MeeSign – Threshold Cryptography Platform

Tartu Workshop 2024

Antonín Dufka (PhD student; threshold cryptography on smartcards)
Jan Kvapil (PhD student; cryptographic verification)
Jakub Janků, Kristián Mika, Jiří Gavenda, Petr Švenda

Centre for Research on Cryptography and Security

Outline

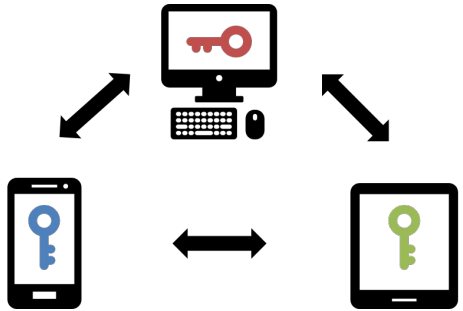
- Part 1 (14:15–15:45)
  - Threshold cryptography
  - MeeSign – threshold cryptography demonstration platform
  - Hands-on and discussion
- Break (15:45–16:15)
- Part 2 (16:15–17:45)
  - Advanced setups hands-on
  - Integrations
  - X-Road

Threshold cryptography

Threshold cryptography

- A concept mainly developed in the 90's is now starting to be used in modern applications
  - Cryptocurrencies (multi-signature wallets)
  - Authentication (eIDAS-compliant – SplitKey)
- A convenient private key protection method
  - Decentralization of storage
  - Elimination of single point of failure
  - Backward compatibility with existing systems

Single-party computation

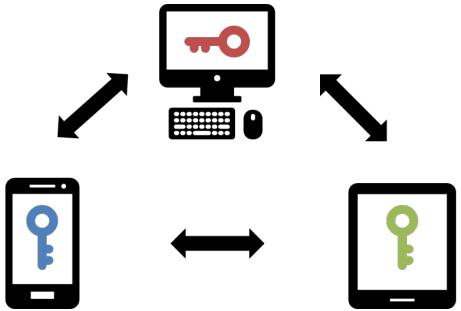


Threshold computation

Secure secret sharing

- Splitting a secret into **n** shares
  - At least **t** shares are needed to reconstruct the secret
  - Less than **t** shares gives *no information* about the secret
  - Can be created “bottom-up” using distributed key generation
- Additive secret sharing (**n**-of-**n**)
  - All shares are needed
  - Can be computed non-interactively
  - More efficient
- Shamir’s secret sharing (**t**-of-**n**)
  - Only **t** shares are needed to reconstruct the secret

Single-party computation



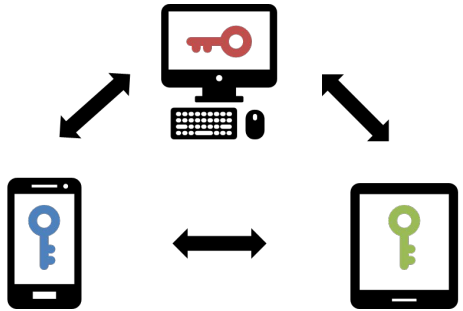
Threshold computation

Threshold protocols

- Devices operate with secret shares directly
  - No need to reconstruct the secret
- Output is indistinguishable from a single-party output

Single-party computation

- Specialized protocols for cryptographic algorithms
  - Signing: RSA, ECDSA, EdDSA, …
  - Decryption: ElGamal, ECIES, …
- Protocols typically require
  - multiple communication rounds
  - more complex computation compared to the base operation



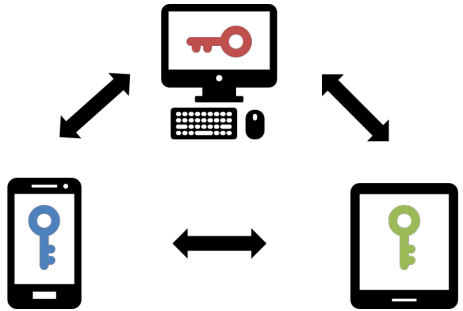
Threshold computation

Practical problems of threshold protocols

- Network communication
  - Peer discovery
  - Authentication of parties (or other trust mechanism)
- Protocol coordination
  - Selection of protocol participants
  - Synchronization of protocol rounds
  - Relaying of messages
- Multi-platform support
  - Running protocols on various devices (smartphones, PCs, …)
- Integration with existing applications

- Trying threshold protocols out is not trivial

Single-party computation

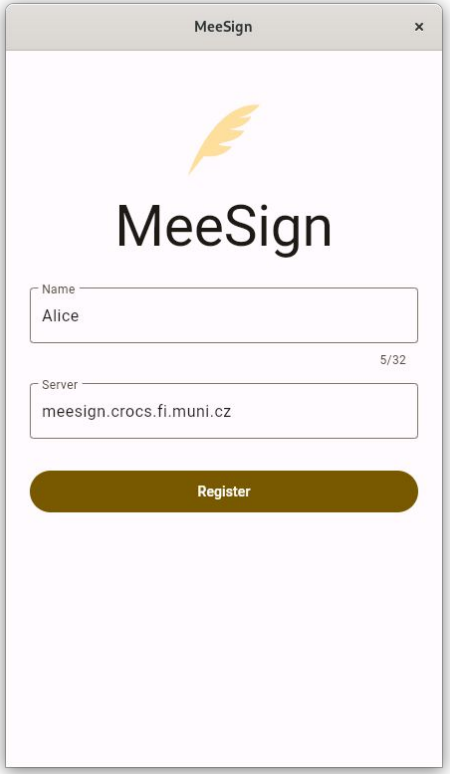


Threshold computation

MeeSign

MeeSign

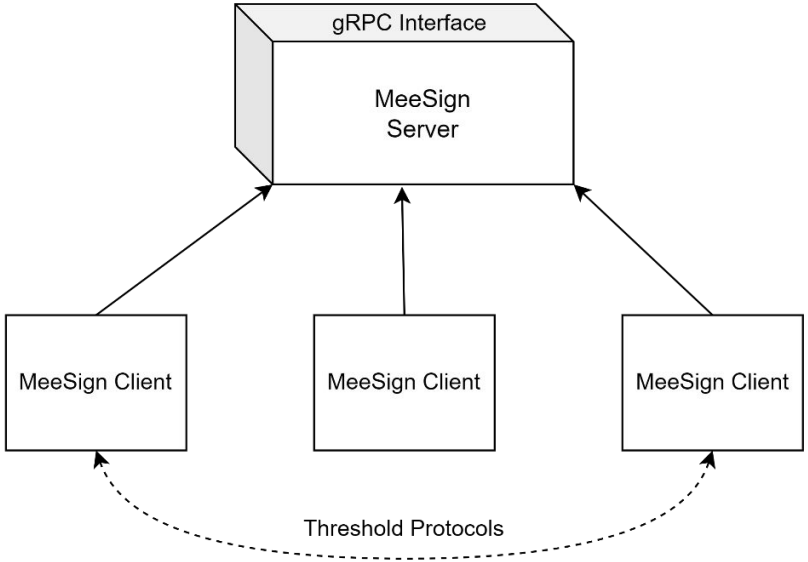
- Threshold cryptography demonstration platform
- User-friendly client application
  - Multi-platform (Android, Windows, MacOS, Linux, soon iOS)
  - Performs threshold cryptography protocols
  - Stores the secret keys (or shares)
- Semi-trusted intermediating server
  - Peer discovery
  - Authentication
  - Protocol coordination
  - Interface for operation requests



Architecture

MeeSign Server

A server facilitating communication among clients and coordination of threshold protocols.



gRPC Interface

A message-based interface for easy integration of MeeSign with other systems.

MeeSign Client

A client application which is registered with a MeeSign Server. The application controls users' private keys and uses them in threshold cryptography protocols with other MeeSign Clients.

Threshold Protocols

Multi-party protocols for signing, decryption, randomness generation running on MeeSign Clients coordinated by MeeSign Server.

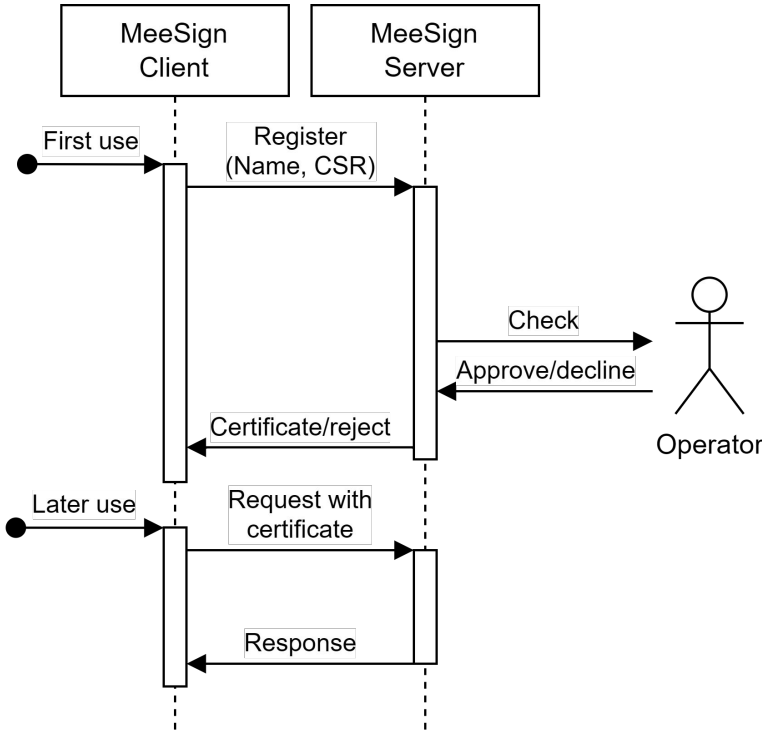
Network communication

- gRPC service
  - Request management
  - Client communication
- Binary protocol
- Easy integration with all common languages using gRPC compiler
- HTTP/2 bidirectional streams used for reducing client latency

```
service MeeSign {
  rpc GetServerInfo(ServerInfoRequest) returns (ServerInfo);
  rpc Register(RegistrationRequest) returns (RegistrationResponse);
  rpc Sign(SignRequest) returns (Task);
  rpc Group(GroupRequest) returns (Task);
  rpc Decrypt(DecryptRequest) returns (Task);
  rpc GetTask(TaskRequest) returns (Task);
  rpc UpdateTask(TaskUpdate) returns (Resp);
  rpc DecideTask(TaskDecision) returns (Resp);
  rpc AcknowledgeTask(TaskAcknowledgement) returns (Resp);
  rpc GetTasks(TasksRequest) returns (Tasks);
  rpc GetGroups(GroupsRequest) returns (Groups);
  rpc GetDevices(DevicesRequest) returns (Devices);
  rpc Log(LogRequest) returns (Resp);
  rpc SubscribeUpdates(SubscribeRequest) returns (stream Task);
}
```

