IPFS/CDN incentives

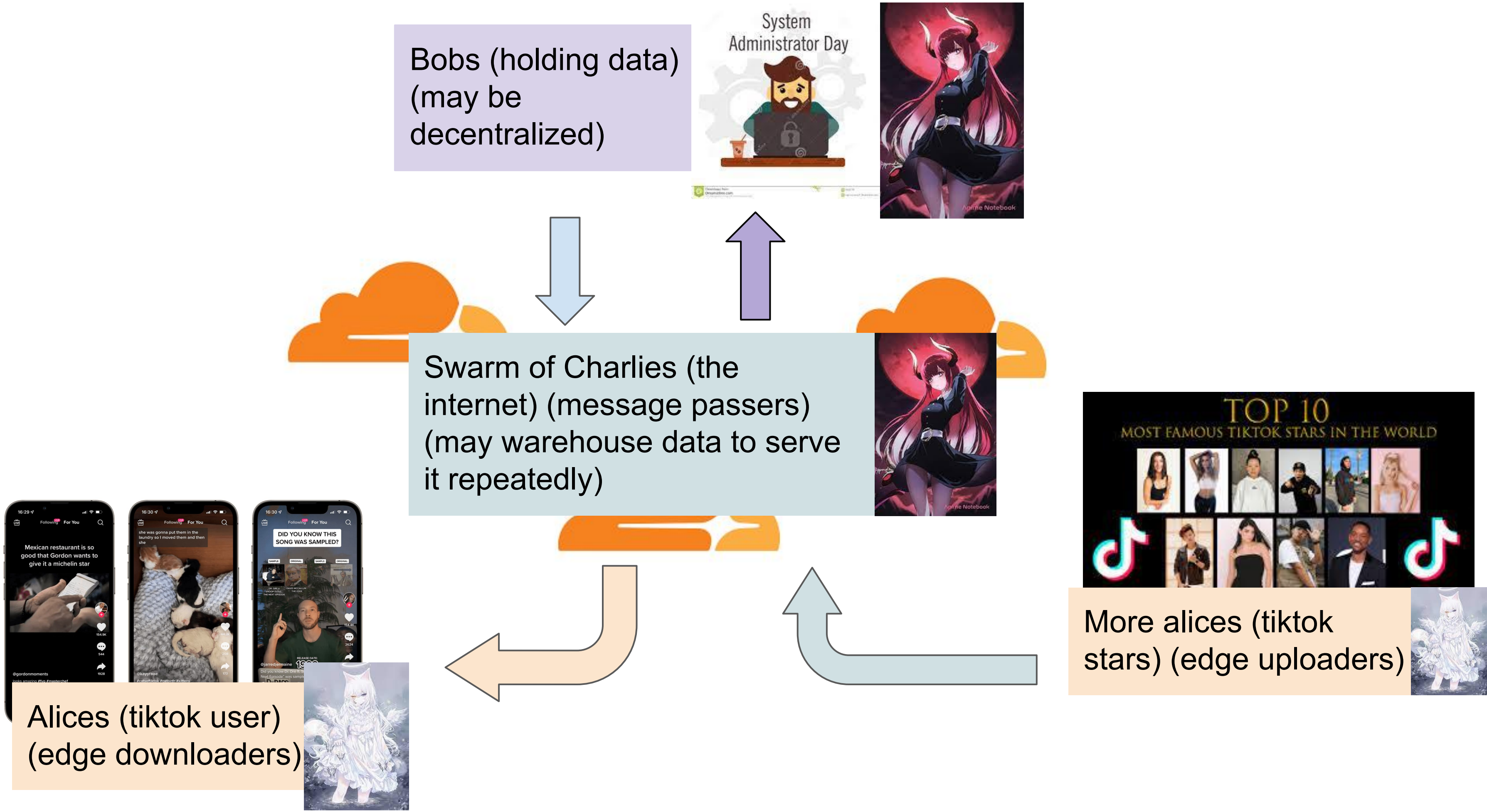
Claudia Richoux
https://w.laudiacay.cool
c@banyan.computer

hi i’m claudia from banyan and retrievals are broken

Problem description

why do we care? / anatomy of a CDN

- people want their data, and quickly! We need to get it to them
- I talk to potential Filecoin users for my company.
  - *giant* objection is whether we can get speed that rivals AWS
- data is distributed all over the planet and we need to get it to them both correctly and quickly
  - It gets to you quicker if it’s closer to you, but the origin could be anywhere!
- durability under attempted censorship by dishonest nodes: a serious concern!
- People are willing to pass it around, maybe
- I did art to show you how this works



