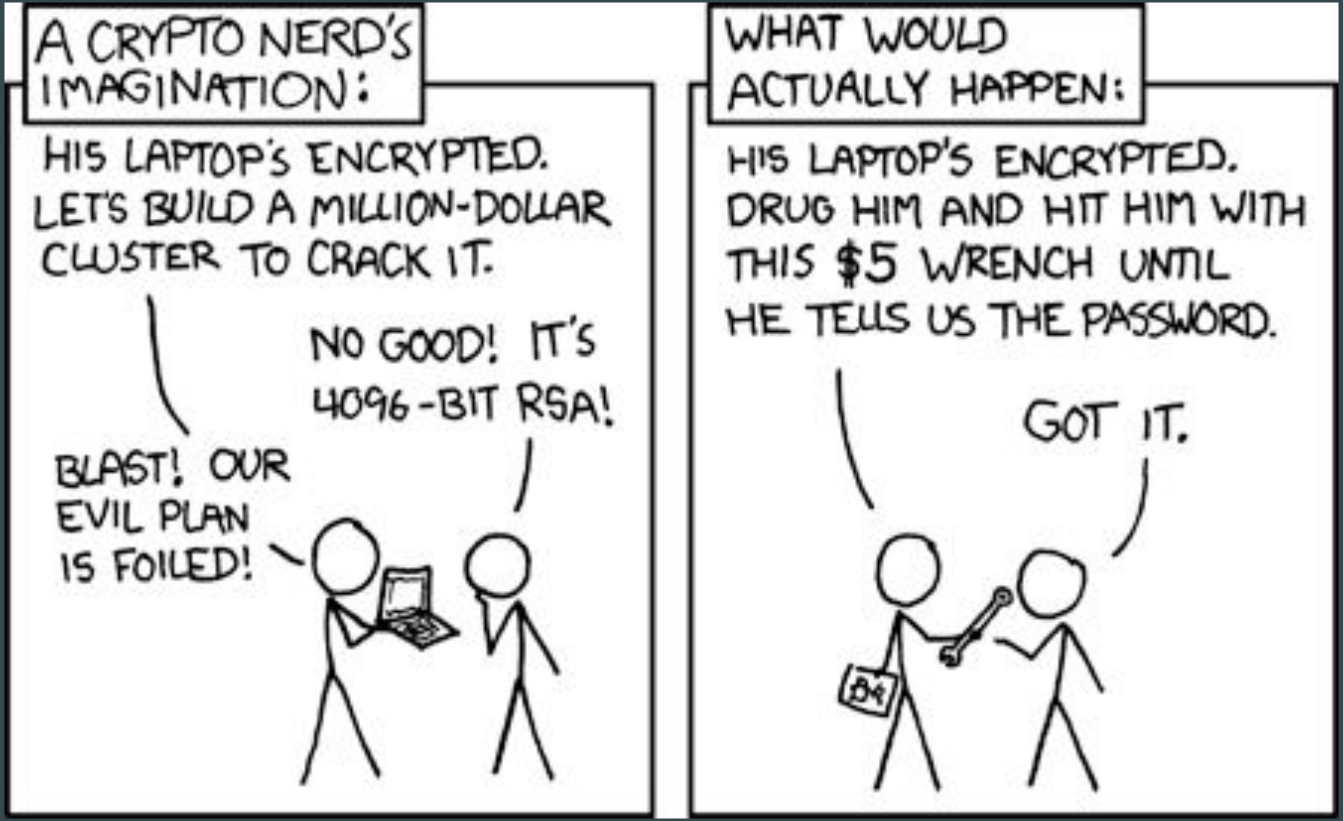


A CRYPTO NERD'S IMAGINATION:
HIS LAPTOP'S ENCRYPTED. LET'S BUILD A MILLION-DOLLAR CLUSTER TO CRACK IT.
NO GOOD! IT'S 4096-BIT RSA!
BLAST! OUR EVIL PLAN IS FOILED!
WHAT WOULD ACTUALLY HAPPEN:
HIS LAPTOP'S ENCRYPTED. DRUG HIM AND HIT HIM WITH THIS $5 WRENCH UNTIL HE TELLS US THE PASSWORD.
GOT IT.

@GusGorman... Because you asked so nicely!

Failings of the Public Key Infrastructure

•••

Jacob Mohrbutter (Qu3b4l1)

Philosophical Point of View

- My background stems from development
    - Mainly low level stuff, C/C++, Assembly

Philosophical Point of View

- My background stems from development
    - Mainly low level stuff, C/C++, Assembly
- In my opinion there are three big phases that all software or software systems experience

Philosophical Point of View

- My background stems from development
  - Mainly low level stuff, C/C++, Assembly
- In my opinion there are three big phases that all software or software systems experience
  - Design and Implementation

Philosophical Point of View

- My background stems from development
  - Mainly low level stuff, C/C++, Assembly
- In my opinion there are three big phases that all software or software systems experience
  - Design and Implementation
  - Maintenance

Philosophical Point of View

- My background stems from development
  - Mainly low level stuff, C/C++, Assembly
- In my opinion there are three big phases that all software or software systems experience
  - Design and Implementation
  - Maintenance
  - Obsoletion

Philosophical Point of View: Design and Implementation

- The act of Designing Software is by its very nature anti-entropic. One takes data and orders it into the desired state

Philosophical Point of View: Design and Implementation

- The act of Designing Software is by its very nature anti-entropic. One takes data and orders it into the desired state
- A newly implemented software system exists at its most ordered state

Philosophical Point of View: Maintenance

- Software, or a software system must meet the demands of the present

Philosophical Point of View: Maintenance

- Software, or a software system must meet the demands of the present
    - Fixing bugs introduces new states to a software system

Philosophical Point of View: Maintenance

- Software, or a software system must meet the demands of the present
    - Fixing bugs introduces new states to a software system
    - News features have to be added to meet evolving demands

