DELEGATEE: BROKERED DELEGATION USING TRUSTED EXECUTION ENVIRONMENTS

Sinisa Matetic, Moritz Schneider, Andrew Miller,

Ari Juels, Srdjan Capkun

OVERVIEW

1. Introduction
2. Motivations and Problem Statement
3. DelagaTEE
4. Security Analysis
5. Prototype
6. Performance
7. Limitations
8. Conclusion

INTRODUCTION

WHAT IS DELEGATION?

- Sharing a portion of one's authority with another
- Allowing other applications access to the user's privileges
- Existing delegation between platforms is somewhat limited
- Ex: Third-party app access to Facebook or Google services, posting to Facebook wall or accessing data on Google drive
- Granularity of this delegation is large; no way to limit number of third-party Facebook posts per day, for example
- Some services have no delegation at all, such as email

WHAT IS DELEGATION?

- Credential sharing is a common way for delegation to be done
- Delegatees gain full access to the user's account
- This can only be done in a safe manner if delegatees are fully trusted

BROKERED DELEGATION

- Enables fine-grained delegation between the owner and delegatees
- Uses trusted execution environments (TEEs) such as SGX
- Delegation policy enforced by a TEE enclave holding the credential
- Allows users to delegate access without the support or knowledge of service providers

BROKERED DELEGATION

- Brokered delegation with DelagaTEE requires no changes to legacy infrastructure, the service, or the user's account
- Two design variations for DelegaTEE, peer-to-peer and third-party credential brokeer
- Alters access-control policy of services in a way that can both provide additional utility or subvert these policies
- For example, resale of paid subscription services

MOTIVATIONS & PROBLEM STATEMENT

MOTIVATIONS

- Two major motivations: new service functionality from brokered delegation, and transforming mandatory access control into discretionary access control
- Mandatory access control: service enforced access restriction based on user credentials
- Discretionary access control: user may give ownership to and determine access type of other users

APPLICATION SCENARIO 1: MAIL/OFFICE

- Mailbox or web office delegation for administrative workers and virtual-assistant services may be desirable for users
- Can restrict access based on parameters, e.g. read-only access, read/send access to a set of domains
- Limited access for law-enforcement, reading emails from a specificed time period relevant to a legal case
- Existing services require full access

APPLICATION SCENARIO 2: PAYMENTS

- Allows for employee use of payment methods (bank accounts, credit cards, PayPal) with restrictions
- Restrictions placed on payments; expenditure limits per transaction, merchant selection
- Currently, trust is placed in certain employees that make all transactions - this is inefficient
- Also allows for "under-banked" populations to use payment methods of friends or family members

APPLICATION SCENARIO 3: WEBSITE ACCESS

- Most versatile form of delegation – web services authenticate users with password and HTTPS cookies
- Social media, music and video streaming services, paywalled academic papers
- Current delegation is only done through sharing of login credentials
- This is not secure and sharing access cannot be done with fine granularity

APPLICATION SCENARIO 4: SHARING ECONOMY

- Allows for delegation to other users on a profit basis
- Access can be sold on an open market
- Subscription services can be resold in areas where they are not normally sold or where they are not economically viable
- Social media account access can be sold to advertisers, with the user being able to restrict the volume and content of posts made in their name

PROBLEM STATEMENT

- Most service providers do not offer fine-grained and secure delegation options
- DelegaTEE allows users to remedy this in a way that:
  - Owner account information remains confidential
  - The owner can restrict access to the account in terms of schedule, duration, reads/ writes, etc.
  - Actions of the owner and delegatees are logged
  - The ability of the service to distinguish between usage of the legitimate owner and delegatees is minimized (not possible for all services)

DELAGATEE

DELAGATEE

- Main concept of the system is to store the owner's credentials in a TEE implementing the delegation policy
- The delegatee communicates with the service indirectly, with the TEE as a proxy

TRUSTED EXECUTION ENVIRONMENTS AND SGX

- Enables isolated code execution in the user's system
- Application split into trusted and untrusted parts
- Application launches enclave, which is stored in protected memory
- Only code inside the enclave can access data in the enclave

