namecoin

Namecoin and Tor as a Public Key Infrastructure

Jeremy Rand
Lead Application Engineer, The Namecoin Project
https://www.namecoin.org/

PGP: 1D04 FB9D 50BF 2A8E 9F3E 58AD DC7E 7F8A E30E 73E6

Presented at FOSDEM 2026
Decentralized Internet and Privacy Devroom

What is TLS?

- Transport Layer Security.
  - Encryption and authentication for TCP services.
  - It’s the “S” in “HTTPS”.
- Based on the 1990’s era Netscape protocol SSL.
  - Occasionally people still incorrectly call it SSL.
- Uses certificate authorities (CA’s) to establish trust.

Public CA’s are a Censorship Risk

- Centralized, trusted 3rd parties.
- Sci-Hub has had their certs revoked (copyright lawsuit).
- Let’s Encrypt says they revoke (censor) about one website certificate per month due to U.S. sanctions compliance reasons.

Public CA’s are a Wiretapping Risk

- DigiNotar CA compromised in 2011, Iranian users wiretapped.
- Moxie Marlinspike 2013 leak:
  - Saudi government intended to legally compel a public CA (Saudi or UAE) to wiretap users.
- Saudi intelligence paid Twitter mole to wiretap me in 2014.
  - Likely because my TLS research threatened their CA-related efforts.
  - Trump DoJ declined to prosecute Wiretap Act violation.

Inspiration from DNSSEC

- In the DNS world, there are TLSA records.
- You look up a TLSA record for a domain name…
  - You get back a public key.
- This public key can be used as a domain-specific CA to authenticate TLS certs for that domain.
- Effectively trades trusting public CA’s for trusting DNS registrars + registries + ICANN.

Self-Authenticating Domain Names?

- Domain names inherently tied to cryptographic material.
  - Authenticating doesn’t involve a third party.

Self-Authenticating Domain Names? Tor Onion Services

- Anonymously hosted.
- Rather slow.
- `2gzyxa5ihm7nsggfxnu52rck2vv4rvmdlkiu3zzui5du4xyclen53wid.onion`
- Ed25519 pubkey embedded inside domain name.

Self-Authenticating Domain Names? Namecoin

- Like the DNS but on a blockchain.
- Doesn’t require Tor.
- Lookups are about as fast as DNS.
- `federalistpapers.bit`
- Owned by a Namecoin address (same format as a Bitcoin address).

Self-Authenticating Domain Names? Implications

- We could do something similar to TLSA records.
- Avoids public CA’s.
- Also avoids trusting DNS.
- Seems like a win.

Wait, but why onion services with TLS?

- Onions are already encrypted….
- But TLS has better end-to-end security.
  - Onion service encryption is terminated at the Tor daemon.
  - TLS is terminated at the application.

Why does the endpoint matter?

- Tor daemon and application may be in different trust domains.
  - Whonix – Tor Browser runs in one VM; Tor daemon runs in another VM.
  - Web server could also be on Whonix, not just the client.
  - Client and server won’t know whether each other has such a threat model.

Why does the endpoint matter? (2)

- Tor daemon and application may have an insecure network connection to each other.
  - Running Tor daemon and an HTTPS server on different cloud infrastructure IP’s.
  - Using Tor Browser on a laptop with a Tor daemon on a WiFi router.
  - Again, client and server won’t know whether each other has such a threat model.

TLS Helps Compatibility

- Many applications expect TLS to be there.
- Web browsers show scary warnings without TLS.
  - And they restrict features too (e.g. webcam access).
- Giving them TLS is easier than patching them.
  - Tor Browser’s relevant patches are a maintenance nightmare.

How do we get TLS implementations to do this?

- Virtually no TLS implementations support TLSA records.
- If we had a pluggable certificate trust API, we could do something?
- There is no WebExtensions API for pluggable certificate trust.

But there’s another kind of browser add-on

- PKCS#11 modules.
- Well-known: it’s how smartcards and HSM’s are added to browsers.
- Less well-known: browsers like Firefox use PKCS#11 as a database query API for TLS certificates.