Philosophical Point of View: Maintenance

- Software, or a software system must meet the demands of the present
  - Fixing bugs introduces new states to a software system
  - News features have to be added to meet evolving demands
- By its very nature maintenance is entropic
  - It's not a linear decay, it's more exponential

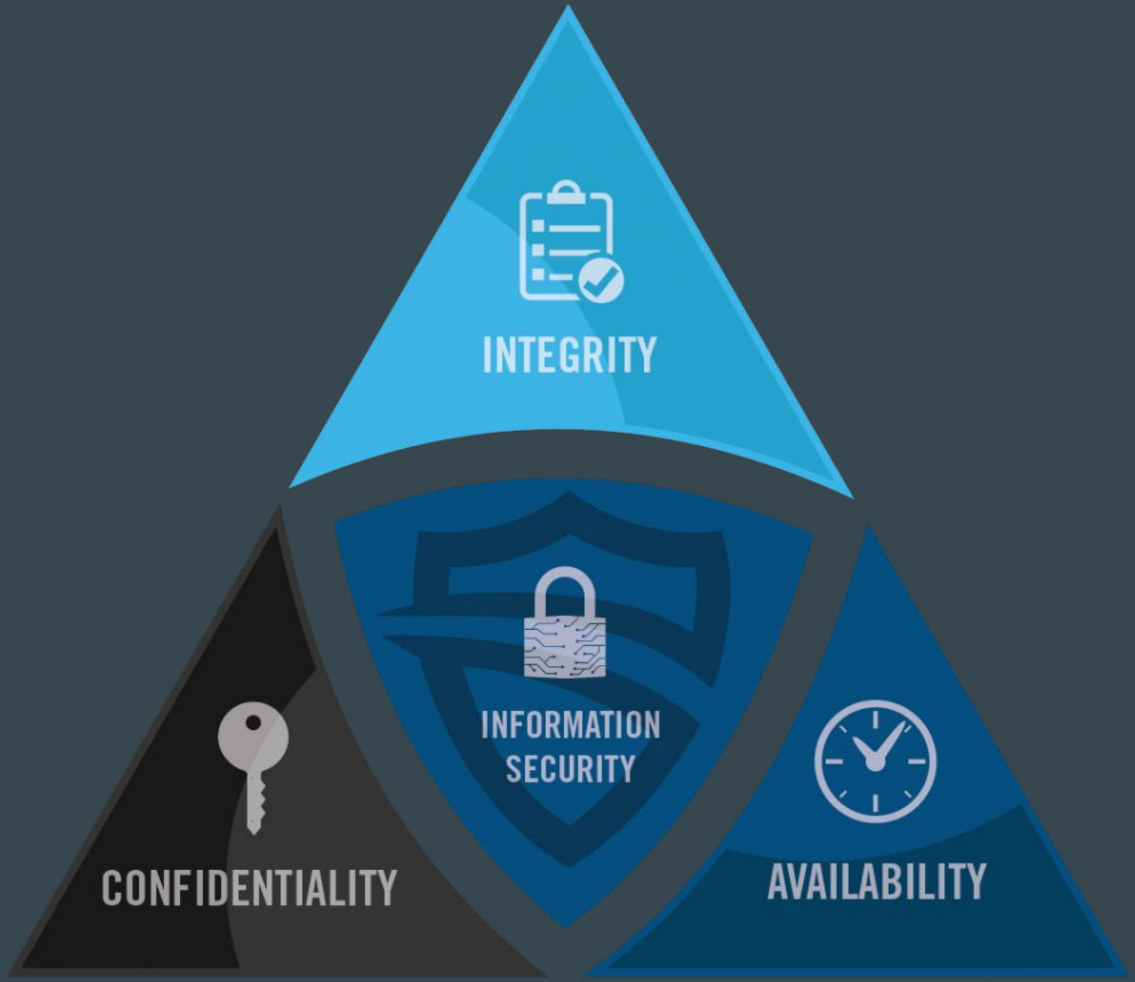
Philosophical Point of View: Obsoletion

- There comes a point where Maintenance becomes the act of deteriorating the functionality of a piece of software, or a software system

The CIA triangle

Integrity implies authenticity

Maybe the original designers of ssl took that into account



The CIA triangle

Integrity implies authenticity

Maybe the original designers of ssl took that into account

INTEGRITY
SECURITY
CONFIDENTIALITY
AVAILABILITY

> “That whole authenticity thing, Yeah we just threw that in at the end”
>
> --Moxie Marlinspike

Certificate Revocation Lists (CRL’s)

Certificates need to be revoked some times.

How did we deal with this?

Hmm what if we DDOS that certificate authority!

RFC-5280: Certificate Revocation Lists

“

When there are no revoked certificates, the revoked certificates list MUST be absent.

Otherwise, revoked certificates are listed by their serial numbers. Certificates revoked by the CA are uniquely identified by the certificate serial number. The date on which the revocation occurred is specified. The time for revocation Date MUST be expressed as described in Section 5.1.2.4. Additional information may be supplied in CRL entry extensions; CRL entry extensions are

