Anonymous Transactions with Revocation and Auditing in Hyperledger Fabric

Anonymous Credentials, Revocation, Auditing, Blockchain

[10] DOI: 10.1007/978-3-030-92547-5_23

Dmytro Bogatov, Angelo De Caro, Kaoutar Elkhiyaoui, Björn Tackmann
dmytro@bu.edu, adc@zurich.ibm.com, kao@zurich.ibm.com
bjoern@dfinity.org

Built from *120359ce* on November 21, 2021

Boston University
Graduate School of Arts and Sciences
Department of Computer Science

Background

Motivation

- Permissioned blockchains explicitly register their participants
  logistics network, bank transactions, know-your-customer and anti-money-laundering regulations
- Prove the permission to post a transaction without revealing identity
  - arbitrary length of delegation chain
  - dynamically embedding an arbitrary number of attributes
  - efficient and integrated with the blockchain
- Revocation and audit
  - Prove AGE $\geq$ 21 using driving license without reveling your name and issuing state
    delegatable anonymous credentials
  - Allow a state to suspend (revoke) the driving license at any time
    privacy-preserving revocation
  - Allow a federal investigator to inspect the entire license at a later date with a warrant
    auditing along with proving that the credential is "auditable"

Motivation

- Permissioned blockchains explicitly register their participants

  logistics network, bank transactions, know-your-customer and anti-money-laundering regulations

- Prove the permission to post a transaction without revealing identity
  - arbitrary length of delegation chain
  - dynamically embedding an arbitrary number of attributes
  - efficient and integrated with the blockchain
- Revocation and audit
  - Prove **AGE** $\geq$ 21 using driving license without reveling your name and issuing state

    delegatable anonymous credentials

  - Allow a state to suspend (revoke) the driving license at any time

    privacy-preserving revocation

  - Allow a federal investigator to inspect the entire license at a later date with a warrant

    auditing along with proving that the credential is "auditable"

Motivation

- Permissioned blockchains explicitly register their participants

  logistics network, bank transactions, know-your-customer and anti-money-laundering regulations
- Prove the permission to post a transaction without revealing identity
  - arbitrary length of delegation chain
  - dynamically embedding an arbitrary number of attributes
  - efficient and integrated with the blockchain
- Revocation and audit
  - Prove $\mathsf{AGE} \geq 21$ using driving license without reveling your name and issuing state

    delegatable anonymous credentials
  - Allow a state to suspend (revoke) the driving license at any time

    privacy-preserving revocation
  - Allow a federal investigator to inspect the entire license at a later date with a warrant

    auditing along with proving that the credential is “auditable”

Motivation

- Permissioned blockchains explicitly register their participants

  logistics network, bank transactions, know-your-customer and anti-money-laundering regulations
- Prove the permission to post a transaction without revealing identity
  - arbitrary length of delegation chain
  - dynamically embedding an arbitrary number of attributes
  - efficient and integrated with the blockchain
- Revocation and audit
  - Prove $\mathsf{AGE} \geq 21$ using driving license without reveling your name and issuing state

    delegatable anonymous credentials
  - Allow a state to suspend (revoke) the driving license at any time

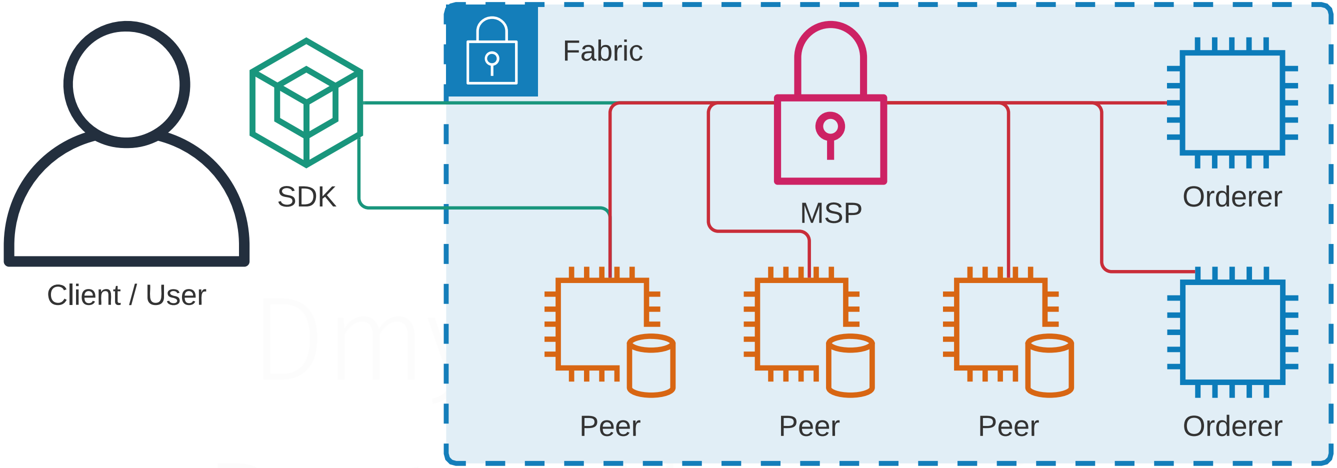
    privacy-preserving revocation
  - Allow a federal investigator to inspect the entire license at a later date with a warrant

    auditing along with proving that the credential is "auditable"

Motivation

- Permissioned blockchains explicitly register their participants
  logistics network, bank transactions, know-your-customer and anti-money-laundering regulations
- Prove the permission to post a transaction without revealing identity
  - arbitrary length of delegation chain
  - dynamically embedding an arbitrary number of attributes
  - efficient and integrated with the blockchain
- Revocation and audit
  - Prove $\mathsf{AGE} \geq 21$ using driving license without reveling your name and issuing state
    delegatable anonymous credentials
  - Allow a state to suspend (revoke) the driving license at any time
    privacy-preserving revocation
  - Allow a federal investigator to inspect the entire license at a later date with a warrant
    auditing along with proving that the credential is "auditable"

Hyperledger Fabric [8]



Components

Clients invoke TXs and observe their results

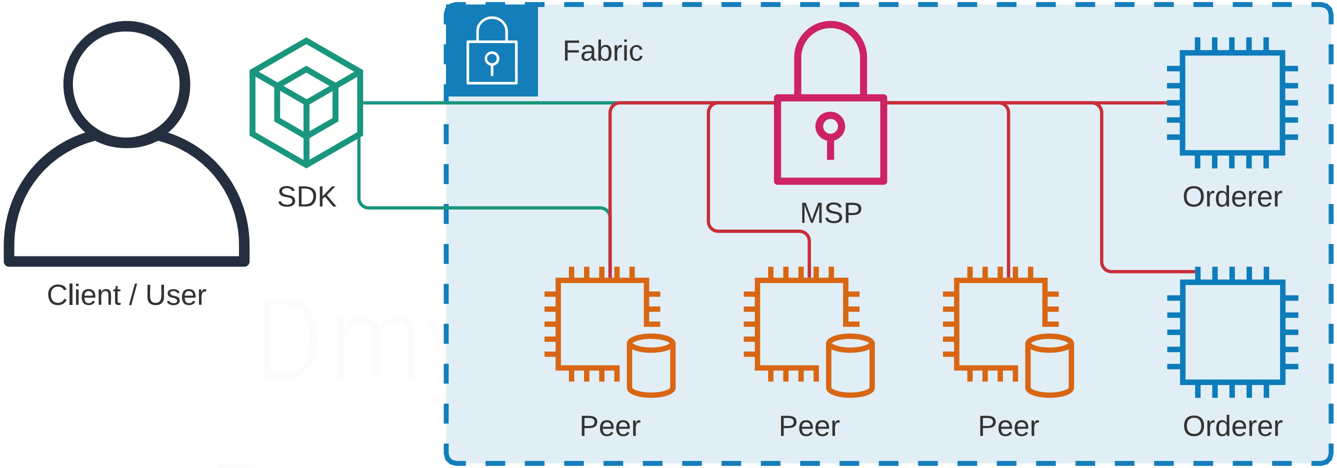
Peers execute and validate TXs

Orderers determine the order of TXs and distribute the blocks to peers

MSP maintains & manages IDs of all members

Execute-Order-Validate

- Client sends TX proposal to endorsers
- Endorsers execute TX, sign read/write sets
- Client prepares TX, sends to orderers
- Orderers puts TX in block, distributes
- All peers validate TX