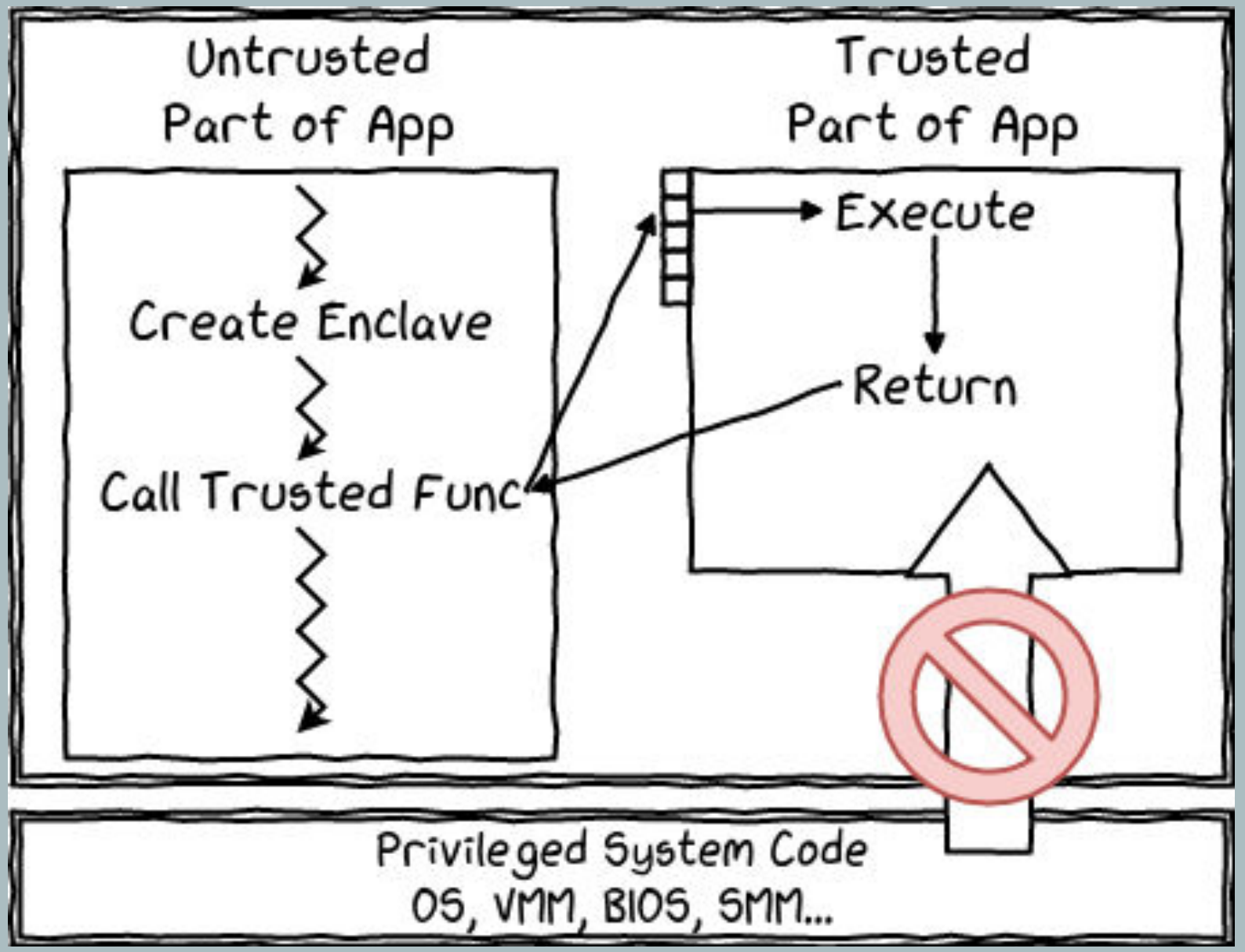


Image: Alexandre Adamski, Blog, quarkslab.com

SYSTEM DESIGN

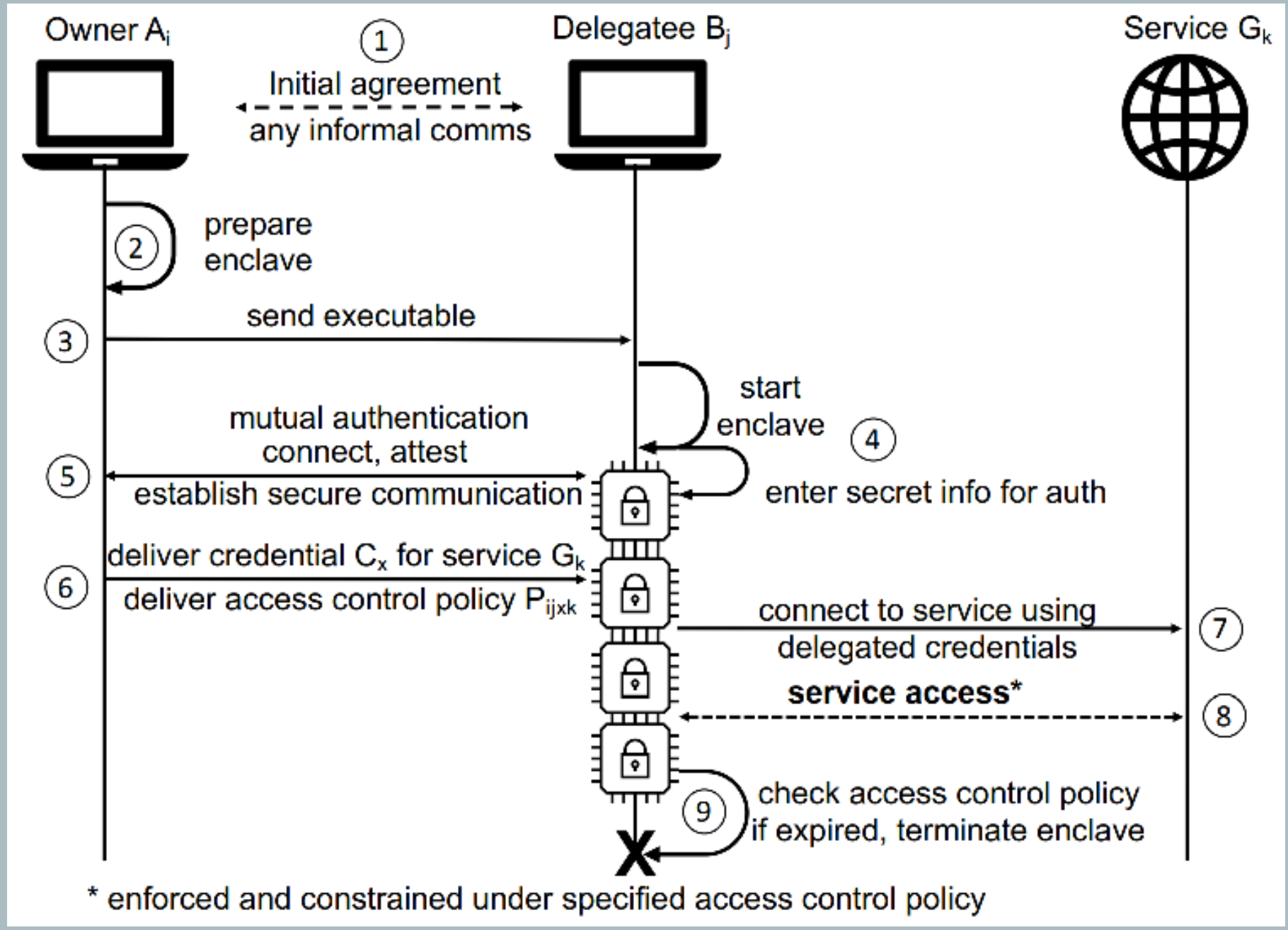
- Two system architectures: centrally brokered and peer-to-peer (P2P)
- Centrally brokered architecture uses a third-party management entity to run the enclaves
- P2P architecture does not use a management entity, instead the delagatee coordinates directly with the owner to gain access to a specific service

PEER-TO-PEER DESIGN

- Supports many owners and delegatees
- Requires a delegatee to have Intel SGX support
- Owner and delegatee first communicate through available communication channels, e.g. email, phone, in person
- Users need to establish a method for authentication upon enclave start (pre-shared key, certificates, etc.)

PEER-TO-PEER DESIGN

- Owner agrees with delegatee on service that will be accessed with the owner's credentials
- Owner prepares the enclave
- Owner sends the executable to delegatee
- Delegatee starts the enclave and authenticates with pre-shared information
- Owner connects to the enclave, verifies correctness of code for the agreed upon service, and establishes secure communication channel
- Owner sends the credentials for the service with the access control policy via the secure channel
- The delgatee uses the enclave as a proxy to connect to the service using the secure channel
- Usage is strictly limited by the access control policy and the delegatee cannot parts of the service not allowed by the owner
- If the policy has a time limit, the delegatee's access to the service is terminated appropriately