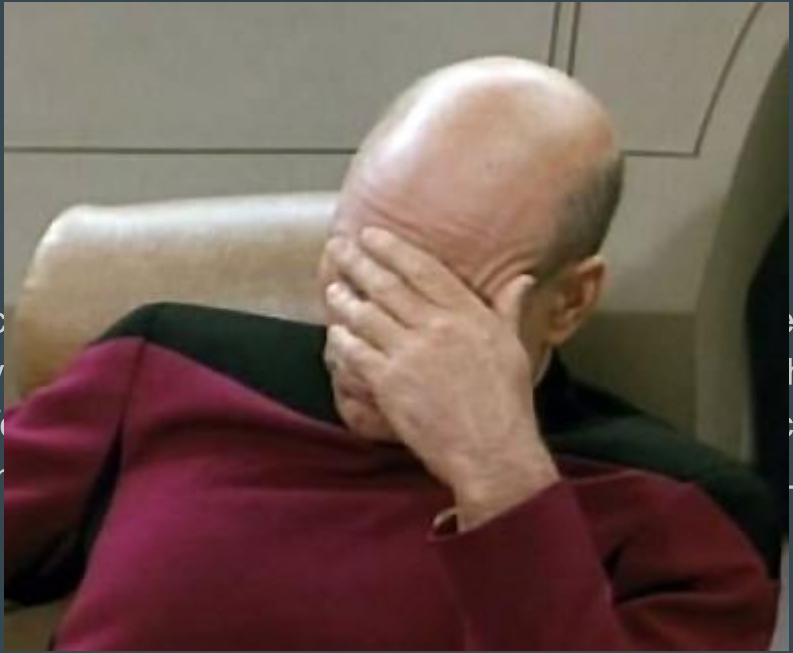
”

discussed in Section 5.3.

RFC-5280: Certificate Revocation Lists

“

When there are



list MUST be absent.

Otherwise, revoked c ertificates revoked by the CA are uniquely identified by h the revocation occurred is specified. The time f cribed in Section 5.1.2.4. Additional informatio L entry extensions are

discussed in Section 5.3.

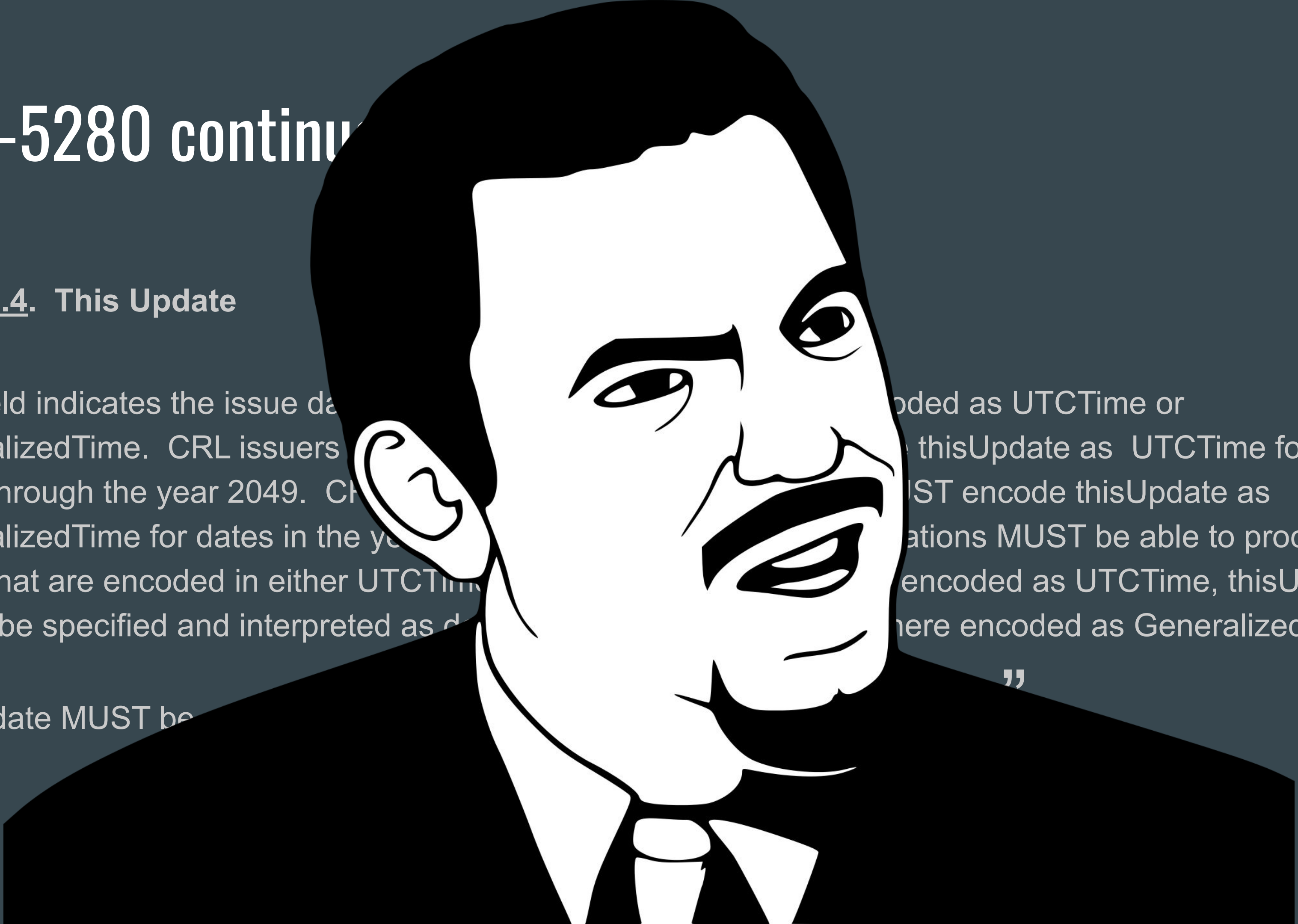
RFC-5280 continued

> “
> **5.1.2.4. This Update**
>
> This field indicates the issue date of this CRL. thisUpdate may be encoded as UTCTime or GeneralizedTime. CRL issuers conforming to this profile MUST encode thisUpdate as UTCTime for dates through the year 2049. CRL issuers conforming to this profile MUST encode thisUpdate as GeneralizedTime for dates in the year 2050 or later. Conforming applications MUST be able to process dates that are encoded in either UTCTime or GeneralizedTime. Where encoded as UTCTime, thisUpdate MUST be specified and interpreted as defined in Section 4.1.2.5.1. Where encoded as GeneralizedTime,
>
> thisUpdate MUST be specified and interpreted as defined in Section 4.1.2.5.2.”