Example PKCS#11 modules

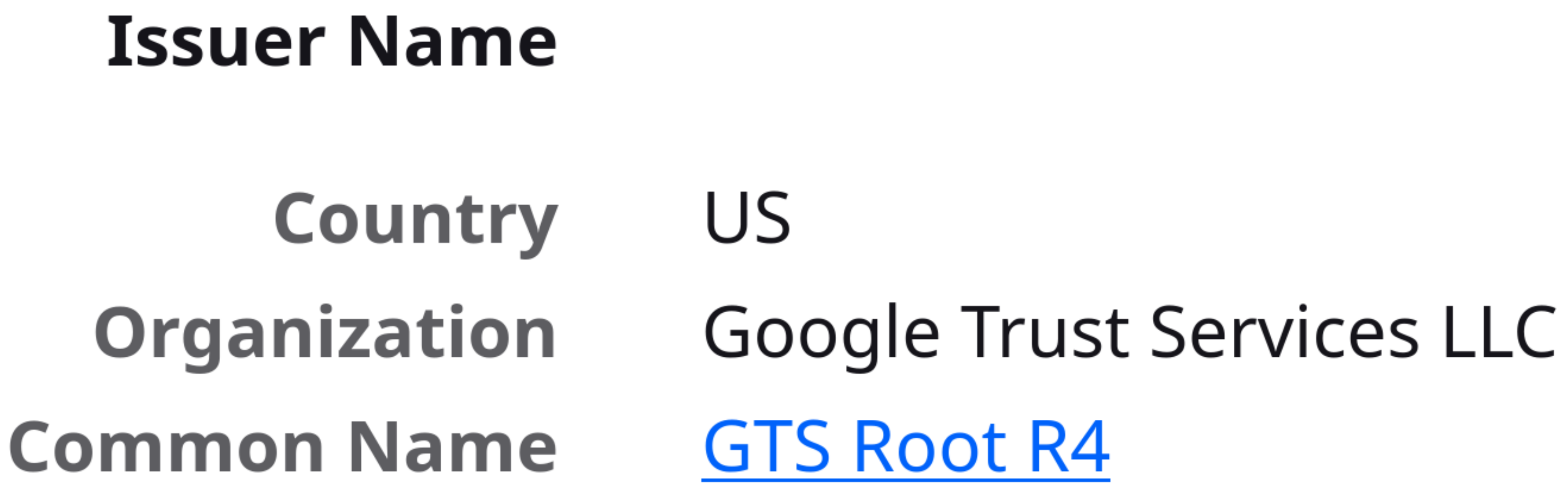
- Mozilla’s built-in root CA list.
  - `libnssckbi.so` – it’s where Let’s Encrypt, Comodo, etc. are.
  - This is a PKCS#11 module!
- SQLite-based trust database in Firefox.
  - “Softoken” – it’s where CA’s are stored that you added in the Firefox UI.
  - This is also a PKCS#11 module!

How do PKCS#11 queries work?

- Firefox sends PKCS#11 module the Issuer Name of a certificate signed by an unknown CA.
- PKCS#11 module responds with the full certificate of that CA.

What’s an Issuer Name?

- A Issuer Name looks like this (in Firefox GUI):



