ASU DEVILSEC / LUG
14 OCT 2022

CRLITE: ALL THE WEB PKI'S TRUST REVOCATIONS, COMPRESSED

FIXING LONGSTANDING WEB PKI PROBLEMS, FINALLY

Hi, I'm

J.C. JONES

- Cryptography Engineer & SRE @ Let's Encrypt

BUT NOT HERE ON THEIR BEHALF

- Formerly: Mozilla Crypto Engineering

@CIPHERCOFFEE

JC@INSUFFICIENT.COFFEE
LETSENCRYPT.ORG

HTTPS://INSUFFICIENT.COFFEE/

ETIQUETTE

I can talk about this stuff all day.

DON'T LET ME

Unclear? Question?

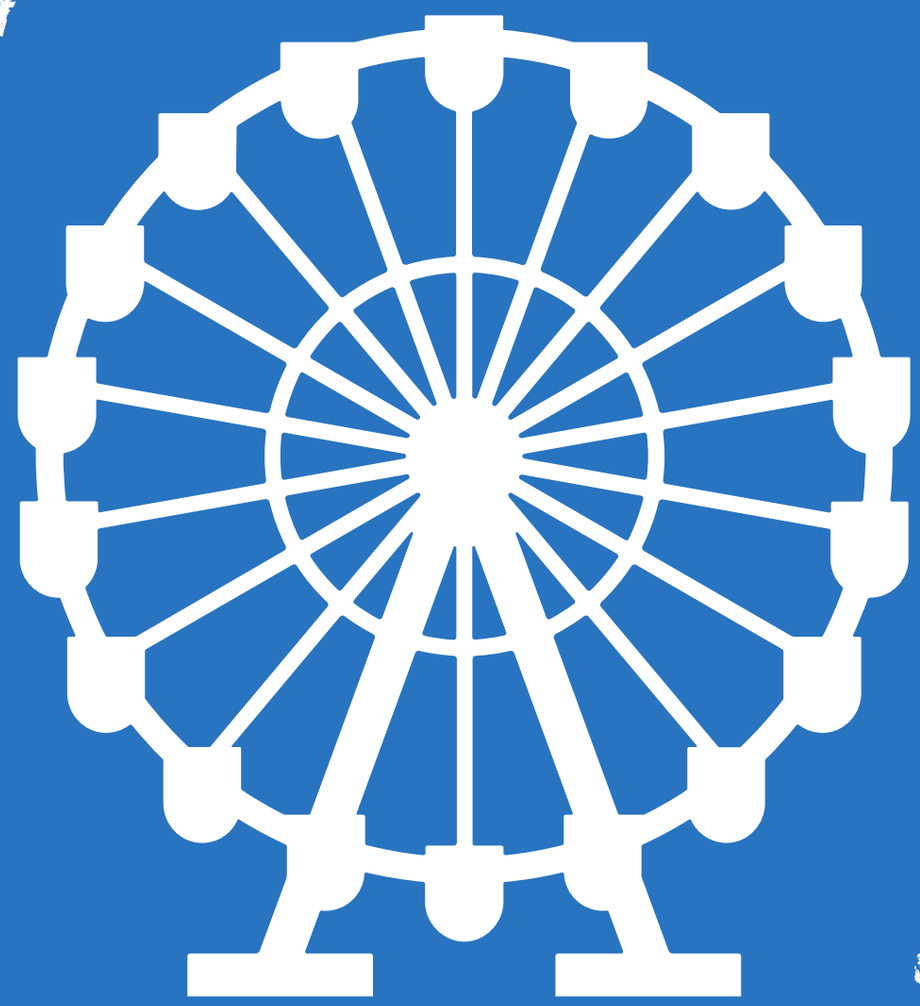
INTERRUPT ME!

Q&A IS MY GOAL

I HAVE LOTS OF SLIDES...
BUT BONUS IF WE USE THEM DURING Q&A!

MAP

Problems with Revocations in the Web PKI



A Quick Intro to the Web PKI

CRLite and its techniques

War Stories?

THE WEB PKI

ALAS, FOR THE GLOSS HAS LONG AGO GONE DARK

PUBLIC KEY INFRASTRUCTURE

FOR THE WEB

PUBLIC KEY INFRASTRUCTURE

"Can I trust this assertion?"

- Asymmetric cryptography
- Digital Signatures
- Transitive Trust

ACTORS IN A PKI

- Certificate Authorities
- Browsers
- Relying Parties (users)

WHAT IS A PUBLIC KEY INFRASTRUCTURE?

1. MY BROWSER TRUSTS THIS CONNECTION ...
2. BECAUSE IT TRUSTS THE SIGNATURE OF THIS CERTIFICATE AUTHORITY ...
3. BECAUSE MY BROWSER VENDOR TRUSTS THE AUTHORITY'S AUDITS ...
4. AND I TRUST MY BROWSER VENDOR'S JUDGEMENT.

WHAT IS A PUBLIC KEY INFRASTRUCTURE?

1. MY BROWSER TRUSTS THIS CONNECTION …
2. BECAUSE IT TRUSTS THE SIGNATURE OF THIS CERTIFICATE AUTHORITY …
3. BECAUSE MY BROWSER VENDOR TRUSTS THE AUTHORITY'S AUDITS …
4. AND I TRUST MY BROWSER VENDOR'S JUDGEMENT.

WHAT IS A PUBLIC KEY INFRASTRUCTURE?

1. MY BROWSER TRUSTS THIS CONNECTION ...
2. BECAUSE IT TRUSTS THE SIGNATURE OF THIS CERTIFICATE AUTHORITY ...
3. BECAUSE MY BROWSER VENDOR TRUSTS THE AUTHORITY'S AUDITS ...
4. AND I TRUST MY BROWSER VENDOR'S JUDGEMENT.

WHAT IS A PUBLIC KEY INFRASTRUCTURE?

1. MY BROWSER TRUSTS THIS CONNECTION …
2. BECAUSE IT TRUSTS THE SIGNATURE OF THIS CERTIFICATE AUTHORITY …
3. BECAUSE MY BROWSER VENDOR TRUSTS THE AUTHORITY'S AUDITS …
4. AND I TRUST MY BROWSER VENDOR'S JUDGEMENT.

WEB PKI: GRANDIOSE BEGINNINGS

THE "DIRECTORY" WILL SAVE US

A BRIEF HISTORY OF THE WEB PKI

- 1988-1990 - X.500 shall be everything and X.509 shall define certificates
- 1995 - Netscape creates SSL, shoehorns in X.509 as the guiding format
- 1995 to ~1998 - Certificate Authorities that pay Netscape get added Navigator's trust anchors
- Aug 2000 - AICPA and CICA publish first WebTrust standard

A BRIEF HISTORY OF THE WEB PKI

- 2001 - Microsoft requires WebTrust for all included CAs
- March 2004 - Mozilla CA Certificate Policy public drafts begin, first public audit reviews
- Nov 2005 - First Mozilla CA Certificate Policy published, referencing WebTrust, ETSI, and ANSI X9.79-1
- 2005 - CA/Browser Forum formed, partly to homogenize Microsoft and Mozilla audits

WEB PKI AFTER 2010: BATTEN DOWN THE HATCHES

WAIT, BEING A CA IS NOT JUST A MONEY-PRINTING PRESS?

WEB PKI IN THE 2010S

- Sept 2011 - DigiNotar gehackt!
- Nov 2011 - CA/Browser Forum's First Baseline Requirements published
- 2012 - Breaking SHA1 signatures becomes feasible
- Dec 2012 - Turktrust MITM intermediate
- March 2013 - First Certificate Transparency log goes online
- Dec 2013 - ANSSI MITM intermediate
- 2014 - Common CA Database, Mozilla CA audits begin, Heartbleed
- January 2015 - Certificate Transparency required for EV in Chrome
- 2016 - SHA1 retired from signatures in the Web PKI
- April 2018 - Certificate Transparency required for all certs in Chrome
- Sept 2019 - Chrome removes EV information from the nav bar

WHAT DOESN'T SUCK ABOUT THE WEB PKI

IT WORKS SO WELL ...
FOR HOW BROKEN IT IS

WEB PKI STUFF WHAT **AIN'T** BROKE

- Certificate Transparency
  - Public Auditing
- Tightened Validation Mechanisms
- Increased Agility
  - Shorter certificate lifetimes
  - Automation
- Certificate Authority Distrust (Revocation)

WHAT ARE SOME OF THE BROKEN THINGS ABOUT THE WEB PKI?

ACTUALLY, JUST TELL US ONE THING, WE DON'T WANT TO BE STUCK HERE ALL NIGHT, IT'S FRIDAY AND WE HAVE THINGS TO DO

WEB PKI STUFF THAT'S **BROKEN**

- End-entity revocation (OCSP)
  - "End-entity" means the lowest certificate in the trust chain. E.g., the website's certificate.
- …and other stuff not relevant to this talk… but … please don't get me started right now or we won't get where we need to go

ONLINE CERTIFICATE STATUS PROTOCOL (OCSP)

What's the status of Certificate X?

Valid! (Also, this response is good for 7 days)

WHAT'S WRONG WITH REVOKING CERTS?