System
Administrator Day
Anime Notebook
TOP 10
MOST FAMOUS TIKTOK STARS IN THE WORLD
Mexican restaurant is so good that Gordon wants to give it a michelin star
@gordonmoments
DID YOU KNOW THIS SONG WAS SAMPLED?
@jarredjermaine
@kayyraae

Bobs (holding data) (may be decentralized)

Swarm of Charlies (the internet) (message passers) (may warehouse data to serve it repeatedly)

Alices (tiktok user) (edge downloaders)

More alices (tiktok stars) (edge uploaders)

…unlike anything we’ve seen before…

- leslie lamport’s byzantine generals (coordination/communication with dishonest or faulty actors)
    - (no incentives)
- …which leads into bitcoin
    - (a bit of incentive for the last mile, nothing in between, public)
- centralized CDNs
    - (legal and reputational incentives)
- IPFS
    - (volunteer)
- Filecoins/Sias/Arweaves
    - (rudimentary / bad decentralized distribution incentives, mostly storage incentives)

Tension!

- incentivizing CDN-speed trustless retrievals suffers from a tension between private information and public information
- any friction in the process (publication/verification) destroys the valuable good
- having a third party, or multiple third parties, validate post facto… not just utterly infeasible at scale, but also, *i can just lie to them*.

Prior work:
incremental verification, p2p privacy

Prior work: incremental verification + p2p privacy

- Bao: blake3 verified streaming (Rudiger’s talk from Saturday!)
  - Bake a merkle inclusion proof into content as you send it
  - 6-7% bandwidth overhead, as you transmit data, to prove it’s properly formed as part of a CID
  - Fast! Really fast!
  - **Win: can ensure each packet is well-formed as a tiny piece of a known piece of content.**
- Wireguard / BTC lightning transport
  - Peer-to-peer VPN/tunnel technology
  - Verifies and encrypts packets from a PKI-identified peer (natural decentralization/P2P)
  - Fast! Well-supported! Already in the Linux kernel!
  - Ripe for adding a well-designed payment channel- periodic setup and teardown of a communication channel in a thread, with fast communications in the middle.
  - **Wins: ensure nobody tampers with the content along the way. Safe P2P encryption. Timer system is perfect for periodically open/closing and renewing payment channels.**

Prior work: Retrieval pinning

Scientists recently discovered that rats love driving tiny cars, even when they don't get treats. When put in mazes adapted to tiny cars, the rats just enjoyed cruising around.

Retriev protocol/retrieval pinning

Retriev protocol/retrieval pinning (I think of it as CDN police!)