Client registration and authentication

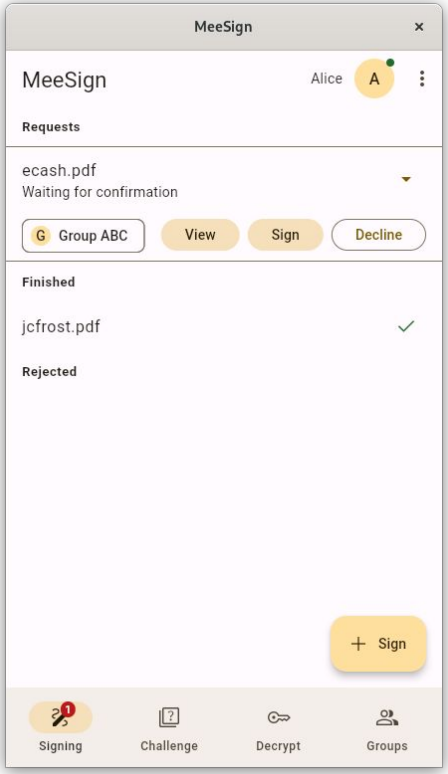
- TLS-based authentication
- Client needs to register on the first use:
  - Client sends its name and CSR to the server
  - Operator checks the request and decides if a certificate should be issued
  - Certificate is sent to the user
- Further communication performs mutual authentication using the certificate and the corresponding private key



Communicating operation requests

- Server provides API for issuing operation requests

- Operation requests are relayed to clients
  - Group establishment (distributed key generation)
  - Signing (request threshold signature by a group)
  - Decryption (request threshold decryption by a group)
- Users may view the received request
- Users can cast their vote (approve / decline)
- The protocol task starts being executed when sufficient number of votes is received

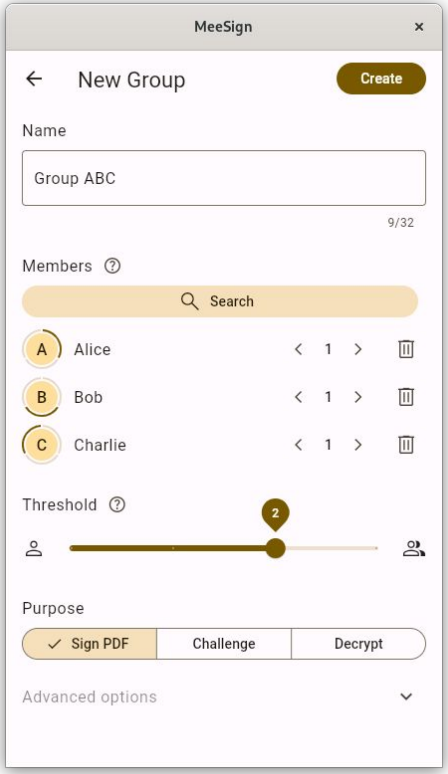


Protocol coordination

- Protocols supported by the platform are
  - threshold (with arbitrary threshold)
  - non-robust (can fail if someone refuses to cooperate)
  - synchronous (require synchronization between rounds)
- The server coordinates a protocol execution:
  - Selects a threshold of participants which will be involved in its execution
  - Waits until all selected parties respond and exchanges their messages
  - Repeats until the protocol is completed (depends on the number of rounds)
- Robustness is approached by restarts
  - In case some parties do not cooperate, server restarts the protocol with a new set of parties
  - Future improvements:
    - Keep state in case some of the parties start cooperating again
    - Some protocols support constructions like ROAST wrapper (Ruffing et al., 2023)

Group establishment

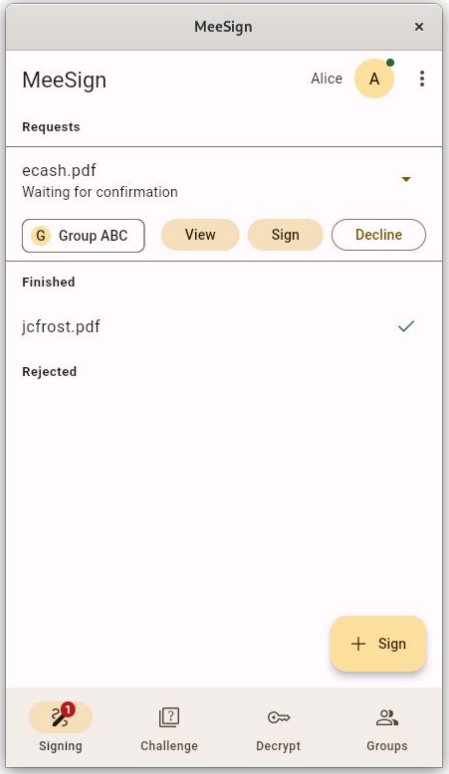
- Groups are sets of devices that can be issued requests
- A group establishment requires setting following properties:
  - name – human-readable identifier of the group
  - members – selection of devices that will participate in the group
  - threshold – the minimal number of parties needed
  - purpose – the type of tasks the group supports
    - signing a PDF, signing a challenge, decryption
  - protocol – the underlying threshold protocol that is used
    - GG18 (ECDSA), FROST (Schnorr), threshold ElGamal
- Advanced features
  - share weighting – a party can have more shares (higher voting rights)
  - policy-based automated parties – configurable during group creation
  - smartcard support – storing shares on a smartcard to improve security



Issuing operation requests

- Operations can be requested from multiple sources

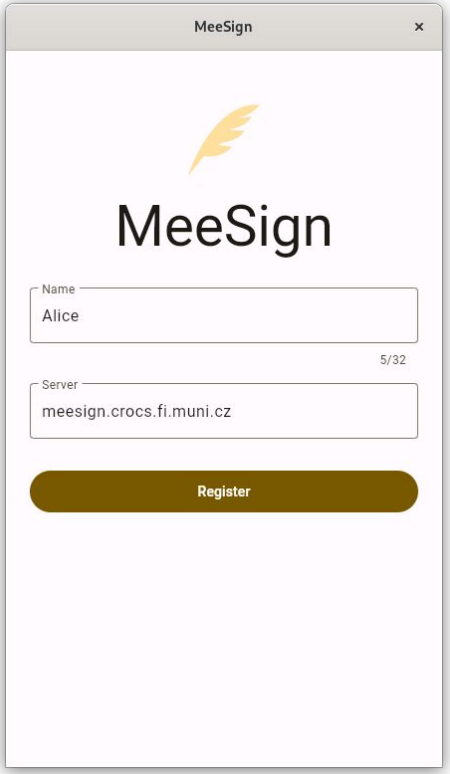
- From the MeeSign client application
  - Sending requests to groups in which a client is involved
  - Outputs are accessible only within the application
- From other applications (integrations)
  - Custom gRPC-based extensions (NextCloud)
  - Virtual tokens for cryptographic interfaces (Cryptoki, FIDO2, Web eID)



Hands-on

Setup

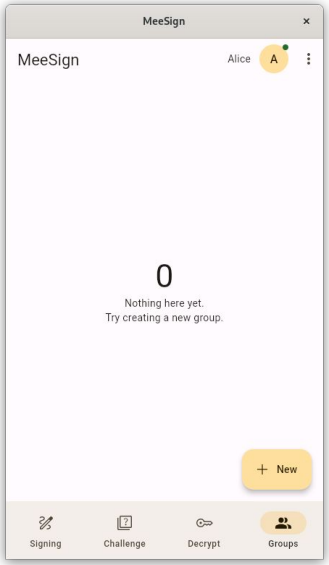
- Run MeeSign client application
  - Available from https://meesign.crocs.fi.muni.cz/
  - Or use the provided VM https://is.muni.cz/go/meesign-vm
- Register into the server instance running at MUNI
  - Input your name
  - Keep server set to `meesign.crocs.fi.muni.cz`
  - Press the Register button

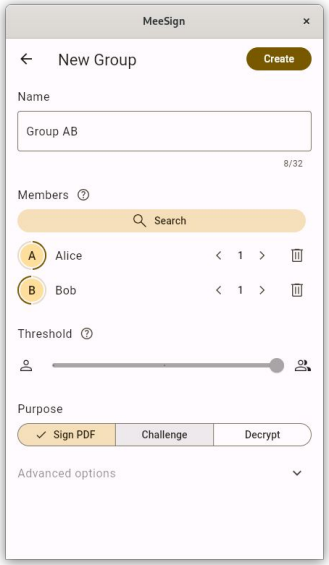


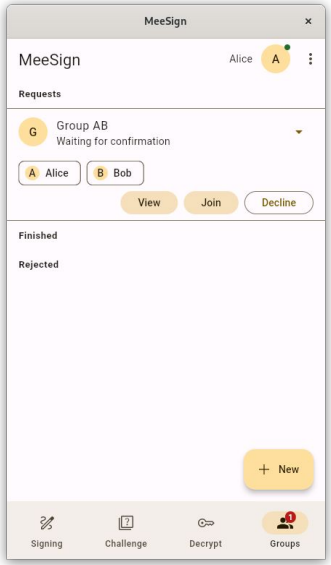
- Download these slides from https://meesign.crocs.fi.muni.cz/