OH NO HE'S GETTING STARTED

ars TECHNICA

BIZ & IT TECH SCIENCE POLICY CARS GAMING & CULTURE S

HTTPSUCKS —

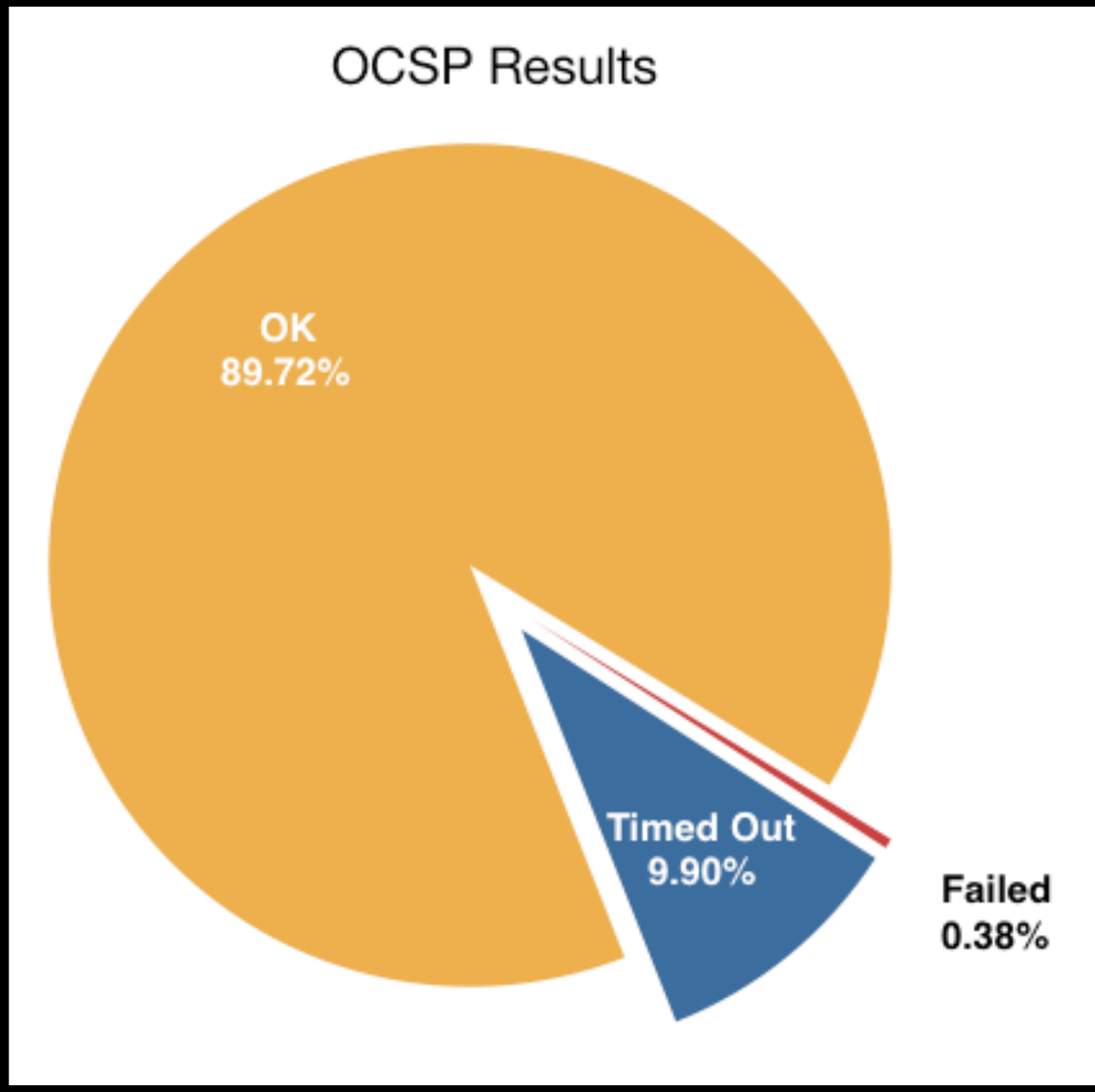
HTTPS Certificate Revocation is broken, and it's time for some new tools

Certificate Transparency and OCSP Must-Staple can't get here fast enough.

SCOTT HELME - 7/3/2017, 5:00 AM

STATE OF THE ART

- If you're not an important brand on the Internet, certificate revocation does nothing.
  - Only Firefox tries to check every site, and 10% of checks time out (which keeps Firefox slower than Safari or Chrome, ow!)
  - OCSP lifetimes mean that even a revoked cert can appear valid for up to additional 7 days.



BROWSERS

- Browsers check “stapled” **OCSP** responses, ***if*** servers provide them.
- Chromium browsers check **CRLSets**.
- Apple’s **Valid** system checks **OCSP** if they get a Bloom filter hit, and fails open.
- Firefox checks **CRLite** and falls back to **OCSP**, which fails open.

THINGS THAT AREN'T BROWSERS

- Command line utilities (curl, etc) often do not check **anything**.
- And that's pretty representative, smh. No one wants their stuff to break because OCSP was flakey.

REVOCATION REDUX: CRLITE

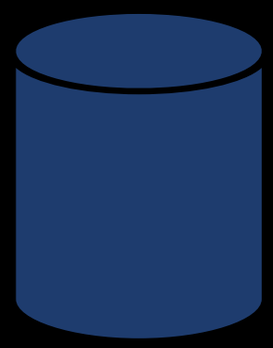
ALL THE BENEFITS OF NOT CHECKING REVOCATION WHILE ALSO CHECKING REVOCATION.

WHAT'S NOT TO LOVE?

CRLITE

- Step 1: Gather all revoked-but-unexpired certificates on the Web from Certificate Revocation Lists (CRLs).
- Step 2: Gather all known unexpired certificates on the web from Certificate Transparency (an oracle).
- Step 3: For each CA, derive the sets UNEXPIRED_VALID and UNEXPIRED_REVOKED
- Step 4: Losslessly compress the sets with cascading Bloom filters.
- Step 5: Ship it regularly.

BACK-END ARCHITECTURE: CRLS



List

CCADB.ORG

ALL CRLS

Download

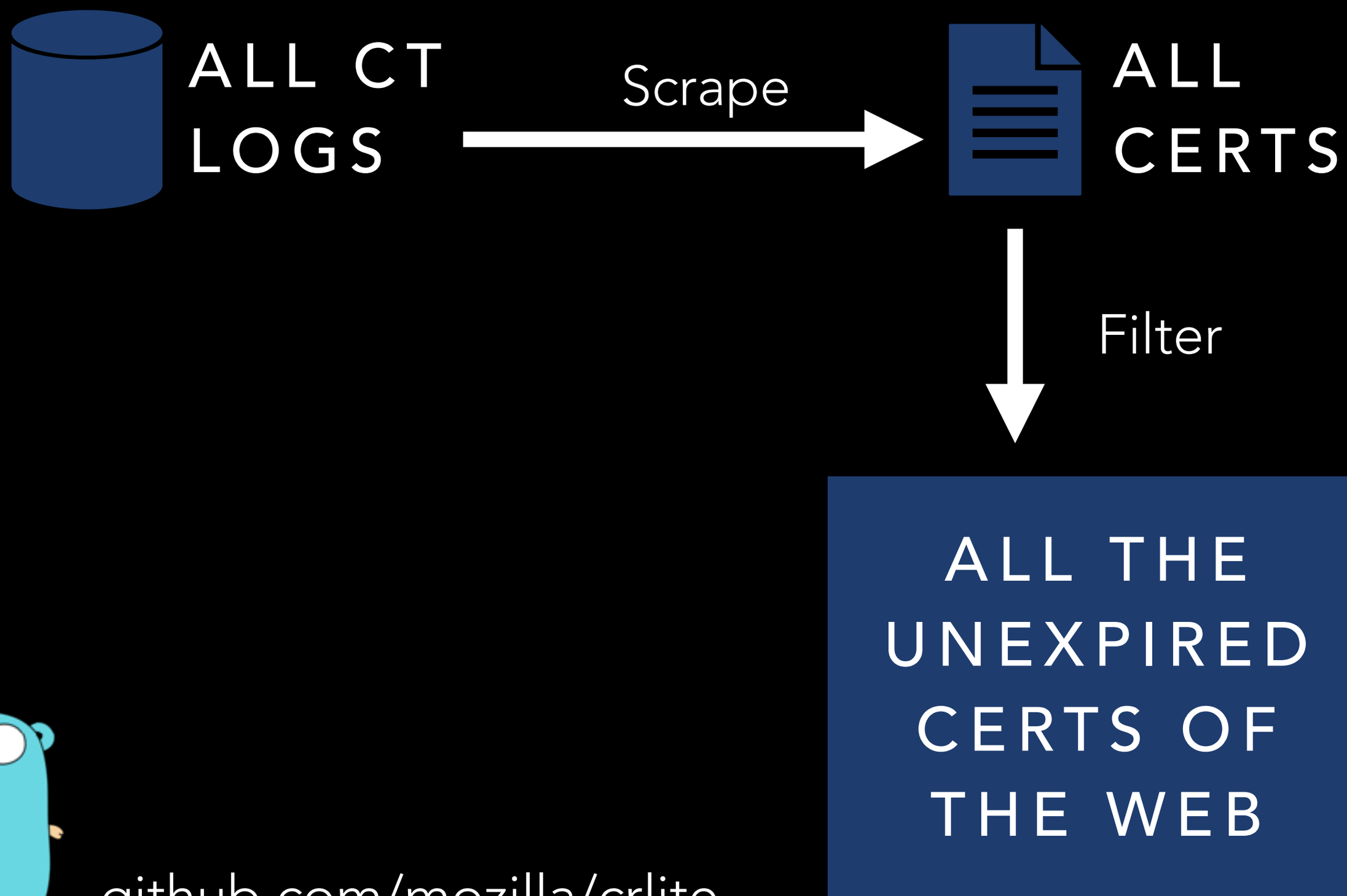
ALL OF THE REVOCATIONS OF THE WEB

github.com/mozilla/crlite

CRLITE

- Step 1: Gather all revoked-but-unexpired certificates on the Web from Certificate Revocation Lists (CRLs).
- Step 2: Gather all known unexpired certificates on the web from Certificate Transparency (an oracle).
- Step 3: For each CA, derive the sets **UNEXPIRED_VALID** and **UNEXPIRED_REVOKED**
- Step 4: Losslessly compress the sets with cascading Bloom filters.
- Step 5: Ship it regularly.

BACK-END ARCHITECTURE: CERTS



github.com/mozilla/crlite

CRLITE

- Step 1: Gather all revoked-but-unexpired certificates on the Web from Certificate Revocation Lists (CRLs).
- Step 2: Gather all known unexpired certificates on the web from Certificate Transparency (an oracle).
- Step 3: For each CA, derive the sets UNEXPIRED_VALID and UNEXPIRED_REVOKED
- Step 4: Losslessly compress the sets with cascading Bloom filters.
- Step 5: Ship it regularly.