Issuer Name

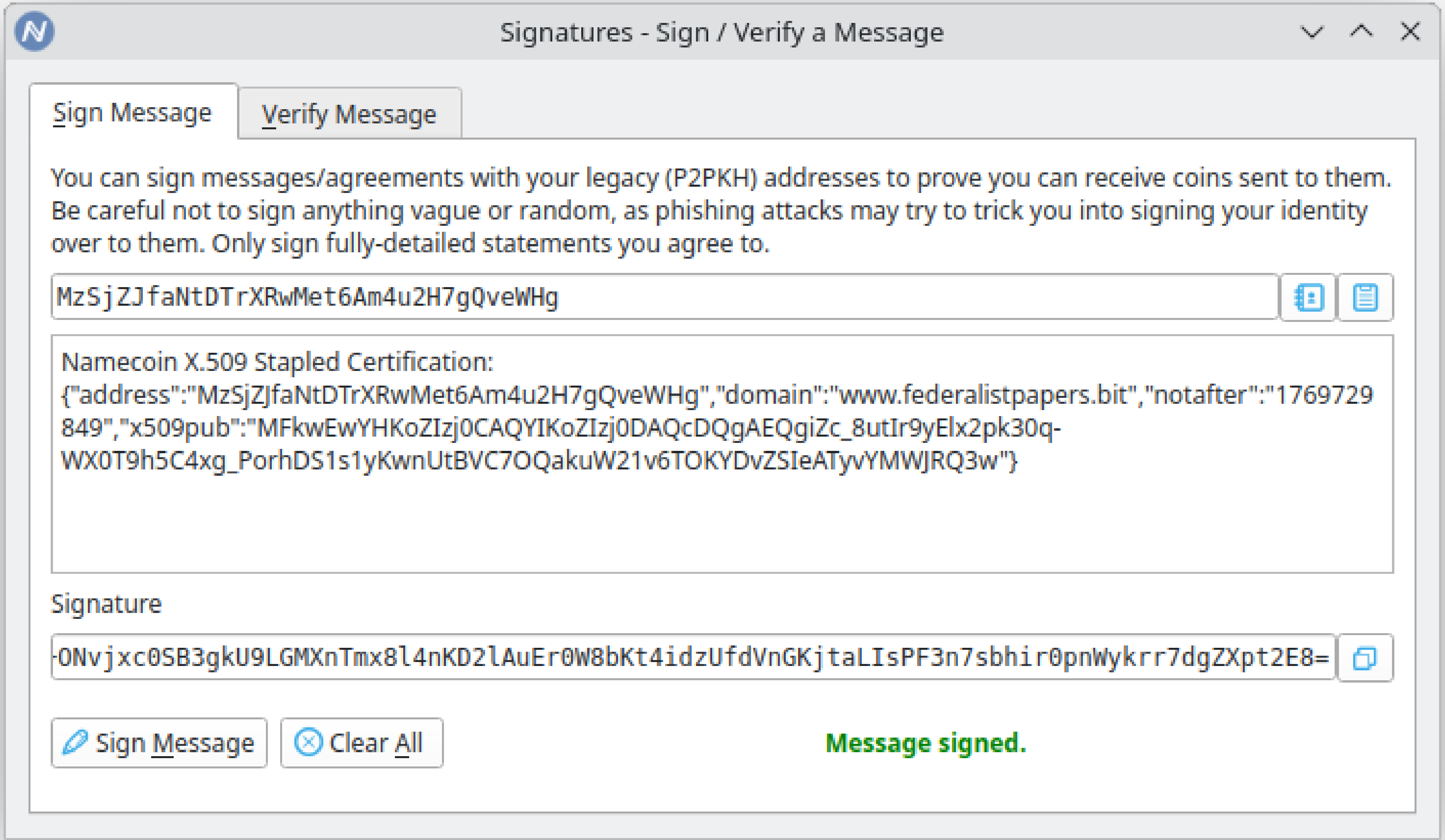
Country	US
Organization	Google Trust Services LLC
Common Name	GTS Root R4

What *isn’t* in an Issuer Name?

- Public key or signature.
  - These are elsewhere: our PKCS#11 module can make them up!
- Domain name that it’s valid for.
  - Instead found in the Name Constraints field: our PKCS#11 module can make this up too.

But how does our module know what to make up?

- What if we put a hint in the Issuer Name of our TLS certificate?
  - Stuff a domain name into the Common Name subfield.
  - Stuff a public key into the Serial Number subfield.
  - Sign the public key with an onion key or a Namecoin address.
    - Stuff that signature into the Serial Number subfield too.