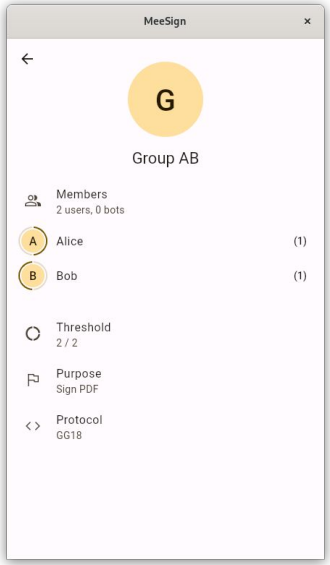
Task: Two-party PDF signing

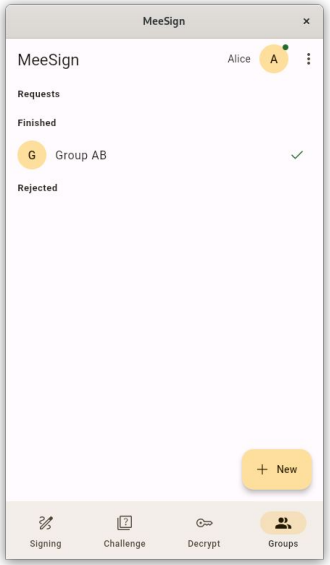
- Create a 2-of-2 group for PDF signing





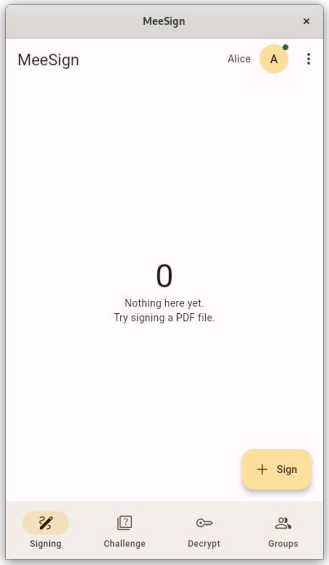


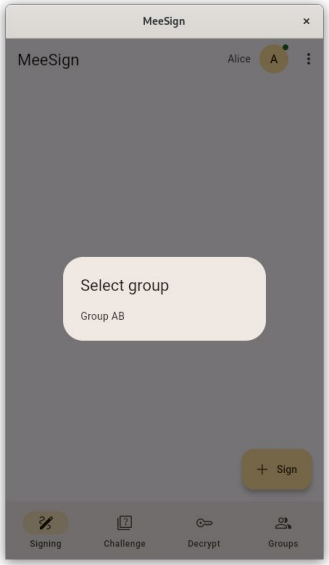


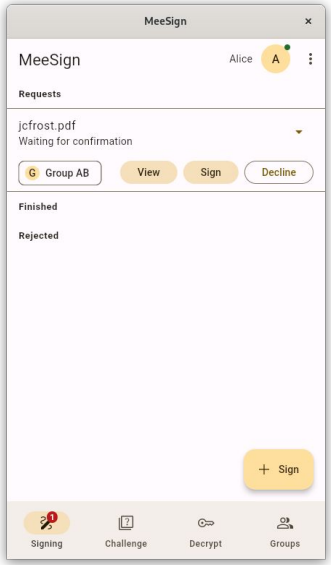


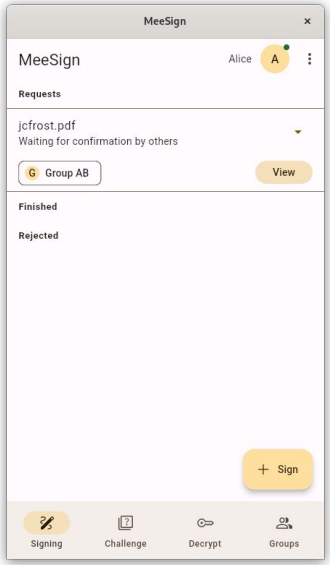
Task: Two-party PDF signing

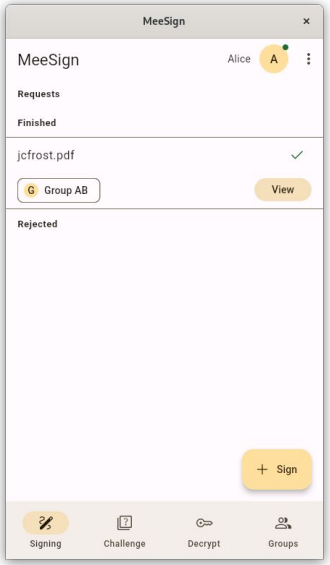
- Sign a PDF with the two-party group









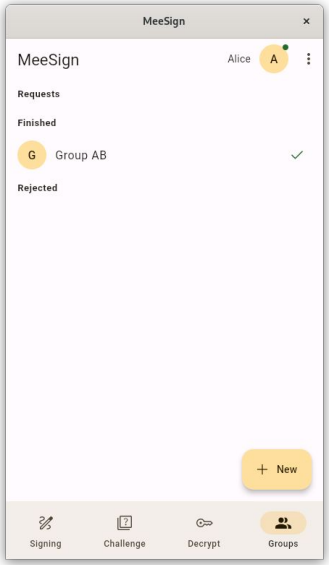


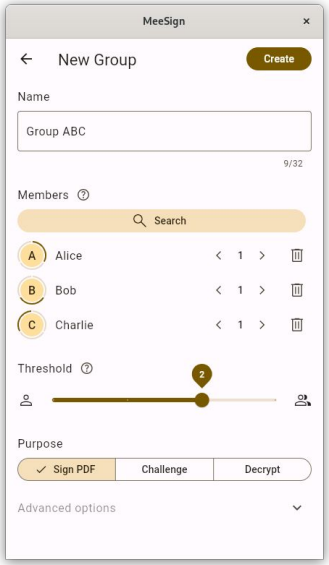
Discussion: Two-party PDF signing

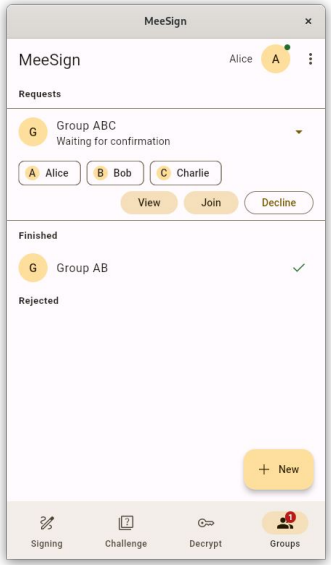
1. Try to verify the signature using standard tools (Adobe Reader, pdfsig, online)
   - Is the signature trusted by the tools? If not, why?
2. How many signatures are visible in the PDF?
3. What practical use cases do you see for two-party PDF signing?

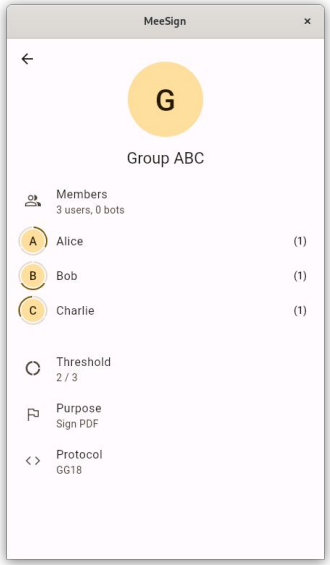
Task: Setting up a threshold

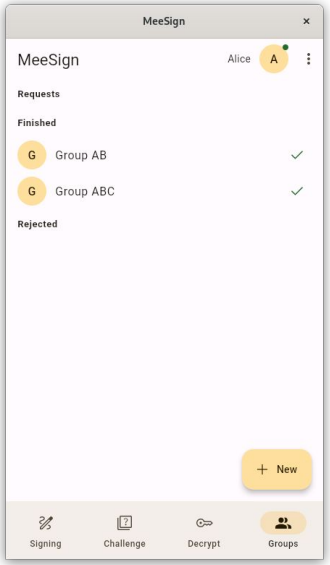
- Create a 2-out-of-3 group for PDF signing