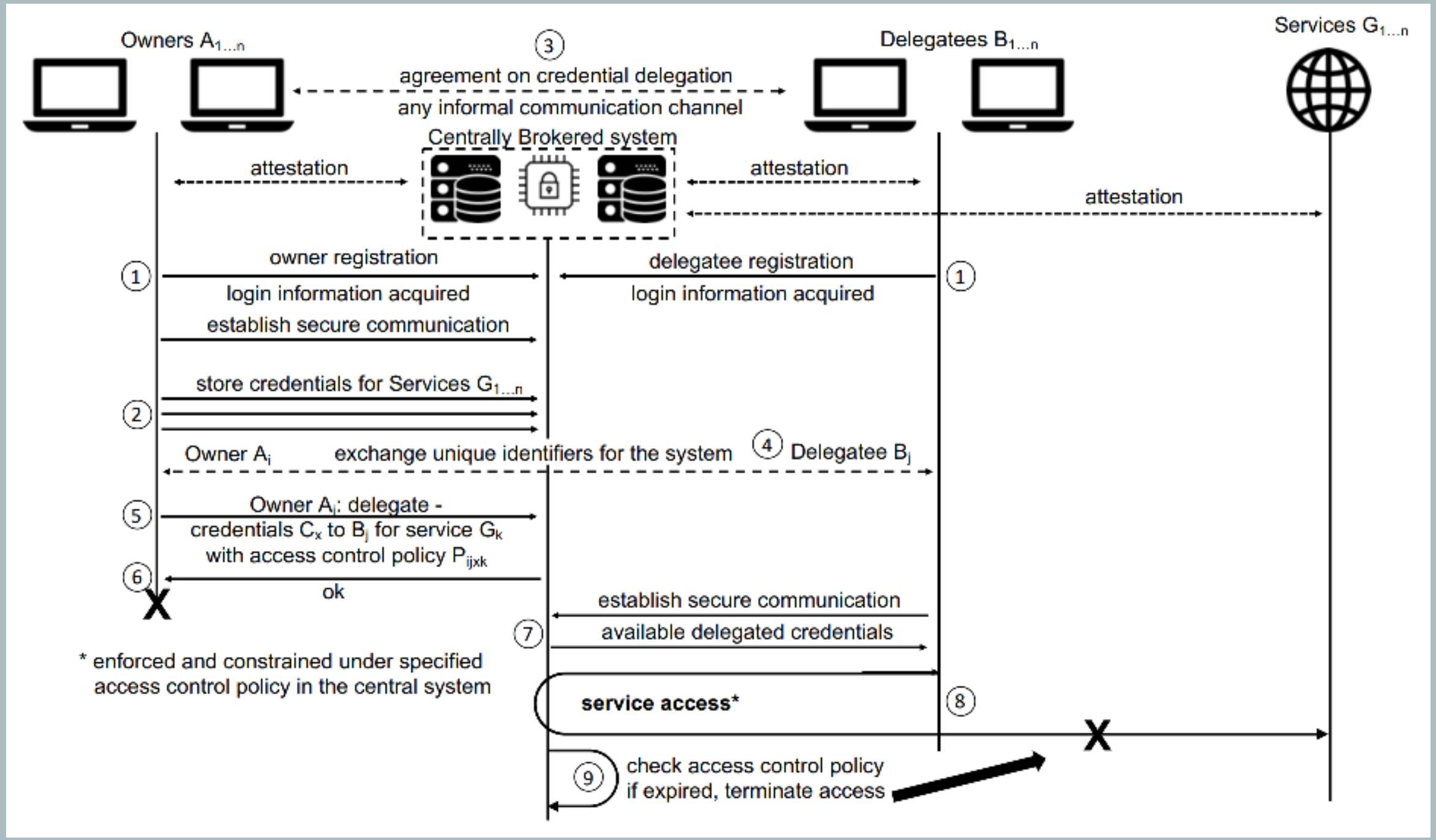

Owner A_i
1
Initial agreement
any informal comms
Delegatee B_j
Service G_k
2
prepare
enclave
send executable
3
start
enclave
4
mutual authentication
connect, attest
5
establish secure communication
enter secret info for auth
deliver credential C_x for service G_k
6
deliver access control policy P_{ijxk}
connect to service using
delegated credentials
7
service access*
8
9
check access control policy
if expired, terminate enclave
* enforced and constrained under specified access control policy

CENTRALLY BROKERED DESIGN

- Uses a central server to manage transactions and communications between all clients
- Requires server to support SGX enclaves, not required for owners or delegatees
- System can verify the running code and service provider

CENTRALLY BROKERED DESIGN

- Owners and delegatees need to register with the system and acquire login information for access
- Owners establish a secure channel to the system and store credentials for services
- Owners may agree with delegatees on the service the owner will grant credentials to, done using other means of communication
- Owner specifies to the system which credentials are to be used for delegation for a service to a particular delegatee, along with the policy
- After receiving confirmation, owner disconnects
- Delegatee can now connect to system and see which services they have been delegated credentials for
- Access to the service is always proxied through the central broker, no direct communication between delegatee and service
- After policy expires, the delegatee loses access and credentials are no longer delegated