BACK-END ARCHITECTURE: SETS

ALL THE UNEXPIRED CERTS OF THE WEB

Intersection

ALL OF THE REVOCATIONS OF THE WEB

UNEXPIRED_REVOKED

UNEXPIRED_VALID

github.com/mozilla/crlite

CRLITE

- Step 1: Gather all revoked-but-unexpired certificates on the Web from Certificate Revocation Lists (CRLs).
- Step 2: Gather all known unexpired certificates on the web from Certificate Transparency (an oracle).
- Step 3: For each CA, derive the sets **UNEXPIRED_VALID** and **UNEXPIRED_REVOKED**
- Step 4: Losslessly compress the sets with cascading Bloom filters.
- Step 5: Ship it regularly.

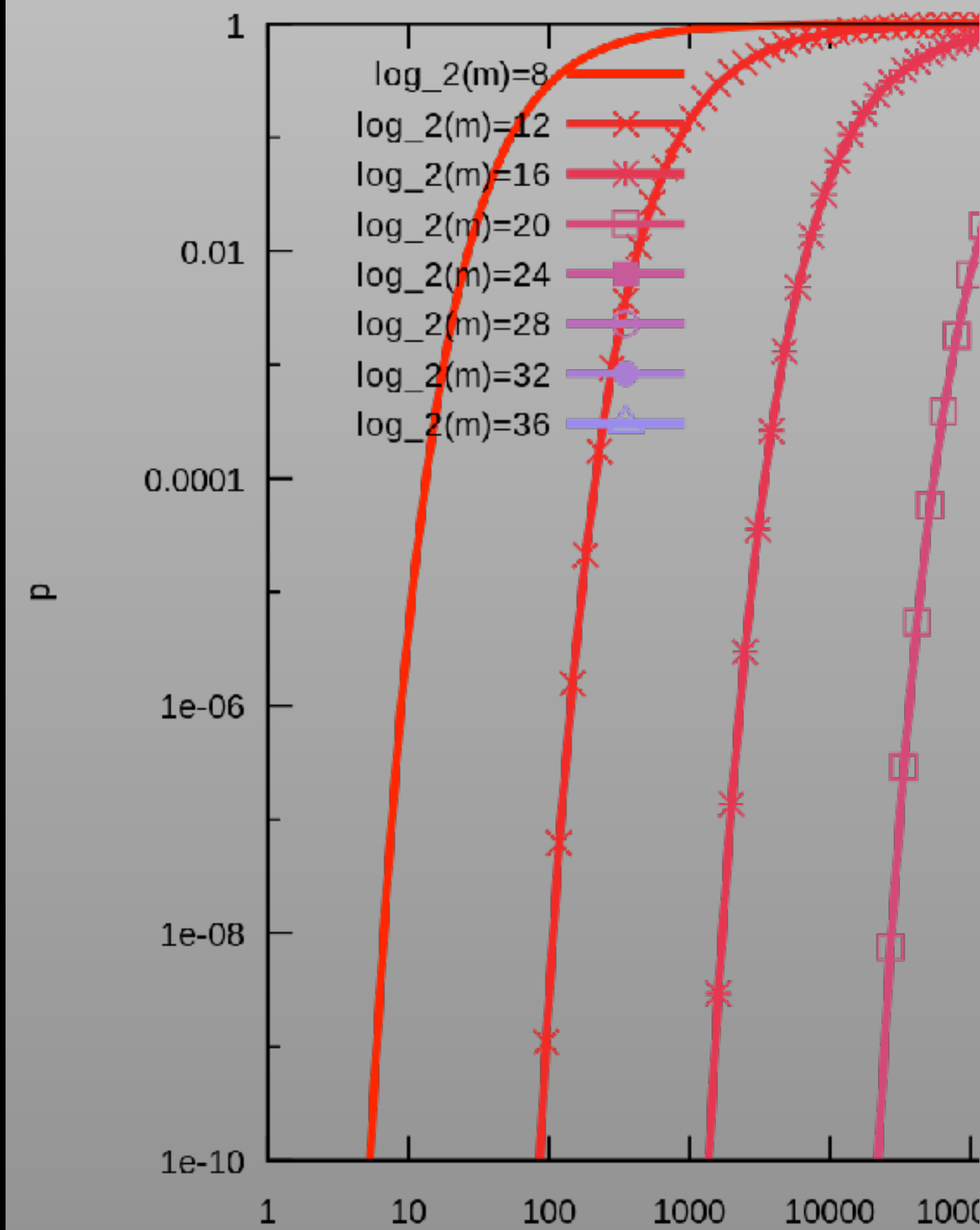
CRLITE AS A 'BLACK BOX'

When presented with a certificate, CRLite has to respond **authoritatively** with one of:

- This certificate is too new
- This certificate is revoked
- This certificate is not-revoked
- This certificate is not part of the CRLite dataset

A QUICK GUIDE TO

BLOOM FILTERS



CONSTRUCTION OF BLOOM FILTERS

1	0	1	0	0	1	1	0

$m = 8$ $k = 4$

To put data **d** into the Bloom filter, we compute k hashes:

hash(d, 0) mod m = *3* → *set bit in index 3 to 1*

hash(d, 1) mod m = *1* → *set bit in index 1 to 1*

hash(d, 2) mod m = *7* → *set bit in index 7 to 1*

hash(d, 3) mod m = *6* → *set bit in index 6 to 1*

HOLD UP

The unspecified things there are all tunable to adjust the **false positive rate**:

- *m:* The number of bits in the filter
- *k:* The number of times an input is hashed
- hash function: has to vary with the input and with *k*
  - `hash(input || hash_iteration[0..k])`
  - CRLite paper used MurmurHash3, Firefox uses SHA2-256

CONSTRUCTION OF BLOOM FILTERS

1	0	1	0	0	1	1	1

$m = 8$ $k = 4$

To put data **d′** into the Bloom filter, we compute k hashes again:

hash(**d′**, 0) mod $m = 1$ → *set bit in index 1 to 1*

hash(**d′**, 1) mod $m = 1$ → *set bit in index 1 to 1*

hash(**d′**, 2) mod $m = 8$ → *set bit in index 8 to 1*

hash(**d′**, 3) mod $m = 7$ → *set bit in index 7 to 1*

SOMETIMES THIS IS CALLED
"TRAINING A BLOOM FILTER"

CHECKING THE FILTER

CHECKING A BLOOM FILTER

To see if data **c′** may be in the Bloom filter, we compute *k* hashes for it:

hash(**c′**, 0) mod *m* = *1* → *check bit in index 1 is 1*

hash(**c′**, 1) mod *m* = *3* → *check bit in index 3 is 1*

hash(**c′**, 2) mod *m* = *2* → *check bit in index 2 is 1*

hash(**c′**, 3) mod *m* = *8* → *check bit in index 8 is 1*

c′ is <u>definitely not</u> in the filter.

CHECKING A BLOOM FILTER

To see if data **c** may be in the Bloom filter, we compute *k* hashes for it:

hash(**c**, 0) mod *m* = *3* → *check bit in index 3 is 1*

hash(**c**, 1) mod *m* = *8* → *check bit in index 8 is 1*

hash(**c**, 2) mod *m* = *3* → *check bit in index 3 is 1*

hash(**c**, 3) mod *m* = *7* → *check bit in index 7 is 1*

c <u>might</u> be in the filter.

"MIGHT BE"

WHILE SUS, IS THE NATURE OF PROBABILISTIC DATA STRUCTURES

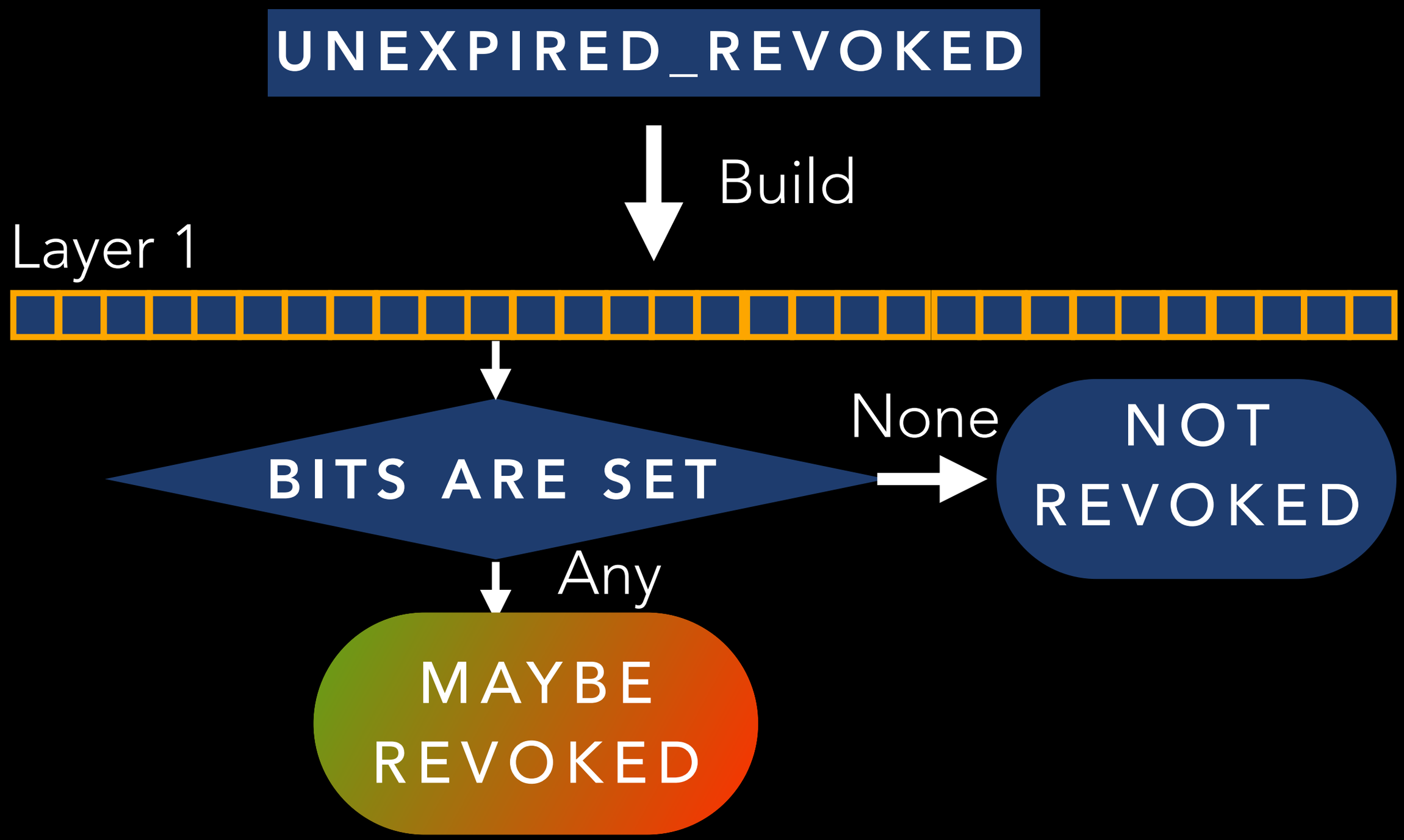
If we know **all** of the inputs, then we can change "might be" to "absolutely is."