Signatures - Sign / Verify a Message

Sign Message | Verify Message

You can sign messages/agreements with your legacy (P2PKH) addresses to prove you can receive coins sent to them. Be careful not to sign anything vague or random, as phishing attacks may try to trick you into signing your identity over to them. Only sign fully-detailed statements you agree to.

```
MzSjZJfaNtDTrXRwMet6Am4u2H7gQveWHg
```

```
Namecoin X.509 Stapled Certification:
{"address":"MzSjZJfaNtDTrXRwMet6Am4u2H7gQveWHg","domain":"www.federalistpapers.bit","notafter":"1769729849","x509pub":"MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEQgiZc_8utIr9yElx2pk30q-WX0T9h5C4xg_PorhDS1s1yKwnUtBVC7OQakuW21v6TOKYDvZSIeATyvYMWJRQ3w"}
```

Signature

```
-0NvjxcOSB3gkU9LGMXnTmx8l4nKD2lAuEr0W8bKt4idzUfdVnGKjtaLIsPF3n7sbhir0pnWykrr7dgZXpt2E8=
```

Sign Message | Clear All

Message signed.

Issuer Name

Common Name	www.federalistpapers.bit Domain AIA Parent CA
Serial Number	Namecoin TLS Certificate Stapled: {"pubb64":"MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEQgiZc_8utIr9yElx2pk30q-WX0T9h5C4xg_PorhDS1s1yKwnUtBVC7OQakuW21v6TOKYDvZSIeATyvYMWJRQ3w","sigs":"[{\"blockchainaddress\":\"MzSjZJfaNtDTrXRwMet6Am4u2H7gQveWHg\",\"blockchainsig\":\"H+ONvjxc0SB3gkU9LGMXnTmx8l4nKD2lAuEr0W8bKt4idzUfdVnGKjtaLIsPF3n7sbhir0pnWykrr7dgZXpt2E8=\"}]\n"}

Validation Workflow

- PKCS#11 module checks the pubkey+signature in the Issuer Serial Number against the domain name in the Issuer Common Name.
  - Confirms that the pubkey was signed by the owner of the domain name.
- Returns a newly generated CA with the given public key.
  - Name constraint for the given domain name.
  - Cross-signed by a locally trusted root CA.
- TLS client is happy!

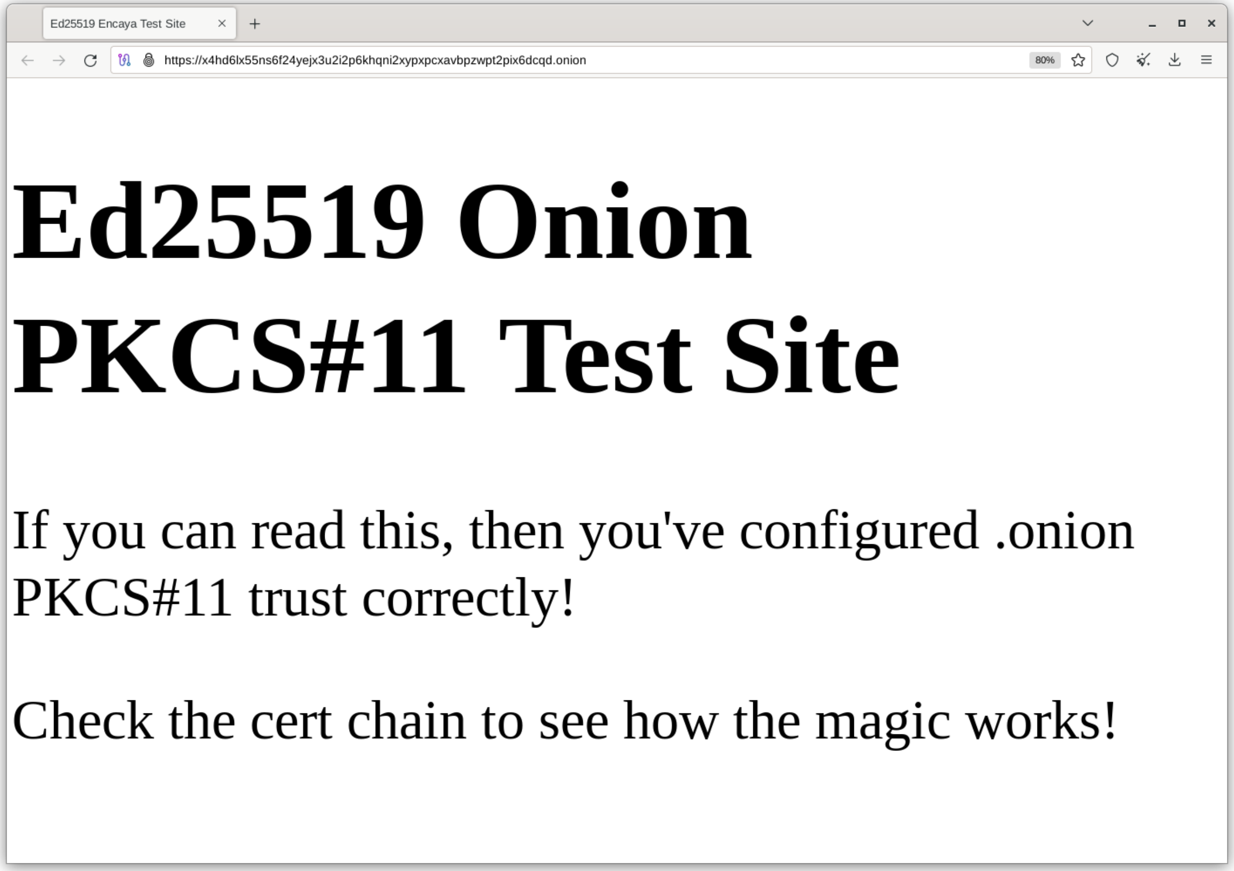
Deployment (Server-Side)

