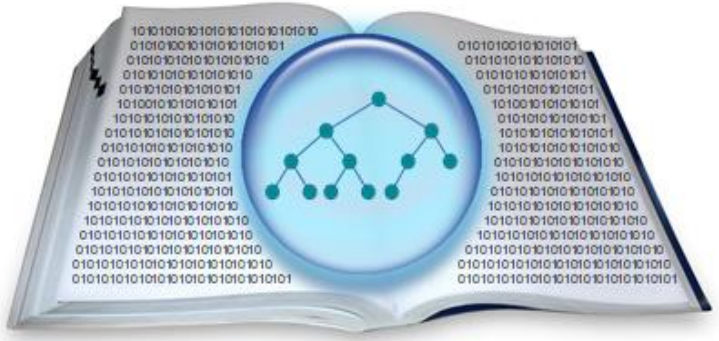


Blockchain, Criptomoedas & Tecnologias Descentralizadas

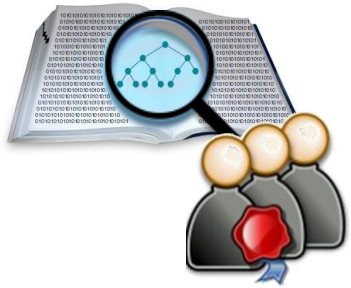
Blockchain sem o hype:

Logs Transparentes

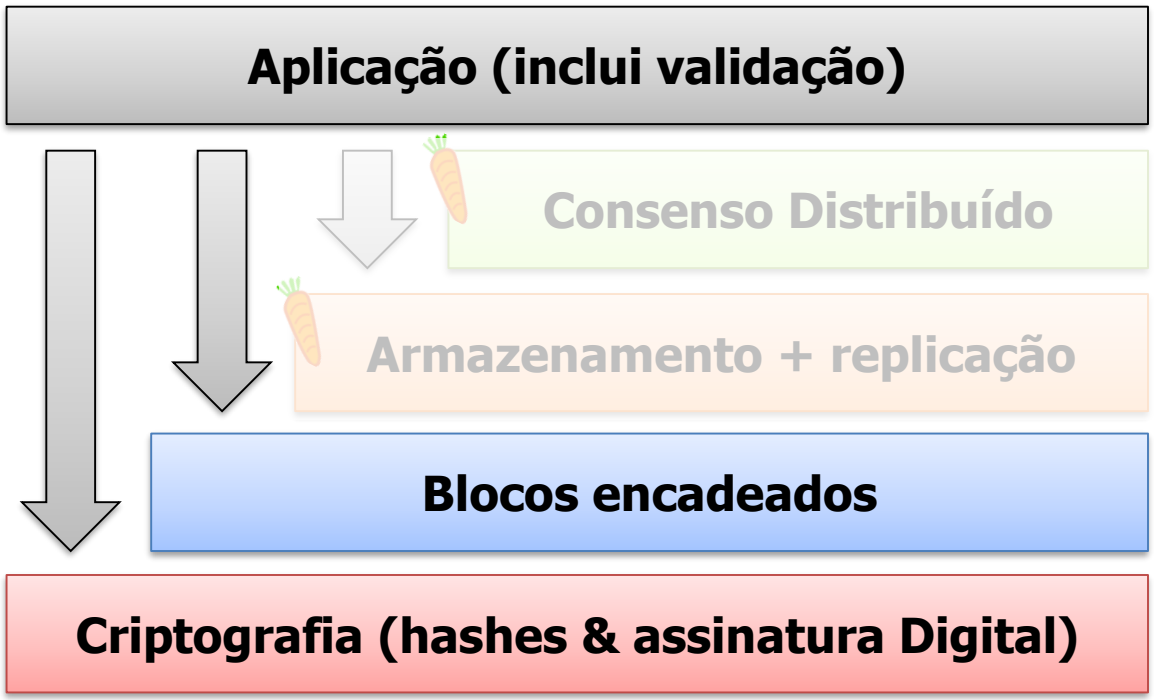
Prof. Dr. Marcos A. Simplicio Jr. – mjunior@larc.usp.br
Escola Politécnica, Universidade de São Paulo

Objetivos

- Discutir o que é um log transparente
  - Parente próximo de blockchains
- Discutir algumas aplicações
  - Transparência de certificados
  - Registro de ativos em plataforma não confiável
  - ...



Log transparente (relembrando)



Ex.: registro com garantia de integridade

Visão uniforme por toda a rede

Disponibilidade, resistência a censura

Ordenação de dados

Detecção de alterações, Irretratabilidade

Blockchain: *"A structure for storing data in which groups of valid transactions, called blocks, form a chronological chain, with each block cryptographically linked to the previous one."* **MIT Technology Review**

Fonte: https://www.technologyreview.com/s/610885/a-glossary-of-blockchain-jargon/

Log transparente (relembrando)

valid ação

Aplicação — Ex.: transferência de ativos digitais

Consenso Distribuído — Visão uniforme por toda a rede

Armazenamento + replicação — Disponibilidade, resistência a censura

Blocos encadeados — Ordenação de dados

Criptografia (hashes & assinatura Digital) — Detecção de alterações, Irretratabilidade

Mágica

Aplicações de blockchain ➔

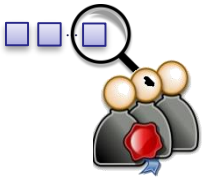
Log transparente pode ser útil

Soluções no mercado (minha experiência):
10%: fazem bastante sentido
50%: bastam assinaturas digitais
20-30%: basta sist. arquivos distribuído
10-20%: requerem "mágica"

Log Transparente

- Há cenários em que replicação total e consenso não são críticos ➔ blockchain como **estrutura de dados**

  - Uma única entidade gerencia **entrada de blocos** e **publica atualizações** periódicas: leva a **visão uniforme** naturalmente



  - **Monitores** armazenam "cauda" do blockchain
    - Opcional: também podem armazenar conteúdo completo, ou partes de interesse: **disponibilidade + resistência a censura**
  - **Tentativas de alteração** de dados pretéritos é facilmente detectada por monitores: **auditabilidade**

    - Dados alterados podem então ser recuperados de monitores que porventura o armazenem