THE "BLOOM FILTER CASCADE"

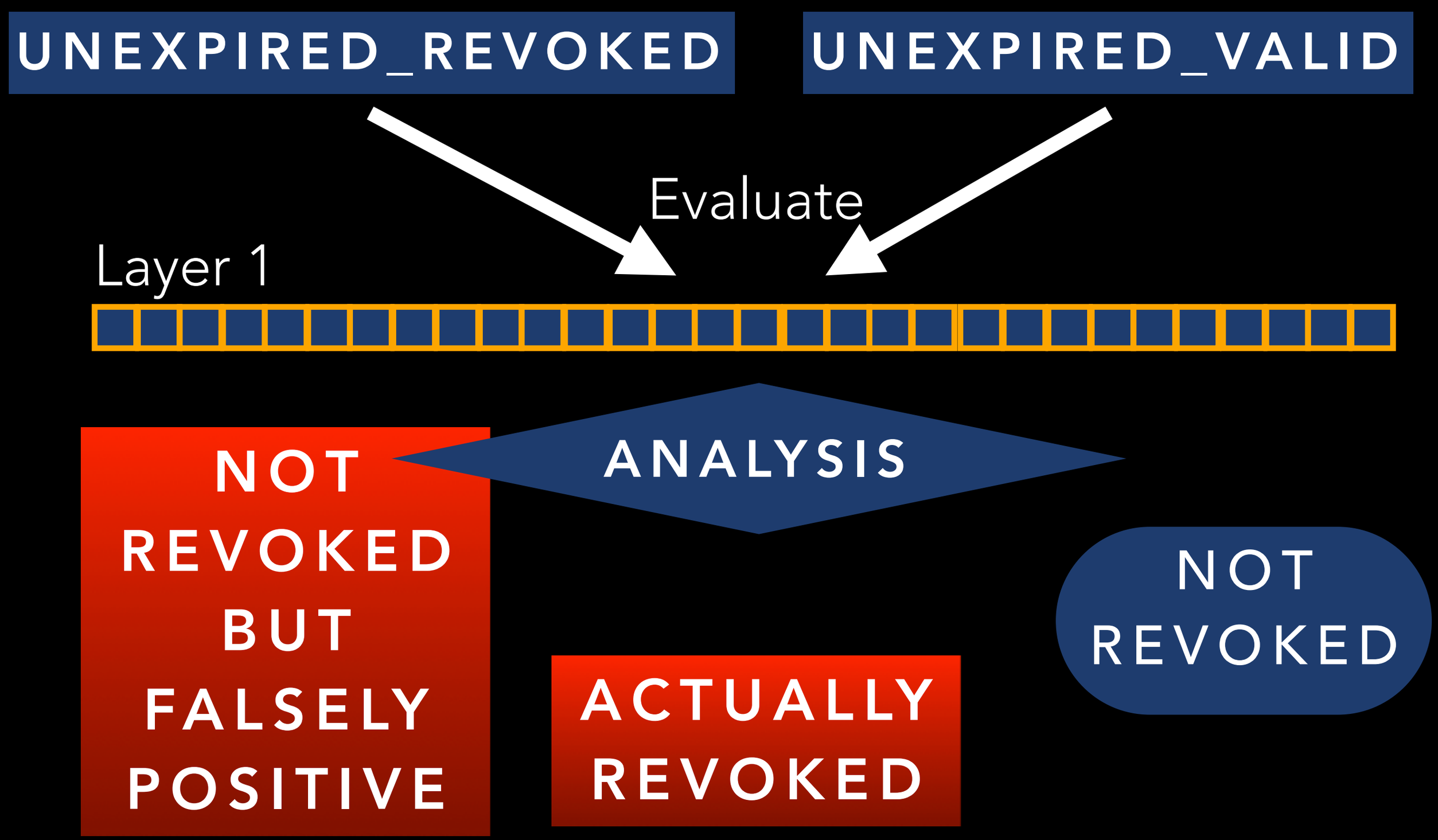
LOSSLESS BLOOM FILTER COMPRESSION

- Using your oracle, evaluate **every unexpired certificate in the Web PKI.**
  - From those evaluations, collect the set of all false positives.
    - Construct **another filter layer, inverting its logic.**
      - And **repeat** until there are **no false positives**.

