, this guy is trying to do a PhD...
  - And you are the (indirect) study subjects
- Related to my Debian task as keyring-maint
  - But will not be applied to Debian
  - At least not in a forseeable future ☺
- Related to the wider OpenPGP ecosystem and community
  - So, relevant to the Free Software world
- In the hopes it will be interesting / entertaining!
  - Most of the introduction will be *old news* for most of you

Once upon a time, there was a happy and naïve network. . .

HAPPY COMPUTER
DM 5,-
2 84 FEBRUAR
5 Heimcomputer der Oberklasse im Vergleichstest
So zaubert man Comics auf den Bildschirm
War Games – der Kino-Hit als Spielprogramm
Listing des Monats: Zauberschloß
Musikbaukasten für den Apple II
mit VGA-Grafik

freepngimg.com (Attribution)

) Happy Computer, archive.org (CC BY-NC-ND)

But the world is full of evil. . .

positek.net / shutterstock (CC BY-SA)

Fortunately, Internet has evolved: We now have cryptography everywhere!

But. . . What does this cryptography really give us?

Privacy News Online (CC BY-SA)

Protection against eavesdropping

Alex Briseño, Flickr (CC BY-SA-NC)

What do we get from the simple use of *public-key cryptography*? And what is still not covered?

We get

- Strong cryptography
  - Impossible to break in a reasonable time, even with current Nation-State resources
- Uses algorithms that have received public, expert scrutiny
  - ElGamal, DSA, RSA, EC
- Works over preexisting protocols
  - E-mail, local storage

We do not get

- Hiding the *fact there is communication* ocurring between two participants
  - Metadata analysis
- Verification of correct identity
  - *Equivocation* attacks
  - *Man in the Middle* (MITM)

PGP: Pretty Good Privacy

> 30 years flying high

Errol Cavit, Flickr (CC BY-NC-ND)

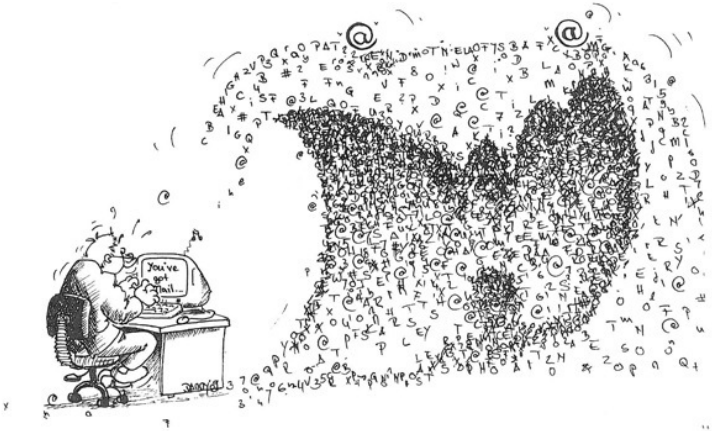
Construction blocks for *identity verification*

Andy Hay (CC-BY)

What does it mean to *verify an identity*?

Aldon Hynes (CC-BY-SA)

Internet is too big to *know* everybody I interact with!



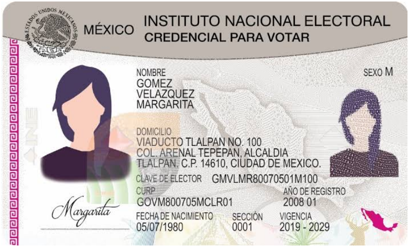
Bob Doyle (CC BY)

Transitive trust distribution mechanisms

. . . But we can trust *somebody*, right?

and we can trust on the *truth* of the identities they are willing to back. . .

① Centralized trust



ESTADOS UNIDOS MEXICANOS

MÉXICO INSTITUTO NACIONAL ELECTORAL

CREDENCIAL PARA VOTAR

NOMBRE
GOMEZ
VELAZQUEZ
MARGARITA

SEXO M

DOMICILIO
VIADUCTO TLALPAN NO. 100
COL. ARENAL TEPEPAN, ALCALDIA
TLALPAN, C.P. 14610, CIUDAD DE MEXICO.