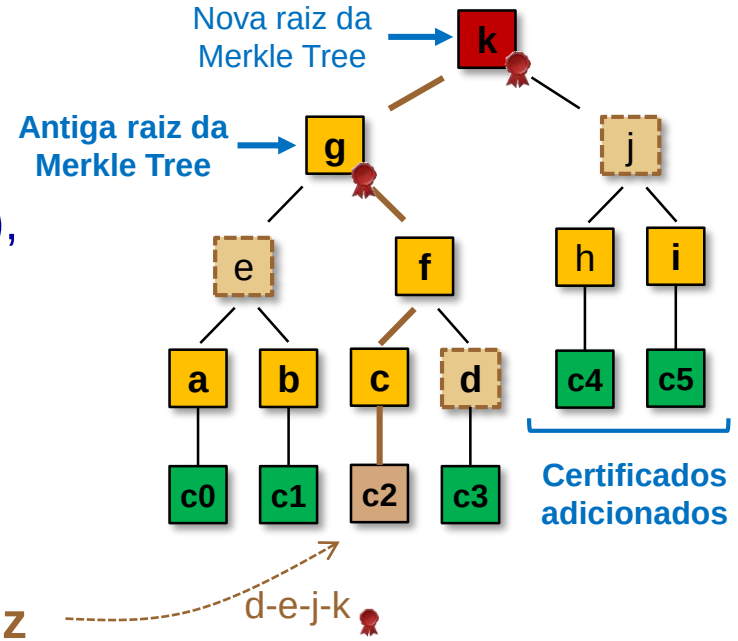
- Permite criar **logs transparentes e verificáveis**
  - Mesmo se entradas não são verificáveis (e.g., ativo real)

Transparência de Certificados

- Aplicação interessante de logs transparentes
- Cenário: certificados web, emitidos por **Infraestrutura de Chaves Públicas (ICP)**
  - Segurança do sistema depende da segurança de Autoridades Certificadores (ACs): “Em ACs nós **confiamos**!”
- Mas... E se uma **CA for comprometida**...?
  - Ou simplesmente não seguir a **devida diligência**...?
- Isso nunca deveria acontecer... mas acontece...
  - DigiNotar (2011): comprometida por hackers
  - Symantec (2015): certificados inválidos p/ Google e Opera
  - Outros: https://www.certificate-transparency.org/what-is-ct

Transparência de Certificados

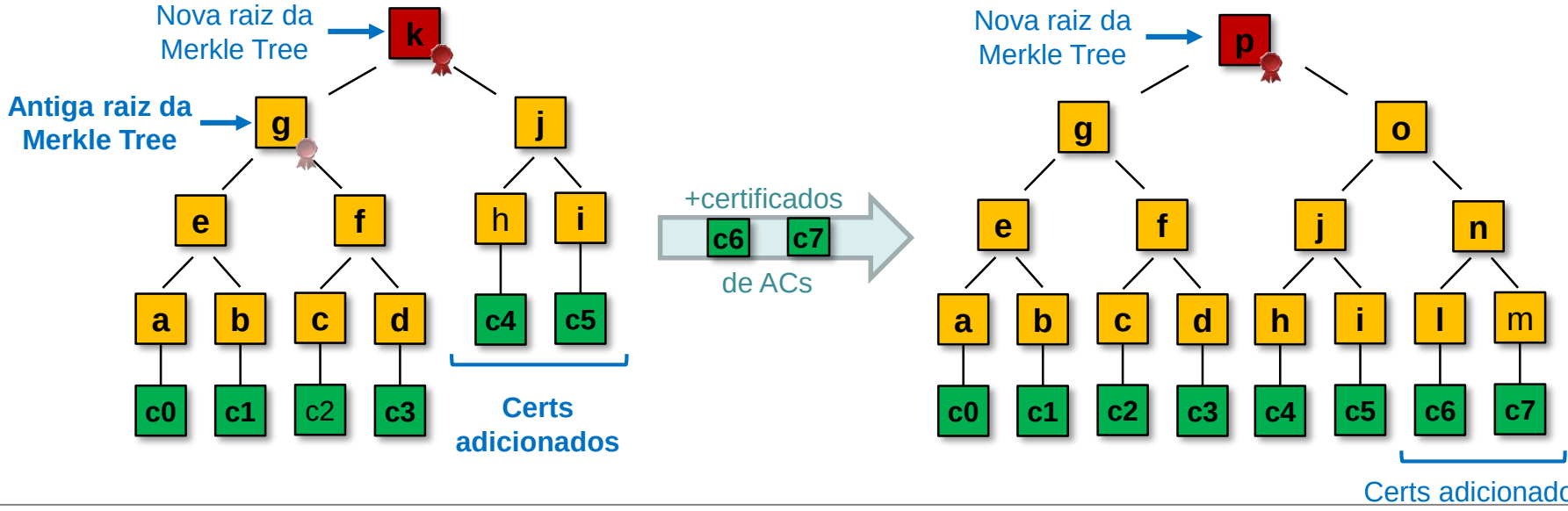
- **Logs públicos** de certificados, com política de "**adição apenas**"
  - **Árvore de Merkle** (melhor desempenho), mas poderia ser um blockchain "linear"
  - **Verificável**, mesmo se não confiável
- **Navegador** só aceita **certs logados**
  - Força CAs a **publicar** suas ações
  - Verificação: vizinhos ao **caminho até raiz**



- **Monitores** verificam conteúdo, e mantêm **raiz da árvore**
  - Detectam certificados com extensões estranhas, ou tentativa de deleção por Servidor que mantém log público
- Donos de domínios têm **visibilidade dos certificados**
  - Podem pedir revogação em caso de emissão inválida