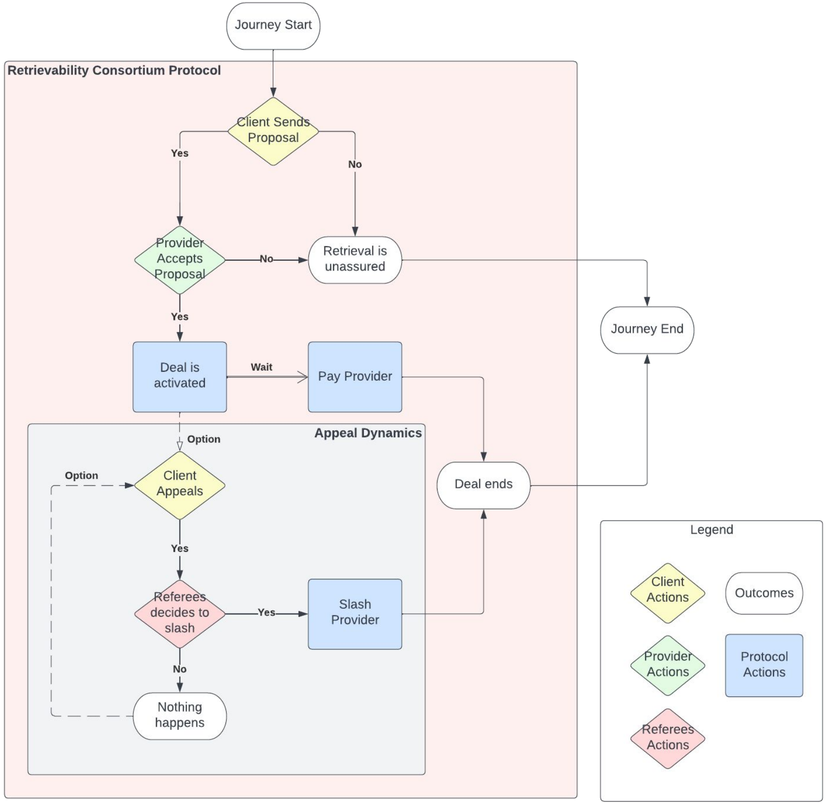
- By cryptonetlab- nicola and irene

Alice pays a group of “referees” to “referee” that Bob sends her her file when she wants it. Bob agrees to submit to being refereed.

All you need to remember is:

- if Bob is fooling around and not sending the file,
- Alice petitions referees,
- who obtain the file from Bob (or slash him),
- validate it,
- and forward it to Alice.

Check out: https://github.com/cryptonetlab/retriev/blob/main/PAPER.md

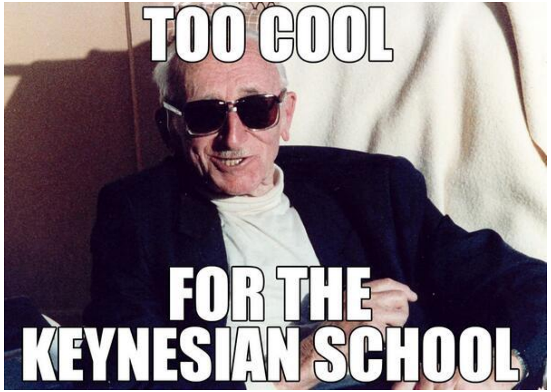


Problems with retriev…

- **Slow**! Need to send to a middleman and then forward to alice
- Have to pass file around between middlemen to validate
  - *(*this part could probably be cleaned up with succinct-er proofs depending on security model)*
- Bob needs to collateralize
  - (nobody wants to collateralize… as we see in Filecoin, doesn't scale)
  - OBSCENELY high collateral multipliers from simulation…
  - *"Time value of money was not taken into consideration. Eg, all decisions can be understood as being immediate."* 😬😬😬
- Alice has to pay middlemen, who have to run servers… ugh.
- There are other problems too…

Elegantish but probably impractical?

Prior work: Payment Drips/tit-for-tat



More good prior work: Skynet in-band incentivization

- Skynet incentivization is in-band incentives over a payment channel
- They have a little payment channel. You bump the amount of money in it with every sector you download and validate.
- You pay the host directly.

Why isn't this great?

- Charlies unpaid!
- One elliptic curve signature per sector (oof) (slow!)
- Not programmable, not transferable, not protocol-layer

A few relevant snippets to orient u (from a pretty big amount of code)
https://gitlab.com/SkynetLabs/skyd/-/blob/master/skymodules/renter/proto/downloader.go#L39
https://gitlab.com/SkynetLabs/skyd/-/blob/master/skymodules/gouging/gouging.go

things i did

Concept: improve skynet with state channel protocol

Set up state channel between client + server- one elliptic curve signature

- Init with client posting a commitment to some keyed RNG output or some sequence of commitments to random values.
- On receipts, with client ACKs, add the preimage of a hash to pay a little to server
- Hash reveals on-chain are quick “monopoly money” to pay server for a packet
- To claim, server proves tickets are correct preimages or valid chunks of keystream :)…

Concept: ramp-up / delegation on this?

Can ramp up to “more packets -> bigger denominations and a longer wait”

