Marcus Dansarie

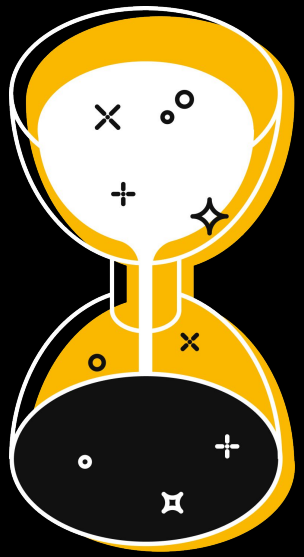
Roughtime: Securing time for IoT devices

Correct time is important

- Many security-critical applications require accurate time

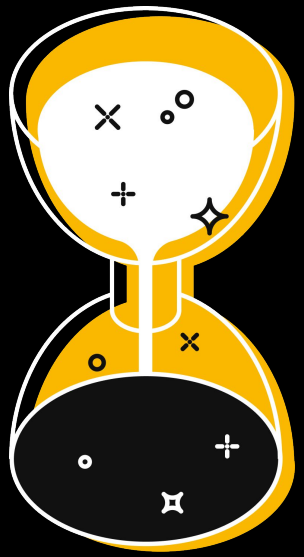
- DNSSEC – enables secure DNS lookups
- TLS – the foundation of many other protocols
  - HTTPS – everything on the web
  - SMTPS, IMAPS, POP3S – secure mail
  - …
- Authentication tokens and two-factor authentication
- Cyber-physical systems
- Logs

- Accuracy requirements vary: sub-second to hours



Consequences of not having correct time

- **Loss of confidentiality and integrity**: Accepting expired or revoked certificates and tokens
- **Loss of availability**: Certificate & token validation failures
- **Loss of traceability**: Inaccurate or ambiguous log files

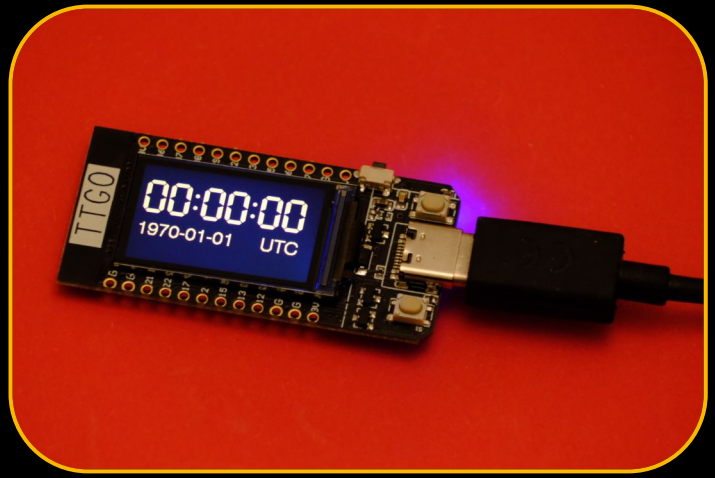


Keeping time

- All devices
  - need time on first use
  - can keep time when powered on

- Real time clocks (RTC)
  - limited accuracy
  - IoT devices may not have an RTC

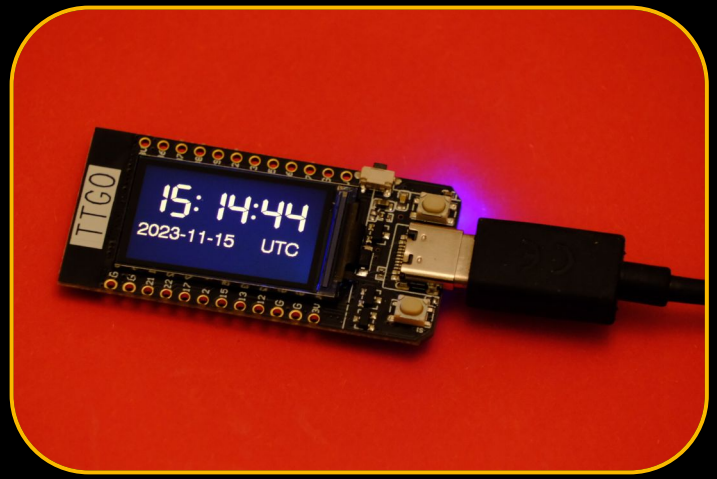
- Conclusion: external time updates are required



Getting time over the network

- **NTP – Network Time Protocol**
  - Often used with a single server (SNTP)
  - Often used without authentication
  - Symmetric authentication
  - Autokey (broken)

- **NTS – Network Time Security**
  - Adds scalable security to NTP
  - Depends on TLS
    - Requires correct time
    - Requires up-to-date CA certificates
  - Low adoption

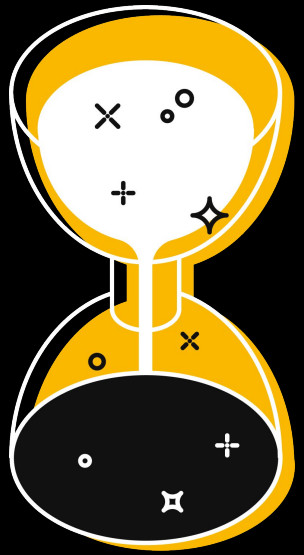


Possible solution: Roughtime

- Protocol is an IETF Draft
  - Watson Ladd (Akamai)
  - Marcus Dansarie (Netnod)

- Started out as a way to verify system time
  - Not intended to replace NTP or NTS

- Netnod received RIPE community funding to help kickstart the development of Roughtime and the IETF draft