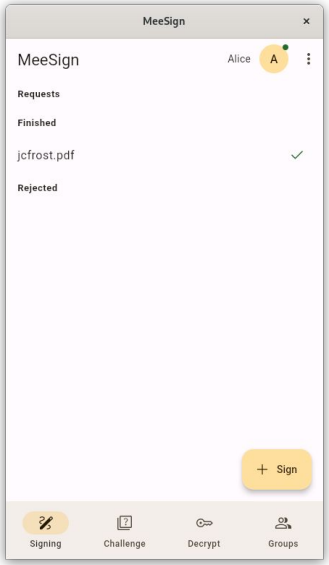


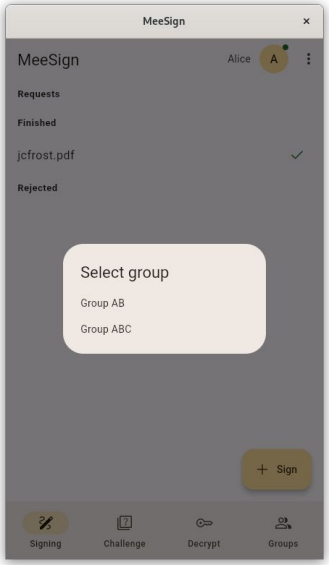


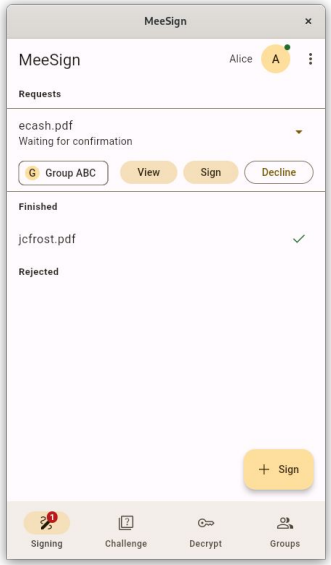


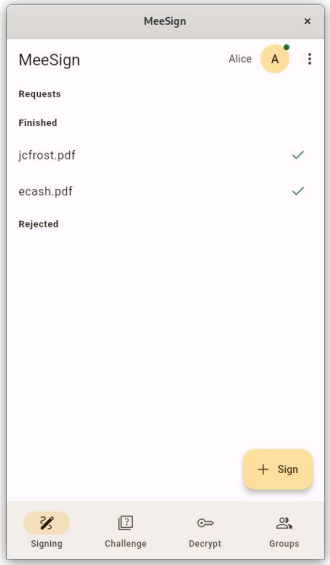
Task: Setting up a threshold

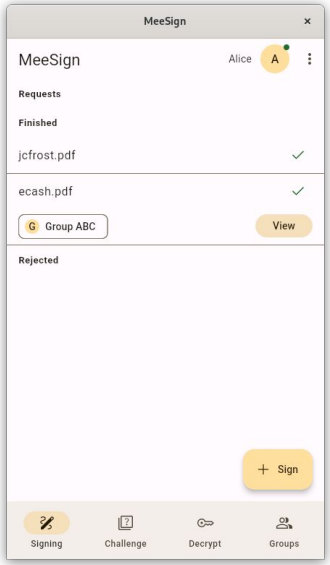
- Sign a PDF with the threshold group









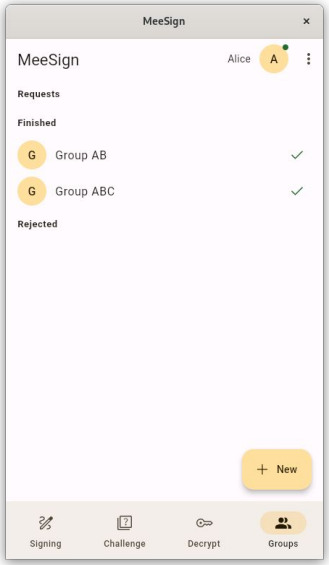


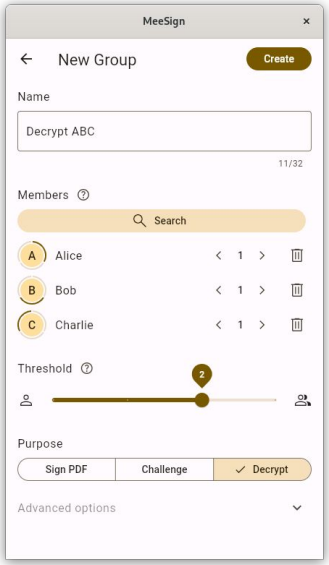
Discussion: Setting up a threshold

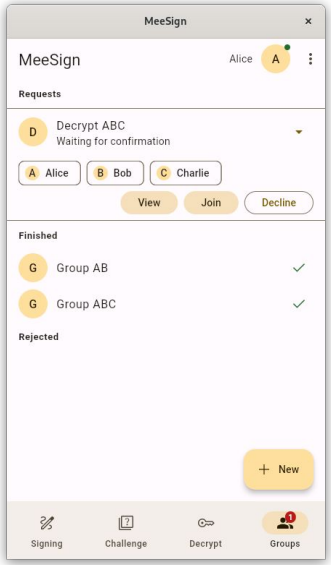
1. How did the group creation behave?
   - Did all of you need to approve it? Why?
2. How did the signing behave?
   - Did all of you need to approve it? Why?
3. Can such a signature be considered a signature (from law perspective)?
4. Where do you see a use case for using threshold signatures?

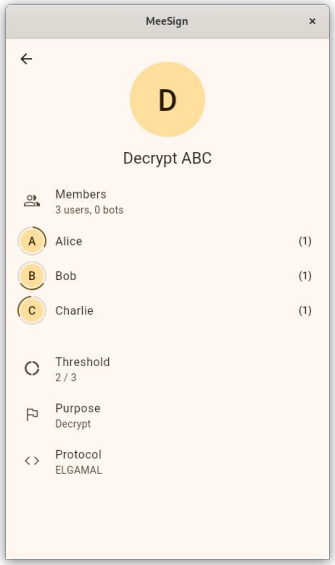
Task: Threshold decryption

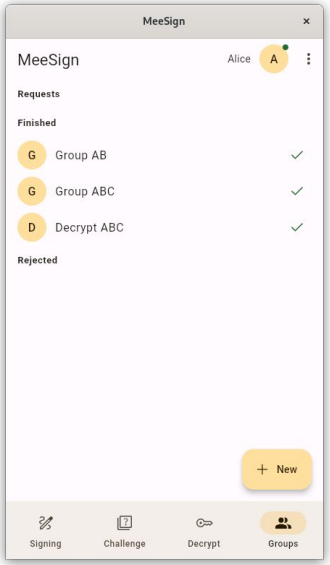
- It is not just signing – create a decryption group





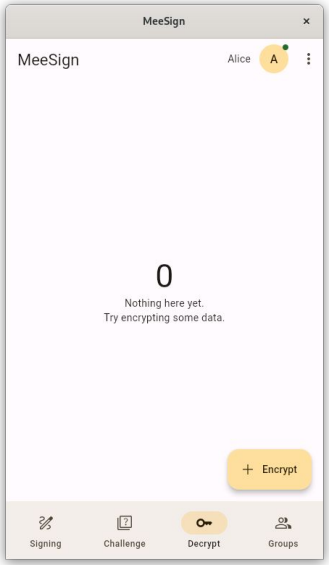


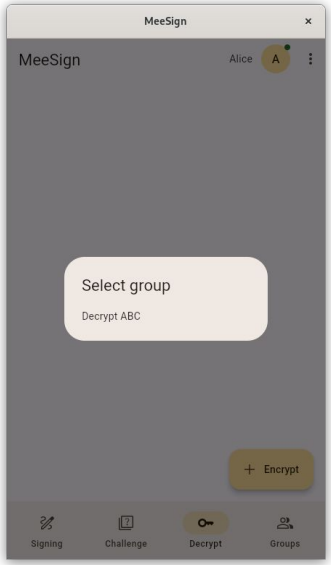


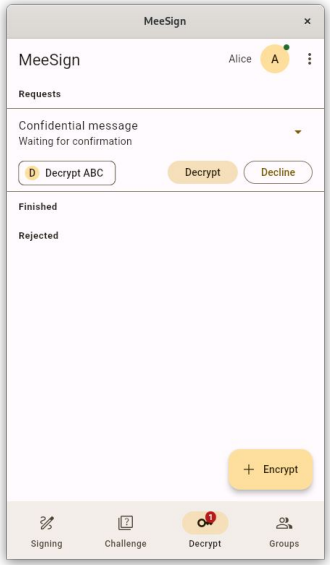


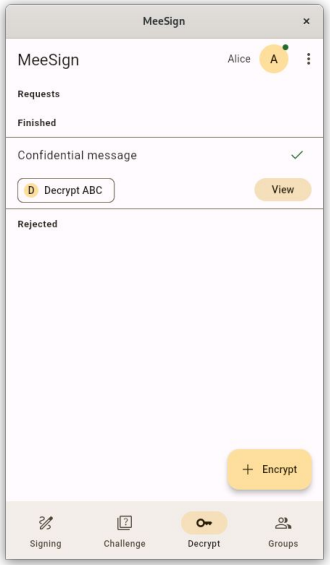
Task: Threshold decryption

- Send an encrypted image or a message to the group







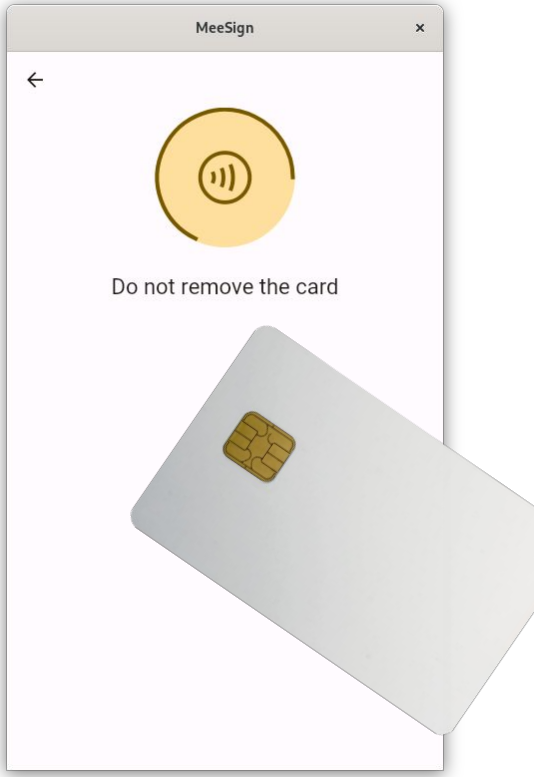


Discussion: Threshold decryption

1. Who from the group can see the decrypted message?
2. Where would you use threshold decryption?

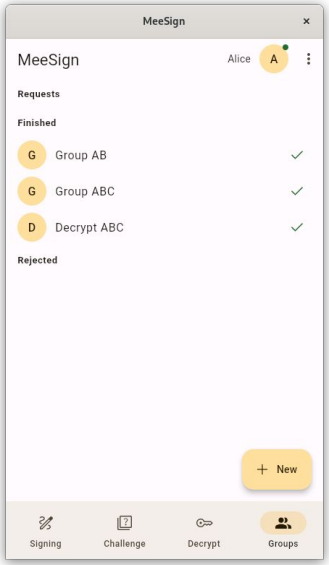
Storing shares on a smartcard

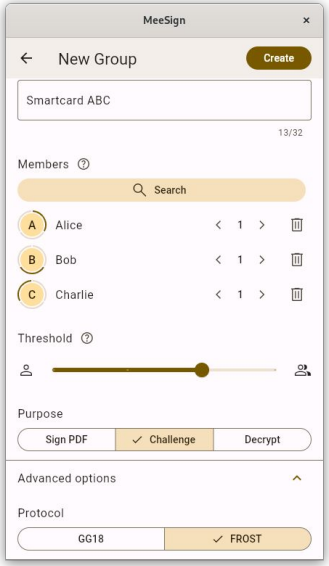
- Client application secrets are stored in user-space
  - A potentially higher risk of compromise
  - Compromising a single share is not fatal (with secret sharing)
- Secrets are often protected using secure hardware
  - TPM, Android Keystore, iOS Secure enclave, etc.
  - But current secure hardware available in consumer devices provides a very limited interface – no support of threshold cryptography
- Smartcards are more flexible
  - We developed an applet that can run one of the protocols
  - A smartcard with the applet can be used directly from the client application