Components

Clients invoke TXs and observe their results

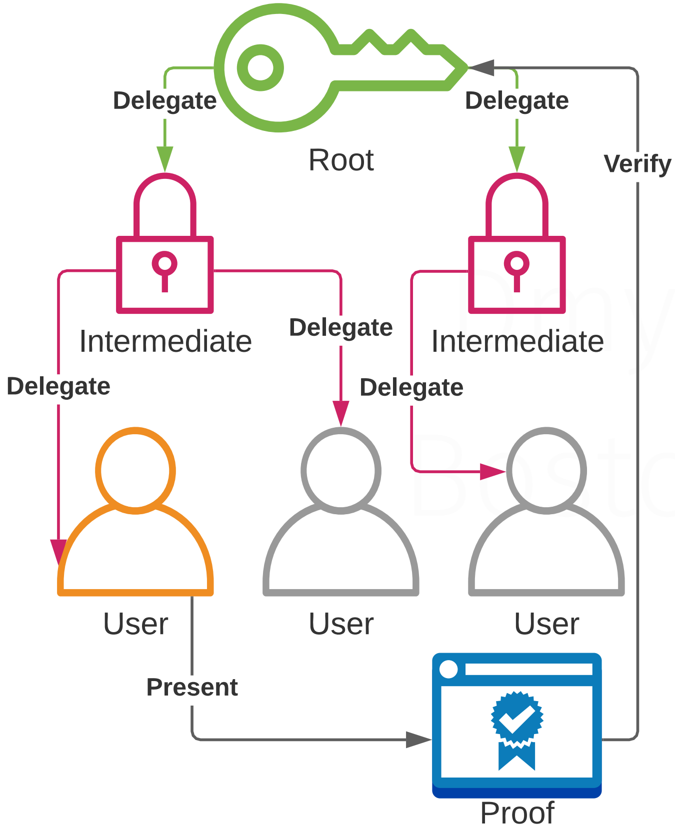
Peers execute and validate TXs

Orderers determine the order of TXs and distribute the blocks to peers

MSP maintains & manages IDs of all members

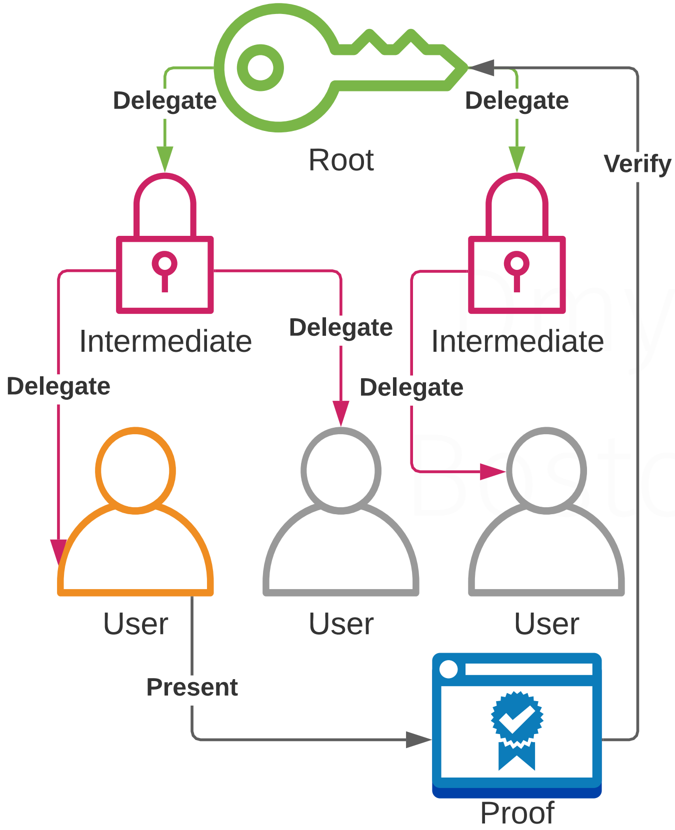
Execute-Order-Validate

- Client sends TX proposal to endorsers
- Endorsers execute TX, sign read/write sets
- Client prepares TX, sends to orderers
- Orderers puts TX in block, distributes
- All peers validate TX



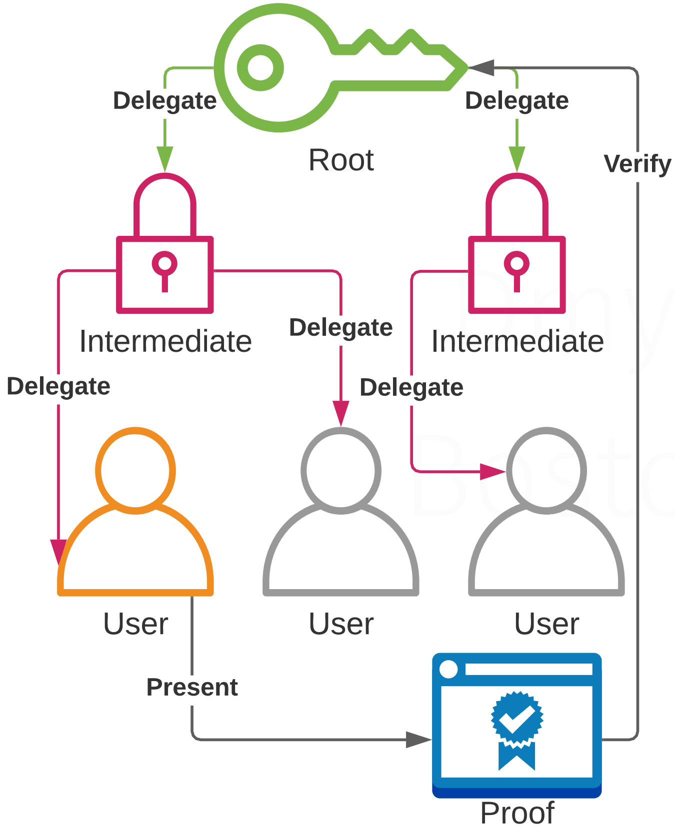
- $\textsc{KeyGen}(\mathrm{sp}) \rightarrow_{\$} (\mathrm{csk}, \mathrm{cpk})$
  generate a pair of keys for the caller (root, intermediate, user)
- $\textsc{Delegate}(\mathrm{csk}_i, \mathrm{cred}_i, \mathrm{cpk}_{i+1}, \vec{a}_{i+1}) \rightarrow_{\$} \mathrm{cred}_{i+1}$
  Level-i authority produces credentials of the *Level-$(i+1)$* binding attributes $\vec{a}_{i+1}$ to public key cpk_{i+1}
- $\textsc{Present}(\mathrm{csk}_L, \mathrm{cred}_L, \mathrm{cpk}_0, \langle a_{i,j} \rangle_{(i,j) \in D}, m) \rightarrow_{\$} \mathfrak{P}_{\mathrm{cred}}$
  shows the validity of cred_L under cpk_0, proves that secret key csk_L matches cred_L and disclosed attributes $\langle a_{i,j} \rangle_{(i,j) \in D}$, signs m
- $\textsc{Verify}(\mathfrak{P}_{\mathrm{cred}}, \mathrm{cpk}_0, \langle a_{i,j} \rangle_{(i,j) \in D}, m) \rightarrow \{0, 1\}$
  verifies the correctness of $\mathfrak{P}_{\mathrm{cred}}$ relative to disclosed attributes $\langle a_{i,j} \rangle_{(i,j) \in D}$, message m and public key cpk_0
- Instantiated with Groth [5] and Schnorr [1] signature schemes in [7]