Building trust over time with server

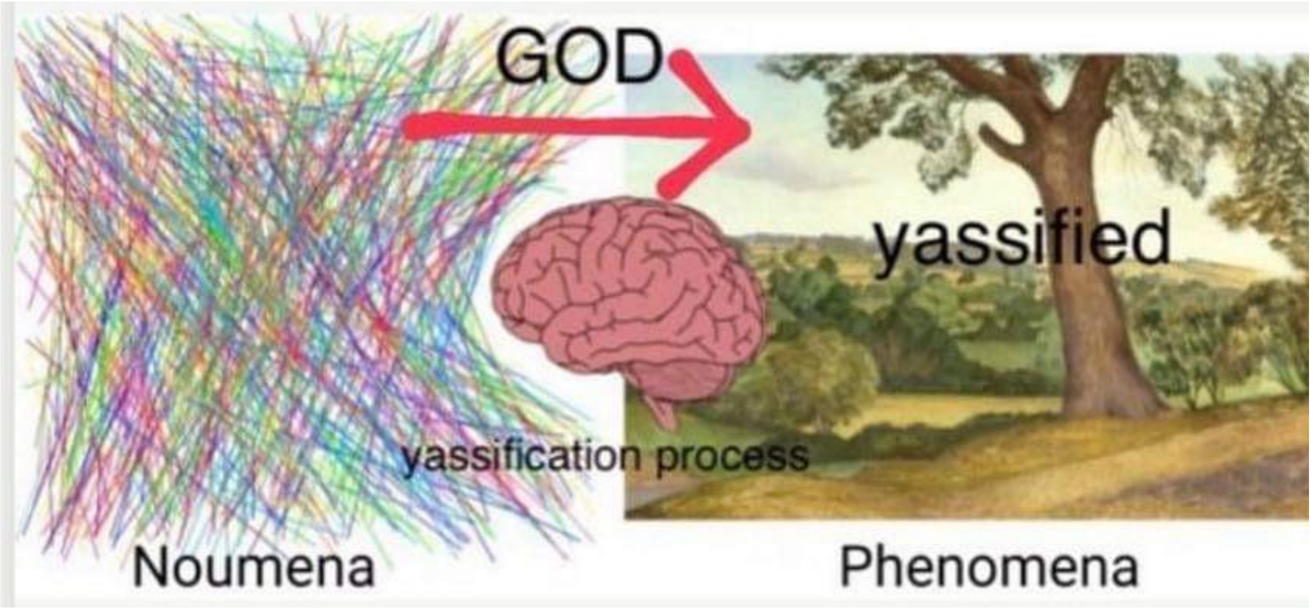
*Economic future work- what’s the right rate of payment? Iterated prisoner’s dilemma

**Can reduce overhead for UDP-based protocols. Doesn’t matter for TCP.*

Weird new topologies of skynet incentives

- Caching Charlies: Charlies can cache and then serve the data, if they’re seeing a lot of requests for it and want to compete/share in profits.
    - How do you pay non-caching Charlies who are message-passing?
- Prepay/pay for others’ bandwidth (Netflix giving clients bandwidth for movies)
    - Give “download coupons” for a specific server and piece of content

Modeling!



Game theory of in-band incentives

- For both parties: generalized multi-armed bandit mixes with iterated prisoner's dilemma mixes with LOTS of competition on both sides
- Server/Bob
    - has its pick of clients/connections to serve data to with fixed bandwidth for maximal payoff
    - switching clients has a warm-up cost in terms of discovery, building trust, establishing connection
    - can stop serving ("betray trust") at any point
- Client/Alice
    - has its pick of servers to get its data from, within the set of servers that have its data, within reasonable RTT
    - Switching servers has the same warm-up cost as switching clients
    - can stop serving to "betray trust" at any point
- Market naturally incentivizes sharing and replication for high-demand data
    - if I'm seeing a lot of cache misses on a given CID, I want to pay Bob once to download a copy, and then I will be his competitor (and profit from this CID)

Game theoretic failure modes

- Alices are unlikely to start a “cartel” where they collectively refuse to pay for their requested data, seeing as they all want their data, and
  - Bob has many profit opportunities, so Alice lacks leverage.
- However, if Bob is insufficiently decentralized…
  - No competition for prices- data hostage situation!
  - We can fix this…

Putting it all together

Filecoin-style storage incentives with collateral slashing

- + retriev-style “you must make this retrievable” slashing incentives
- + In-band skynet-style “you should make it retrievable with competitive QoS” incentives across all Saturn SPs

= the file is kept at multiple collateralized locations (no loss)

- + It is always accessible, or SOME collateralized SP is slashed
- + It is accessible quickly, because ALL SPs are competing for client bandwidth payments. market incentivizes new, geolocalized replications.

= YAY! Decentralized CDN POPs.