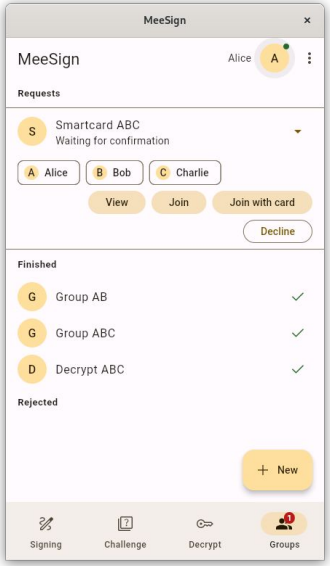


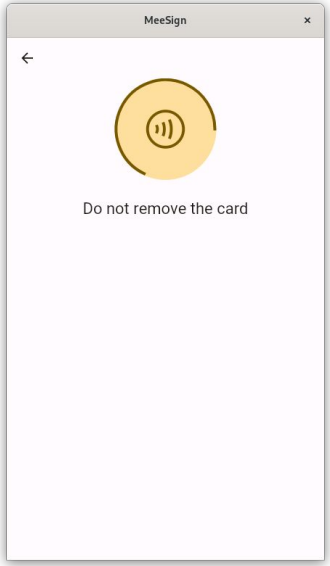
Task: Signing with a smartcard

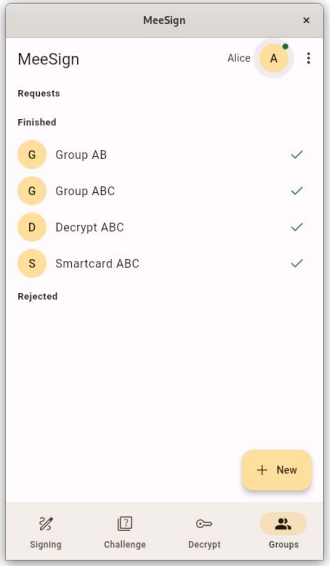
- FROST protocol supports smartcards – join a group with a smartcard





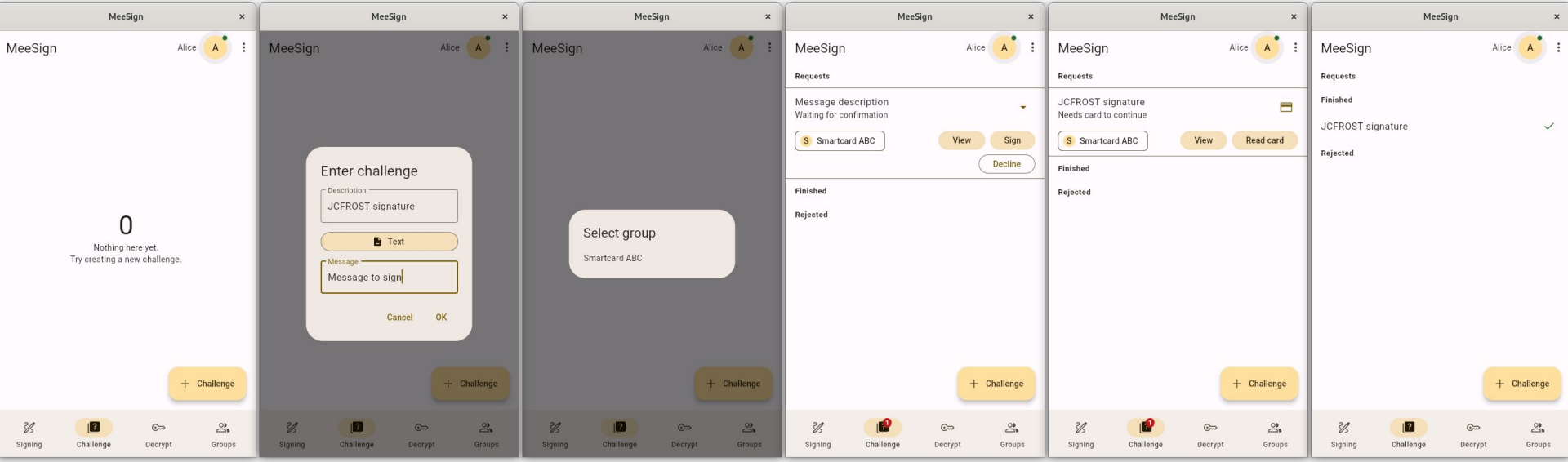






Task: Signing with a smartcard

- Sign a message in a group with a smartcard



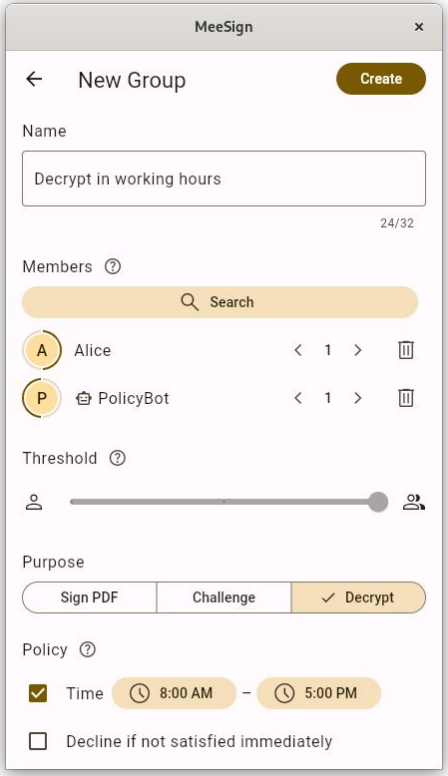
Discussion: Signing with a smartcard

1. What are the security benefits of using a smartcard?
2. Why we cannot use this signature in PDFs?
3. Try signing with a smartcard in 2-of-2, 2-of-5, and 5-of-5 settings.
   - When do you notice longer computation on smartcard?

Break

Policy-based parties

- Interesting threshold cryptography usage scenarios can be enabled by automated parties:
  - 2-of-2 a company HSM and a client smartphone
  - 2-of-3 a server and two users
    - the server signs only during working hours
    - two users together can overrule it
- We provide a policy-based self-deployable bot



Policy-based parties

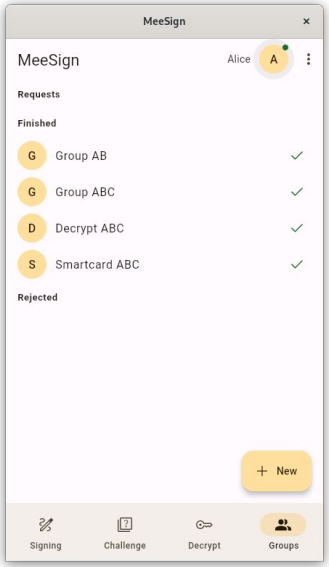
- Configured using a base policy (JSON)
  - can be overridden by group policy
  - supports setting time range
  - flag whether to decline immediately or wait for validity

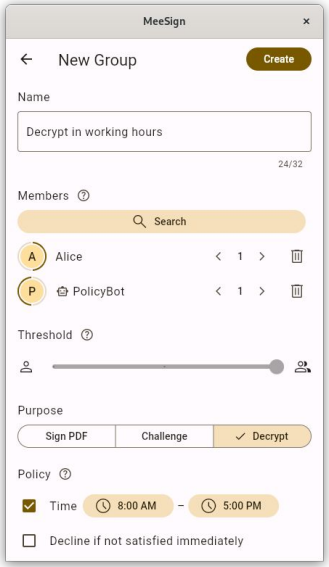
```
{
    "decline": true,
    "after": "08:00",
    "overridable": true
}
```

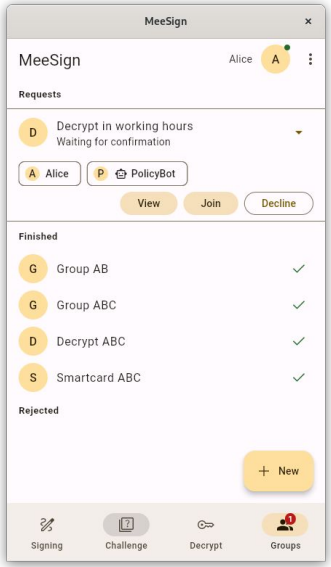
```
↳ ./policy --policy policy.json --host meesign.crocs.fi.muni.cz
No credentials found, registering as PolicyBot
Logged in as PolicyBot#a998
Base policy: {decline: true, after: 08:00, overridable: true}
Approved task: Sample task {}
Declined task: meesign-workshop.pdf {after: 00:00, before: 14:00, decline: true}
```

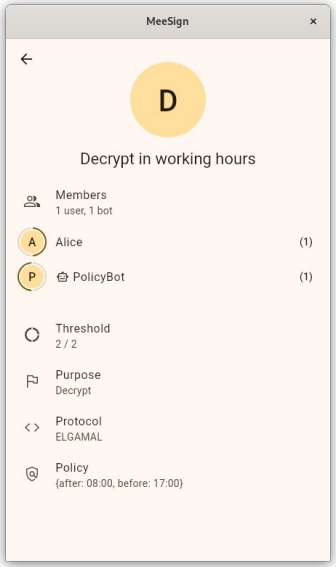
Task: Policy-based automated parties

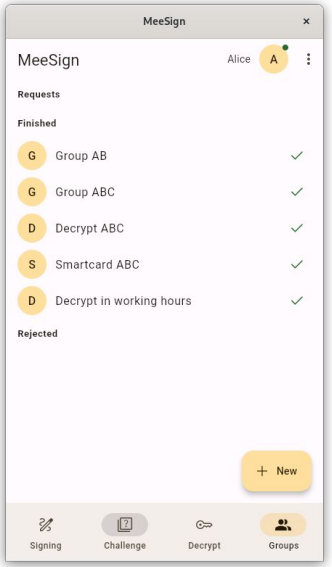
- Create a group which will start approving messages in 3 minutes from now