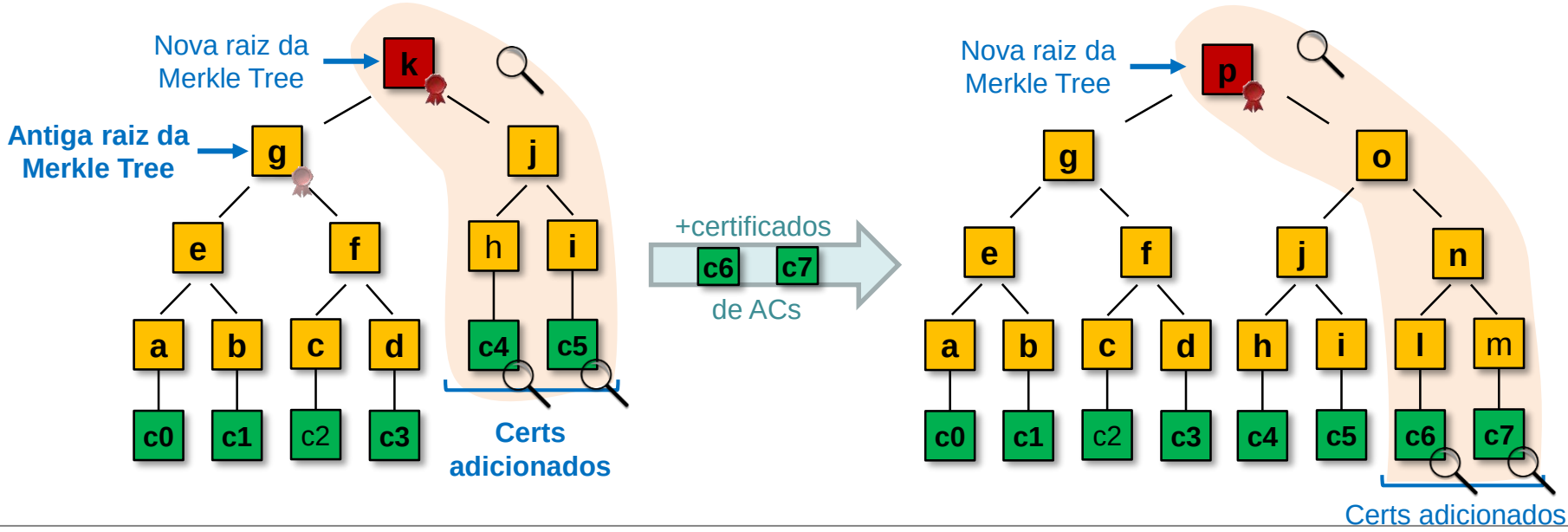
Transparência de Certificados

- Funcionamento: **inserção**
  - AC envia “pré-certificado” a **Servidor de Log:** Google (2013), Cloudflare (2018),…
  - Servidor de log responde com SCT (*Signed Certificate Timestamp*)
    - Promessa de registrar certificado dentro de MMD (*maximum merge delay*): e.g., 24h
  - AC insere SCT no certificado (extensão X.509).
  - Servidor de log insere cerificados em Árvore de Merkle (balanceada) e assina a raiz: **não requer mecanismo de consenso**



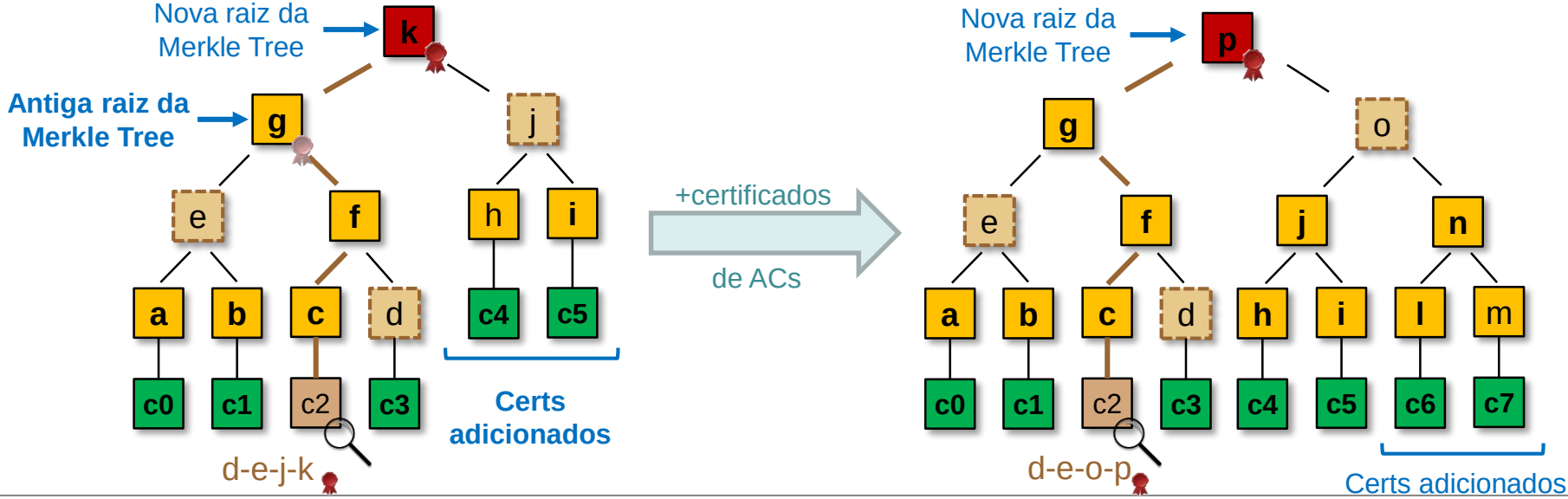
Transparência de Certificados

- Funcionamento: **verificação**
  - **Monitores**: fazem download periódico e revisão dos logs
    - Ex.: “nova raíz bate com (raíz antiga + novos certificados)?”
    - Ex.: certificados emitidos por ACs válidas? Têm estrutura correta?
  - Podem armazenar apenas **raíz** da árvore (**detecção** de alterações) ou **replicar árvore** completa (**recuperação** em caso de alteração)
  - Qualquer interessado: empresas, governo, serviços contratados (**incentivo**), …