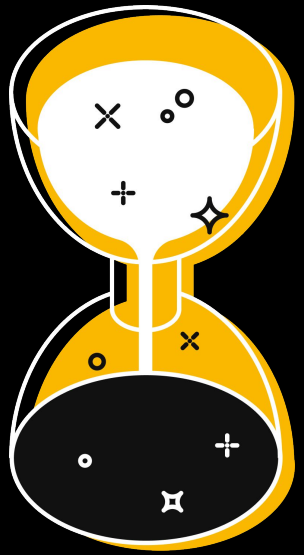


Roughtime: concepts

- Servers have **long-term public keys**
  - Tradeoff: turns time bootstrapping problem into a key distribution problem
  - Uses Ed25519 signatures & Merkle trees
  - Intended for devices where the server list can be updated

- Client asks **multiple servers** for time
  - Checks that responses are consistent
  - Removes single point of failure/attack

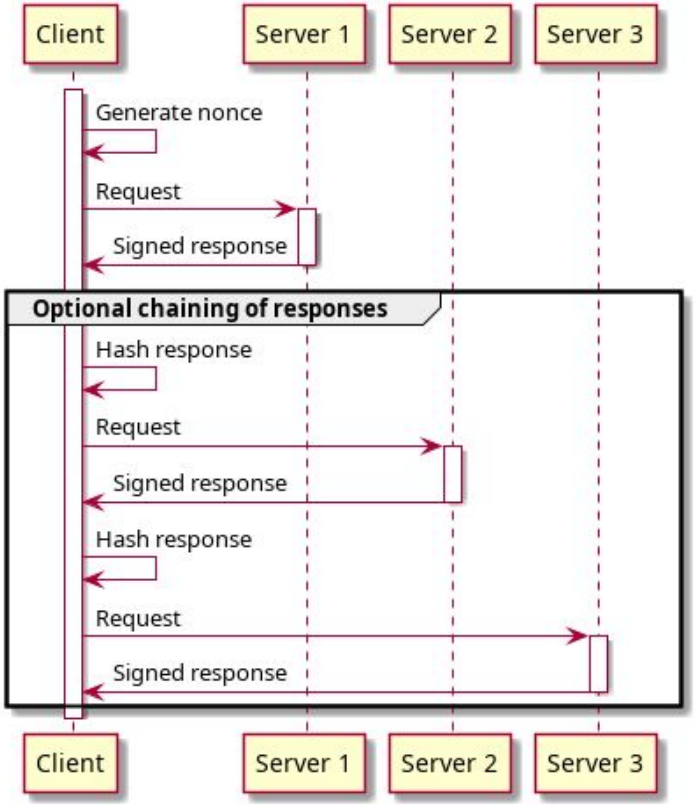
- **Possible to cryptographically prove server malfeasance**



Roughtime: details

- Responses include a time and radius
  - Single-second resolution
  - Servers guarantee that true time is within radius

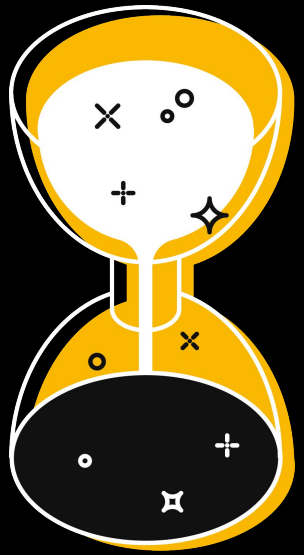
- A 32-byte hash of the request is included in the Merkle tree
  - Allows timestamping of arbitrary data
  - Chaining of responses



Roughtime: evolution

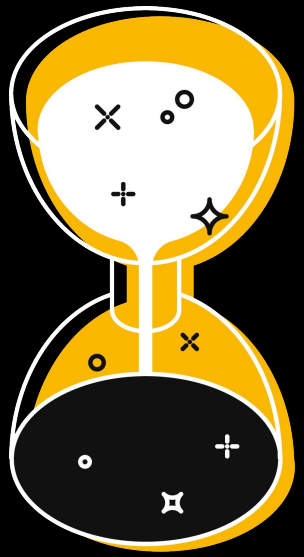
- It is now a decent generic time protocol
  - Secure by default
  - Fairly low CPU usage and small memory footprint
  - Can prove server malfeasance
  - Can timestamp arbitrary data

- Hackathon at IETF 121 in November 2024
  - Discovered and fixed security issues
  - Multiple interoperating implementations



Roughtime: next steps

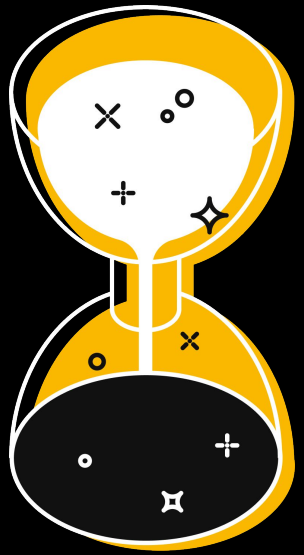
- Intended status: experimental RFC
- IETF working group last call
- Updating implementations
- Building a robust ecosystem of servers and implementations



Roughtime: test it

- Cloudflare (server & client)
  - https://github.com/cloudflare/roughtime
- Craggy (client)
  - https://github.com/nahojkap/craggy
- Pyroughtime (server & client)
  - https://github.com/dansarie/pyroughtime
- Roughenough (server & client)
  - https://github.com/int08h/roughenough
- Roughtimed (server)
  - https://github.com/dansarie/roughtimed

- Most come with a list of servers (ecosystem.json)
  - If not, just use roughtime.se



Thank you!