RFC-5280 continu

> “
> 5.1.2.4. This Update
>
> This field indicates the issue da oded as UTCTime or GeneralizedTime. CRL issuers e thisUpdate as UTCTime for dates through the year 2049. C ST encode thisUpdate as GeneralizedTime for dates in the ye ations MUST be able to process dates that are encoded in either UTCTim encoded as UTCTime, thisUpdate MUST be specified and interpreted as d here encoded as GeneralizedTime,
> ”

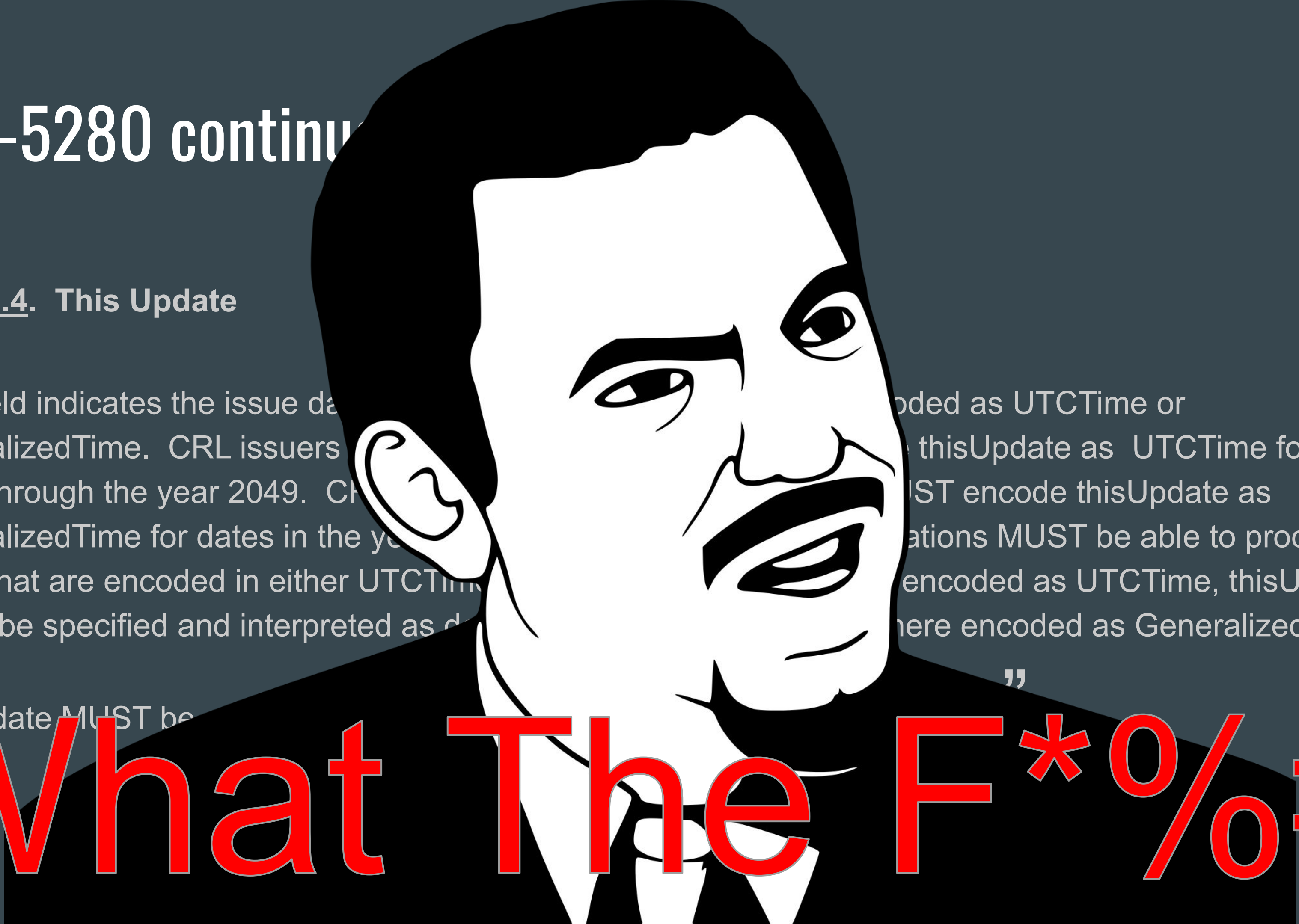
thisUpdate MUST be



RFC-5280 continu

> “
> **5.1.2.4. This Update**
>
> This field indicates the issue da oded as UTCTime or
> GeneralizedTime. CRL issuers thisUpdate as UTCTime for
> dates through the year 2049. CR ST encode thisUpdate as
> GeneralizedTime for dates in the ye ations MUST be able to process
> dates that are encoded in either UTCTim encoded as UTCTime, thisUpdate
> MUST be specified and interpreted as d here encoded as GeneralizedTime,
> ”

thisUpdate MUST be



Philosophical Point of View Revisited

- Programmers are happy to fill in the blanks
    - If the spec is not written well, no one will be working under the same assumptions
- Specifications need to be written so that there is no underlying assumptions needed
- Communication, or lack thereof, breaks shit

Somehow this all worked up until ~2011

What changed?

Somehow this all worked up until ~2011

What changed?

Etisalat: Thank you! Blackberrys were definitely better after receiving spyware from a trusted third party update.

Somehow this all worked up until ~2011

What changed?

Etisalat: Thank you! Blackberrys were definitely better after receiving spyware from a trusted third party update.

map of certificate authorities in ~2010

Somehow this all worked up until ~2011

What changed!