Deleagtable Anonymous Credentials [7]



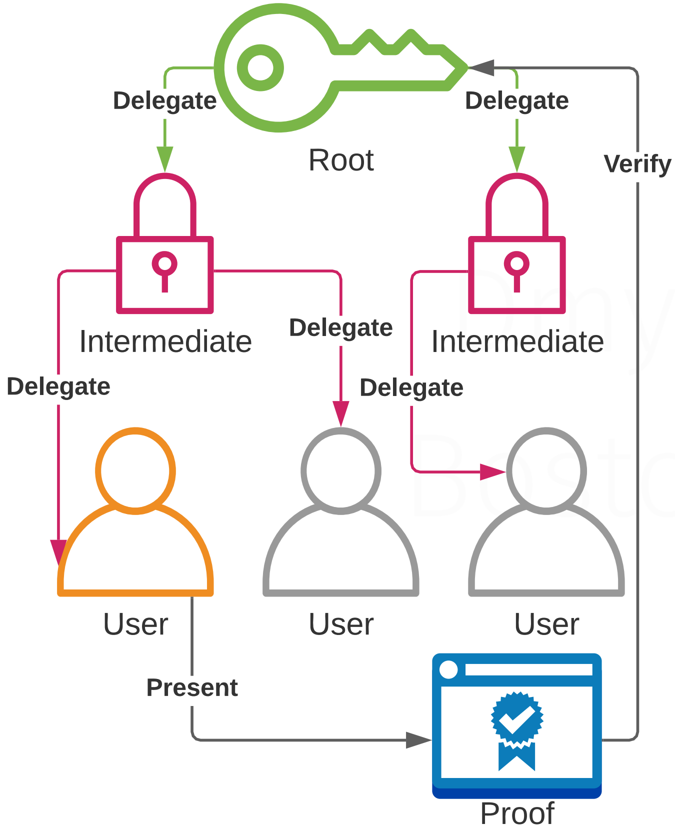
- $\textsc{KeyGen}(\mathsf{sp}) \rightarrow_{\$} (\mathsf{csk}, \mathsf{cpk})$
  generate a pair of keys for the caller (root, intermediate, user)
- $\textsc{Delegate}(\mathsf{csk}_i, \mathsf{cred}_i, \mathsf{cpk}_{i+1}, \vec{a}_{i+1}) \rightarrow_{\$} \mathsf{cred}_{i+1}$
  Level-i authority produces credentials of the *Level-*$(i+1)$ binding attributes $\vec{a}_{i+1}$ to public key cpk_{i+1}
- $\textsc{Present}(\mathsf{csk}_L, \mathsf{cred}_L, \mathsf{cpk}_0, \langle a_{i,j} \rangle_{(i,j) \in D}, m) \rightarrow_{\$} \mathfrak{P}_{\mathsf{cred}}$
  shows the validity of cred_L under cpk_0, proves that secret key csk_L matches cred_L and disclosed attributes $\langle a_{i,j} \rangle_{(i,j) \in D}$, signs m
- $\textsc{Verify}(\mathfrak{P}_{\mathsf{cred}}, \mathsf{cpk}_0, \langle a_{i,j} \rangle_{(i,j) \in D}, m) \rightarrow \{0, 1\}$
  verifies the correctness of $\mathfrak{P}_{\mathsf{cred}}$ relative to disclosed attributes $\langle a_{i,j} \rangle_{(i,j) \in D}$, message m and public key cpk_0
- Instantiated with Groth [5] and Schnorr [1] signature schemes in [7]

Deleagtable Anonymous Credentials [7]



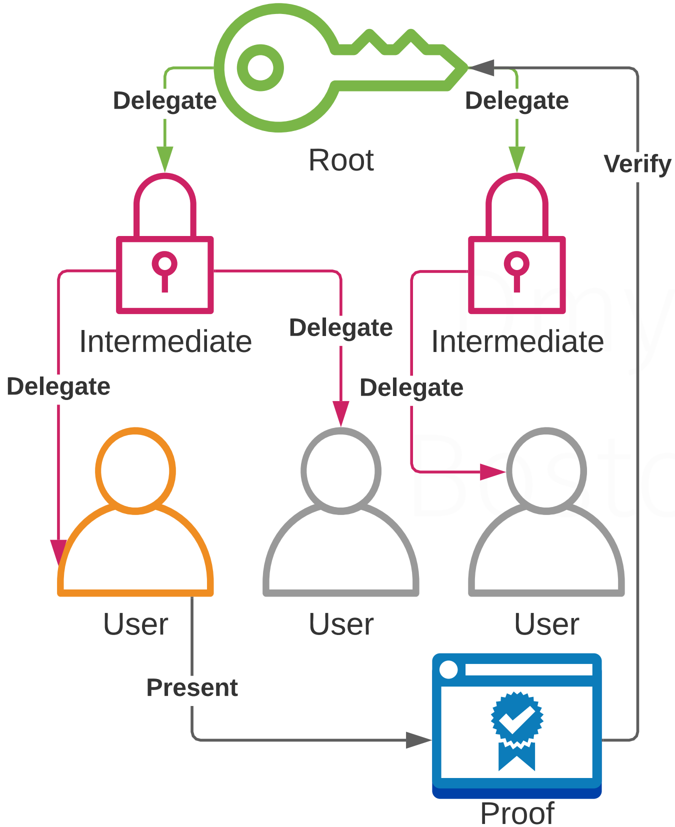
- $\textsc{KeyGen}(\mathsf{sp}) \rightarrow_{\$} (\mathsf{csk}, \mathsf{cpk})$
  generate a pair of keys for the caller (root, intermediate, user)
- $\textsc{Delegate}(\mathsf{csk}_i, \mathsf{cred}_i, \mathsf{cpk}_{i+1}, \vec{a}_{i+1}) \rightarrow_{\$} \mathsf{cred}_{i+1}$
  Level-i authority produces credentials of the *Level*-$(i+1)$ binding attributes $\vec{a}_{i+1}$ to public key cpk_{i+1}
- $\textsc{Present}(\mathsf{csk}_L, \mathsf{cred}_L, \mathsf{cpk}_0, \langle a_{i,j} \rangle_{(i,j)\in D}, m) \rightarrow_{\$} \mathfrak{P}_{\mathsf{cred}}$
  shows the validity of cred_L under cpk_0, proves that secret key csk_L matches cred_L and disclosed attributes $\langle a_{i,j} \rangle_{(i,j)\in D}$, signs m
- $\textsc{Verify}(\mathfrak{P}_{\mathsf{cred}}, \mathsf{cpk}_0, \langle a_{i,j} \rangle_{(i,j)\in D}, m) \rightarrow \{0, 1\}$
  verifies the correctness of $\mathfrak{P}_{\mathsf{cred}}$ relative to disclosed attributes $\langle a_{i,j} \rangle_{(i,j)\in D}$, message m and public key cpk_0
- Instantiated with Groth [5] and Schnorr [1] signature schemes in [7]