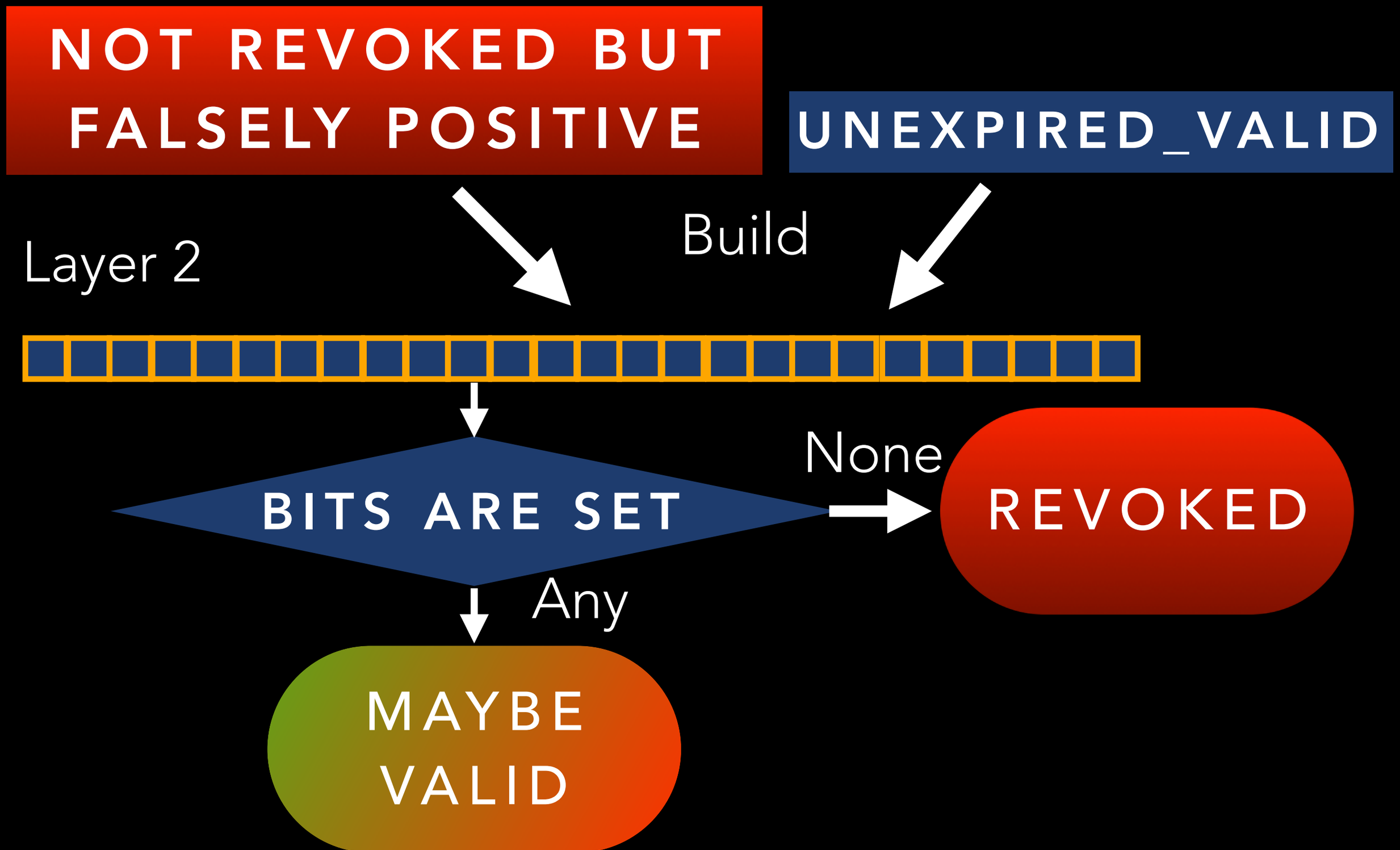
LOSSLESS BLOOM FILTER COMPRESSION



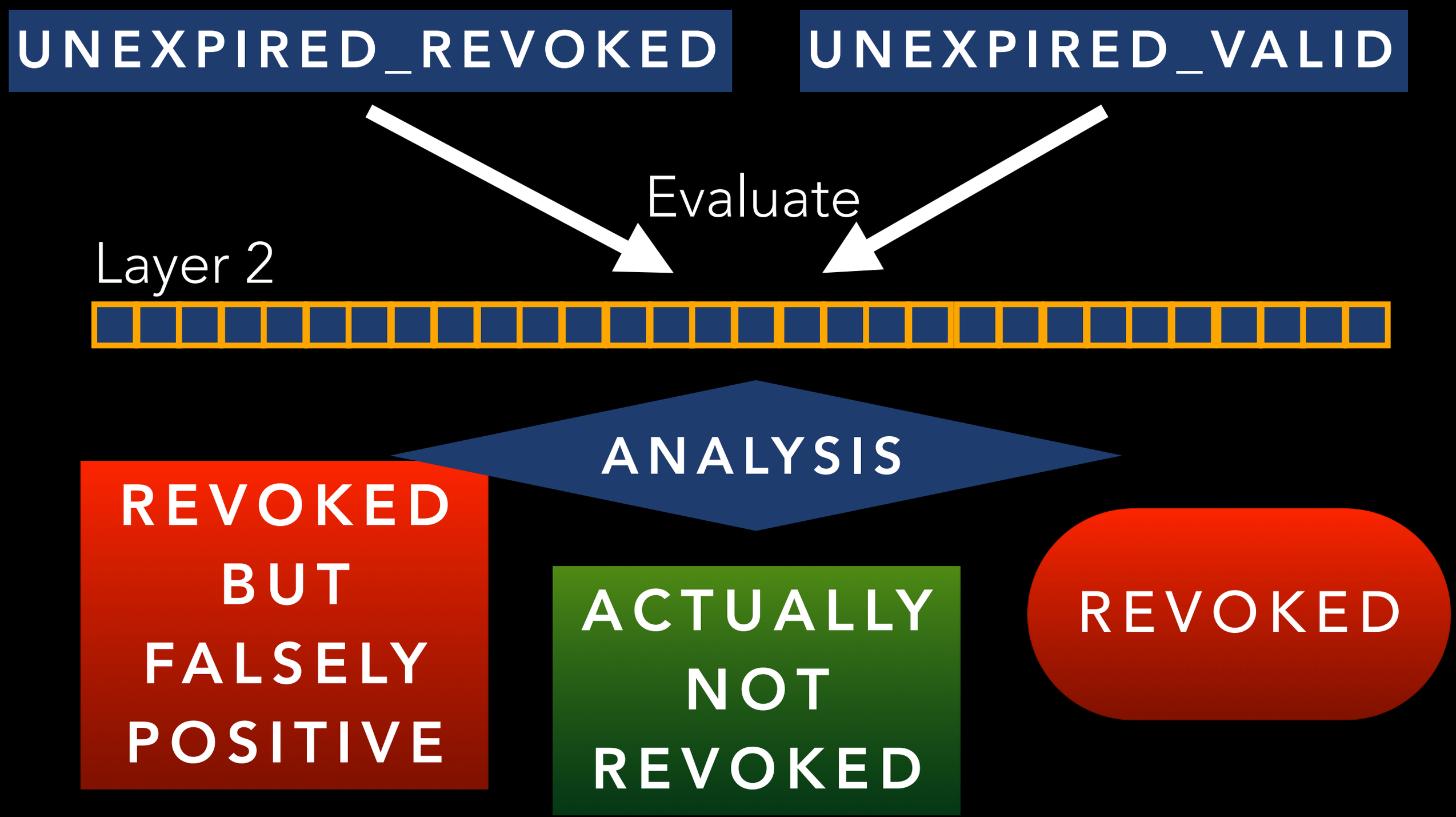
LOSSLESS BLOOM FILTER COMPRESSION



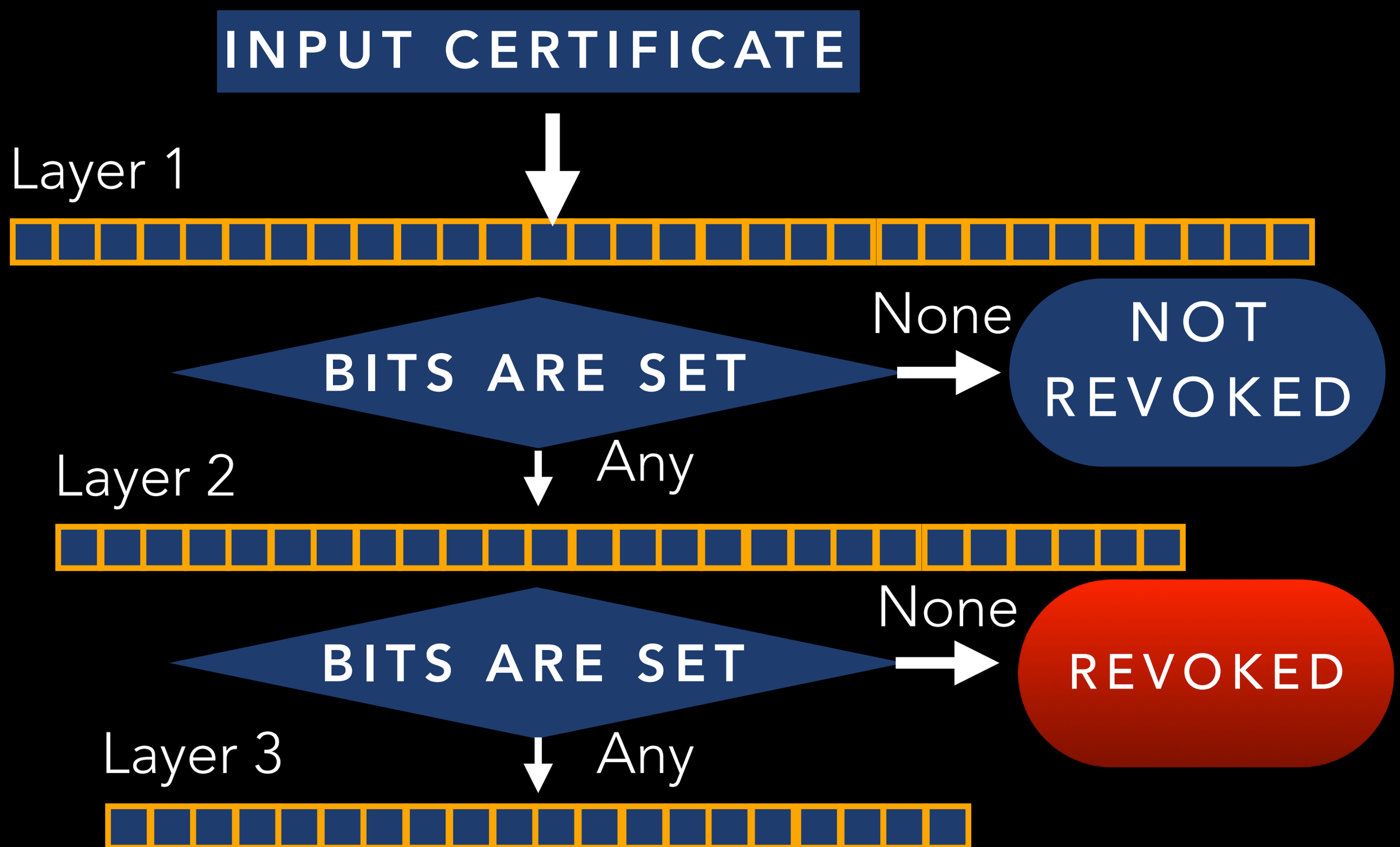
LOSSLESS BLOOM FILTER COMPRESSION



LOSSLESS BLOOM FILTER COMPRESSION



LOSSLESS BLOOM FILTER COMPRESSION



BUT WE STILL NEED TO KNOW
EVERY CERTIFICATE ON THE WEB

WE NEED AN ORACLE

CERTIFICATE TRANSPARENCY
GIVES US OUR ORACLE

CERTIFICATE TRANSPARENCY

- Append-only Merkle trees
- Decentralized, run by different browser vendors and CAs
- In aggregate, contain a copy of every website certificate issued in the Web PKI
- Publicly searchable. See crt.sh/ and censys.io/

CERTIFICATE TRANSPARENCY

- All Certificates in the Web PKI must:
  - Be disclosed to multiple **Certificate Transparency** (CT) logs before issuance (as "precertificates").
  - Embed in the final certificate multiple **Signed Certificate Timestamps (SCT)**, which prove *submission at a certain time to the signing log*.

CERTIFICATE TRANSPARENCY

- Using knowledge of each **SCT's** log's **Maximum Merge Delay:**
  - When examining a certificate, you know when it appeared in CT. Thus, *whether or not it is included in the Filter.*

WHAT'S THE DECISION TREE?

CRLITE CHECK FLOWCHART

Is this certificate's issuer enrolled in CRLite?