CLAVE DE ELECTOR GMVLMR80070501M100

CURP
GOVM800705MCLR01

AÑO DE REGISTRO
2008 01

FECHA DE NACIMIENTO
05/07/1980

SECCIÓN
0001

VIGENCIA
2019 - 2029

Margarita

Robbie Sproule, Wikipedia (CC BY)
Francis Sarahi Castro Ponce, Wikipedia (CC 0)

) everystockphoto.com, Wikipedia (DP)

2 Distributed trust

Miren Pardo, Juegos de asamblea: Conocemos a nuestros compañeros (CC BY-SA-NC)

Formalizing a little bit. . .

Centralized mechanisms

- A set of *ultimate roots of trust* are *centrally* defined
- Each *Root of trust* can *delegate* trust on several *Ceritifation Authorities* (CA)
- Communication parties (i.e. servers) provide their public key and a CA-signed *certificate*

⇓

PKI-CA model

Distributed mechanisms

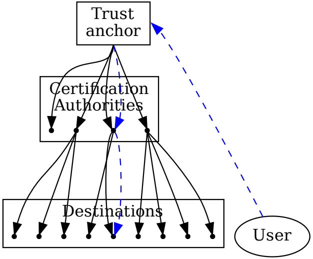
- Centered in *each user*
- Every user can *emit ceritifcations* for whom they personally know
  - Signing policies?
  - What does it mean to *know*?
  - Can I trust *your* criteria?
- A global *Web of Trust* global is *woven*

⇓

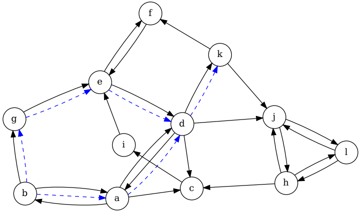
WoT model

Note, of course, there are other models...

Transitive trust distribution models



Centralized: Certification Authorities (PKI-CA)



Distributed: Web of Trust (WoT)

Focus of the work: Distributed model (WoT)

. . . But that requires *many people* to know *many people*!

Steven Fruitsmaak, WikiNews (CC BY)

So, we only need to *grow* the size of the WoT?

- Everybody verifies each other's documents (government-issued ID?)
- *Certifies* the keys of the rest of the group
- Network tust strongly increases!

Steven Fruitsmaak, WikiNews (CC BY)

So, we only need to *grow* the size of the WoT?

- Everybody verifies each other's documents (government-issued ID?)
- *Certifies* the keys of the rest of the group
- Network tust strongly increases!
- . . . In >300 people gatherings. . .

SRSLY?

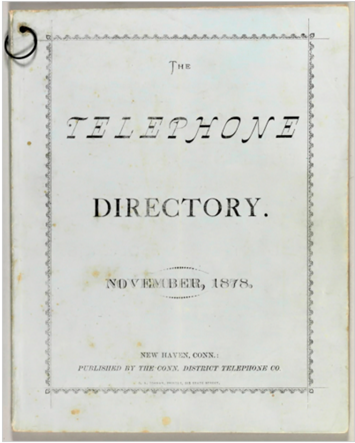
Steven Fruitsmaak, WikiNews (CC BY)

The public key distribution problem

A key distribution *infrastructure* is now needed…

- Under TLS (PKI-CA), key+certificates are presented upon session establishment
  - Watch out for MitM and revocations!
  - Do you *really* trust the *trusted introducers*?
- Under OpenPGP (WoT), the destination key must be obtained *before sending a message*
  - Asynchronous operation

⇒ PKS keyservers



The TELEPHONE DIRECTORY.

NOVEMBER, 1878.