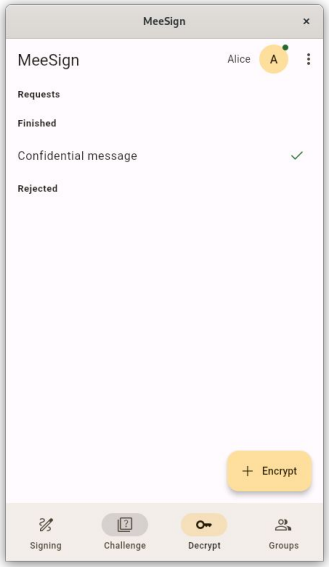


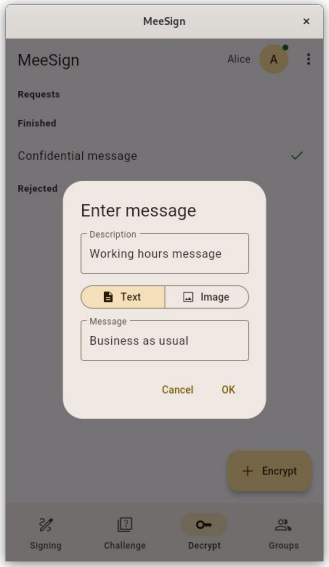


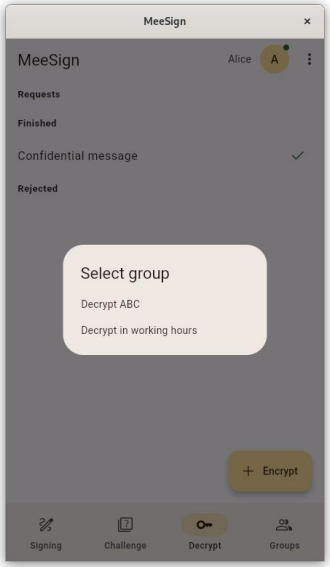


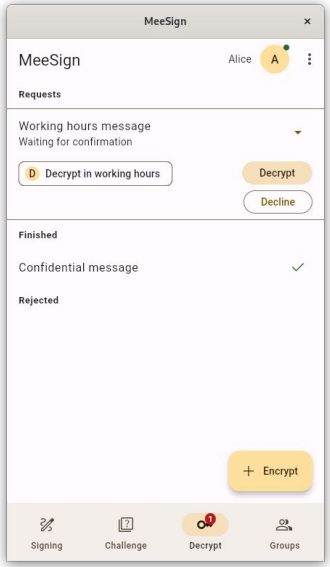
Task: Policy-based automated parties

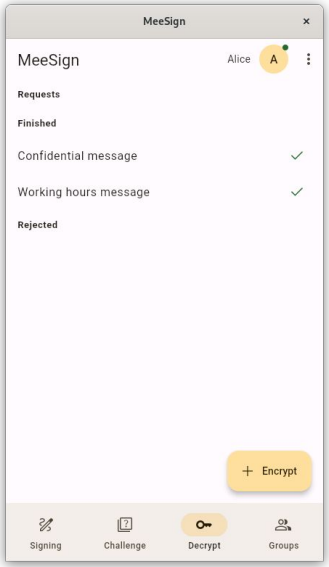
- Send a request to the group











Discussion: Policy-based automated parties

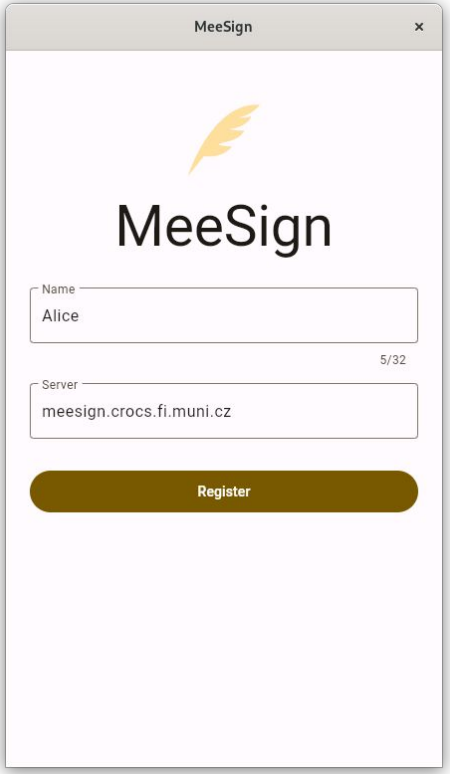
1. Does the automated party follow the policy even during group establishment?
2. What is the difference between using an automated party provided as a service and your self-deployed one?
3. What other policies could be enforced through the automated parties?

Task: Try running your own policy bot

- Requirements
  - a UNIX machine (e.g., the VM we provided)
- Download a release build of the policy bot:
  - https://github.com/crocs-muni/meesign-client/releases/download/v0.4.2/meesign_policybot.tar.gz
- Extract the tar archive: `tar xf meesign_policybot.tar.gz`
- Try to run it (see help first): `./meesign_policybot --help`
- Add your bot to a group (optionally with a custom policy)

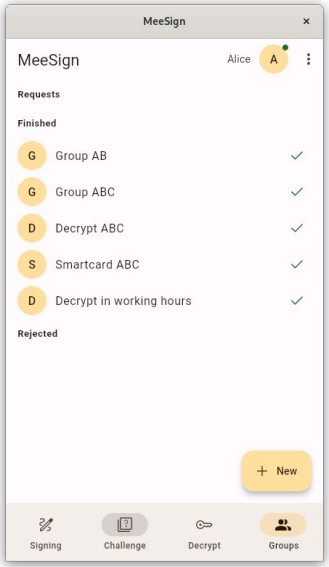
Share weighting

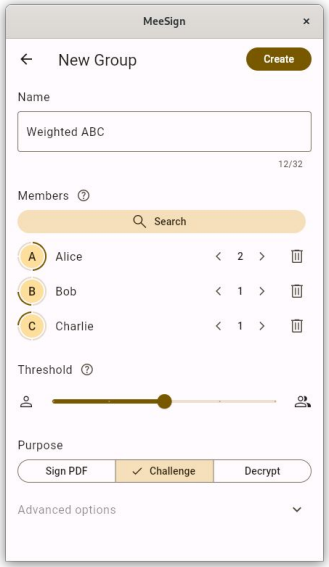
- Giving higher voting rights to certain parties (by providing them more shares)
- Can be constructed at two levels:
  - a wrapper over a protocol where a party acts as multiple participants
  - directly on the protocol level without additional communication overhead
- We support the wrapper-based approach

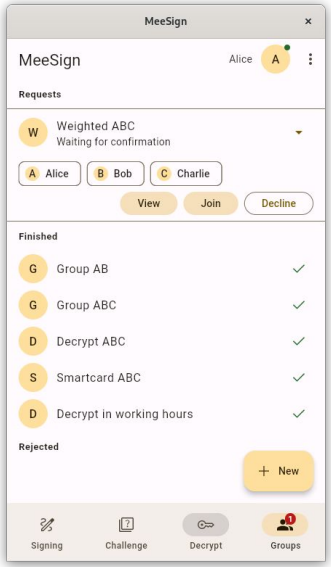


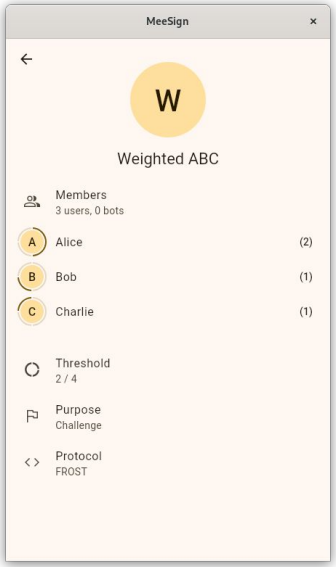
Task: Share weighting

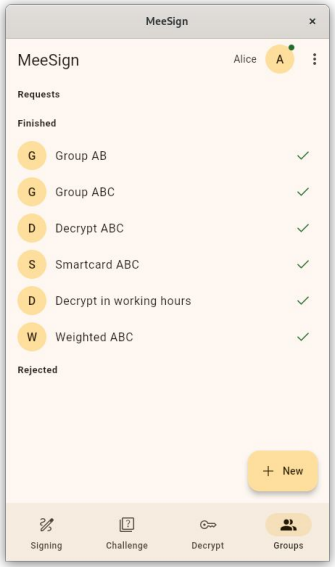
- Create a 2-of-4 group with just 3 people





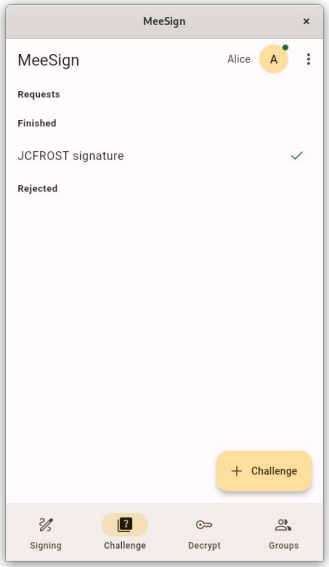


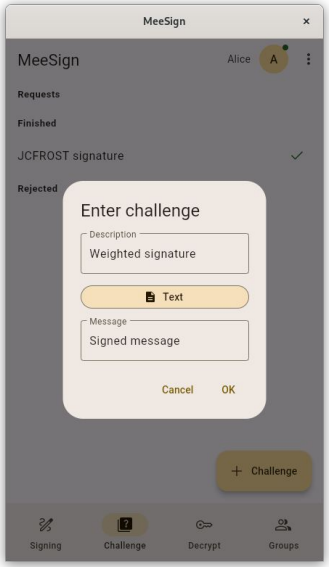


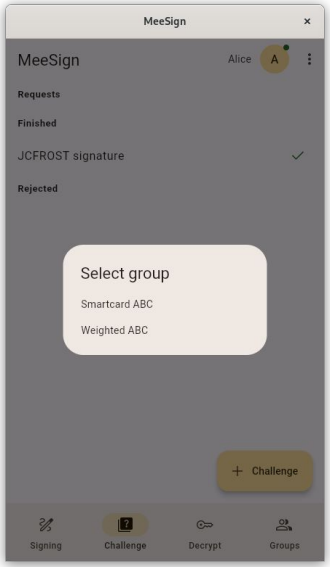


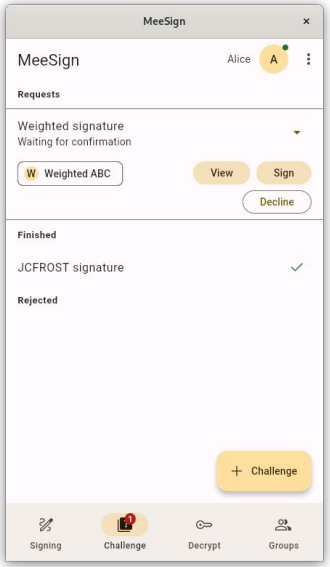
Task: Share weighting

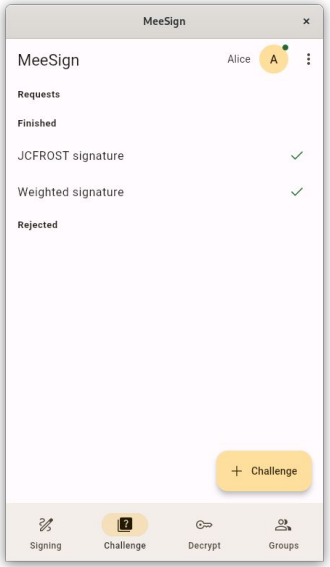
- Create a 2-of-4 group with just 3 people











Discussion: Share weighting

1. What are some new use-cases that weren't possible before?
2. Can you think of some interesting combinations with policies?