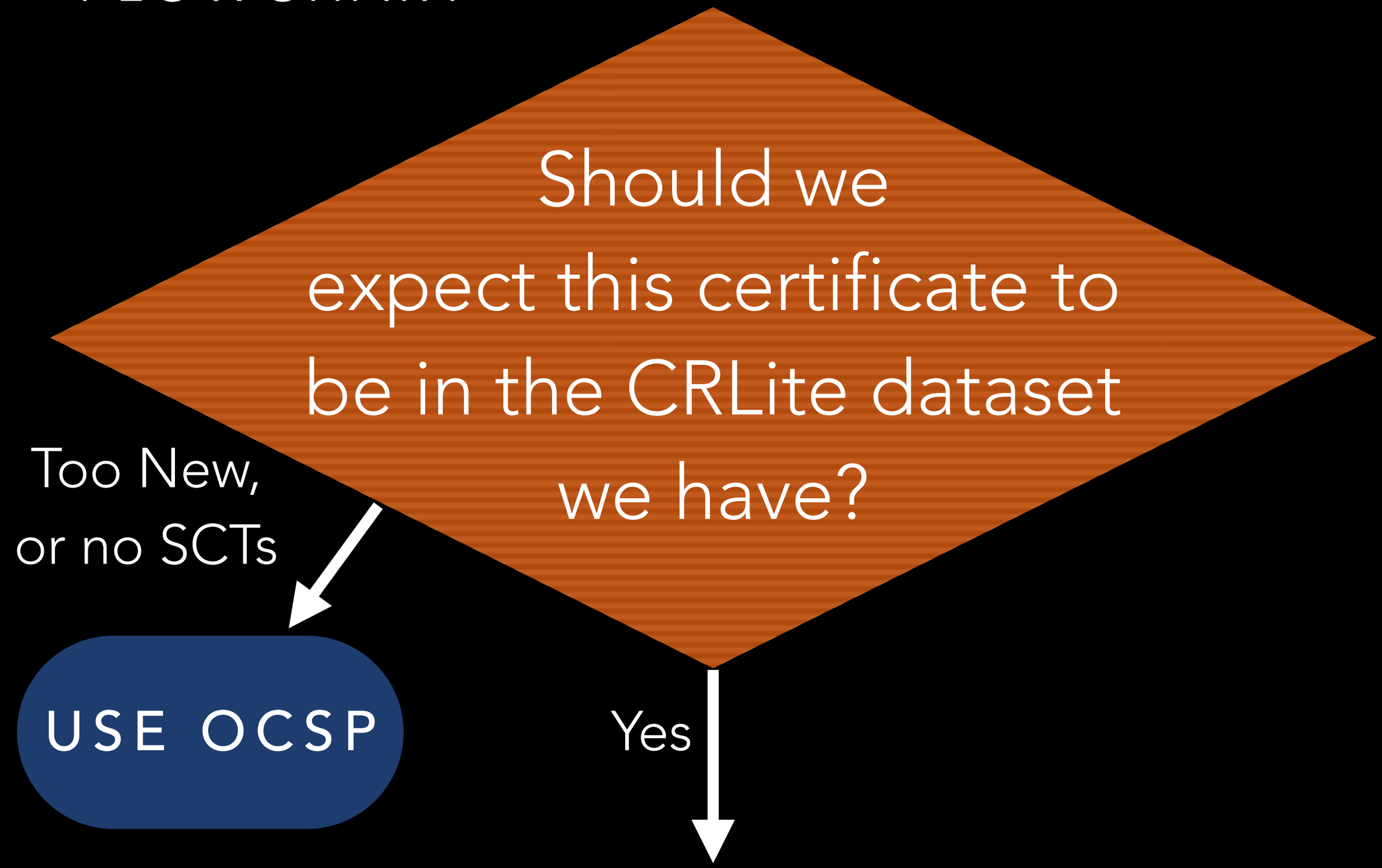
No

USE OCSP

Yes

AS OF 1 OCTOBER 2022, THAT'S THE WHOLE WEB PKI!