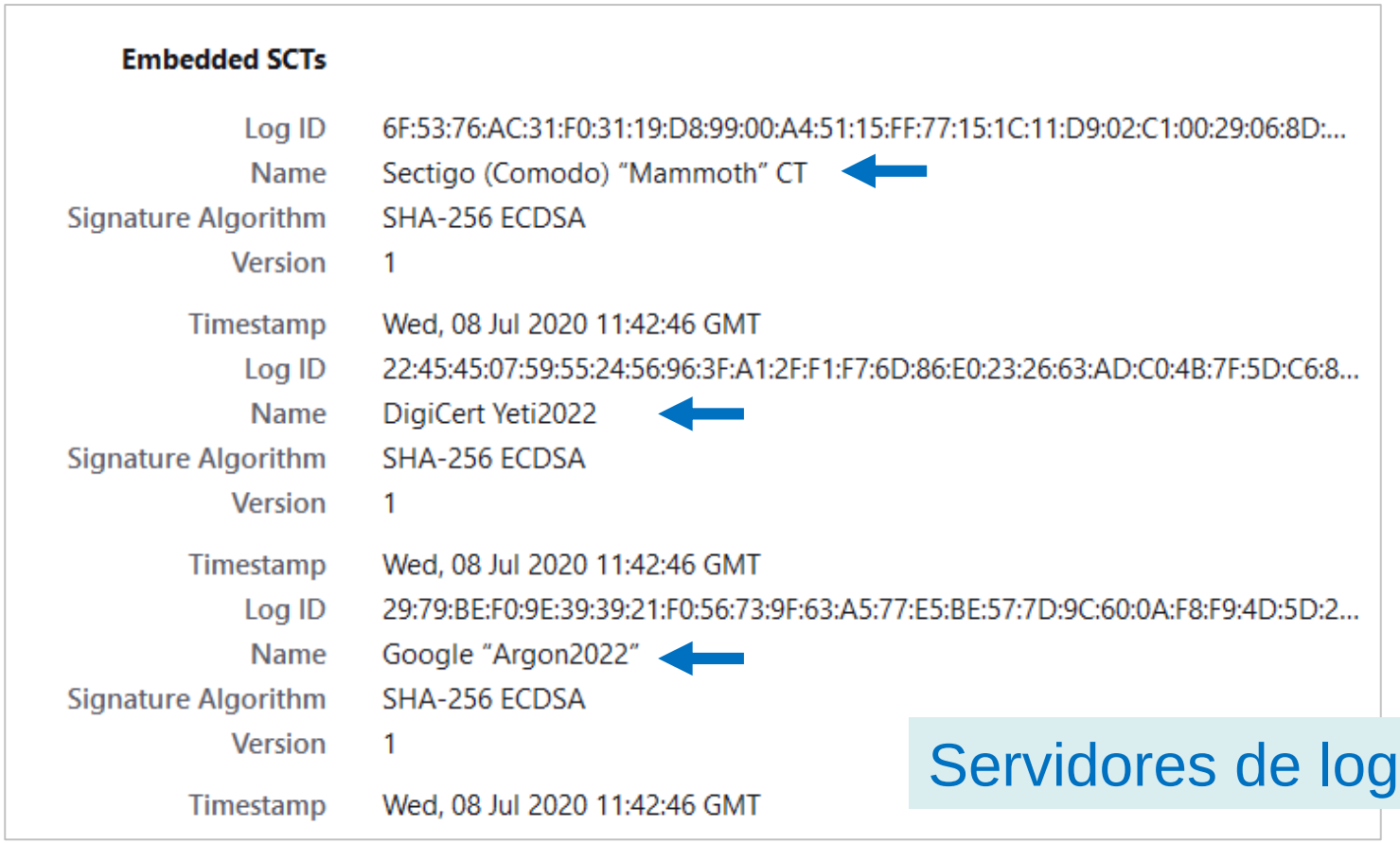
Transparência de Certificados

- Funcionamento: **operação**
  - **Auditores**: verificam se certificado encontra-se no log
  - "**Prova de auditoria de Merkle**": raiz + caminho da folha até raiz
    - Motivação para Merkle Tree: **O(lg n)** nós, vs. O(n) em blockchain linear
    - Donos de domínios podem armazenar provas e entregá-las com certificados
  - Se **embutido em navegador**: recusa conexão TLS de certificado não logado
    - Genericamente: aplicação verifica se **transparência** está sendo respeitada



Transparência de Certificados

- Exemplo: certificado em https://www5.usp.br/



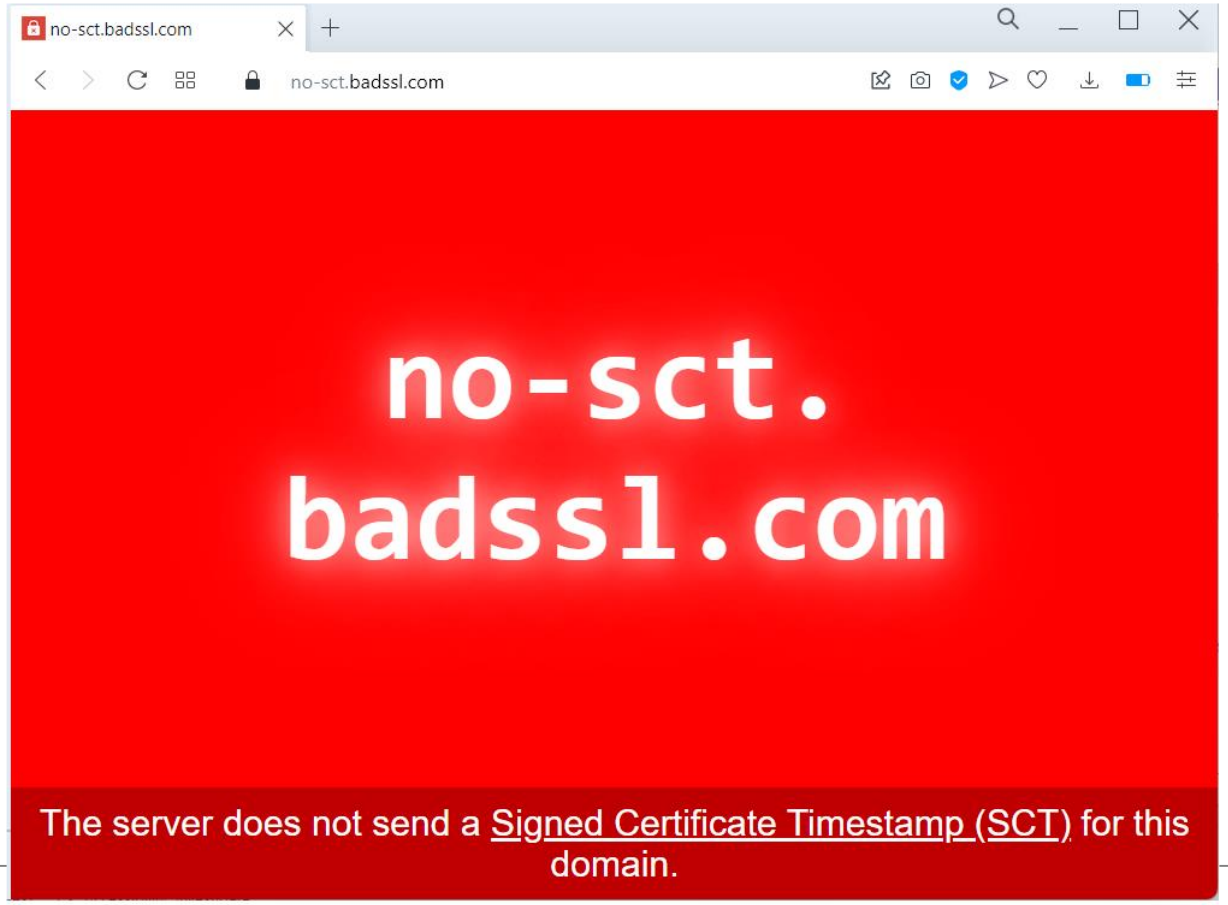
Embedded SCTs

Log ID	6F:53:76:AC:31:F0:31:19:D8:99:00:A4:51:15:FF:77:15:1C:11:D9:02:C1:00:29:06:8D:...
Name	Sectigo (Comodo) "Mammoth" CT
Signature Algorithm	SHA-256 ECDSA
Version	1
Timestamp	Wed, 08 Jul 2020 11:42:46 GMT
Log ID	22:45:45:07:59:55:24:56:96:3F:A1:2F:F1:F7:6D:86:E0:23:26:63:AD:C0:4B:7F:5D:C6:8...
Name	DigiCert Yeti2022
Signature Algorithm	SHA-256 ECDSA
Version	1
Timestamp	Wed, 08 Jul 2020 11:42:46 GMT
Log ID	29:79:BE:F0:9E:39:39:21:F0:56:73:9F:63:A5:77:E5:BE:57:7D:9C:60:0A:F8:F9:4D:5D:2...
Name	Google "Argon2022"
Signature Algorithm	SHA-256 ECDSA
Version	1
Timestamp	Wed, 08 Jul 2020 11:42:46 GMT