- `ncgencert -host x4hd6lx55ns6f24yejx3u2i2p6khqni2xypxpcxavbpzwpt2pix6dcqd.onion -grandparent-tor-key /var/lib/tor/hidden_service/hs_ed25519_secret_key`
  - Put the resulting `chain.pem` and `key.pem` into Caddy.

Deployment (Client-Side)

- Installing a PKCS#11 module in Firefox is easy.
  - Just go to “Security Devices” settings….
  - Click “Load” and choose the path of the PKCS#11 module….
    - It’s a `.so`, `.dylib` or `.dll` file depending on OS.
  - You’re done. TLS with onion services and Namecoin will work now.

It works in Tor Browser!


Ed25519 Encaya Test Site
https://x4hd6lx55ns6f24yejx3u2i2p6khqni2xypxpcxavbpzwpt2pix6dcqd.onion
80%

Ed25519 Onion PKCS#11 Test Site

If you can read this, then you've configured .onion PKCS#11 trust correctly!

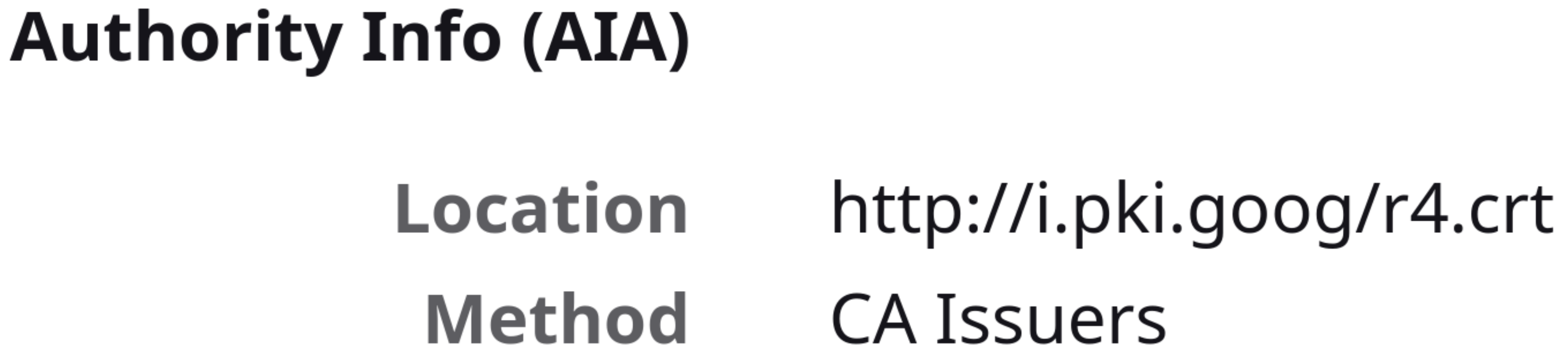
Check the cert chain to see how the magic works!