CRLITE CHECK FLOWCHART



COMPARE TIMES BETWEEN **SCT** AND **CRLITE**

CERTIFICATE

SCT SIGNATURE:
28 SEPTEMBER AT 23:44 UTC

CERTIFICATE

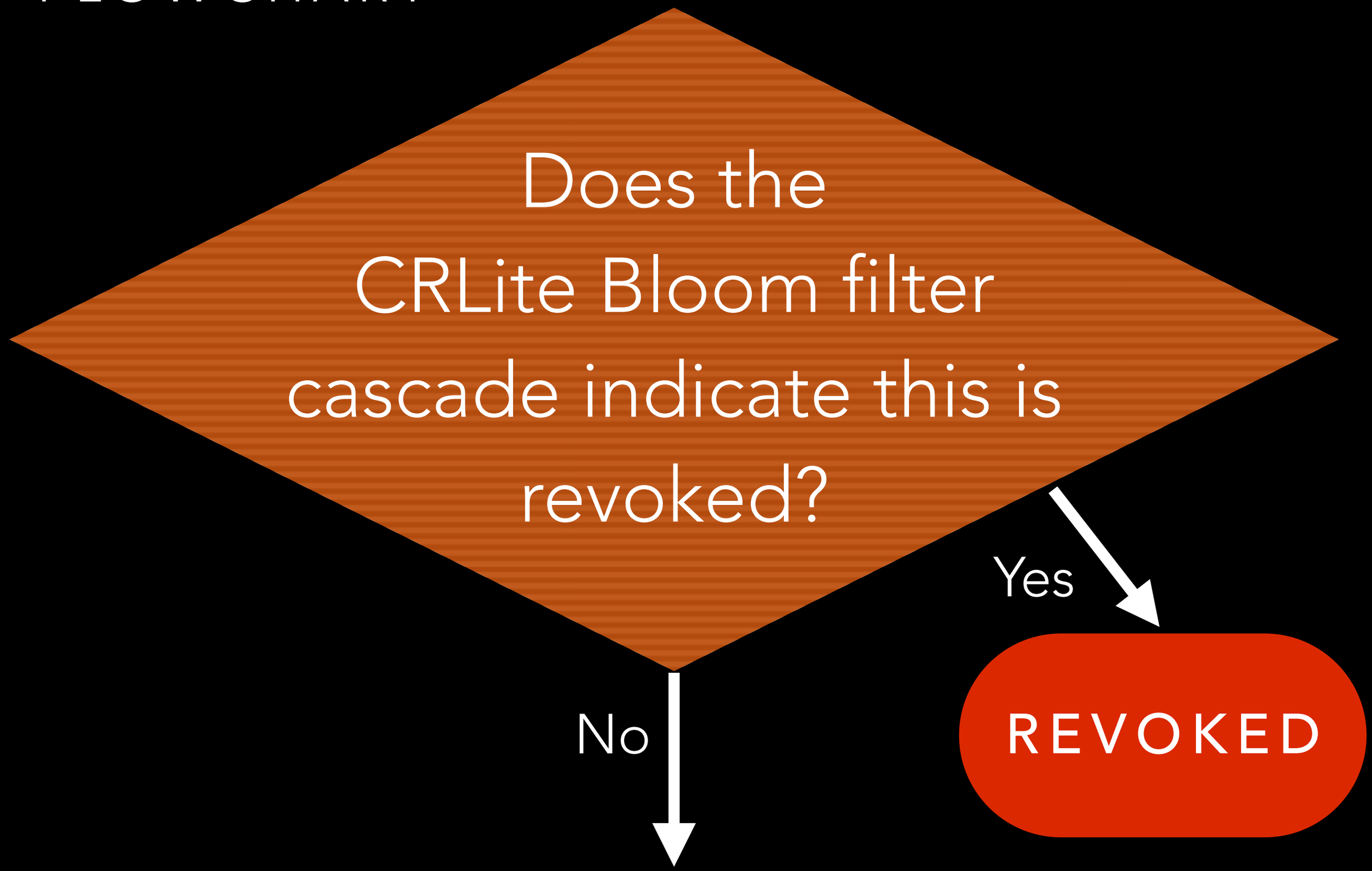
SCT SIGNATURE:
14 OCTOBER AT 03:19 UTC

CRLITE DATA

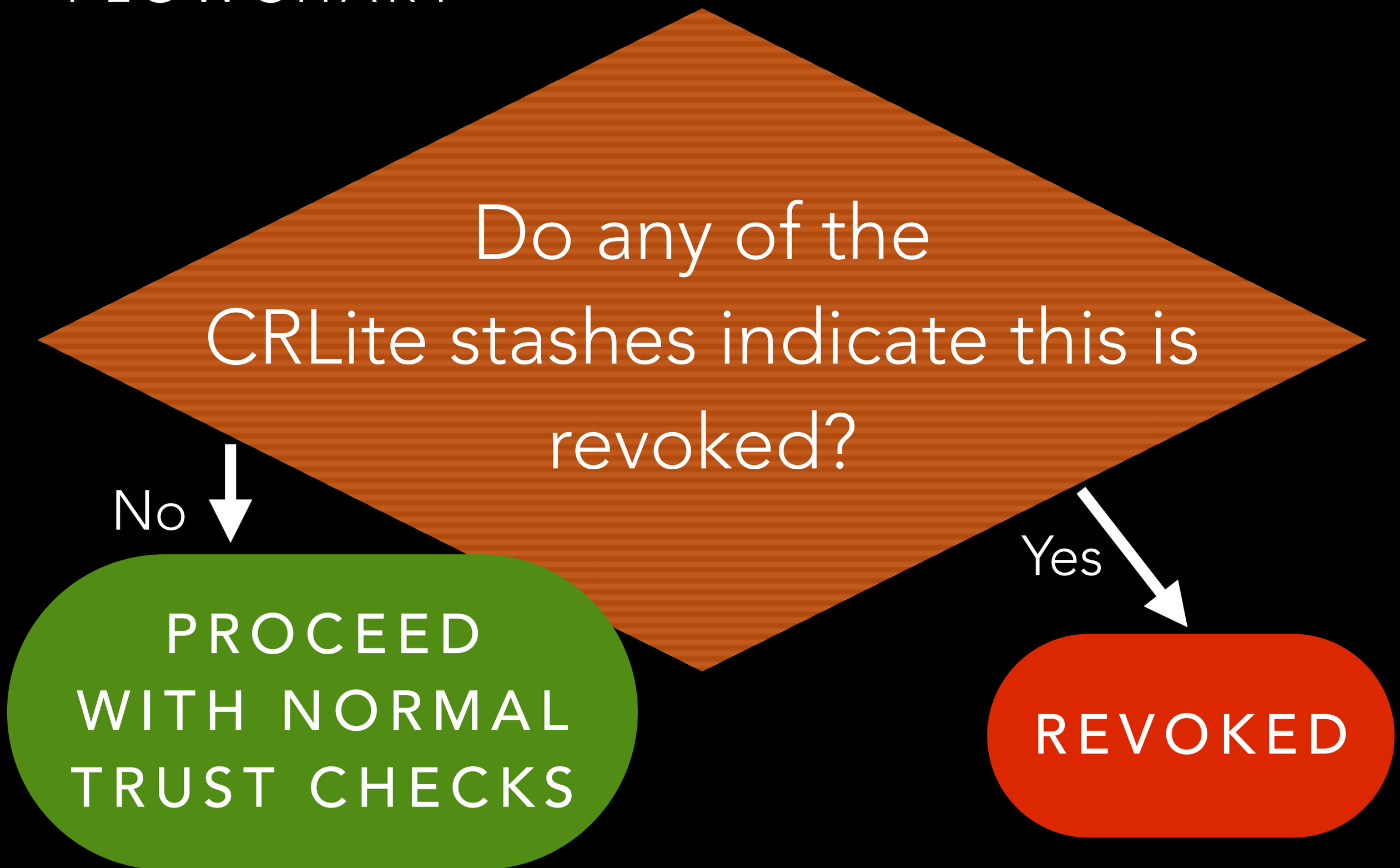
FILTER:
2 OCTOBER 00:29 UTC

LATEST STASH:
14 OCTOBER AT 00:19 UTC

CRLITE CHECK FLOWCHART



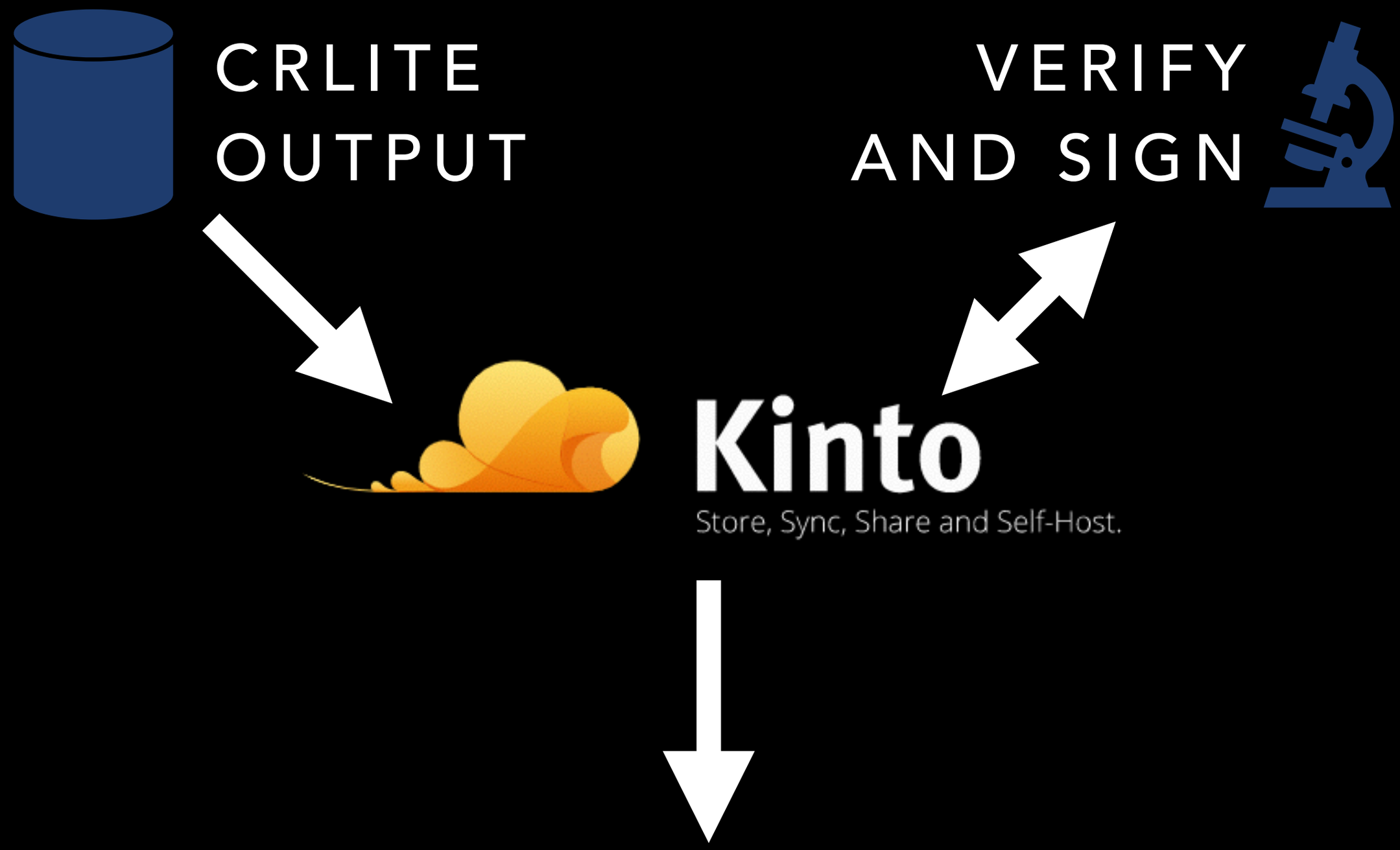
CRLITE CHECK FLOWCHART



CRLITE

- Step 1: Gather all revoked-but-unexpired certificates on the Web from Certificate Revocation Lists (CRLs).
- Step 2: Gather all known unexpired certificates on the web from Certificate Transparency (an oracle).
- Step 3: For each CA, derive the sets UNEXPIRED_VALID and UNEXPIRED_REVOKED
- Step 4: Losslessly compress the sets with cascading Bloom filters.
- Step 5: Ship it regularly.

DISTRIBUTION ARCHITECTURE

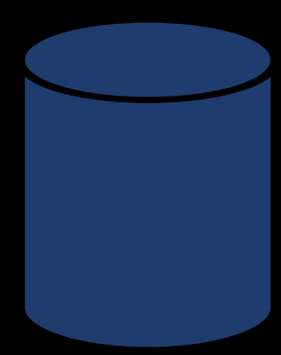


CONTENT DELIVERY NETWORK

BROWSER-SIDE UPDATE ARCHITECTURE

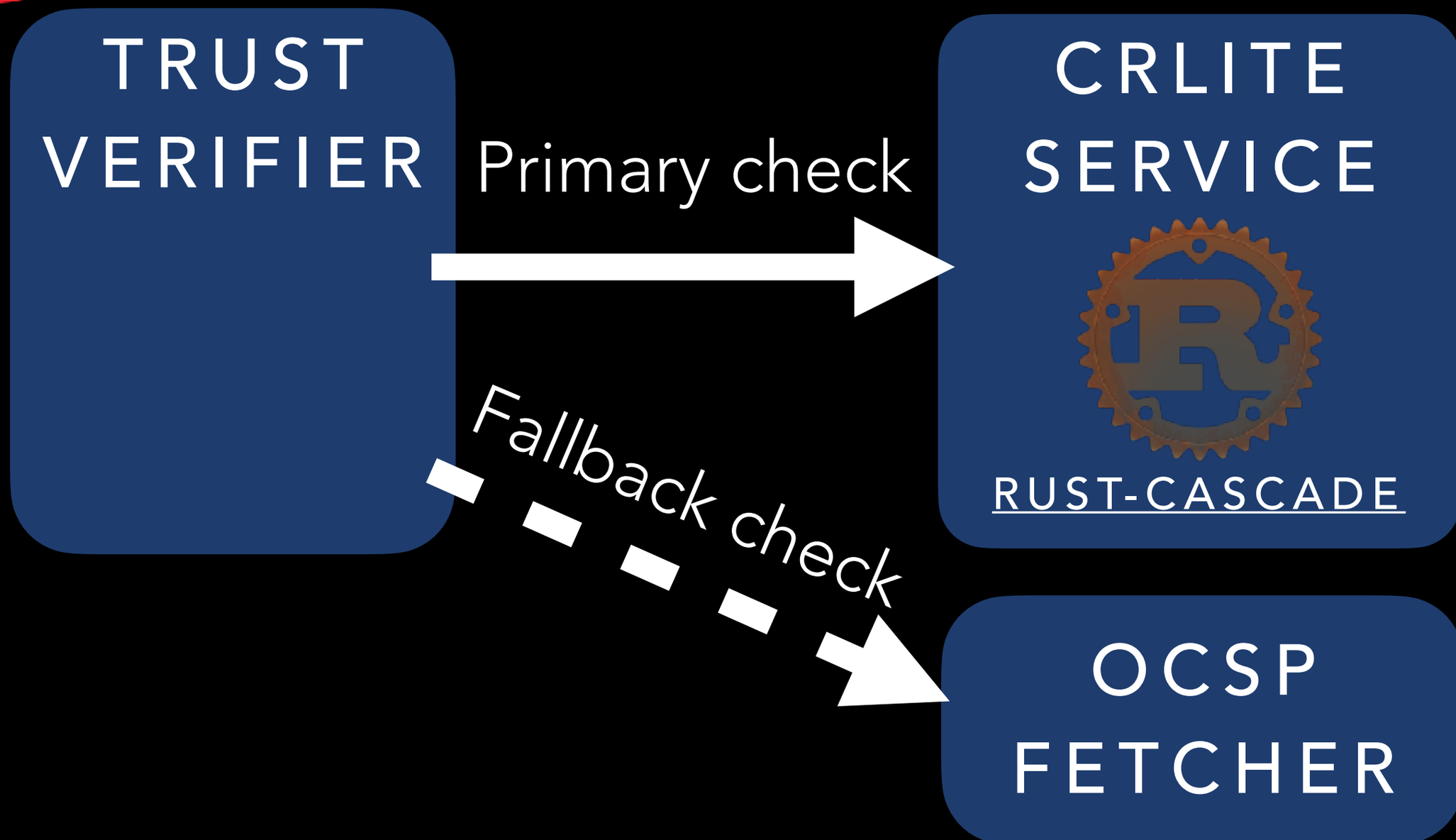
PUSH:
UPDATES AVAILABLE

FILTER AND
STASH UPDATES



FIREFOX
PROFILE
DIRECTORY

BROWSER-SIDE VERIFY ARCHITECTURE



IT'S THAT TIME

FREQUENTLY ASKED QUESTIONS

Doesn't OCSP Stapling solve this problem?

What about command line utilities? Curl and whatnot?

Use of OS-level SecureTransport / Schannel SSP

It'd be super cool to keep CRLite data as a system resource, huh?

Why not XOR Filters or any of the fancier PDSes?

How big are the filters and the updates?

FILTER, UPDATED ~7X/YEAR	6 MB
STASH, UPDATED 4X/DAY	5-40 KB

~150 kB a day is a Lot!