ADD bao at or just below the application layer for streaming applications (video, webRTC, anything where you’re sending parts of a whole)

ADD in-band incentives to wireguard: state channel close out and renegotiation happen with handshake timer every 5 minutes at key deprecation time

For UDP protocols, add extra control messages to WG to pass payments

For TCP protocols, tack the payments onto the ack:)

Tl;dr: you need a backstop, plus competition as gasoline

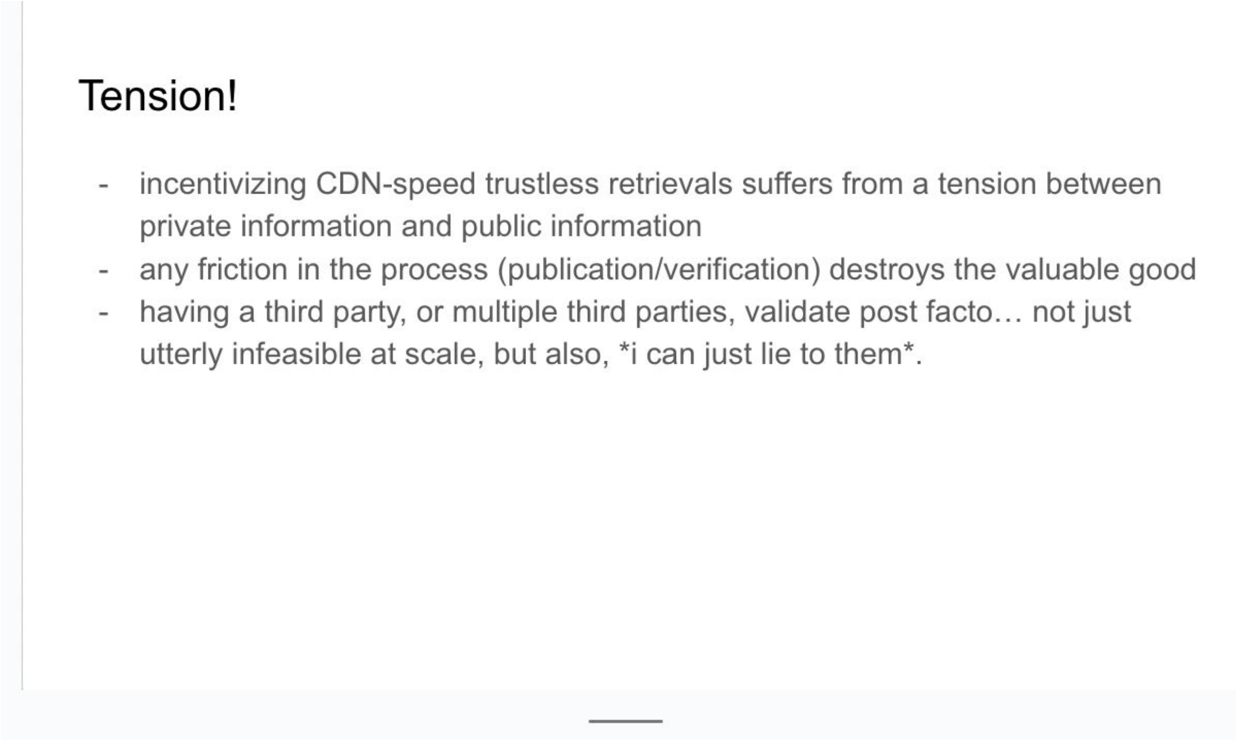
- You need to eliminate situations where *nobody* will store or serve the data (censorship, data hostage, simple loss), using a game theoretic “stick”.
  - This has to be ensured by imposing slashing on individual responsible SPs for failing to meet minimum QoS.
  - Filecoin, Sia, Retrieval pinning all do this.
- You need to create incentives for the data to be replicated, served rapidly, and generally “kept hot” to meet demand, using a game theoretic “carrot”.
  - This is done by creating a market and forcing SPs to compete on latency and replications to adaptively meet customer demand in the moment.
  - Skynet incentives do this.
- Layering incentives across potentially-overlapping sets of SPs allows for strong performance in all cases.

Resolves the tension from the first slide :)

You get both in-band fast local knowledge for fine-tuning (hayek)

And you get “The Government” on the side to make sure nothing catastrophic happens if things get unworkable internally

win!



Tension!

- incentivizing CDN-speed trustless retrievals suffers from a tension between private information and public information
- any friction in the process (publication/verification) destroys the valuable good
- having a third party, or multiple third parties, validate post facto… not just utterly infeasible at scale, but also, *i can just lie to them*.