Etisalat: Thank you! Blackberrys were definitely better after receiving spyware from a trusted third party update.

map of certificate authorities in ~2010

Stuxnet - Was a valid Realtek device driver from January 25th, 2010 to July 22nd, 2010

Somehow this all worked up until ~2011

What changed?

Etisalat: Thank you! Blackberrys were definitely better after receiving spyware from a trusted third party update.

map of certificate authorities in ~2010

Stuxnet - Was a valid Realtek device driver from January 25th, 2010 to July 22nd, 2010

Flame - Had a valid certificate from December 28th, 2010 to june 4th, 2012

Oh.. and then We have the CA’s themselves

Comodo: private keys compromised in 2012 by “Nation State actors”… cough cough, script kiddie, cough.

Oh.. and then We have the CA’s themselves

Comodo: private keys compromised in 2012 by “Nation State actors”... cough cough, script kiddie, cough.

And here is for the lolz https://pastebin.com/u/ComodoHacker

Oh.. and then We have the CA’s themselves

Comodo: private keys compromised in 2012 by “Nation State actors”... cough cough, script kiddie, cough.

And here is for the lolz https://pastebin.com/u/ComodoHacker

Diginotar, compromised July 10th, 2011. Removed as a trusted root CA September 2nd, 2011 after issuing 531 invalid certificates.

The Problem: How the F%*# do we Audit this thing

Problem: There are too many certificate authorities, each one has their own fragmented data store.

The Problem: How the F%*# do we Audit this thing

Problem: There are too many certificate authorities, each one has their own fragmented data store.

Solution: Bolt on a new component that enables auditing.

The Problem: How the F%*# do we Audit this thing

Problem: There are too many certificate authorities, each one has their own fragmented data store.

Solution: Bolt on a new component that enables auditing.

Thank you Google for introducing the maintenance patch that finally pushes PKI into the final phase of the software life cycle. Obsoletion!

The Problem: How the F%*# do we Audit this thing

Problem: There are too many certificate authorities, each one has their own fragmented data store.

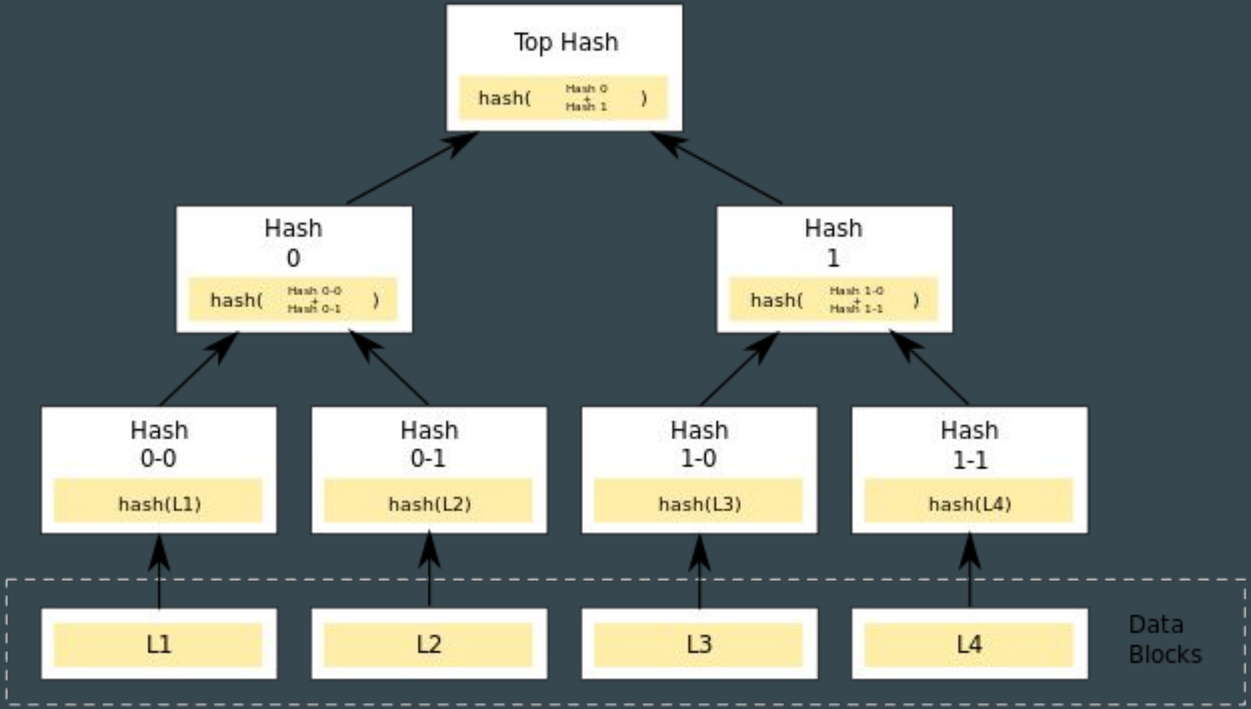
Solution: Bolt on a new component that enables auditing.

Thank you Google for introducing the maintenance patch that finally pushes PKI into the final phase of the software life cycle. Obsoletion!