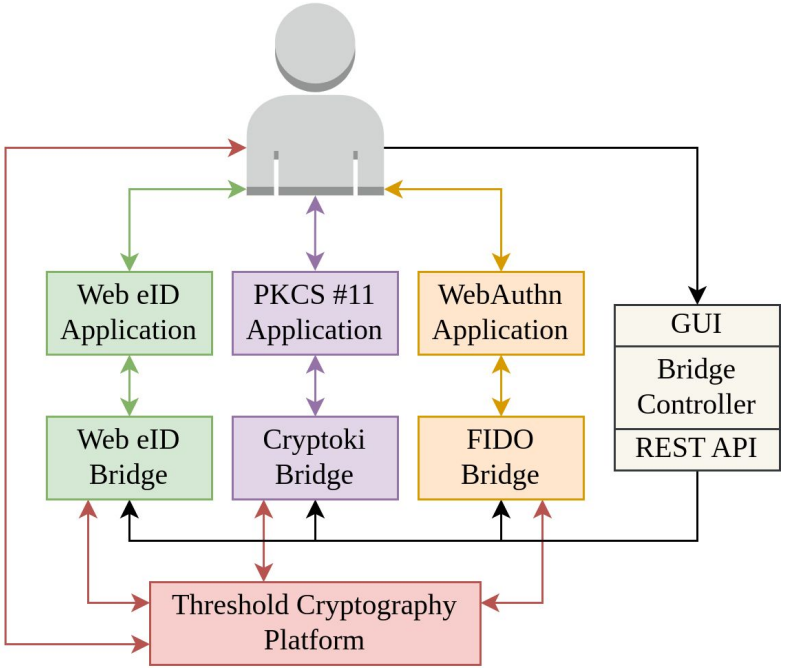
Integrations

Integration via hardware token interfaces

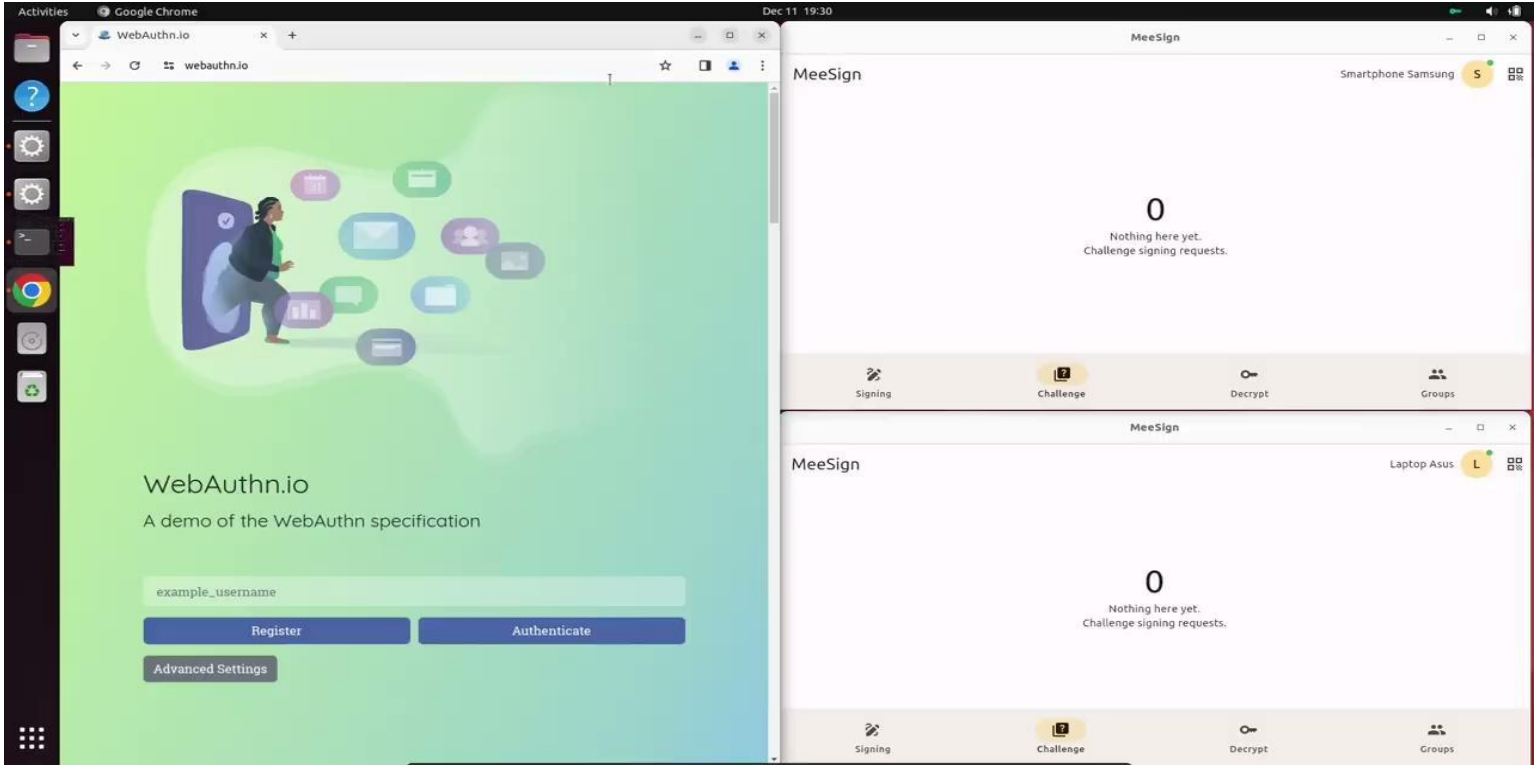
- Applications may support hardware tokens:
  - Smartcards
  - FIDO2 tokens
  - Cryptographic hardware wallets
  - ID cards
- Tokens provide standardized interfaces:
  - Cryptoki (PKCS#11)
  - FIDO2 (WebAuthn / Passkeys)
  - Hardware wallet interface (HWI)
  - Web eID (PC/SC)
- “Virtual tokens” listening on these interfaces can relay requests to a MeeSign server

Bridge Suite

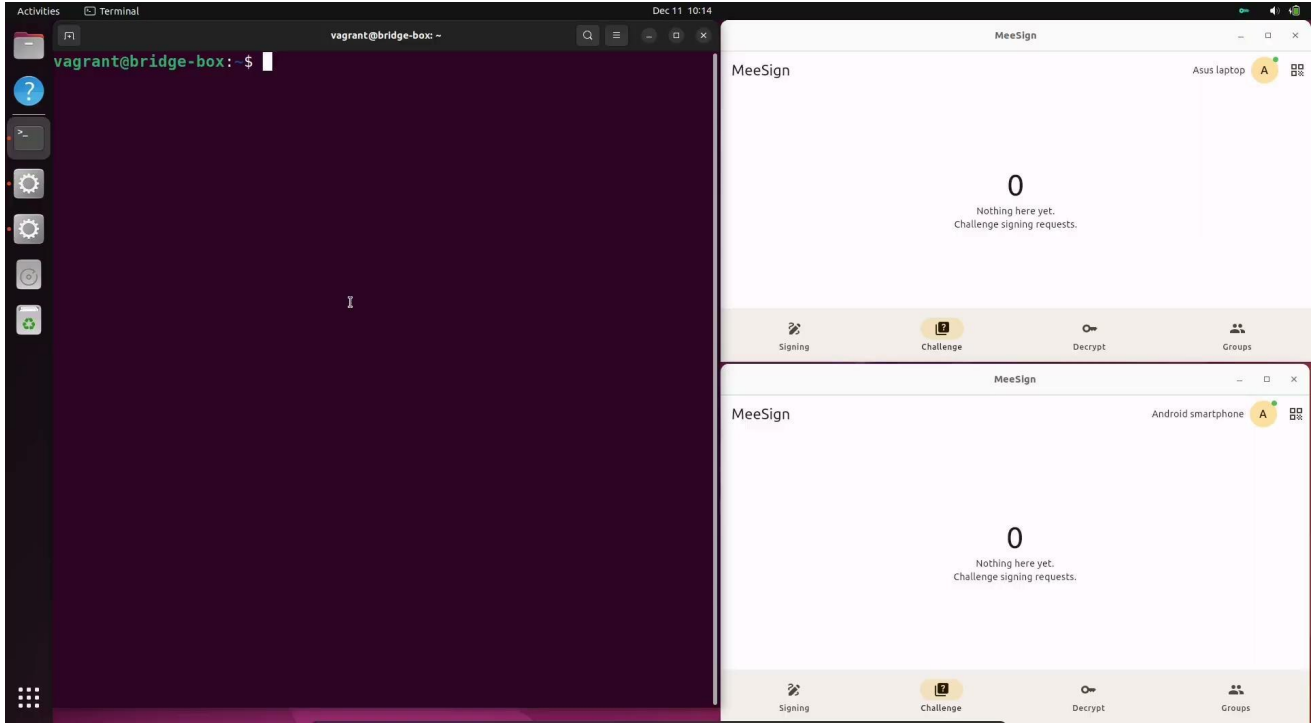
- A suite of virtual tokens for delegating cryptographic operations
- Each virtual token (“bridge”) can be deployed as a standalone component
- We also provide a GUI application for unified setup and configuration:
  - Configuring a target server
  - Choosing a remote certificate
  - Selecting a target group



FIDO Bridge Demo (Webauthn)



SSH Demo (Cryptoki)