Deleagtable Anonymous Credentials [7]

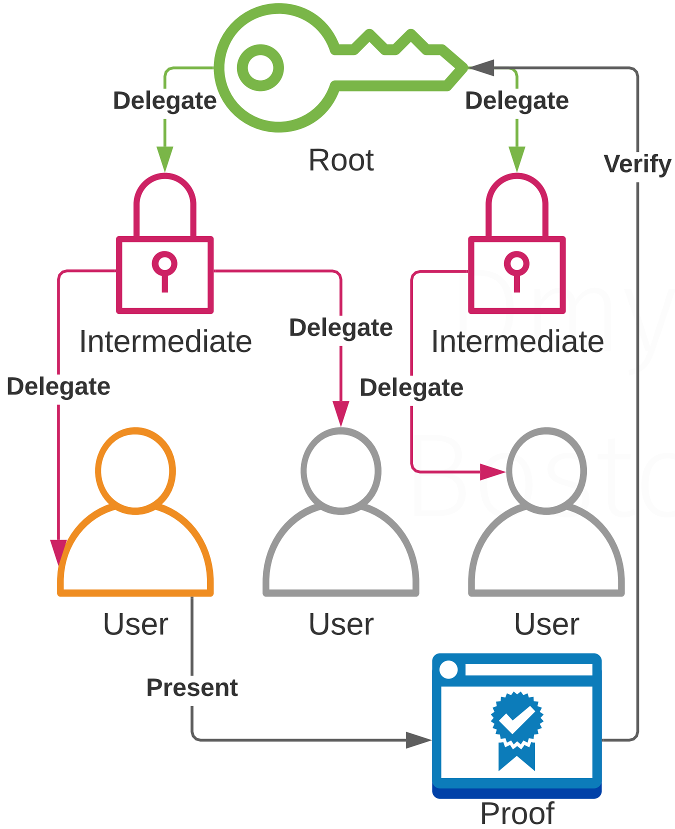


- $\textsc{KeyGen}(\mathsf{sp}) \rightarrow_\$ (\mathsf{csk}, \mathsf{cpk})$
  generate a pair of keys for the caller (root, intermediate, user)
- $\textsc{Delegate}(\mathsf{csk}_i, \mathsf{cred}_i, \mathsf{cpk}_{i+1}, \vec{a}_{i+1}) \rightarrow_\$ \mathsf{cred}_{i+1}$
  Level-i authority produces credentials of the *Level-*$(i+1)$ binding attributes $\vec{a}_{i+1}$ to public key cpk_{i+1}
- $\textsc{Present}(\mathsf{csk}_L, \mathsf{cred}_L, \mathsf{cpk}_0, \langle a_{i,j} \rangle_{(i,j)\in D}, m) \rightarrow_\$ \mathfrak{P}_{\mathsf{cred}}$
  shows the validity of cred_L under cpk_0, proves that secret key csk_L matches cred_L and disclosed attributes $\langle a_{i,j} \rangle_{(i,j)\in D}$, signs m
- $\textsc{Verify}(\mathfrak{P}_{\mathsf{cred}}, \mathsf{cpk}_0, \langle a_{i,j} \rangle_{(i,j)\in D}, m) \rightarrow \{0, 1\}$
  verifies the correctness of $\mathfrak{P}_{\mathsf{cred}}$ relative to disclosed attributes $\langle a_{i,j} \rangle_{(i,j)\in D}$, message m and public key cpk_0
- Instantiated with Groth [5] and Schnorr [1] signature schemes in [7]

Deleagtable Anonymous Credentials [7]



- $\textsc{KeyGen}(\mathsf{sp}) \rightarrow_{\$} (\mathsf{csk}, \mathsf{cpk})$
  generate a pair of keys for the caller (root, intermediate, user)
- $\textsc{Delegate}(\mathsf{csk}_i, \mathsf{cred}_i, \mathsf{cpk}_{i+1}, \vec{a}_{i+1}) \rightarrow_{\$} \mathsf{cred}_{i+1}$
  Level-i authority produces credentials of the *Level-*$(i+1)$ binding attributes $\vec{a}_{i+1}$ to public key cpk_{i+1}
- $\textsc{Present}(\mathsf{csk}_L, \mathsf{cred}_L, \mathsf{cpk}_0, \langle a_{i,j} \rangle_{(i,j) \in D}, m) \rightarrow_{\$} \mathfrak{P}_{\mathsf{cred}}$
  shows the validity of cred_L under cpk_0, proves that secret key csk_L matches cred_L and disclosed attributes $\langle a_{i,j} \rangle_{(i,j) \in D}$, signs m
- $\textsc{Verify}(\mathfrak{P}_{\mathsf{cred}}, \mathsf{cpk}_0, \langle a_{i,j} \rangle_{(i,j) \in D}, m) \rightarrow \{0, 1\}$
  verifies the correctness of $\mathfrak{P}_{\mathsf{cred}}$ relative to disclosed attributes $\langle a_{i,j} \rangle_{(i,j) \in D}$, message m and public key cpk_0
- Instantiated with Groth [5] and Schnorr [1] signature schemes in [7]



- $\textsc{KeyGen}(\mathsf{sp}) \rightarrow_{\$} (\mathsf{csk}, \mathsf{cpk})$
  generate a pair of keys for the caller (root, intermediate, user)
- $\textsc{Delegate}(\mathsf{csk}_i, \mathsf{cred}_i, \mathsf{cpk}_{i+1}, \vec{a}_{i+1}) \rightarrow_{\$} \mathsf{cred}_{i+1}$
  Level-i authority produces credentials of the *Level-*$(i+1)$ binding attributes $\vec{a}_{i+1}$ to public key cpk_{i+1}
- $\textsc{Present}(\mathsf{csk}_L, \mathsf{cred}_L, \mathsf{cpk}_0, \langle a_{i,j} \rangle_{(i,j) \in D}, m) \rightarrow_{\$} \mathfrak{P}_{\mathsf{cred}}$
  shows the validity of cred_L under cpk_0, proves that secret key csk_L matches cred_L and disclosed attributes $\langle a_{i,j} \rangle_{(i,j) \in D}$, signs m
- $\textsc{Verify}(\mathfrak{P}_{\mathsf{cred}}, \mathsf{cpk}_0, \langle a_{i,j} \rangle_{(i,j) \in D}, m) \rightarrow \{0, 1\}$
  verifies the correctness of $\mathfrak{P}_{\mathsf{cred}}$ relative to disclosed attributes $\langle a_{i,j} \rangle_{(i,j) \in D}$, message m and public key cpk_0
- Instantiated with Groth [5] and Schnorr [1] signature schemes in [7]

Improved Construction

Revocation

General approach

- Revocation is inherently at odds with anonymity
- We couple *epoch-based whitelisting* with signatures in a way that yields efficient proofs of non-revocation
- *Epochs* defined in terms of blockchain heigh
- *Epoch handle* (signature) binds public key to epoch

Instantiation

- Simple way: epoch as an attribute
- Explicit proof of non-revocation

  allows to decouple credential issuer and revocation authority

$$\varepsilon := g^{\textsc{Hash}(\text{epoch})}$$

$$\sigma \leftarrow_{\$} \textsc{Groth.Sign}(\text{rsk}; \varepsilon, \text{cpk})$$

$$\mathfrak{P} \leftarrow_{\$} \text{NIZK}\{(\sigma_{1,\ldots,L}, \text{cpk}_{1,\ldots,L}, \langle a_{i,j} \rangle_{(i,j)\notin D}, \sigma_m, \sigma) :$$

$$\bigwedge_{i=2,4,\ldots,L} \textsc{Groth}_1.\textsc{Verify}(\text{cpk}_{i-1}; \sigma_i; \text{cpk}_i, a_{i,1}, \ldots, a_{i,n_i})$$

$$\bigwedge_{i=1,3,\ldots,L} \textsc{Groth}_2.\textsc{Verify}(\text{cpk}_{i-1}; \sigma_i; \text{cpk}_i, a_{i,1}, \ldots, a_{i,n_i})$$

$$\wedge\ \textsc{Schnorr.Verify}(\text{cpk}_L; \sigma_m; m)$$

$$\boxed{\wedge \textsc{Groth.Verify}(\text{rpk}; \sigma; \varepsilon, \text{cpk}_L)}\}$$