Thus we have Certificate Transparency 2011

Certificate Transparency

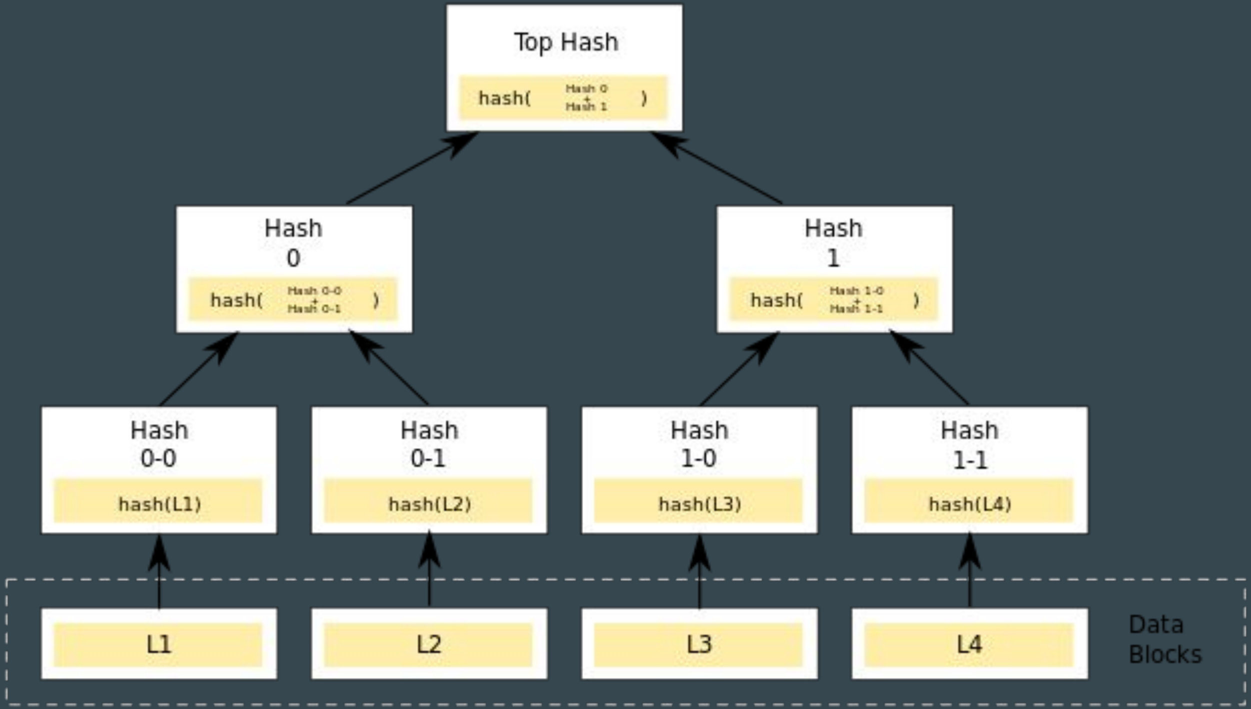
Collect certificates



Certificate Transparency

Collect certificates

Hash them and create a Merkle Hash Tree

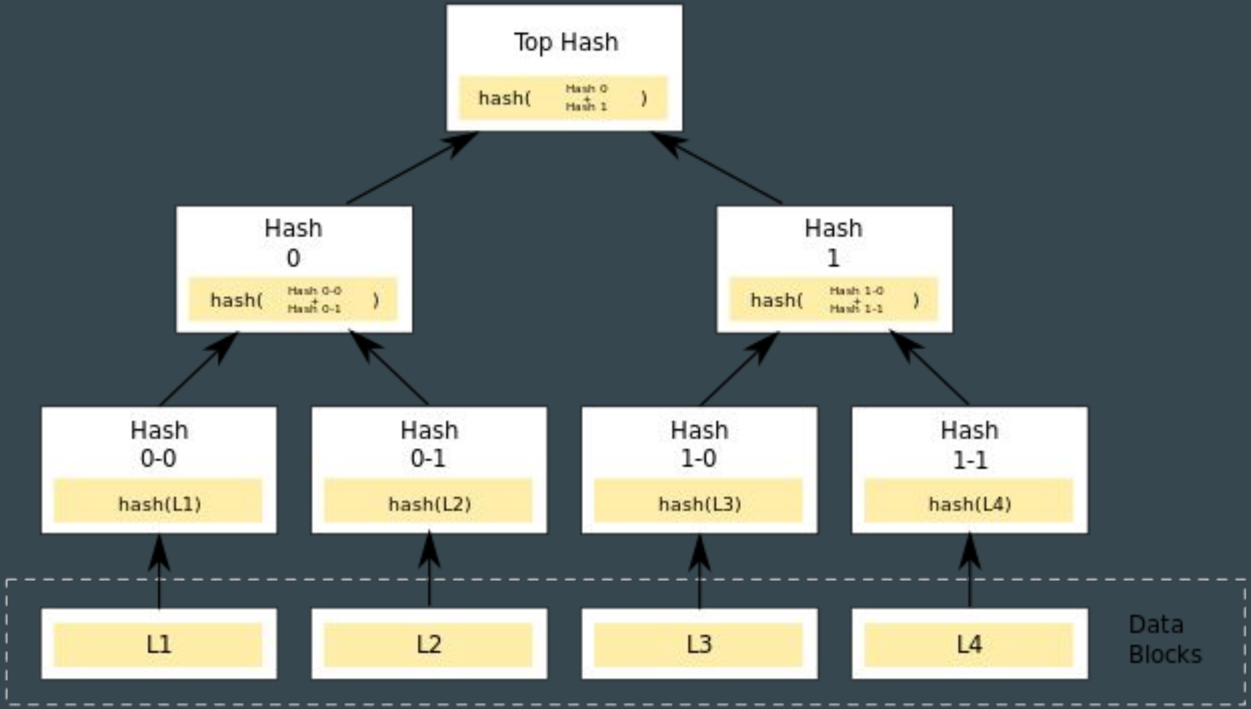


Certificate Transparency

Collect certificates

Hash them and create a Merkle Hash Tree

Sign the head of that tree and add to the certificate log at a pre-conceived time interval