Owners $A_{1...n}$
Delegatees $B_{1...n}$
Services $G_{1...n}$
3
agreement on credential delegation
any informal communication channel
Centrally Brokered system
attestation
attestation
attestation
owner registration
login information acquired
delegatee registration
login information acquired
1
1
establish secure communication
store credentials for Services $G_{1...n}$
2
Owner A_i
exchange unique identifiers for the system
4
Delegatee B_j
5
Owner A_i: delegate -
credentials C_x to B_j for service G_k
with access control policy P_{ijxk}
6
ok
establish secure communication
7
available delegated credentials
* enforced and constrained under specified
access control policy in the central system
service access*
8
9
check access control policy
if expired, terminate access

ANONYMOUS USAGE

- Identity-based usage (non-anonymous) follows directly from the model discussed previously – users know each other, have a communication channel, and can mutually identify
- Owner directly delegates credentials to a delegatee, such as a friend, family member, or colleague

ANONYMOUS USAGE

- DelegaTEE conceals the owner's credentials, preserving anonymity in both P2P and centrallized architectures
- An outside system allowing for anonymity (e.g. a bulletin board) may be used to broker services
- Owners and delegatees can identify themselves with pseudonyms, such as onion addresses or PGP signatures

SECURITY ANALYSIS

SECURITY PRINCIPLES

- The owner's access credentials remain confidential
- The use of delegated credentials is defined by the access control policy, which will not be violated
- Use of credentials can only be granted to the intended delegatee, with authorization of the owner

SECURITY ANALYSIS

- DelegaTEE is designed to provide these guarantees against a strong attacker
- Assumed that the attacker does not corrupt the full software stack of the owner and delegatee machines or the online service
- Assumed that the attacker can control everything else, including reading and manipulating network traffic between parties