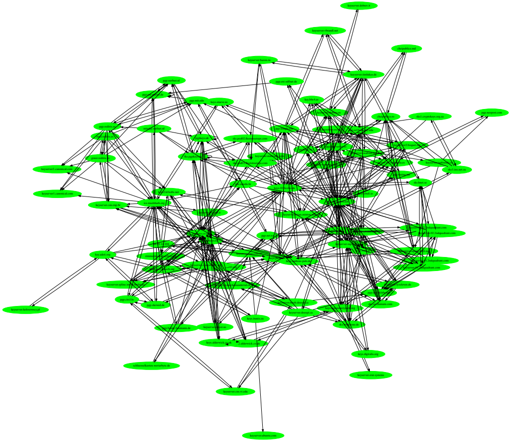
NEW HAVEN, CONN.:
PUBLISHED BY THE CONN. DISTRICT TELEPHONE CO.

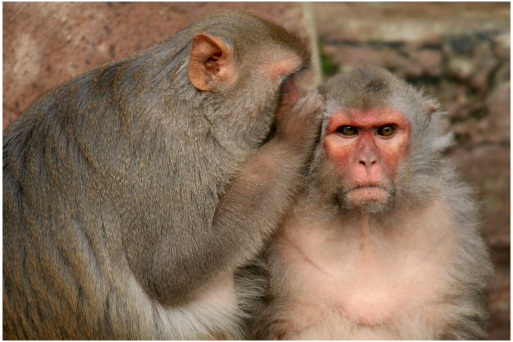
Connecticut District Telephone Company, 1878 (DP)

But... how do we *avoid centralization*?

Javier Álvarez, Flickr (CC BY-ND)

Set of keyservers running an *epidemic* or *gossip protocol* for *large sets reconciliation*. . .





jinterwas, Flickr (CC BY)

Result (1): Binary, non-modifiable, distributed, non-authenticated, eventually consistent storage

Mike Izbicki (CC BY-SA)

Result ②: Attacks on the model ☹

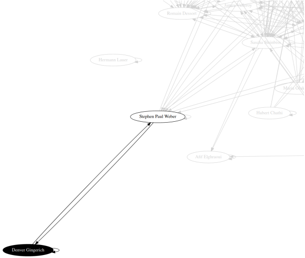
Ben Simon (CC BY)

)

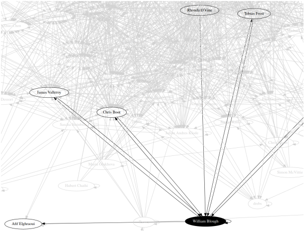
jinterwas, Flickr (CC BY)

What is *certificate poisoning*? ①

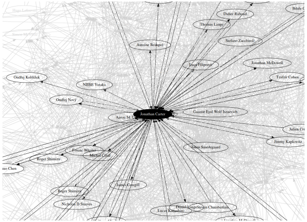
Normally, only my *direct contacts* will certify my key, allowing others to find me in the WoT



I might be little connected. . .



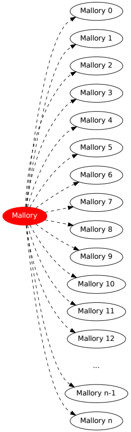
Somewhat more connected. . .



I can be *strongly* connected. . .

Normal keys will have dozens, maybe up to *hundreds* of certifications.

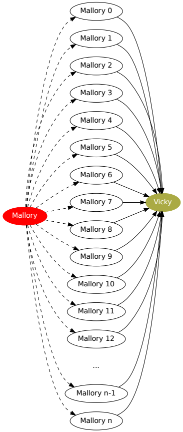
What is *certificate poisoning*? ②



An attacker, *Mallory* (M), can generate *many* throwaway identities $M_1, M_2, M_3, \ldots M_n$ ($n \approx 100\,000$)

These identities are *garbage keys*, they don't even need to be linked to *Mallory*'s real identity.

What is *certificate poisoning*? ③

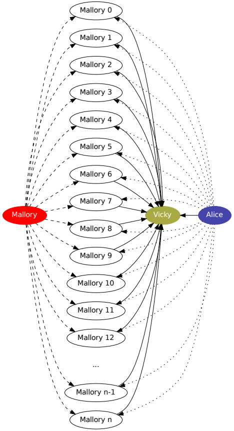


Mallory certifies victim *Vicky*'s key with all their identities — and make *Vicky*'s public key V useless.