Servidores de log

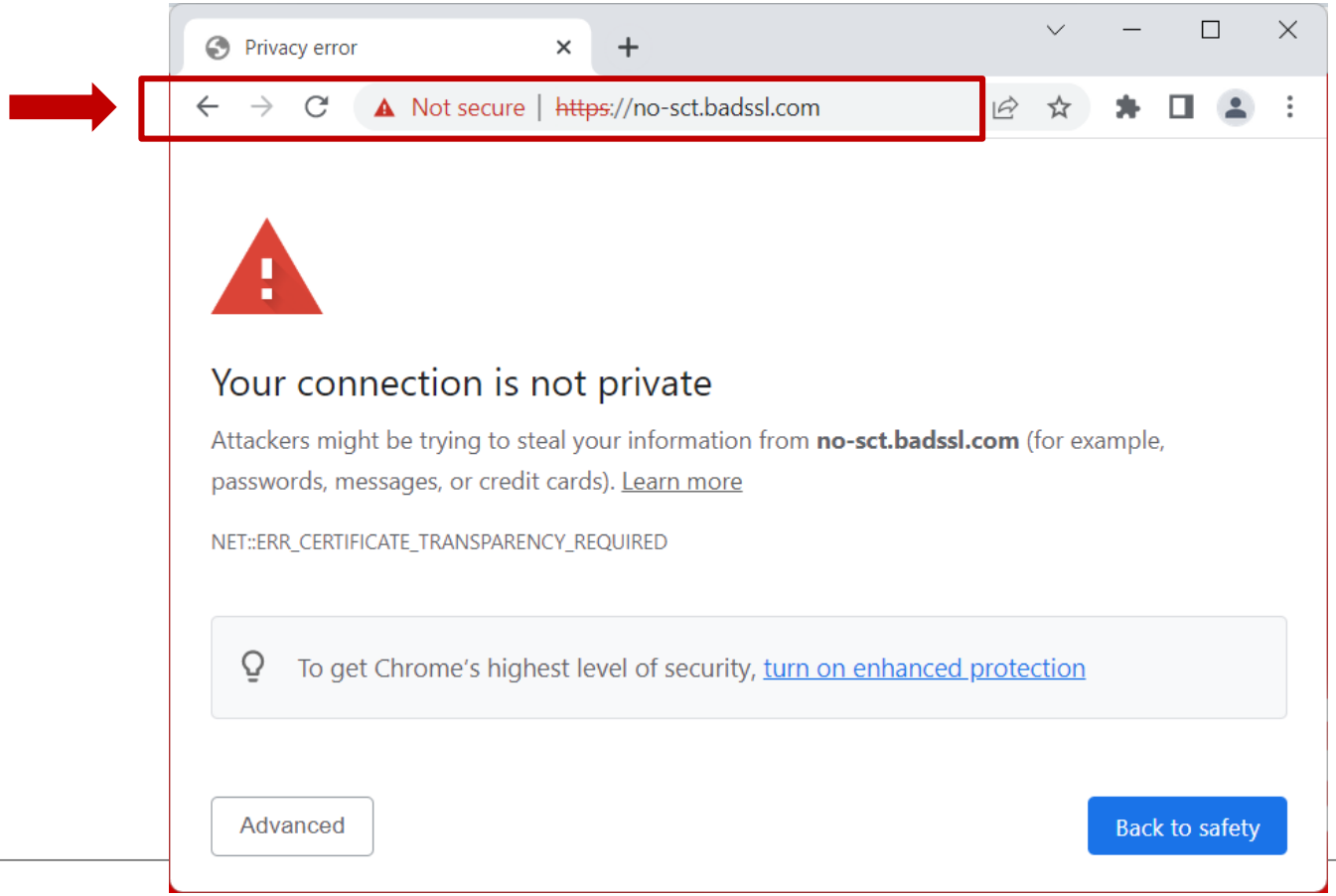
Transparência de Certificados

- Ex.: certificado sem SCT (https://no-sct.badssl.com/)
  - Opera



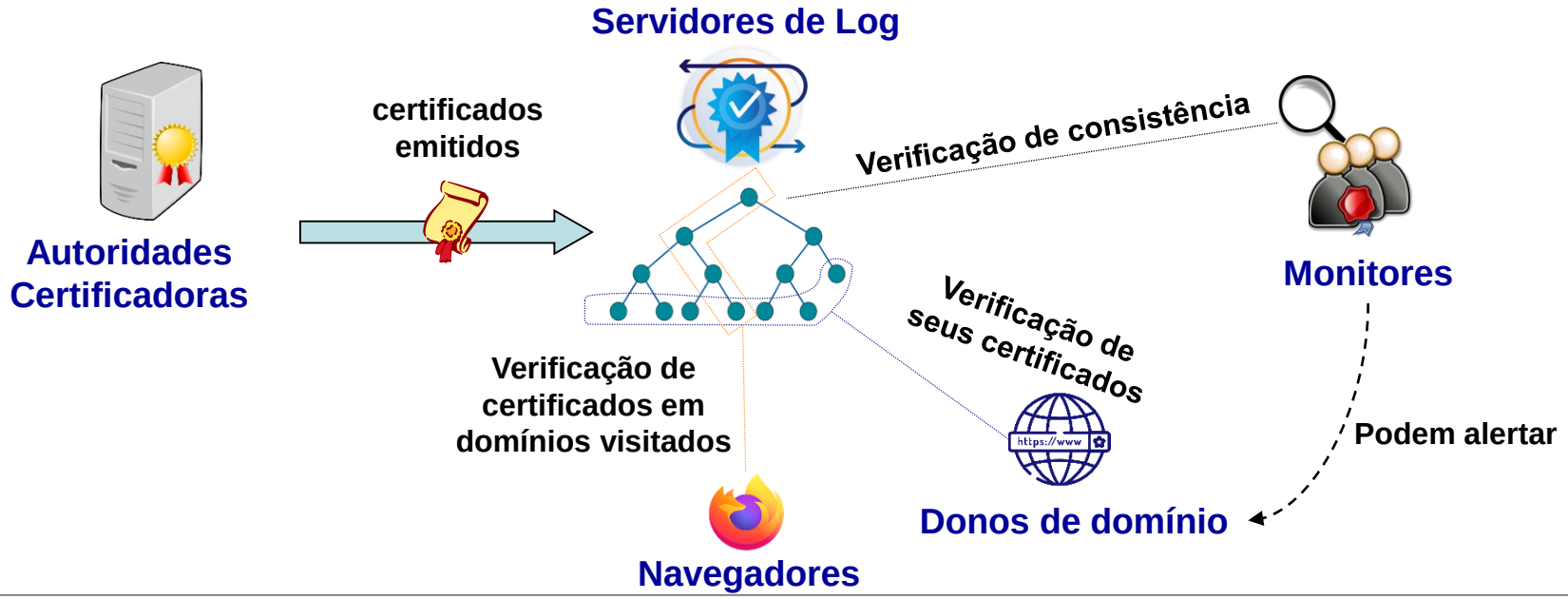
Transparência de Certificados

- Ex.: certificado sem SCT (https://no-sct.badssl.com/)
  - Google Chrome (CT já é mandatória)



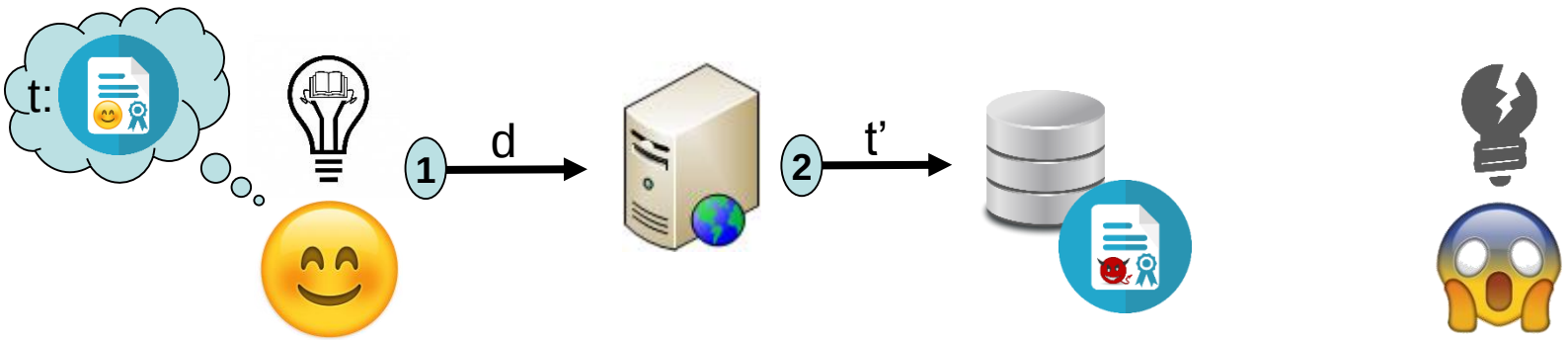
Transparência de Certificados

- **Resumo:**
  - **Logs públicos** de certificados (adição apenas)
  - **Navegador** só aceita **certificados logados**
  - **Monitores** verificam conteúdo, e mantêm **raiz da árvore**