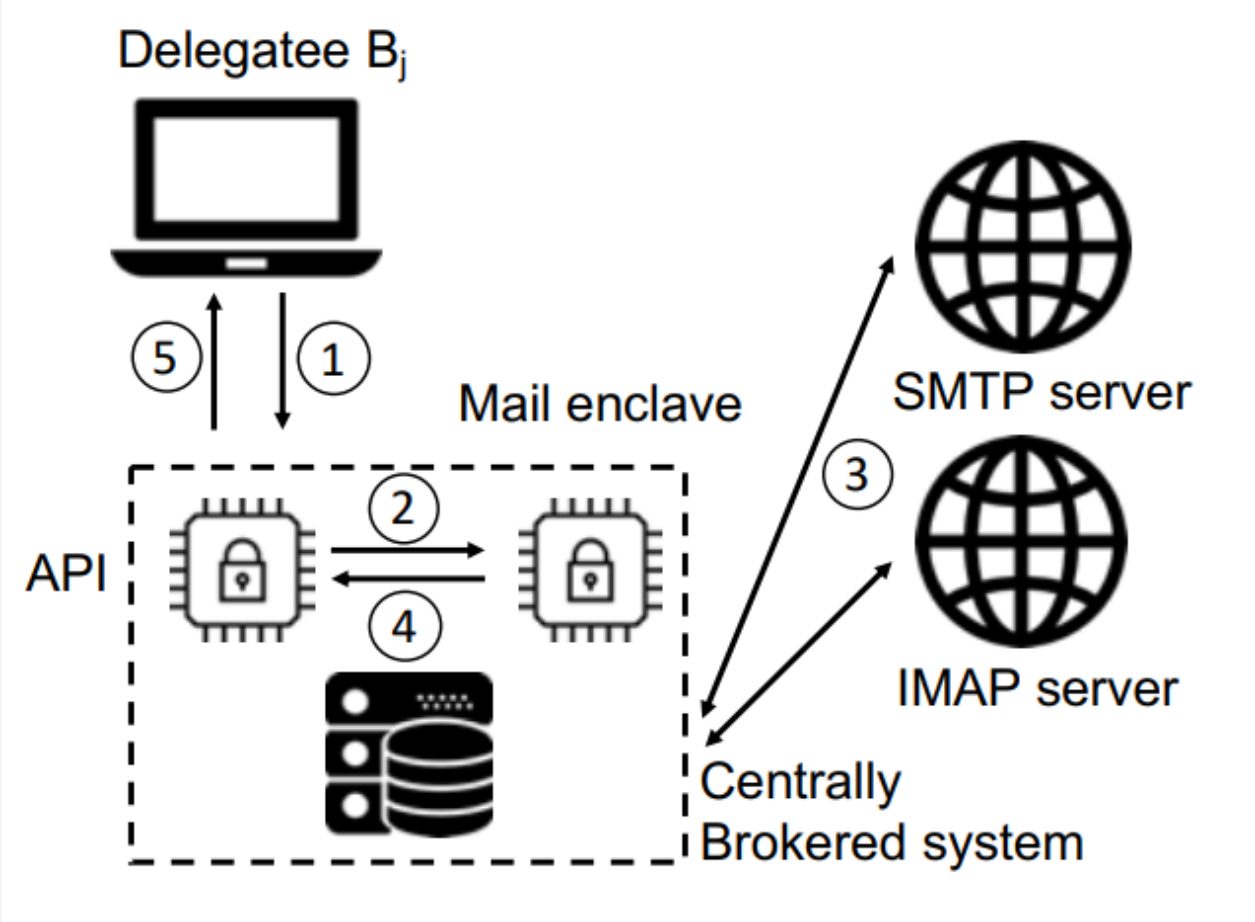
SECURITY ANALYSIS

- Compromise of the SGX enclaves is allowed, as long as the software stack on the enclave machine is not also compromised
- Pre-shared means of authentication allow the owner and user to verify each other before credentials are transmitted
- Side-channel attacks considered to be out of scope for this paper