Vicky sees herself forced to abandon her identity and generate a new pair of keys V', but. . .

- Getting her new identity connected to the WoT has a high cost (time, effort)
- Opens a time window for supplantation / ID theft

What is *certificate poisoning*? ④



When *Alice* (*A*) searches for *Vicky*'s key, upon importing it, she suffers a denial of service (and possibly an OpenPGP database corruption)

What is *certificate poisoning*? ⑤

Peter Krimbacher, Wikipedia (CC BY-SA)

Why don't we delete the spurious certificates?

Jumanji Solar, Flickr (CC BY-NC-SA)

)

Why don't we delete the spurious certificates?

Jumanji Solar, Flickr (CC BY-NC-SA)

)

José-Manuel Benito, Wikimedia (DP)

Why don't we delete the spurious certificates?

And... What about the European **GDPR**?

Right to be forgotten, information deletion orders...

Jumanji Solar, Flickr (CC BY-NC-SA)

José Manuel Benito, Wikimedia (DP)

Why don't we delete the spurious certificates?

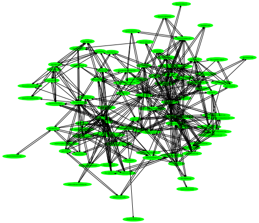
And... What about the European **GDPR**?

Right to be forgotten, information deletion orders...

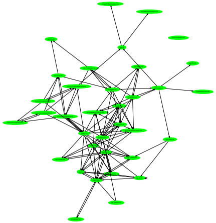
- GDPR imposes *privacy conditions* that are *impossible to comply with* for keyserver network operators
- ...All of this has caused the number of keyservers to decrease strongly... And the outlook is quite bleak ☹

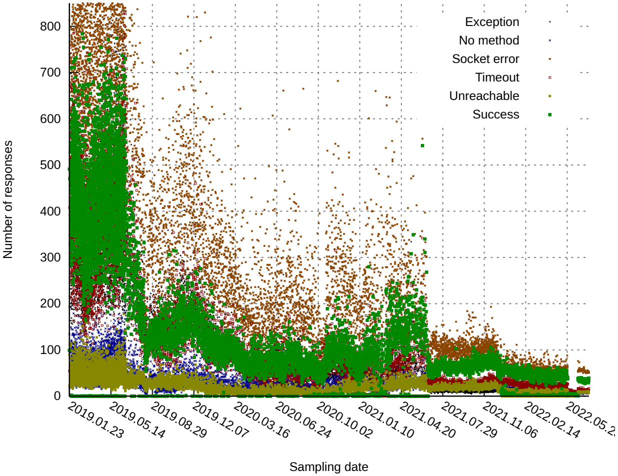
The keyserver network. . . shrinks ☹

2019.01.23



2022.07.16





Data available at https://sks-status.gwolf.org/

Central idea

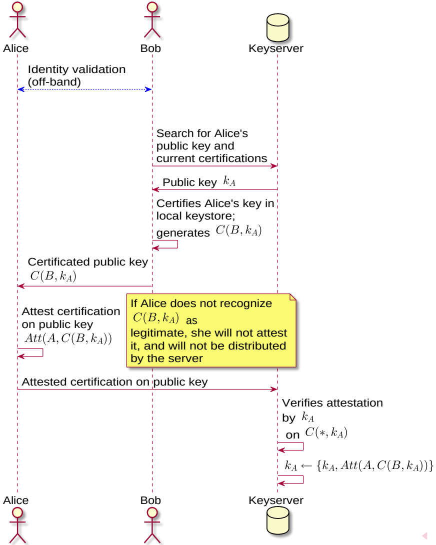
Present a solution that *keeps the distributed model viable*, without requiring centralizing entities.

My main goal is to present a protocol that prevents *certificate poisoning* without compromising WoT's main positive characteristics.