Revocation

General approach

- Revocation is inherently at odds with anonymity
- We couple *epoch-based whitelisting* with signatures in a way that yields efficient proofs of non-revocation
- *Epochs* defined in terms of blockchain heigh
- *Epoch handle* (signature) binds public key to epoch

Instantiation

- Simple way: epoch as an attribute
- Explicit proof of non-revocation allows to decouple credential issuer and revocation authority

$$\varepsilon := g^{\textsc{Hash}(\text{epoch})}$$

$$\sigma \leftarrow_{\$} \textsc{Groth.Sign}(\text{rsk}; \varepsilon, \text{cpk})$$

$$\mathfrak{P} \leftarrow_{\$} \text{NIZK}\{(\sigma_{1,\dots,L}, \text{cpk}_{1,\dots,L}, \langle a_{i,j} \rangle_{(i,j) \notin D}, \sigma_m, \sigma) : \bigwedge_{i=2,4,\dots,L} \textsc{Groth}_1.\textsc{Verify}(\text{cpk}_{i-1}; \sigma_i; \text{cpk}_i, a_{i,1}, \dots, a_{i,n_i}) \bigwedge_{i=1,3,\dots,L} \textsc{Groth}_2.\textsc{Verify}(\text{cpk}_{i-1}; \sigma_i; \text{cpk}_i, a_{i,1}, \dots, a_{i,n_i}) \wedge \textsc{Schnorr.Verify}(\text{cpk}_L; \sigma_m; m) \wedge \textsc{Groth.Verify}(\text{rpk}; \sigma; \varepsilon, \text{cpk}_L)\}$$

Revocation

General approach

- Revocation is inherently at odds with anonymity
- We couple *epoch-based whitelisting* with signatures in a way that yields efficient proofs of non-revocation
- *Epochs* defined in terms of blockchain heigh
- *Epoch handle* (signature) binds public key to epoch

Instantiation

- Simple way: epoch as an attribute
- Explicit proof of non-revocation

  allows to decouple credential issuer and revocation authority

$$\varepsilon := g^{\textsc{Hash}(\text{epoch})}$$

$$\sigma \leftarrow_{\$} \textsc{Groth.Sign}(\text{rsk}; \varepsilon, \text{cpk})$$

$$\mathfrak{P} \leftarrow_{\$} \text{NIZK}\{(\sigma_{1,\ldots,L}, \text{cpk}_{1,\ldots,L}, \langle a_{i,j} \rangle_{(i,j)\notin D}, \sigma_m, \sigma) : \bigwedge_{i=2,4,\ldots,L} \textsc{Groth}_1.\textsc{Verify}(\text{cpk}_{i-1}; \sigma_i; \text{cpk}_i, a_{i,1}, \ldots, a_{i,n_i}) \bigwedge_{i=1,3,\ldots,L} \textsc{Groth}_2.\textsc{Verify}(\text{cpk}_{i-1}; \sigma_i; \text{cpk}_i, a_{i,1}, \ldots, a_{i,n_i}) \wedge \textsc{Schnorr.Verify}(\text{cpk}_L; \sigma_m; m) \boxed{\wedge \textsc{Groth.Verify}(\text{rpk}; \sigma; \varepsilon, \text{cpk}_L)}\}$$

General approach

- TX author embeds her ID (public key) encrypted under the auditor's public key
- Prove that the user **encrypts her own public key** and **uses the public key of the authorized auditor**

Instantiation

- User generates ElGamal secret and public keys $(\mathsf{ask}, \mathsf{apk} = g^{\mathsf{ask}})$
- Encrypts with $\mathsf{enc} = (\mathsf{cpk}_L \cdot \mathsf{apk}^{\rho}, g^{\rho})$
- Auditor decrypts enc

guaranteed to succeed

$$\mathfrak{P} \leftarrow_{\$} \mathsf{NIZK}\{(\sigma_{1,\dots,L}, \mathsf{cpk}_{1,\dots,L}, \langle a_{i,j} \rangle_{(i,j) \notin D}, \sigma_m, \sigma, \rho) :$$

$$\bigwedge_{i=2,4,\dots}^{L} \textsc{Groth}_1.\textsc{Verify}(\mathsf{cpk}_{i-1}; \sigma_i; \mathsf{cpk}_i, a_{i,1}, \dots, a_{i,n_i})$$

$$\bigwedge_{i=1,3,\dots}^{L} \textsc{Groth}_2.\textsc{Verify}(\mathsf{cpk}_{i-1}; \sigma_i; \mathsf{cpk}_i, a_{i,1}, \dots, a_{i,n_i})$$

$$\wedge\ \textsc{Schnorr}.\textsc{Verify}(\mathsf{cpk}_L; \sigma_m; m)$$

$$\wedge\ \textsc{Groth}.\textsc{Verify}(\mathsf{rpk}; \sigma; \varepsilon, \mathsf{cpk}_L)$$

$$\boxed{\wedge \mathsf{enc} = (\mathsf{cpk}_L \cdot \mathsf{apk}^{\rho}, g^{\rho})}\}$$

Audit

General approach

- TX author embeds her ID (public key) encrypted under the auditor's public key
- Prove that the user **encrypts her own public key** and **uses the public key of the authorized auditor**

Instantiation

- User generates ElGamal secret and public keys $(\mathsf{ask}, \mathsf{apk} = g^{\mathsf{ask}})$
- Encrypts with $\mathsf{enc} = (\mathsf{cpk}_L \cdot \mathsf{apk}^{\rho}, g^{\rho})$
- Auditor decrypts enc

guaranteed to succeed

$$\mathfrak{P} \leftarrow_{\$} \mathrm{NIZK}\{(\sigma_{1,\ldots,L}, \mathsf{cpk}_{1,\ldots,L}, \langle a_{i,j} \rangle_{(i,j)\notin D}, \sigma_m, \sigma, \rho) :$$

$$\bigwedge_{i=2,4,\ldots}^{L} \textsc{Groth}_1.\textsc{Verify}(\mathsf{cpk}_{i-1}; \sigma_i; \mathsf{cpk}_i, a_{i,1}, \ldots, a_{i,n_i})$$

$$\bigwedge_{i=1,3,\ldots}^{L} \textsc{Groth}_2.\textsc{Verify}(\mathsf{cpk}_{i-1}; \sigma_i; \mathsf{cpk}_i, a_{i,1}, \ldots, a_{i,n_i})$$

$$\wedge\ \textsc{Schnorr}.\textsc{Verify}(\mathsf{cpk}_L; \sigma_m; m)$$

$$\wedge\ \textsc{Groth}.\textsc{Verify}(\mathsf{rpk}; \sigma; \varepsilon, \mathsf{cpk}_L)$$

$$\boxed{\wedge \mathsf{enc} = (\mathsf{cpk}_L \cdot \mathsf{apk}^{\rho}, g^{\rho})}\}$$

General approach

- TX author embeds her ID (public key) encrypted under the auditor's public key
- Prove that the user **encrypts her own public key** and **uses the public key of the authorized auditor**