PROTOTYPE

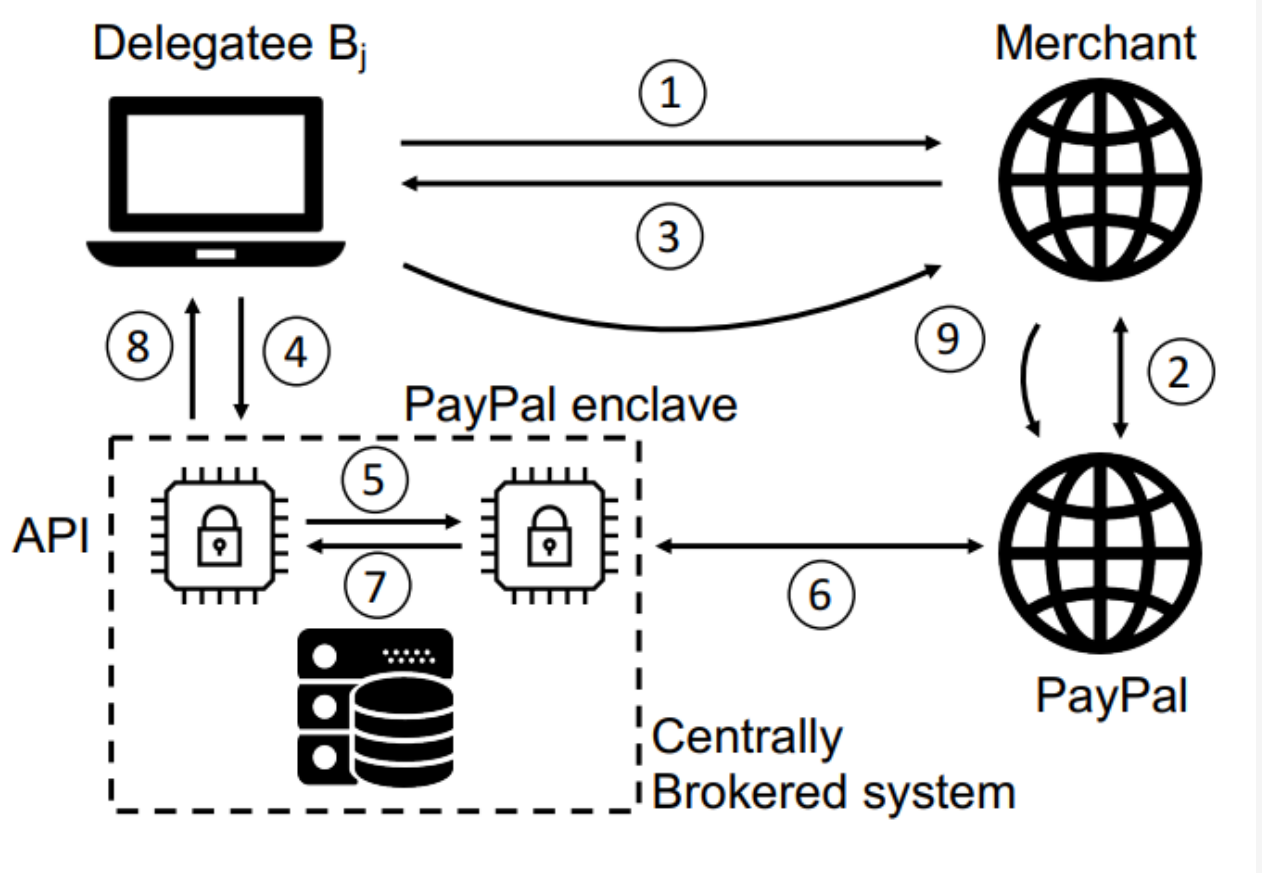
PROTOTYPE 1: MAIL/OFFICE

1. Delegatee wants to use some credentials, connects securely to API, and requests to perform a credentialed action.
2. API verifies that the delegatee has access to the credentials and forwards the request along with the access policy to the mail enclave.
3. Mail enclave connects to the SMTP or IMAP server and executes the operation.
4. Access policy is applied to the response from the mail server and the resulting response is sent to the API.
5. The API delivers the reponse to the delegatee.