First-party attested third party certification (1PA3PC) protocol → Require all OpenPGP packets modifying *k* to be *accepted* (signed) by *k*

- Certificate poisoning no longer possible
- Implementing a decades-long best-practices recommendation that has been unable to be mandated
  - Ever heard of... `caff` ? ☺
  - But push a *best practice* to a *requirement* level

1PA3PC: Key certification protocol



Alice

Bob

Keyserver

Identity validation (off-band)

Search for Alice's public key and current certifications

Public key k_A

Certifies Alice's key in local keystore; generates $C(B, k_A)$

Certificated public key $C(B, k_A)$

Attest certification on public key $Att(A, C(B, k_A))$

If Alice does not recognize $C(B, k_A)$ as legitimate, she will not attest it, and will not be distributed by the server

Attested certification on public key

Verifies attestation by k_A on $C(*, k_A)$

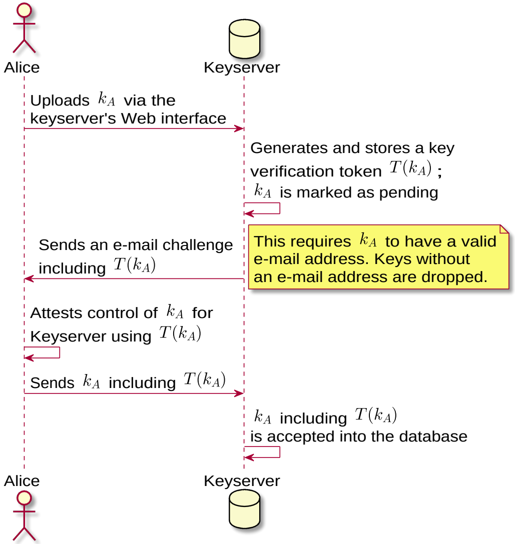
$k_A \leftarrow \{k_A, Att(A, C(B, k_A))\}$

Alice

Bob

Keyserver

Extending 1PA3PC to allow for *UID verification*



Uploads k_A via the keyserver's Web interface

Generates and stores a key verification token $T(k_A)$; k_A is marked as pending

This requires k_A to have a valid e-mail address. Keys without an e-mail address are dropped.

Sends an e-mail challenge including $T(k_A)$

Attests control of k_A for Keyserver using $T(k_A)$

Sends k_A including $T(k_A)$

k_A including $T(k_A)$ is accepted into the database

This is far from well thought-out

- Affixing attestation information to k_A allows keys to be identified as *having been uploaded to* given a server
- Each keyserver operator maintains a list of *locally trusted* keyservers
  - Operators can report keyservers as *rogue*, but decisions are *local*
- Each keyserver can offer *different views* of the database
  - Based on *each operator's trust*

Extending 1PA3PC to allow for *information removal*?

This is even less thought out ☺

And what if *Alice* attests a *specific view* of k_A?

- She can control the information on her key she accepts as valid
- By *hiding* information on non-attested (or *past-attested*) signatures, users can control *what the network says about their social connections*
  - Of course, the old information *is still there* — although somewhat hidden
  - Key bloat is not solved (although should remain *controllable* due to the 1PA3PC key certification protocol)

Expected outcome

This seemingly simple modification to the keyserver network operation pursues to:

- Allow a decentralized, public keyserver network to keep operating, mitigating the effect attacks have had on it, and allowing it to continue to exist with modern privacy expectations
- Keep the WoT decentralized transitive trust model relevant and sustainable for OpenPGP communications
  - Fundamental component for several large-scale, geographically-distributed free software development projects
- Allow for signatures' information *not to be presented* to users if it's no longer desired by key owner
  - (Would that satisfy GDPR? Am quite skeptical, and IANAL, but. . .)
- What about *death by kindness*?
  - OpenPGP + WoT are hard enough to use as it is. Extra hurdles might actually hurt rather than help it!

Thank you very much for your attention.

... And for listening to my half-baked ideas ;-)

Gunnar Wolf
→ `gwolf@debian.org`
`https://people.debian.org/~gwolf/dc22/openpgp.pdf`