  - Donos de domínios têm **visibilidade dos certificados**



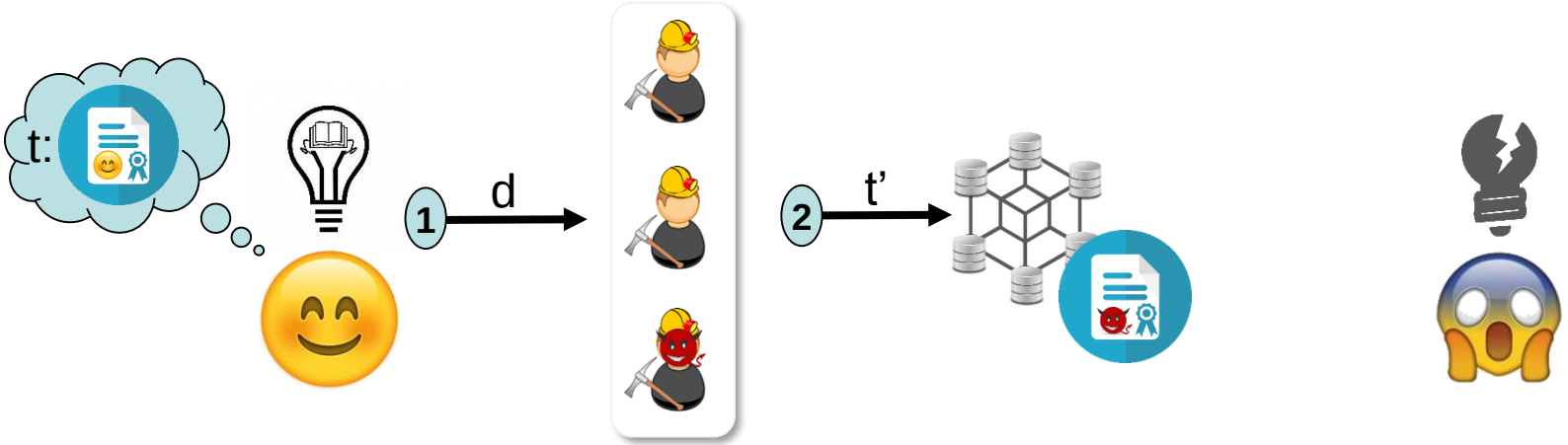
Log transparente: registro de ativo

- Cenário que admite tempo aproximado (definido por servidor), mas auditável (e.g., monotonicamente crescente)
  - **Ex.:** usuário deseja registrar dado *d* (e.g., patente, NFT) em sistema
  - **Abordagem 1**: usuário envia *d* para servidor de registro, e aguarda token *t* referente a *d*, tokenizado em seu nome…
    - … mas *d* é tokenizado no nome de outrém (dono do servidor…?!)