Compatibility?

- Any application using NSS or GnuTLS knows how to use PKCS#11 for this.
- That covers Firefox and Tor Browser.
- What about Chromium?

Chromium and AIA

- Chromium looks up unknown CA’s via the AIA field rather than the Issuer Name field.
- It’s just a URL where Chromium downloads the missing CA’s certificate.

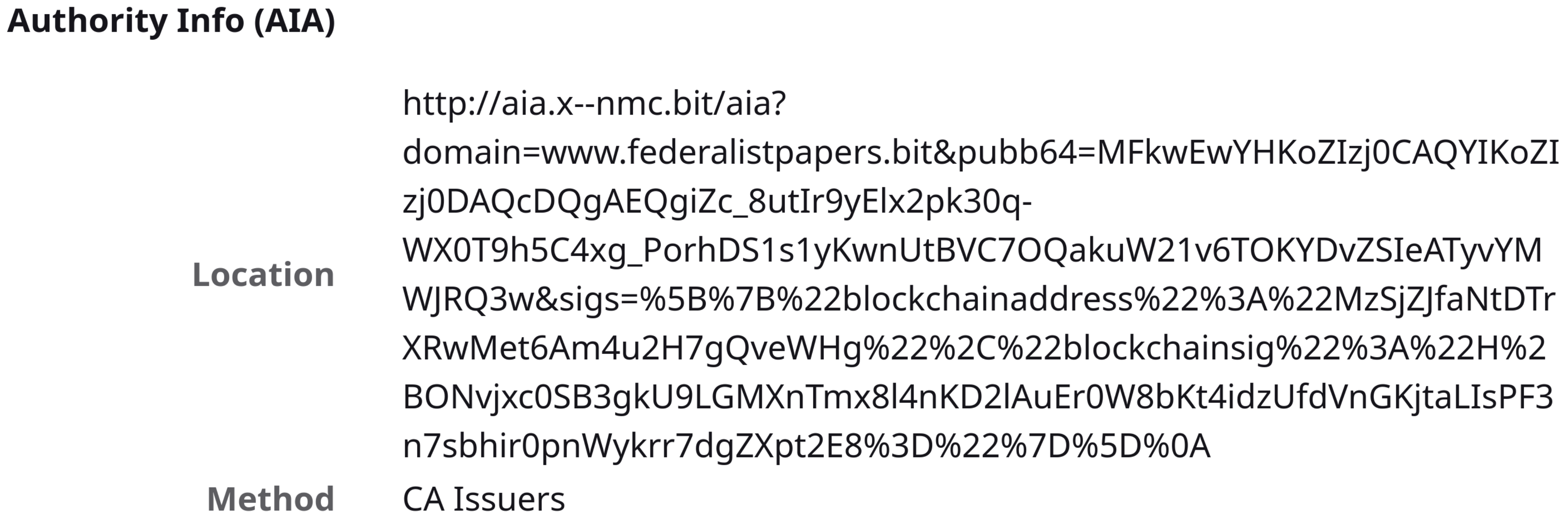


Authority Info (AIA)

Location	http://i.pki.goog/r4.crt
Method	CA Issuers

We can do AIA the same way

- We can run a localhost HTTP daemon that responds to AIA lookups.



Authority Info (AIA)

Location	http://aia.x--nmc.bit/aia?domain=www.federalistpapers.bit&pubb64=MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEQgiZc_8utIr9yElx2pk30q-WX0T9h5C4xg_PorhDS1s1yKwnUtBVC7OQakuW21v6TOKYDvZSIeATyvYMWJRQ3w&sigs=%5B%7B%22blockchainaddress%22%3A%22MzSjZJfaNtDTrXRwMet6Am4u2H7gQveWHg%22%2C%22blockchainsig%22%3A%22H%2BONvjxc0SB3gkU9LGMXnTmx8l4nKD2lAuEr0W8bKt4idzUfdVnGKjtaLIsPF3n7sbhir0pnWykrr7dgZXpt2E8%3D%22%7D%5D%0A
Method	CA Issuers

Compatibility!