Instantiation

- User generates ElGamal secret and public keys $(\mathsf{ask}, \mathsf{apk} = g^{\mathsf{ask}})$
- Encrypts with $\mathsf{enc} = (\mathsf{cpk}_L \cdot \mathsf{apk}^\rho, g^\rho)$
- Auditor decrypts enc
  guaranteed to succeed

$$\mathfrak{P} \leftarrow_{\$} \mathrm{NIZK}\{(\sigma_{1,\ldots,L}, \mathsf{cpk}_{1,\ldots,L}, \langle a_{i,j}\rangle_{(i,j)\notin D}, \sigma_m, \sigma, \rho) :$$

$$\bigwedge_{i=2,4,\ldots}^{L} \textsc{Groth}_1.\textsc{Verify}(\mathsf{cpk}_{i-1}; \sigma_i; \mathsf{cpk}_i, a_{i,1}, \ldots, a_{i,n_i})$$

$$\bigwedge_{i=1,3,\ldots}^{L} \textsc{Groth}_2.\textsc{Verify}(\mathsf{cpk}_{i-1}; \sigma_i; \mathsf{cpk}_i, a_{i,1}, \ldots, a_{i,n_i})$$

$$\wedge\ \textsc{Schnorr}.\textsc{Verify}(\mathsf{cpk}_L; \sigma_m; m)$$

$$\wedge\ \textsc{Groth}.\textsc{Verify}(\mathsf{rpk}; \sigma; \varepsilon, \mathsf{cpk}_L)$$

$$\boxed{\wedge \mathsf{enc} = (\mathsf{cpk}_L \cdot \mathsf{apk}^\rho, g^\rho)}\}$$

Performance optimizations

- Simplified the pseudocode in [7] and corrected mistakes
- Parallelized on the granularity of commitments
- Optimal Miller's loop and final exponentiation

$$\prod_i e\left(a_i, b_i\right)^{c_i} = \text{FEXP}\left(\prod_i \hat{t}(a_i^{c_i}, b_i)\right) = \text{FEXP}\left(\prod_i \hat{t}(a_i, b_i^{c_i})\right)$$

Require: $a_i \in \mathbb{G}_1$, $b_i \in \mathbb{G}_2$, $c_i \in \mathbb{Z}_q \cup \perp$ for $L = 1, \ldots, n$

Ensure: $\text{EPRODUCT}(\langle a_i, b_i, c_i \rangle_{i=1}^{n}) = \prod_{i=1}^{n} e(a_i, b_i)^{c_i}$

procedure $\text{EPRODUCT}(\langle a_i, b_i, c_i \rangle_{i=1}^{n})$
  $r := 1_T \in \mathbb{G}_T$ ▷ an *identity element*
  for $i = (1, \ldots, n)$ **do**
    if $c_i \neq \perp$ **then**
      $a_i := a_i^{c_i}$
  for $i = (1, 3, \ldots, n)$ **do**
    if $a_{i+1} \neq \perp$ **then**
      ▷ $\hat{t}_2$ is a more efficient version of $\hat{t} \cdot \hat{t}$
      $r := r \cdot \hat{t}_2\,(a_i, b_i, a_{i+1}, b_{i+1})$
    else
      $r := r \cdot \hat{t}\,(a_i, b_i)$
  return $\text{FEXP}\,(r)$

Performance optimizations

- Simplified the pseudocode in [7] and corrected mistakes
- Parallelized on the granularity of commitments
- Optimal Miller's loop and final exponentiation

$$\prod_i e\left(a_i, b_i\right)^{c_i} = \text{FEXP}\left(\prod_i \hat{t}(a_i^{c_i}, b_i)\right) = \text{FEXP}\left(\prod_i \hat{t}(a_i, b_i^{c_i})\right)$$

Require: $a_i \in \mathbb{G}_1$, $b_i \in \mathbb{G}_2$, $c_i \in \mathbb{Z}_q \cup \perp$ for $L = 1, \ldots, n$

Ensure: $\text{EPRODUCT}(\langle a_i, b_i, c_i \rangle_{i=1}^{n}) = \prod_{i=1}^{n} e(a_i, b_i)^{c_i}$

```
procedure EPRODUCT(⟨a_i, b_i, c_i⟩_{i=1}^n)
    r := 1_T ∈ G_T                                ▷ an identity element
    for i = (1, ..., n) do
        if c_i ≠ ⊥ then
            a_i := a_i^{c_i}
    for i = (1, 3, ..., n) do
        if a_{i+1} ≠ ⊥ then
            ▷ t̂_2 is a more efficient version of t̂ · t̂
            r := r · t̂_2 (a_i, b_i, a_{i+1}, b_{i+1})
        else
            r := r · t̂ (a_i, b_i)
    return FEXP (r)
```

Performance optimizations

- Simplified the pseudocode in [7] and corrected mistakes
- Parallelized on the granularity of commitments
- Optimal Miller's loop and final exponentiation

$$\prod_i e\left(a_i, b_i\right)^{c_i} = \text{FEXP}\left(\prod_i \hat{t}(a_i^{c_i}, b_i)\right) = \text{FEXP}\left(\prod_i \hat{t}(a_i, b_i^{c_i})\right)$$

Require: $a_i \in \mathbb{G}_1$, $b_i \in \mathbb{G}_2$, $c_i \in \mathbb{Z}_q \cup \perp$ for $L = 1, \ldots, n$

Ensure: $\textsc{eProduct}(\langle a_i, b_i, c_i \rangle_{i=1}^{n}) = \prod_{i=1}^{n} e(a_i, b_i)^{c_i}$

procedure $\textsc{eProduct}(\langle a_i, b_i, c_i \rangle_{i=1}^{n})$
    $r := 1_T \in \mathbb{G}_T$ ▷ an *identity element*
    for $i = (1, \ldots, n)$ **do**
        if $c_i \neq \perp$ **then**
            $a_i := a_i^{c_i}$
    for $i = (1, 3, \ldots, n)$ **do**
        if $a_{i+1} \neq \perp$ **then**
            ▷ $\hat{t}_2$ is a more efficient version of $\hat{t} \cdot \hat{t}$
            $r := r \cdot \hat{t}_2\left(a_i, b_i, a_{i+1}, b_{i+1}\right)$
        else
            $r := r \cdot \hat{t}\left(a_i, b_i\right)$
    return $\text{FEXP}\left(r\right)$

Setup

	Level-i CA		Level-$(i+1)$ CA
1 :	***Level-i* CA**		***Level-$(i+1)$* CA**

. . . . Repeated for L rounds of delegation (from the Root CA to Intermediate CAs to the User) . . . .