…But less than OCSP for more than half of Firefox for Android users

CRLITE FREQUENTLY ASKED QUESTIONS

TYPICAL OCSP FOR AN RSA CERT FROM LET'S ENCRYPT

	HTTP HEADERS	OCSP DATA	
REQUEST	110 BYTES (ACCEPT-ENCODING, HOST, CONNECTION)	118 BYTES (BASE64)	
RESPONSE	386 BYTES (CACHE TAGS, EXPIRY TIME, CONTENT-LENGTH, ...)	543 BYTES (BINARY)	
TOTAL	496 BYTES	661 BYTES	**1157 BYTES**

CRLITE FREQUENTLY ASKED QUESTIONS

OCSP UPDATES	CRLITE UPDATES
1.1 KB/EA	~ 80-200 KB/DAY

SINCE MOST WEB USERS VISIT ~100 ORIGINS/DAY...

CRLite is within the same order of magnitude amount of data.

Can I look at the data?

https://github.com/mozilla/crlite/wiki

#where-can-i-get-the-crlite-data-that-is-used-to-make-filters

RECENT RUNS

Google Cloud Select a project

Search Products, resources, docs (/)

Cloud Storage

Buckets

Monitoring

Settings

Bucket details

REFR

crlite-filters-prod

OBJECTS CONFIGURATION PERMISSIONS PROTECTION LIFECYCLE

Buckets > crlite-filters-prod

UPLOAD FILES UPLOAD FOLDER CREATE FOLDER TRANSFER DATA MANAGE HOLDS DOWNLOAD DELETE

Filter by name prefix only Filter Filter objects and folders

Name	Size	Type	Created	Storage class	Last modified	Public access
20220831-3/	—	Folder	—	—	—	—
20220901-0/	—	Folder	—	—	—	—
20220901-1/	—	Folder	—	—	—	—
20220901-2/	—	Folder	—	—	—	—
20220901-3/	—	Folder	—	—	—	—
20220902-0/	—	Folder	—	—	—	—
20220902-1/	—	Folder	—	—	—	—
20220902-2/	—	Folder	—	—	—	—
20220902-3/	—	Folder	—	—	—	—
20220903-0/	—	Folder	—	—	—	—

A SINGLE RUN

crlite-filters-prod

OBJECTS | CONFIGURATION | PERMISSIONS | PROTECTION | LIFECYCLE

Buckets > crlite-filters-prod > 20220930-3

UPLOAD FILES | UPLOAD FOLDER | CREATE FOLDER | TRANSFER DATA | MANAGE HOLDS | DOWNLOAD | DELETE

Filter by name prefix only | Filter | Filter objects and folders | Show deleted data

Name	Size	Type	Created	Storage class	Last modified	Public access		Version his
completed	46 B	text/plain	Sep 30, 20...	Standard	Sep 30, 20...	Value hidden	Copy URL	–
crl-audit.json	3.4 MB	application/json	Sep 30, 20...	Standard	Sep 30, 20...	Value hidden	Copy URL	–
crls.tar.gz	232.4 MB	application/x-tar	Sep 30, 20...	Standard	Sep 30, 20...	Value hidden	Copy URL	–
ct-logs.json	7.2 KB	application/json	Sep 30, 20...	Standard	Sep 30, 20...	Value hidden	Copy URL	–
enrolled.json	3.2 MB	application/json	Sep 30, 20...	Standard	Sep 30, 20...	Value hidden	Copy URL	–
known/	–	Folder	–	–	–	–		–
log/	–	Folder	–	–	–	–		–
mlbf/	–	Folder	–	–	–	–		–
revoked/	–	Folder	–	–	–	–		–
timestamp	19 B	application/octet-stream	Sep 30, 20...	Standard	Sep 30, 20...	Value hidden	Copy URL	–

FILTER AND STASH

crlite-filters-prod

OBJECTS | CONFIGURATION | PERMISSIONS | PROTECTION | LIFECYCLE

Buckets > crlite-filters-prod > 20220930-3 > mlbf

UPLOAD FILES | UPLOAD FOLDER | CREATE FOLDER | TRANSFER DATA | MANAGE HOLDS

Filter by name prefix only | Filter | Filter objects and folders

Name	Size	Type	Created
filter	6.1 MB	application/octet-stream	Sep 30, 20...
filter.stash	20.2 KB	application/octet-stream	Sep 30, 20...
revset.bin	221.2 MB	application/octet-stream	Sep 30, 20...

Why does revocation matter at all?

Are CRLite updates subject to denial of service?

Isn't Disclosure to Certificate Transparency (SCTs) optional?

THANKS TO THE ORIGINAL CRLITE ACADEMIC TEAM

James Larisch, Northeastern University

David Choffnes, Northeastern University

Alan Mislove, Northeastern University

Christo Wilson, Northeastern University

Dave Levin, University of Maryland

Bruce M. Maggs, Duke University and Akamai Technologies

I HOPE WE'RE DOING Q&A BY NOW

SERIOUSLY HOW DID I GET TO THIS SLIDE

BETTER START TALKING ABOUT CRAZY STUFF

This has been a lot of words from

J.C. JONES

- Cryptography Engineer & SRE @ Let's Encrypt

BUT NOT HERE ON THEIR BEHALF

- Formerly: Mozilla Crypto Engineering

@CIPHERCOFFEE

JC@INSUFFICIENT.COFFEE
LETSENCRYPT.ORG

HTTPS://INSUFFICIENT.COFFEE/

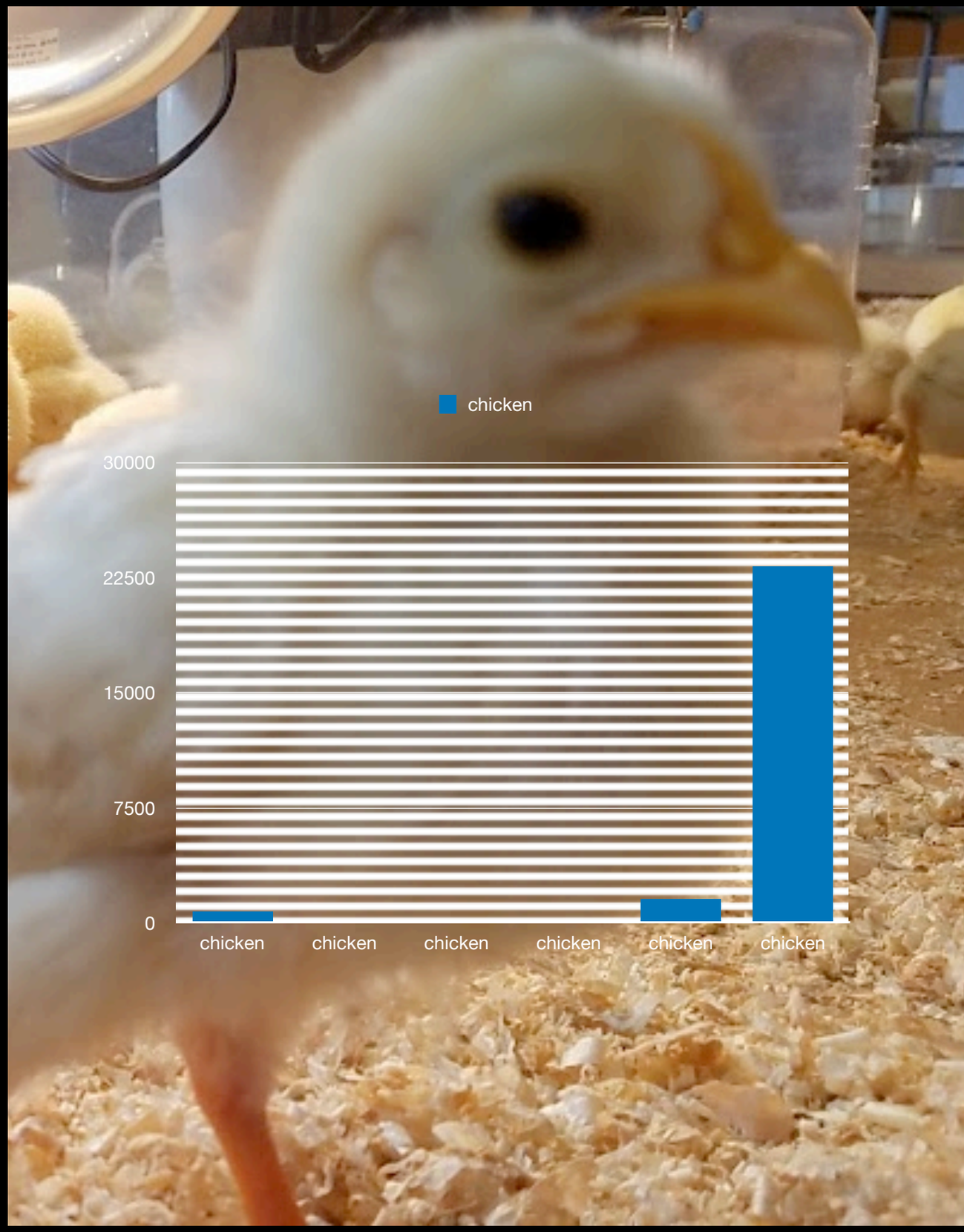
CHICKEN CHICKEN CHICKEN

CHICKEN CHICKEN

CHICKEN CHICKEN

chicken

	chicken	chicken	chicken
chicken	chicken	chicken	chicken
chicken	chicken	chicken	chicken
chicken	chicken	chicken	chicken
chicken	chicken	chicken	chicken
chicken	chicken	chicken	chicken
chicken	chicken	chicken	chicken
chicken	chicken	chicken	chicken
chicken	chicken	chicken	chicken
chicken	chicken	chicken	chicken
chicken	chicken	chicken	chicken
chicken	chicken	chicken	chicken
chicken	chicken	chicken	chicken
chicken	chicken	chicken	chicken
chicken	chicken	chicken	chicken
chicken	chicken	chicken	chicken
chicken	chicken	chicken	chicken
chicken	chicken	chicken	chicken
chicken	chicken	chicken	chicken
chicken	chicken	chicken	chicken
chicken	chicken	chicken	chicken



SPKI

SUBJECT POULTRY KEY INFORMATION