Bonus: Upload is free in this paradigm!

Out-of-band stick: backstopping bob for minimal QoS

Alice makes sure bob posts a filecoin deal / retriev deal / other commitment

This enforces that he’ll start publicly proving deal receipt or get slashed.

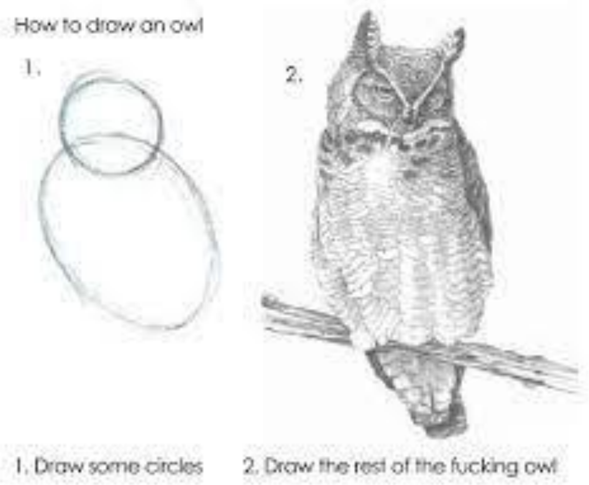
In-band carrot: this is the same as download!

bob pays alice(+charlie?) to get his file over the normal skynet-style protocol. Price may be low or free.

Replications can be incentivized by simply repeating part 1 across multiple Bobs.

nice. some cool thoughts. what now?

homework assignments!



How to draw an owl

1.

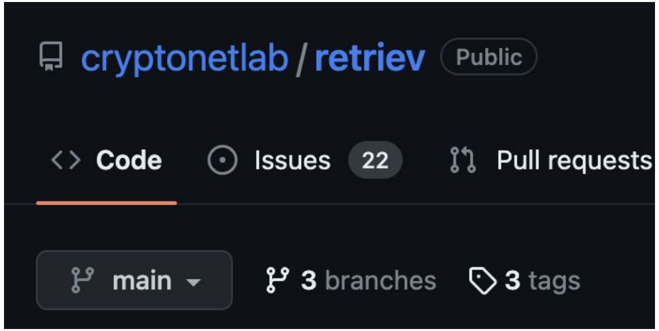
2.

1. Draw some circles

2. Draw the rest of the fucking owl

cadCAD

Edu



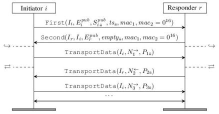
cryptonetlab / retriev Public

<> Code   Issues 22   Pull requests

main   3 branches   3 tags

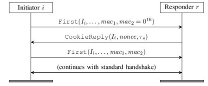
WIREGUARD

FAST, MODERN, SECURE VPN TUNNEL



Initiator i
Responder r
First$(I_i, E_i^{pub}, S_{i🔒}^{pub}, ts_{🔒}, mac_1, mac_2 = 0^{16})$
Second$(I_r, I_i, E_r^{pub}, empty_{🔒}, mac_1, mac_2 = 0^{16})$
TransportData$(I_i, N_1^{\rightarrow}, P_{1🔒})$
TransportData$(I_r, N_2^{\leftarrow}, P_{2🔒})$
TransportData$(I_i, N_3^{\rightarrow}, P_{3🔒})$
...

Figure 1a: WireGuard's protocol messages.



Initiator i
Responder r
First$(I_i, \ldots, mac_1, mac_2 = 0^{16})$
CookieReply$(I_i, nonce, \tau_{🔒})$
First$(I_i, \ldots, mac_1, mac_2)$
(continues with standard handshake)

CookieReply

$$\begin{aligned} R_r &\leftarrow\!\!\$\ \{0,1\}^{256} \quad \text{(refresh only every 2 minutes)} \\ \tau &\leftarrow \mathsf{mac}(R_r, A_i) \quad \text{with } A_i = \mathsf{IP}_i \| \mathsf{Port}_i \\ nonce &\leftarrow\!\!\$\ \{0,1\}^{192} \\ \tau_{🔒} &\leftarrow \mathsf{xaenc}(\mathsf{hash}(\mathsf{label}_{\mathsf{cookie}} \| S_r^{pub}), nonce, \tau, mac_1) \end{aligned}$$

Non-zero mac_2

$$mac_2 \leftarrow \mathsf{mac}(\tau, msg_\beta)$$

Figure 1c: Cookie mechanism under load.