	Level-i CA	Message	Level-$(i+1)$ CA
2 :	$\mathsf{csk}_i \leftarrow\$ \mathbb{Z}_q, \mathsf{cpk}_i := g^{\mathsf{csk}_i}$		$\mathsf{csk}_{i+1} \leftarrow\$ \mathbb{Z}_q, \mathsf{cpk}_{i+1} := f^{\mathsf{csk}_{i+1}}$
3 :	$\mathsf{nonce} \leftarrow\$ \{0,1\}^\lambda$	nonce $\longrightarrow$	$\mathfrak{P}_{\mathsf{pk}} \leftarrow\$ \textsc{ProvePK}(\mathsf{csk}_{i+1}, \mathsf{cpk}_{i+1}, \mathsf{nonce})$
4 :	$\textsc{VerifyPK}(\mathfrak{P}_{\mathsf{pk}}, \mathsf{cpk}_{i+1}, \mathsf{nonce})$	$\longleftarrow$ $\mathfrak{P}_{\mathsf{pk}}, \mathsf{cpk}_{i+1}$	
5 :	$\sigma_{i+1} \leftarrow\$ \textsc{Groth.Sign}(\mathsf{csk}_i; \mathsf{cpk}_{i+1}, \vec{a}_{i+1})$	σ_{i+1} $\longrightarrow$	$\mathsf{cred}_{i+1} := (\sigma_{i+1}, \vec{a}_{i+1}, \mathsf{cpk}_{i+1})$

Revocation

	Revocation authority		User
1 :	**Revocation authority**		**User**
	... On each epoch, user requests a non-revocation handle ...		
2 :	$\mathsf{rsk} \leftarrow_{\$} \mathbb{Z}_q, \mathsf{rpk} := g^{\mathsf{rsk}}$		$\mathsf{csk} \leftarrow_{\$} \mathbb{Z}_q, \mathsf{cpk} := g^{\mathsf{csk}}$
3 :	$\mathsf{nonce} \leftarrow_{\$} \{0,1\}^{\lambda}$	$\xrightarrow{\mathsf{nonce}}$	$\mathfrak{P}_{\mathsf{pk}} \leftarrow_{\$} \textsc{ProvePK}(\mathsf{csk}, \mathsf{cpk}, \mathsf{nonce})$
4 :	$\textsc{VerifyPK}(\mathfrak{P}_{\mathsf{pk}}, \mathsf{cpk}, \mathsf{nonce})$	$\xleftarrow{\mathfrak{P}_{\mathsf{pk}}, \mathsf{cpk}}$	
5 :	$\sigma \leftarrow_{\$} \textsc{NRSign}(\mathsf{rsk}; \mathsf{cpk}, \mathsf{epoch})$	$\xrightarrow{\sigma}$	σ, epoch

Transaction submission

Verifier		User
		cred $:= (\langle \sigma_j, \vec{a}_j, \mathsf{cpk}_j \rangle_{j=1}^{L})$

User submits a transaction

Verifier		User
		$\mathsf{enc}, \rho := \textsc{AuditEnc}(\mathsf{apk}, \mathsf{cpk})$
		$\mathsf{sk}_{\mathsf{nym}}, \mathsf{pk}_{\mathsf{nym}} \leftarrow\$ \textsc{MakeNym}(\mathsf{csk})$
		$\mathfrak{P}_{\mathsf{rev}} \leftarrow\$ \textsc{NRProve}(\sigma, \mathsf{csk}, \mathsf{sk}_{\mathsf{nym}}, \mathsf{epoch})$
		$\mathfrak{P}_{\mathsf{audit}} \leftarrow\$ \textsc{AuditProve}(\mathsf{enc}, \rho, \mathsf{cpk}, \mathsf{csk}, \mathsf{pk}_{\mathsf{nym}}, \mathsf{sk}_{\mathsf{nym}})$
		$\mathfrak{P}_{\mathsf{cred}} \leftarrow\$ \textsc{CredProve}(\mathsf{cred}, D, \mathsf{sk}_{\mathsf{nym}}, \mathsf{csk}, \bot)$
		$\sigma_{\mathsf{nym}} \leftarrow\$ \textsc{SignNym}(\mathsf{pk}_{\mathsf{nym}}, \mathsf{sk}_{\mathsf{nym}}, \mathsf{csk}, \mathsf{tx})$
$(\mathfrak{P}_{\mathsf{cred}}, \mathfrak{P}_{\mathsf{rev}}, \mathfrak{P}_{\mathsf{audit}}, \mathsf{enc}, \mathsf{tx}, \mathsf{pk}_{\mathsf{nym}})$	$\xleftarrow{\ m, \sigma_{\mathsf{nym}}\ }$	$m := (\mathfrak{P}_{\mathsf{cred}}, \mathfrak{P}_{\mathsf{rev}}, \mathfrak{P}_{\mathsf{audit}}, \mathsf{enc}, \mathsf{tx}, \mathsf{pk}_{\mathsf{nym}})$
$\textsc{VerifyNym}(\mathsf{pk}_{\mathsf{nym}}, \mathsf{tx}, \sigma_{\mathsf{nym}})$		
$\textsc{NRVerify}(\mathfrak{P}_{\mathsf{rev}}, \mathsf{pk}_{\mathsf{nym}}, \mathsf{epoch})$		
$\textsc{AuditVerify}(\mathfrak{P}_{\mathsf{audit}}, \mathsf{enc}, \mathsf{pk}_{\mathsf{nym}})$		
$\textsc{CredVerify}(\mathfrak{P}_{\mathsf{cred}}, D, \mathsf{pk}_{\mathsf{nym}}, \bot)$		

Experimental Evaluation

Setup

- Implemented a stand-alone production-ready library in Go [9]
  470 tests cover 100 % of the code
- Apache Milagro Cryptographic Library (AMCL) [11] with a 254-bit Barreto-Naehrig curve [2]
- Benchmarks run on `c2-standard-60` GCE VM running Ubuntu 18.04
  all benchmarked operations were run 100 times
- Default number of levels and attributes per level are $L = 2$ and $n = 2$
- Implemented a distributed prototype of Fabric using our scheme
- Setup is different from the base scheme in [7]
- Aiming to answer 6 evaluation questions

Question 1: what is the optimizations' performance benefit?

e-product	Parallelization	CredProve Big	CredProve Small	CredVerify Big	CredVerify Small
disabled	disabled	2 873	843	1 523	948
enabled	disabled	1 312	341	853	372
disabled	enabled	1 480	357	890	352
enabled	enabled	890	191	391	197
Improvement ($\approx$ times)		3.2	4.4	3.9	**4.8**

Optimizations benchmark for $L = 2$ and $n = 2$ (small) and $L = 5$ and $n = 3$ (big).
The values are in milliseconds.

Question 2: how does the scheme scale with the number of levels and attributes?

L \ n	0	1	2	3	4
1	41 ms 89 ms 398 B	51 ms 110 ms 534 B	63 ms 116 ms 670 B	72 ms 153 ms 806 B	82 ms 173 ms 942 B
2	94 ms 124 ms 801 B	138 ms 158 ms 1.2 kB	192 ms 198 ms 1.6 kB	255 ms 262 ms 2.0 kB	315 ms 310 ms 2.4 kB
3	173 ms 188 ms 1.2 kB	273 ms 249 ms 1.7 kB	367 ms 329 ms 2.3 kB	516 ms 387 ms 2.8 kB	616 ms 427 ms 3.3 kB
5	333 ms 276 ms 2.0 kB	542 ms 342 ms 2.9 kB	661 ms 391 ms 3.9 kB	891 ms 500 ms 4.8 kB	1 146 ms 648 ms 5.7 kB

Parameters benchmark. In each cell the top value is a proof generation overhead, the middle value is a proof verification overhead and the bottom value is the proof size.

Question 3: what overhead do our extensions impose?

Procedure	Time $\mathbb{G}_1$	Time $\mathbb{G}_2$	Procedure	Time $\mathbb{G}_1$	Time $\mathbb{G}_2$
Groth.KeyGen	1.6	4.7	Groth.Sign	16	41
Groth.Randomize	11	23	Groth.Verify	53	62
Schnorr.Sign	1.6	4.8	Schnorr.Verify	2	9.6
AuditEncrypt	3	9.4	NRSign	14	30
AuditProve	5.8	24	NRProve	66	88
AuditVerify	9.2	39	NRVerify	127	149
MakeNym	2.1	9.4	ProvePK	3.1	9.4
SignNym	2.2	9.9	VerifyPK	2	9.5
VerifyNym	3.5	14	KeyGen	1.5	4.2

Running time of extensions in milliseconds.