Task: Setup SSH authentication

- Requirements
  - a UNIX machine (e.g., the VM we provided)
  - a server with existing SSH access (optionally, see next slide a local Docker SSH)
- Setup a cryptoki-bridge dynamic library
  - Follow https://github.com/KristianMika/bridge-suite/wiki/Cryptoki-Bridge
  - Set the `GROUP_ID="<group-id>"` environment variable
    - The group ids can be found at https://meesign.crocs.fi.muni.cz/groups.txt
      `[<group-id>] <group-name> (t-of-n; purpose)`
    - The group needs to have purpose set to `Challenge` and protocol to `GG18`
- Setup an SSH authentication according to the instructions https://github.com/KristianMika/bridge-suite/wiki/Applications

[Optional] Docker SSH 1/2

- Create a Dockerfile in a folder:

```
FROM ubuntu:24.04
RUN apt update && apt install -y openssh-server
RUN sed -i 's/PermitRootLogin prohibit-password/PermitRootLogin yes/'
/etc/ssh/sshd_config
RUN useradd -m -s /bin/bash username
RUN echo "username:password" | chpasswd
EXPOSE 22
ENTRYPOINT service ssh start && bash
```

- Source: https://circleci.com/blog/ssh-into-docker-container/

[Optional] Docker SSH 2/2

- In that folder build with
  `docker build . -t sshd-image`
- Run the image with
  `docker run -t sshd-image:latest`
- Find <containerID> with
  `docker container ls`
- Find the <containerIP> with
  `docker inspect --format='{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' <containerID>`
- SSH with to the container
  `ssh username@<containerIP>`

X-Road

MeeSign use-case in X-Road

CHESS

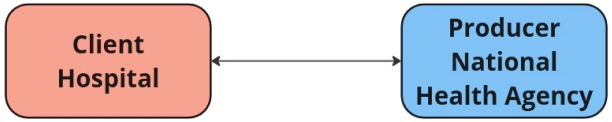
Funded by the European Union

- CHESS project collaboration
  - Mariia Bakthina, Raimundas Matulevičius (UT) and Petr Švenda
- Paper: The Power of Many: Securing Organisational Identity Through Distributed Key Management
  - To be presented at CAiSE 2024, June 3–7
- Core idea:
  - Use threshold signatures to enforce custom policies in Information Systems

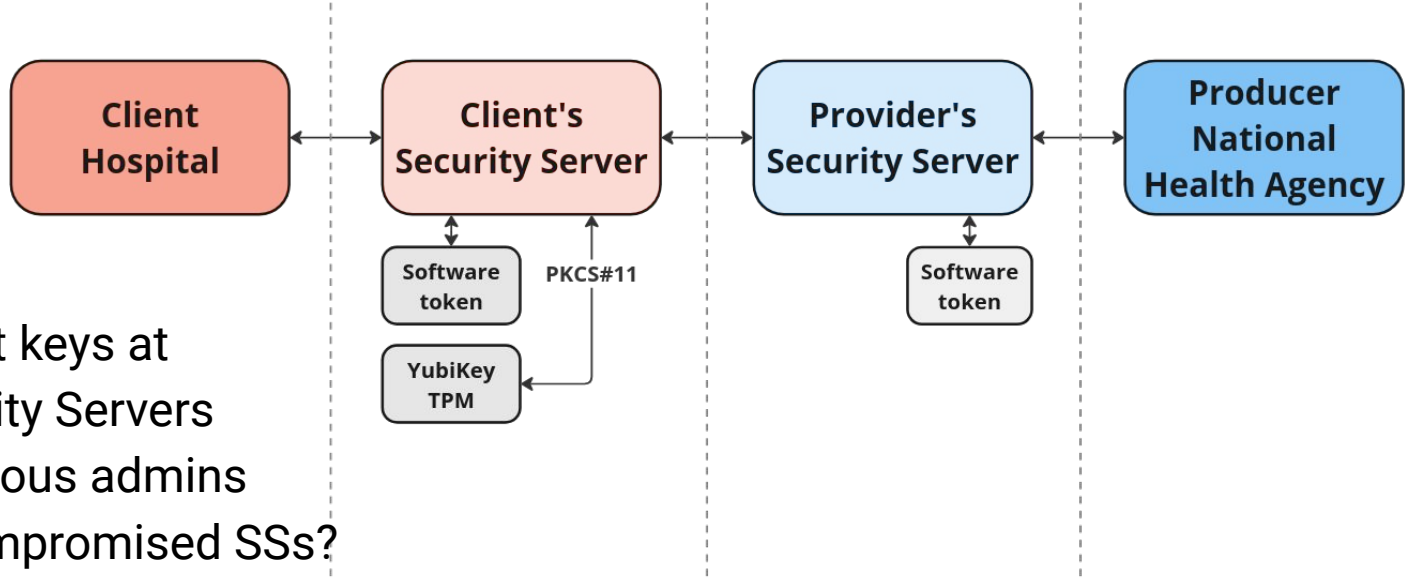
https://chess-eu.cs.ut.ee/

X-Road system

- Comprising of various members
  - Companies, national and governmental institutions, etc.
  - Cross-national connection
- Data exchange layer between members
  - Digitally signed tunnel for REST/SOAP API messages
- Security requirements
  - Integrity (of data) and origin (of sender) of messages
- Digital signatures provide both
  - However…

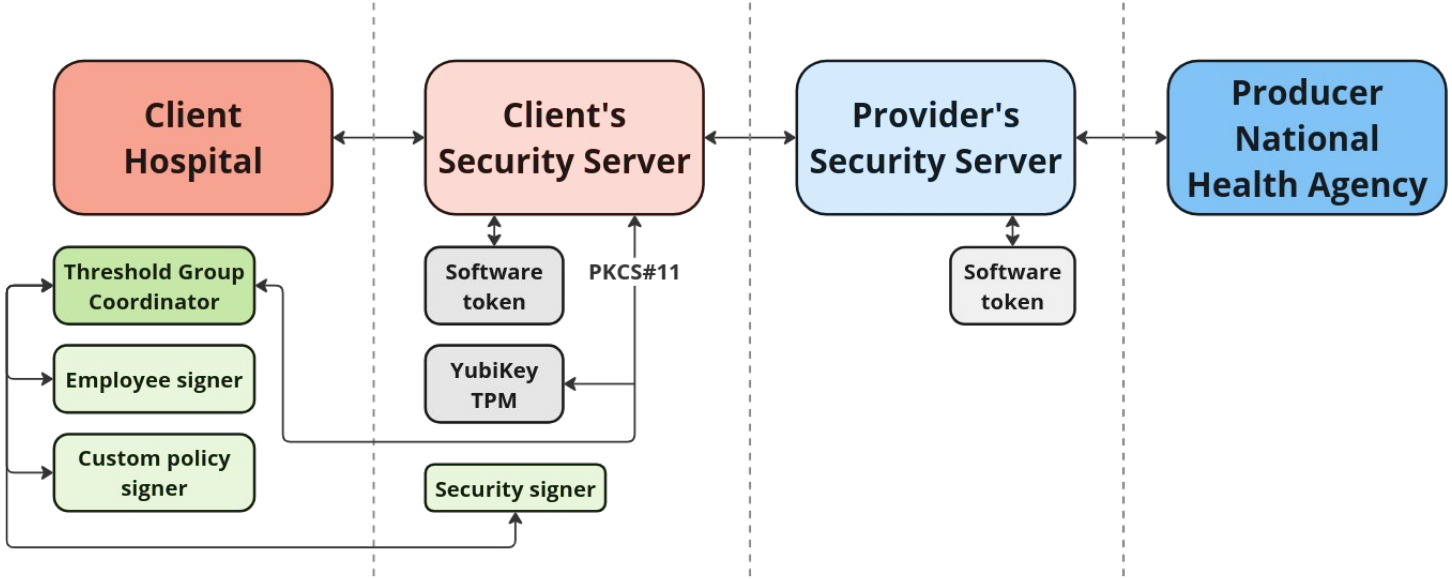


X-Road architecture



- Secret keys at Security Servers
- Malicious admins or compromised SSs?

X-Road architecture with threshold signatures



- Custom policies enforced by automated signers

Custom policies

- Variations
  - Time-based, data-based policies
- Currently
  - Either you trust Security Server with enforcing them
  - Or you enforce them locally, but still SPoF
- Manual policy signers or automated signers
- Examples
  - 3-of-3
    - Doctor, Hospital, Security Server
  - 2-of-4
    - CEO with two shares, automated signer, an employee
    - Any catch?

Efficiency of RSA threshold signatures

- Client–Provider data exchange
  - Provider uses SoftToken and the Client's token varies
  - Mean from 1000 measurements
  - SoftToken used as a baseline

Client's token:	SoftToken	SoftHSM	YubiKey 5	TPM NTC 7.2.3.1	*this work*
mean RTT	82ms	75ms	216ms	260ms	276ms
mean slowdown	1.0x	0.92x	2.65x	3.18x	3.38x

X-Road Demo

- Full walkthrough available at
  - https://github.com/crocs-muni/xroad-threshold-signatures

Limitations and possible improvements

- Currently, trusted key dealer
  - First party generates the secret and shares it
  - Could be mitigated with DKG
- Extra network communication
  - Could be improved with message presigning
- RSA – Shoup's Practical Threshold Signatures
  - https://github.com/crocs-muni/pretzel
  - Elliptic Curve Cryptography support on X-Road's roadmap for 2025

Thank you!

- We welcome all collaboration
  - MeeSign is still under active development
  - Possible BSc or MSc thesis topics available
- Contacts

Antonín Dufka
dufkan@mail.muni.cz

Jan Kvapil
kvapil@mail.muni.cz

https://meesign.crocs.fi.muni.cz/

References

- Gennaro, Rosario, and Steven Goldfeder. "Fast multiparty threshold ECDSA with fast trustless setup." Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. 2018.
- Komlo, Chelsea, and Ian Goldberg. "FROST: Flexible round-optimized Schnorr threshold signatures." Selected Areas in Cryptography: 27th International Conference, Halifax, NS, Canada (Virtual Event), October 21-23, 2020, Revised Selected Papers 27. Springer International Publishing, 2021.
- Ruffing, Tim, et al. "ROAST: Robust asynchronous Schnorr threshold signatures." Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security. 2022.
- Shoup, Victor. "Practical Threshold Signatures". Eurocrypt 2000.