Log transparente: registro de ativo

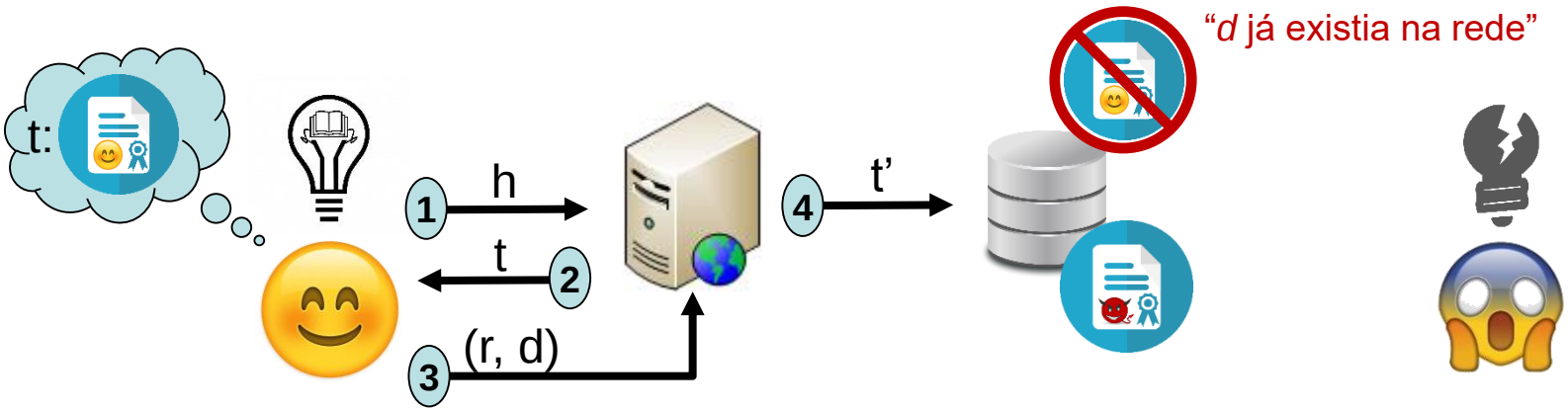
- Cenário que admite tempo aproximado (definido por servidor), mas auditável (e.g., monotonicamente crescente)
  - **Ex.:** usuário deseja registrar dado *d* (e.g., patente, NFT) em sistema
  - **Abordagem 2**: usuário envia *d* para rede blockchain de registro, e aguarda token *t* referente a *d*, tokenizado em seu nome…
    - … mas *d* é tokenizado no nome de outrém (minerador malicioso) após consenso…



Log transparente: registro de ativo

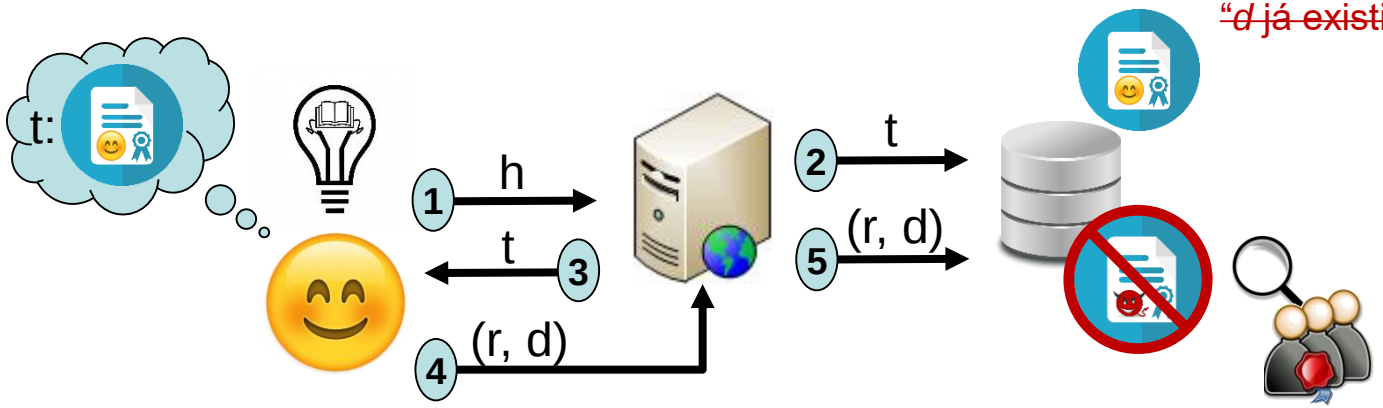
- Cenário que admite tempo aproximado (definido por servidor), mas auditável (e.g., monotonicamente crescente)
  - **Ex.:** usuário deseja registrar dado *d* (e.g., patente, NFT) em sistema
  - **Abordagem 3**: usuário envia *h* = Hash(*r*, *d*) para servidor de registro com *r* aleatório, aguarda confirmação, e depois envia (*r, d*) ao servidor

    - … mas servidor alega que *d* é uma cópia, e mostra t' tokenizado no nome de outrém (dono do servidor…?!)



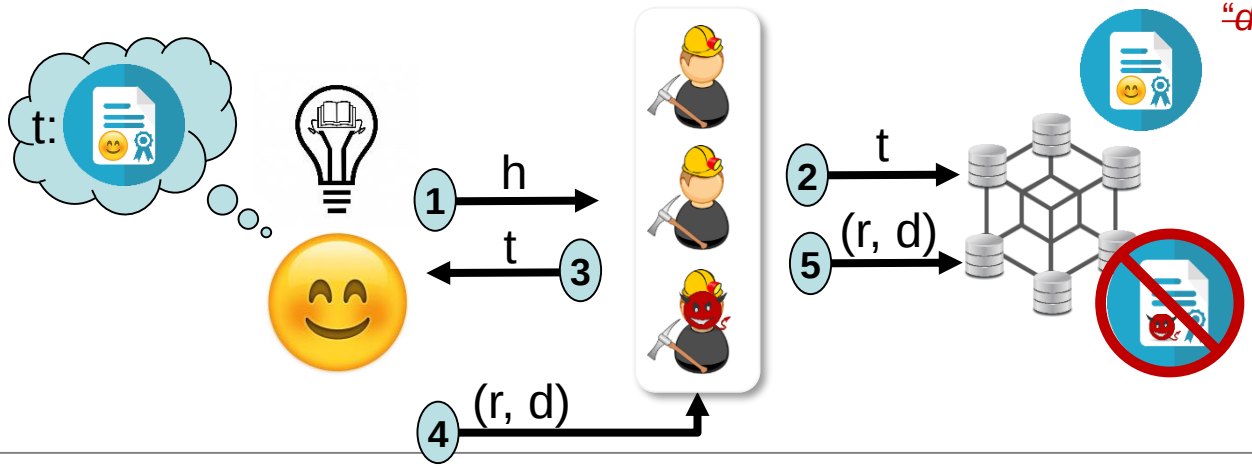
Log transparente: registro de ativo

- Cenário que admite tempo aproximado (definido por servidor), mas auditável (e.g., monotonicamente crescente)
  - **Ex.:** usuário deseja registrar dado *d* (e.g., patente, NFT) em sistema
  - **Abordagem 4**: usuário envia *h* = Hash(*r*, *d*) para servidor de registro, com *r* aleatório; aguarda confirmação em log transparente; e depois envia (*r, d*) ao servidor
    - Se servidor alegar que *d* é uma cópia, precisa mostrar t' tokenizado no nome de outrém já registrado no log transparente



Log transparente: registro de ativo

- Cenário que admite tempo aproximado (definido por servidor), mas auditável (e.g., monotonicamente crescente)
  - **Ex.:** usuário deseja registrar dado *d* (e.g., patente, NFT) em sistema
  - **Abordagem 5**: usuário envia *h* = Hash(*r*, *d*) para rede blockchain de registro, com *r* aleatório; aguarda confirmação na rede após consenso; e depois envia (*r, d*) à rede blockchain
    - Não é possível alegar que *d* é uma cópia, pois seria necessário haver t' tokenizado no nome de outrém já registrado no blockchain



~~"*d* já existia na rede"~~

Log transparente: Exercício

- Cenário: assinatura digital para **credenciais de longa duração**, como **diplomas universitários**
- **Pergunta:** quais desses problemas têm relação com logs transparentes?