Question 4: how does the system compare to the old non-delegatable idemix?

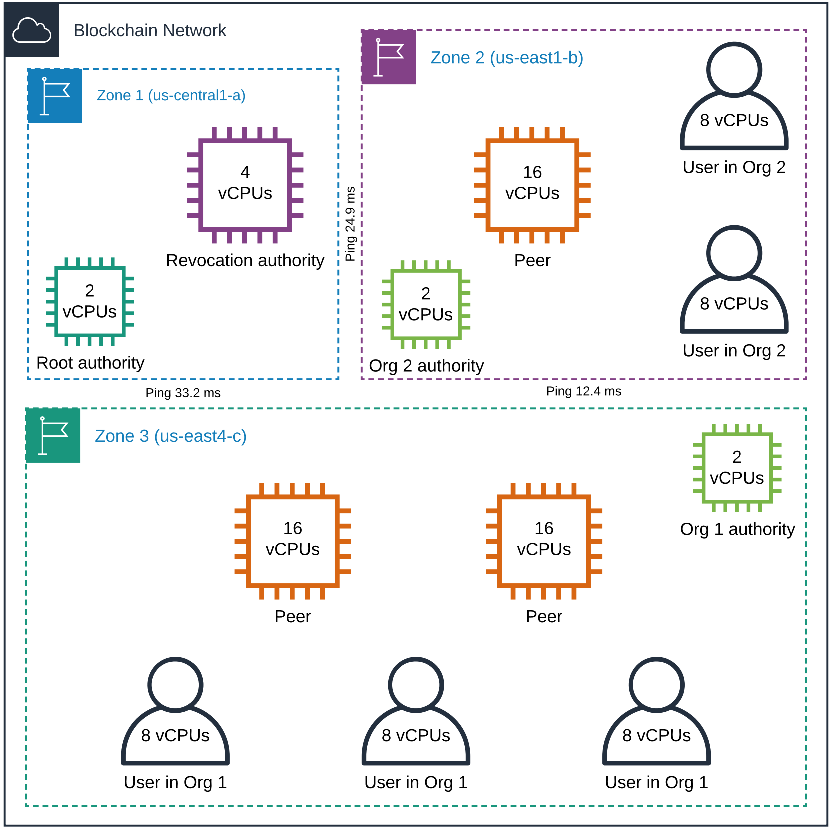
- Ran workloads against current (non-delegatable) Fabric idemix and regular MSP
  generating secrets, signing and verifying identities
- Idemix in Fabric [6] uses BBS+ signatures [3]
  ran actual Fabric code
- Default (not-idemix) Fabric MSP simply uses X.509 certificates and ECDSA algorithms [4]
  ran ECDSA routines in Go `crypto` module using P-384 curve — most secure option in Fabric
- Results show the relative costs of using more privacy-preserving solutions
  - default MSP takes 21 ms
  - idemix MSP in Fabric takes 108 ms
  - our solution takes 210 ms

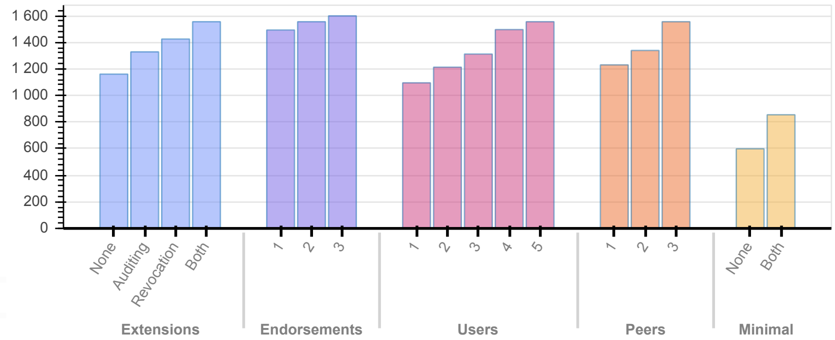
Question 5: how practical is maintaining a single and possibly distributed revocation authority?

Network log visualization (subset is shown, 18 transactions). Interval size is 20 ms. Experiment involves 5 users, 3 peers, 2 endorsements, 20 KiB/s and 50 KiB/s local and global bandwidths, and epoch length 5 seconds. Bars show objects in the network, lines show latencies (green for ideal, red for real). Latency scale is logarithmic.

- Despite short epochs (5 seconds), revocation requests do not result in any spikes in latency
- Overhead of issuing the non-revocation handle is 15 ms–30 ms — a fraction of total overhead
- We observed a stable 200 requests per second throughput on our testing machine

Question 6: what is the efficacy of the entire blockchain stack using our protocol?





- Number of endorsements does not significantly affect the overhead
  endorsements are processed in parallel
- Number of users influences the overhead substantially
  Each user increases TXs validated by a single peer
- Number of peers is positively correlated with the overhead
  TX is completed when the *last* peer validates it

Anonymous Transactions with Revocation and Auditing in Hyperledger Fabric

Anonymous Credentials, Revocation, Auditing, Blockchain

[10] DOI: 10.1007/978-3-030-92547-5_23

Dmytro Bogatov, Angelo De Caro, Kaoutar Elkhiyaoui, Björn Tackmann
dmytro@bu.edu, adc@zurich.ibm.com, kao@zurich.ibm.com
bjoern@dfinity.org

Built from *120359ce* on November 21, 2021

Boston University
Graduate School of Arts and Sciences
Department of Computer Science

References

[1] Claus P. Schnorr. "Efficient identification and signatures for smart cards". In: *Advances in Cryptology — CRYPTO*. Ed. by Gilles Brassard. Vol. 435. LNCS. Springer, 1989, pp. 239–252.

[2] Paulo SLM Barreto and Michael Naehrig. "Pairing-friendly elliptic curves of prime order". In: *International Workshop on Selected Areas in Cryptography*. Springer. 2005, pp. 319–331.

[3] Man Ho Au, Willy Susilo, and Yi Mu. "Constant-size dynamic k-TAA". In: *International conference on security and cryptography for networks*. Springer. 2006, pp. 111–125.

[4] National Institute of Standards and Technology. *FIPS PUB 186-4: Digital Signature Standard*. National Institute for Standards and Technology, July 2013. URL: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf.

[5] Jens Groth. “Efficient fully structure-preserving signatures for large messages”. In: *International Conference on the Theory and Application of Cryptology and Information Security*. Springer. 2015, pp. 239–259.

[6] Jan Camenisch, Manu Drijvers, and Anja Lehmann. “Anonymous Attestation Using the Strong Diffie-Hellman Assumption Revisited”. In: *Trust and Trustworthy Computing*. Springer International Publishing, 2016, pp. 1–20.

[7] Jan Camenisch, Manu Drijvers, and Maria Dubovitskaya. “Practical UC-secure delegatable credentials with attributes and their application to blockchain”. In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM. 2017, pp. 683–699.

[8] Elli Androulaki *et al.* “Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains”. In: *Proceedings of the Thirteenth EuroSys Conference*. EuroSys ’18. 2018, 30:1–30:15. DOI: 10.1145/3190508.3190538.

[9] Dmytro Bogatov. *Delegatable Anonymous Credentials library*. https://github.com/dbogatov/dac-lib. 2021.

[10] Dmytro Bogatov *et al.* “Anonymous Transactions with Revocation and Auditing in Hyperledger Fabric”. In: *Cryptology and Network Security*. Springer International Publishing, 2021. DOI: 10.1007/978-3-030-92547-5_23.

[11] Michael Scott. “The Apache Milagro Crypto Library”. In: (). URL: https://github.com/MIRACL/amcl.