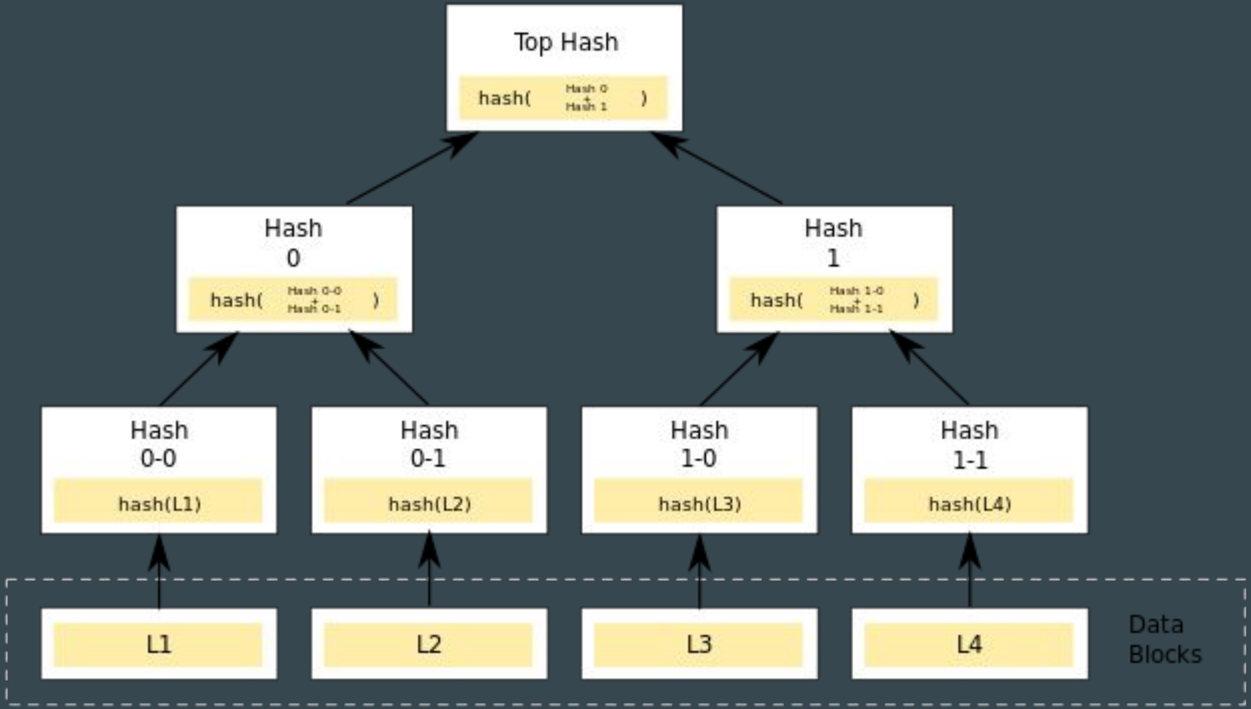
Certificate Transparency

Collect certificates

Hash them and create a Merkle Hash Tree

Sign the head of that tree and add to the certificate log at a pre-conceived time interval

Certificates submitted to the log receive a Signed Certificate Timestamp (SCT)



Certificate Transparency

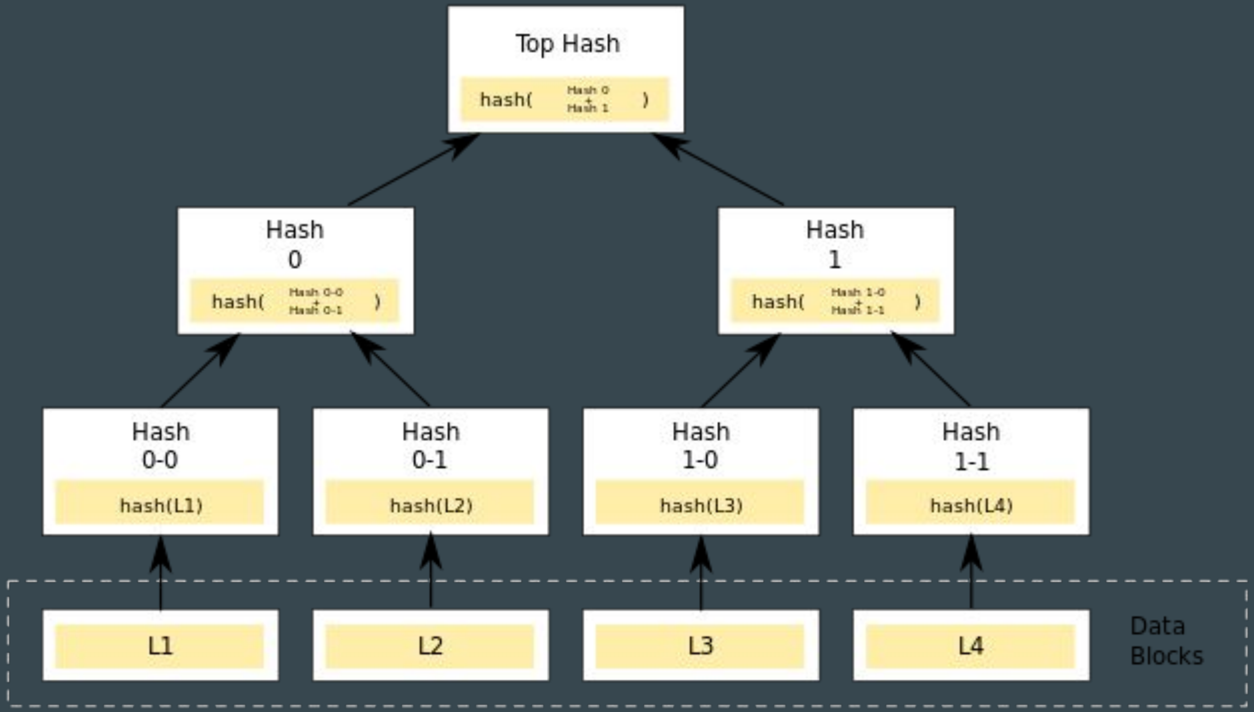
Collect certificates

Hash them and create a Merkle Hash Tree

Sign the head of that tree and add to the certificate log at a pre-conceived time interval

Certificates submitted to the log receive a Signed Certificate Timestamp (SCT)

Certificate and SCT are served to clients.