First

$$\begin{aligned} E_i^{pub}: \quad (E_i^{priv}, E_i^{pub}) &\leftarrow\!\!\$\ \mathsf{keygen}() \\ C_0 &\leftarrow \mathsf{hash}(\mathsf{protocol\_name}) \\ H_0 &\leftarrow \mathsf{hash}(C_0 \| \mathsf{prologue}) \\ H_1 &\leftarrow \mathsf{hash}(H_0 \| S_r^{pub}) \\ C_1 &\leftarrow \mathsf{hkdf}_1(C_0, E_i^{pub}) \\ H_2 &\leftarrow \mathsf{hash}(H_1 \| E_i^{pub}) \\ C_2 \| k_1 &\leftarrow \mathsf{hkdf}_2(C_1, \mathsf{dh}(E_i^{priv}, S_r^{pub})) \\ S_{i🔒}^{pub}: \quad S_{i🔒}^{pub} &\leftarrow \mathsf{aenc}(k_1, 0, S_i^{pub}, H_2) \\ H_3 &\leftarrow \mathsf{hash}(H_2 \| S_{i🔒}^{pub}) \\ C_3 \| k_2 &\leftarrow \mathsf{hkdf}_2(C_2, \mathsf{dh}(S_i^{priv}, S_r^{pub})) \\ ts_{🔒}: \quad ts_{🔒} &\leftarrow \mathsf{aenc}(k_2, 0, \mathsf{timestamp}(), H_3) \\ H_4 &\leftarrow \mathsf{hash}(H_3 \| ts_{🔒}) \\ mac_1: \quad mac_1 &\leftarrow \mathsf{mac}(\mathsf{hash}(\mathsf{label}_{\mathsf{mac1}} \| S_r^{pub}), msg_\alpha) \end{aligned}$$

Second

$$\begin{aligned} E_r^{pub}: \quad (E_r^{priv}, E_r^{pub}) &\leftarrow\!\!\$\ \mathsf{keygen}() \\ C_4 &\leftarrow \mathsf{hkdf}_1(C_3, E_r^{pub}) \\ H_5 &\leftarrow \mathsf{hash}(H_4 \| E_r^{pub}) \\ C_5 &\leftarrow \mathsf{hkdf}_1(C_4, \mathsf{dh}(E_r^{priv}, E_i^{pub})) \\ C_6 &\leftarrow \mathsf{hkdf}_1(C_5, \mathsf{dh}(E_r^{priv}, S_i^{pub})) \\ C_7 \| \pi \| k_3 &\leftarrow \mathsf{hkdf}_3(C_6, psk) \\ H_6 &\leftarrow \mathsf{hash}(H_5 \| \pi) \\ empty_{🔒}: \quad empty_{🔒} &\leftarrow \mathsf{aenc}(k_3, 0, empty, H_6) \\ H_7 &\leftarrow \mathsf{hash}(H_6 \| empty_{🔒}) \\ mac_1: \quad mac_1 &\leftarrow \mathsf{mac}(\mathsf{hash}(\mathsf{label}_{\mathsf{mac1}} \| S_i^{pub}), msg_\alpha) \end{aligned}$$

Key Derivation

$$T^{\rightarrow} \| T^{\leftarrow} \leftarrow \mathsf{hkdf}_2(C_7, empty)$$

TransportData

$$\begin{aligned} P_{1🔒} &\leftarrow \mathsf{aenc}(T^{\rightarrow}, N_1^{\rightarrow} = 0, P_1, empty) \\ P_{2🔒} &\leftarrow \mathsf{aenc}(T^{\leftarrow}, N_2^{\leftarrow} = 0, P_2, empty) \\ P_{3🔒} &\leftarrow \mathsf{aenc}(T^{\rightarrow}, N_3^{\rightarrow} = 1, P_3, empty) \end{aligned}$$

Figure 1b: Cryptographic Computations for Protocol Messages.

Questions!

(*Thanks to dig, b5, david vorick from skynet, marten seeman, will Scott, Matt Stephenson, nicola for letting me bounce ideas off of them… you all rock :D)

(Thanks 2 Olive n alex n rob for everything)