- Virtually all TLS implementations support either PKCS#11 or AIA.
  - OpenSSL uses a different API.
  - We don’t support OpenSSL’s API yet, but it’s analogous to PKCS#11 and should not be problematic to support.

Signature Scheme Support

- Tor uses Ed25519.
- Namecoin uses Bitcoin Script (secp256k1 with multisig, timelocks, etc).
- TLS implementations only support ECDSA with NIST curves.
  - Totally fine – the signature is validated by the PKCS#11 module.

Namecoin Edge Cases

- Multiple Namecoin addresses can control a Namecoin domain name simultaneously.
  - They can also control different subdomains.
- So the Issuer Name / AIA URL has to encode which Namecoin address and subdomain was involved.

Scalability Benefits

- Signing a TLS public key doesn't require any broadcasting.
  - Improves scalability.
  - TLS handshake bytes are cheap.
  - Namecoin blockchain storage and onion service DHT storage are much more expensive.
  - This will be magnified once PQ signatures like Dilithium become commonplace.

SSH

- SSH can use SSHFP records from DNS.
  - But that means broadcasting the SSHFP records.
    - Not as scalable as what we do for TLS.
  - Can we keep it inside the SSH handshake like we did for TLS?

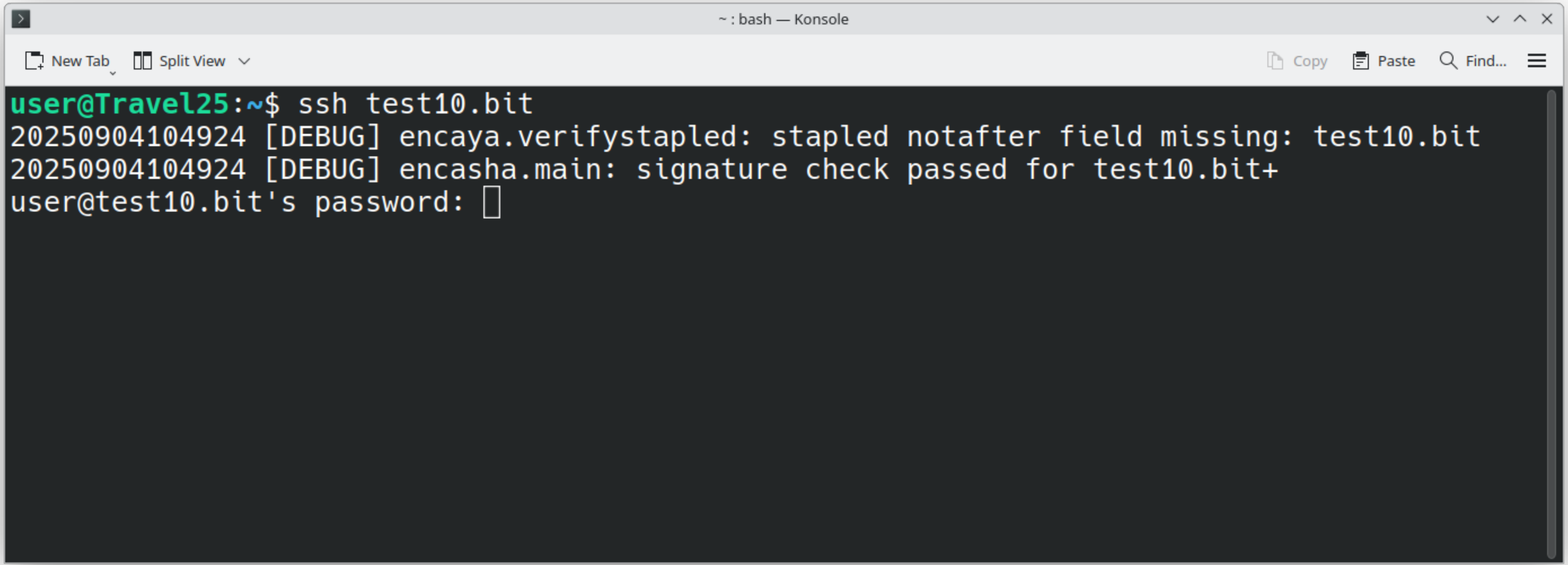
Fun with SSH Host Keys

- SSH has certificates and CA’s.
  - Not used very often, but they’re a thing.
  - Generally used in enterprise environments.
- SSH certificates can have “extensions”.
  - Similar role as the Issuer Name and AIA.
  - We can embed a signature here.

Validating SSH Host Keys

- OpenSSH has the “KnownHostsCommand” option.
  - It runs a custom command, and passes the SSH certificate it received to that command.
  - The command can choose to mark it as trusted.
- So we made such a command.
  - Shares most of the code in common with our PKCS#11 module and AIA daemon.

Validating SSH Host Keys



```
user@Travel25:~$ ssh test10.bit
20250904104924 [DEBUG] encaya.verifystapled: stapled notafter field missing: test10.bit
20250904104924 [DEBUG] encasha.main: signature check passed for test10.bit+
user@test10.bit's password: 
```

Revoking TLS Certificates

- PKCS#11 supports querying for revocation status too.
  - Query is keyed by Issuer Name and Certificate Serial Number.
  - You can put Issuer/Serial pairs in the Namecoin blockchain or the onion descriptor.