A. “Pessoa mente sobre ter diploma”

B. “Universidade mente ao negar emissão de diploma”

C. “Pessoa perde diploma”

D. “Universidade vende diplomas falsos”

E. “Chave privada antiga é comprometida e usada para forjar diplomas com data em que ela ainda era válida”

Log transparente: Exercício (resp.)

- Cenário: assinatura digital para **credenciais de longa duração**, como **diplomas universitários**
- **Pergunta:** quais desses problemas têm relação com logs transparentes?

A. “Pessoa mente sobre ter diploma” ➔ basta **assinatura digital** do diploma

B. “Universidade mente ao negar emissão de diploma” ➔ basta **assinatura digital** do diploma, mantida em posse do seu dono

C. “Pessoa perde diploma” ➔ resolvido com **backup** (nuvem, IPFS, …)

Log transparente: Exercício (resp.)

- Cenário: assinatura digital para **credenciais de longa duração**, como **diplomas universitários**
- **Pergunta:** quais desses problemas têm relação com logs transparentes?

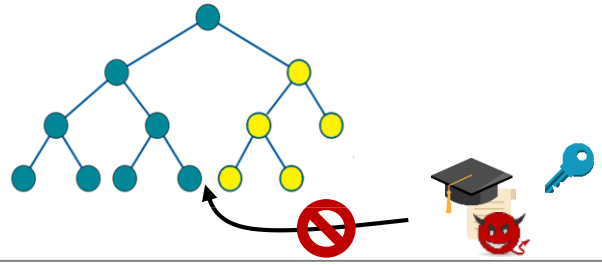
D. “Universidade vende diplomas falsos” ➔ log transparente aumenta custos da fraude
   - Log registra (1) **ingresso** do aluno e (2) **finalização** do curso: permite verificar se diploma condiz com **histórico do curso**.
   - **Não impede pré-venda**: pessoa registrada hoje tem diploma emitido no futuro, mesmo sem cursar disciplinas
   - **Não impede *mea-culpa***: “por falha nossa, esquecemos de registrar o aluno no ingresso, mas aqui está o diploma dele”

Log transparente: Exercício (resp.)

- Cenário: assinatura digital para **credenciais de longa duração**, como **diplomas universitários**
- **Pergunta:** quais desses problemas têm relação com logs transparentes?

E. “Chave privada antiga é comprometida e usada para forjar diplomas com data em que ela ainda era válida” ➔ log transparente mitiga ataques

  - Log registra **data de emissão do diploma**: registro monotonicamente crescente
  - Chave antiga pode ser usada para gerar assinatura válida, mas não para gerar **prova de auditoria**: diploma não pode ser emitido no passado!

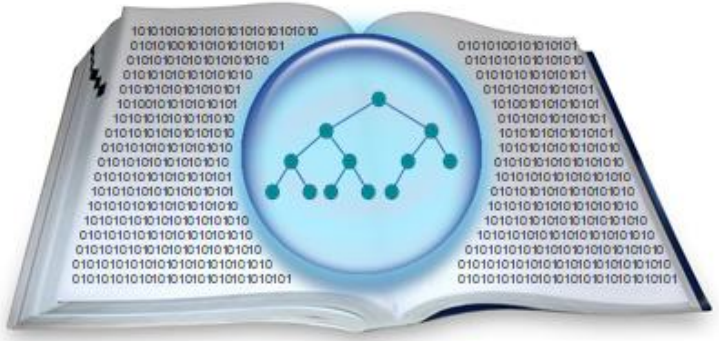


Log transparente: outros

- Log transparente: cenário que admite tempo aproximado (definido por servidor), mas auditável (e.g., monotonicamente crescente)

  - **Registro de previsões por videntes**, evitando tentativas de editar cartas após seu registro em cartório (com espaços em branco…)
    - Ex.: “Eu previ o resultado da loteria: veja carta registrada há 10 dias atrás!”

  - **Registro de ações oficiais** por autoridades no tempo, para evitar tentativas de re-escrever o passado
    - Ex.: “Não houve prevaricação, pois assim que recebi denúncia sobre irregularidade em compra, repassei para as autoridades competentes”
  - Registro de **dados relativos a eleições**, possibilitando verificações e maior confiança no resultado e refutando teorias da conspiração
    - Ex.: “Mas os dados relativos à urna no site da autoridade eleitoral foram alterados após as eleições! Ela foi trocada!”



Blockchain, Criptomoedas & Tecnologias Descentralizadas

Blockchain sem o hype:

Logs Transparentes

Prof. Dr. Marcos A. Simplicio Jr. – mjunior@larc.usp.br
Escola Politécnica, Universidade de São Paulo

Apoio financeiro: d24n.org <https://d24n.org>
ripple impact | university blockchain research initiative

Referências

- B. Laurie & E. Kasper, E. (2012). Revocation transparency. Google Research, September, 33. URL: https://www.links.org/files/RevocationTransparency.pdf
- B. Laurie, B. (2014). Certificate transparency. Communications of the ACM, 57(10), 40-46. See also https://www.certificate-transparency.org/what-is-ct
- Google (online). Trust your data with a tamper-evident log. URL: https://transparency.dev
- B. Li, J. Lin, F. Li, Q. Wang, Q. Li, J. Jing, & C. Wang, (2019). Certificate transparency in the wild: Exploring the reliability of monitors. In Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (pp. 2505-2520). URL: https://doi.org/10.1145/3319535.3345653
- F. Matsumoto, J. Silva, M. Simplicio (2021). Transparência de Domínios: maior auditabilidade para serviços de Transparência de Certificados. In: Salão De Ferramentas - Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais (SBSeg), 21., Online. Porto Alegre: Sociedade Brasileira de Computação,. p.42-49. URL: https://doi.org/10.5753/sbseg_estendido.2021.17338