Certificate Transparency

Collect certificates

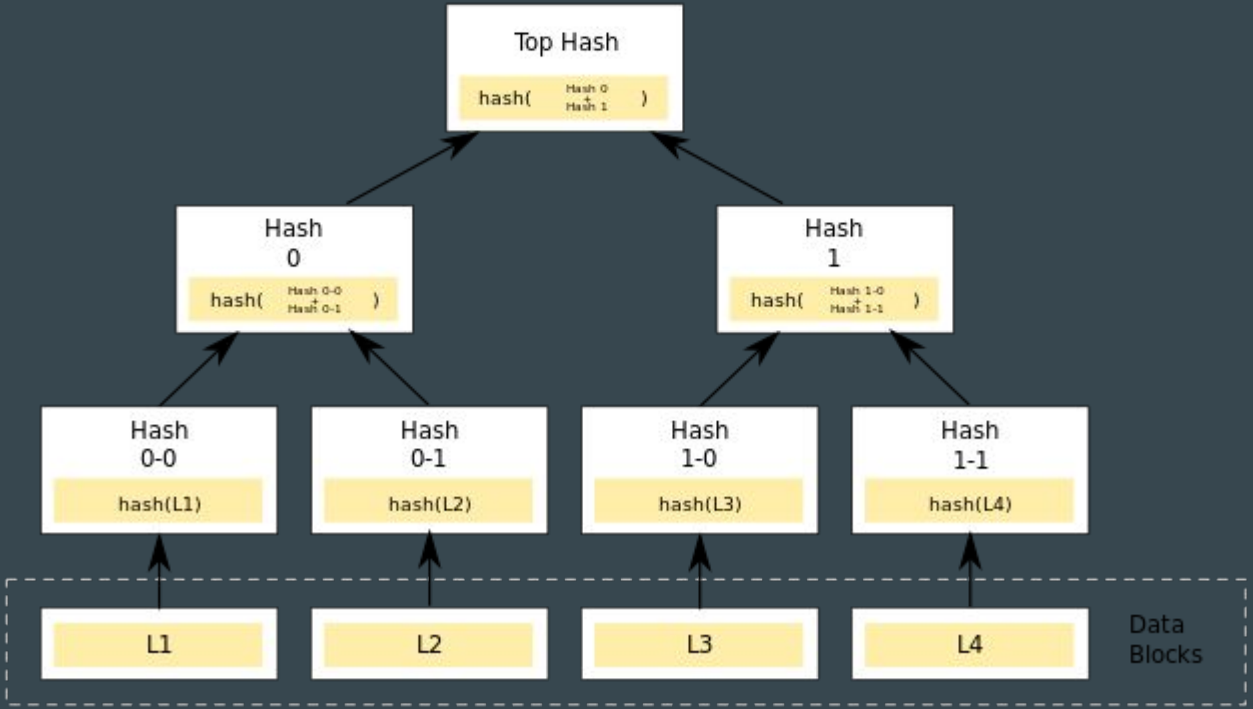
Hash them and create a Merkle Hash Tree

Sign the head of that tree and add to the certificate log at a pre-conceived time interval

Certificates submitted to the log receive a Signed Certificate Timestamp (SCT)

Certificate and SCT are served to clients.

SCT acts to locate the specific entry



Benefits

As long as clients have a valid view of the log, the state of a certificate can be ensured

Benefits

As long as clients have a valid view of the log, the state of a certificate can be ensured

The merkle hash trees ensure that the data within the tree is immutable, no certificate log can deny signing a Merkle Hash Tree

Benefits

As long as clients have a valid view of the log, the state of a certificate can be ensured

The merkle hash trees ensure that the data within the tree is immutable, no certificate log can deny signing a Merkle Hash Tree

Auditing becomes easier and visibility within the public key infrastructure is increased.

WOHOOO! PROBLEM SOLVED
TIME TO GO HOME!
makeameme.org

But wait, We can fuck with this

Lets execute a split-world attack

But wait, We can fuck with this

Lets execute a split-world attack

Attacker gets a valid certificate

But wait, We can fuck with this

Lets execute a split-world attack

Attacker gets a valid certificate

Attacker submits certificate to the certificate log and gets a SCT

But wait, We can fuck with this

Lets execute a split-world attack

Attacker gets a valid certificate

Attacker submits certificate to the certificate log and gets a SCT

Attacker sends victim the certificate and SCT.

But wait, We can fuck with this

Lets execute a split-world attack

Attacker gets a valid certificate

Attacker submits certificate to the certificate log and gets a SCT

Attacker sends victim the certificate and SCT.

The victim can’t see that a revocation was made in a future log because they have a valid but fraudulent SCT

Back to square one

Well, I think it's fairly obvious, there's no way forward without scraping the whole thing.

There are takeaways from this system. Maybe some important design considerations for a next generation public key infrastructure.

Design Considerations

Certificates need to have the ability to assume a revocation state.

Design Considerations

Certificates need to have the ability to assume a revocation state.

Public ledgers such as certificate transparency may have solved visibility but we can do better.

Design Considerations

Certificates need to have the ability to assume a revocation state.

Public ledgers such as certificate transparency may have solved visibility but we can do better.

The design should construct an immutable account of actions that protect the actual state of a certificate.

Design Considerations

Certificates need to have the ability to assume a revocation state.

Public ledgers such as certificate transparency may have solved visibility but we can do better.

The design should construct an immutable account of actions that protect the actual state of a certificate.

The current cryptographic functionality of the public key infrastructure should be maintained. This does not mean it’s bolted on, it needs to be an integrated member of the design.

Thank You for putting up with me for this long

I allude to a viable solution in my paper while addressing a few more design considerations

https://github.com/Qu3b411/Failings-of-the-Public-Key-Infastructure/

Ping me on slack and let me know what you thought!

@Qu3b411