- For Chromium, we haven’t tried yet.
  - Might be able to use OCSP like we use AIA?

Revoking SSH Host Keys

- OpenSSH gives us both the CA pubkey and the host pubkey.
  - Easy to check both against the revocation list in the Namecoin blockchain or onion descriptor.

First-Party Isolation

- Tor Browser enforces FPI (“First-Party Isolation”):
  - A browser tab for “forum.example.bit” and a tab for “wiki.example.bit” can share a Tor circuit…
  - But tabs for “forum.example.bit” and “wiki.example2.bit” must use mutually segregated Tor circuits.
  - Subresources (images, CSS, etc) stay in the circuit of their parent webpage.
  - Same rules apply to browser state in those tabs (e.g. cookies).
- This prevents tracking you across sites.

First-Party Isolation with PKCS#11 and AIA

- PKCS#11 and AIA have no mechanism to determine the first-party domain associated with a certificate lookup request.
  - Which prevents us from isolating Namecoin or onion descriptor lookups on different Tor circuits.
- So how do we support anonymity?

First-Party Isolation (Solution)

- Before the TLS/SSH handshake, Tor’s control port API notifies that a connection to that hostname is about to happen.
  - FPI *is* available there (thanks Tor devs for merging our patches!).
- We can do the Namecoin or onion descriptor lookup at that time, and cache:
  - The domain.
  - (For Namecoin) The list of Namecoin addresses who control the domain.
  - Any revocations in the blockchain or onion descriptor.

First-Party Isolation (Solution) (2)

- PKCS#11 module / AIA server / SSH command can just query that cache.
  - No network traffic.
  - Essentially zero latency overhead.
- The cache isn’t isolated, *but* there’s no state in it that can be observed by a website.

Thanks to funders

- NLnet Foundation’s Internet Hardening Fund / Netherlands Ministry of Economic Affairs and Climate Policy
- NLnet Foundation’s Next Generation Internet Zero Core Fund / European Commission
- Power Up Privacy
- Cyphrs

Contact me at...

- https://www.namecoin.org/
- OpenPGP (after FOSDEM):
  1D04 FB9D 50BF 2A8E 9F3E 58AD DC7E 7F8A E30E 73E6
- jeremyrand@danwin1210.de (after FOSDEM)
- byronlelah@airmail.cc (travel during FOSDEM – no PGP)
- Or just find me in person at FOSDEM. (I’m wearing a Namecoin shirt.)