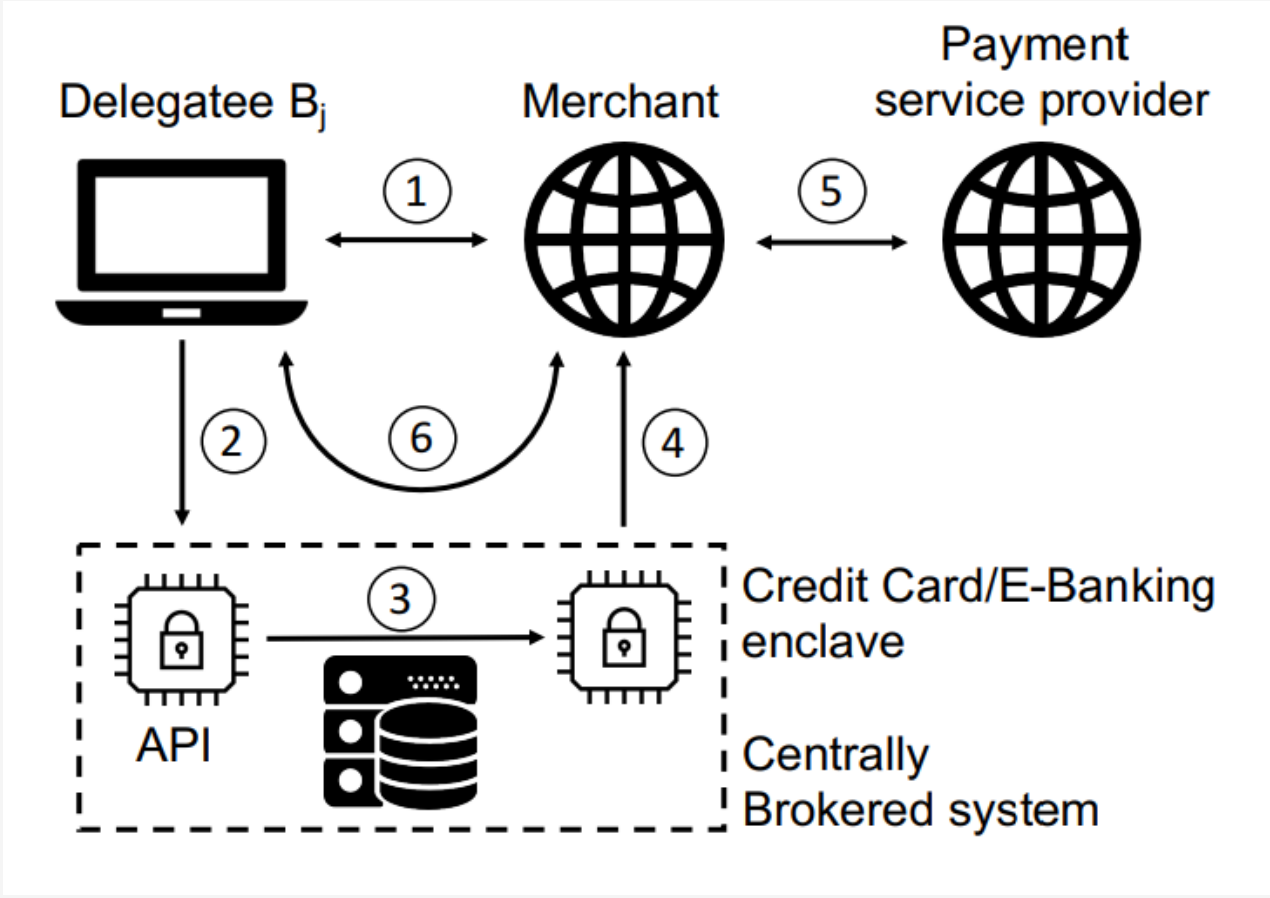
PROTOTYPE 2: PAYPAL

1. Delegatee wishes to buy something from a merchant using credentials delegated by the owner. Delegatee connects to the merchant and asks for a PayPal payment.
2. Merchant uses the PayPal API to create a payment.
3. Payment is forwarded to the delegatee.
4. Delegatee connects/authenticates to the API and requests to pay with owner's credentials.
5. API enclave verifies access to credentials and forwards request, credentials and policy to the PayPal enclave
6. If allowed by the policy, the PayPal enclave makes the payment with the owner's credentials.
7. Confirmation number from payment forwarded to the API.
8. API delivers confirmation number to the delegatee.
9. Delegatee forwards confirmation number to the merchant to finalize and confirm payment.



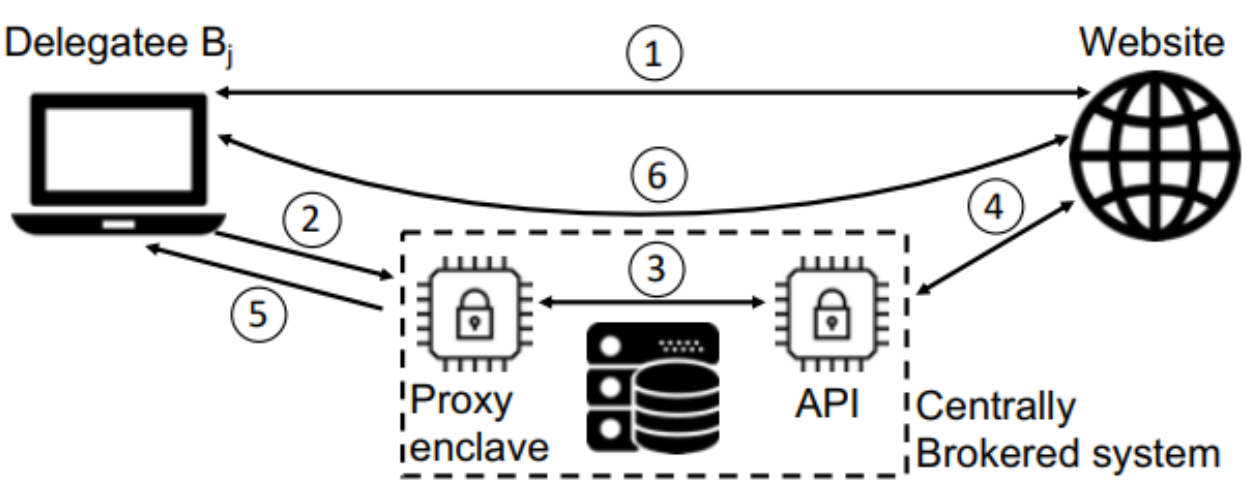
PROTOTYPE 3: CREDIT CARD/BANKING

1. Delegatee wishes to buy something from the merchant using the delegated credentials containing credit card or banking information. The delegatee connects to the website and a browser extension renders a second button next to the normal credit card/banking credentials submission button.
2. On clicking the injected button, the browser extension requests a payment with the delegated credentials from the API.
3. The API verifies that the user has access to the credentials, then forwards the request, credentials and policy to the banking enclave.
4. If allowed by the policy, the enclave fills the credentials into the request from the merchant and then submits it.
5. Payment provider finalizes payment.
6. Response is forwarded back to the delegatee.



PROTOTYPE 4: HTTPS PROXY

1. The delegatee wishes to log into a website using delegated credentials; they connect to the website and a browser extension renders a second button beside the normal login button.
2. On clicking the button, the brower extension changes the URL pointing to the proxy and appends cookies specifying the credentials the delegatee wishes to use.
3. The proxy asks the API for the credentials; if access to the credentials is permitted, the API responds with them.
4. The proxy enclave supplies the username and password to the login request, sends it to the website and receives the response.
5. The proxy rewrites the header of the response to encrypt cookies and forwards it to the delegatee.
6. All subsequent connections must go through the proxy, where the access policy is enforced.



PERFORMANCE

PERFORMANCE

- Tests done using two machines - i7-7700 with 16GB RAM; able to serve approximately 100 concurrent users
- Overhead is approximately 50ms for an SSL handshake inside an enclave
- Mail enclave has minimal overhead, approximately .07ms longer. The centrally brokered system is slightly slower, as it involves additional communication with the API

PERFORMANCE

- PayPal delegation has a negligible performance impact; increasing from ~26 seconds to 27 seconds – most of this time is spent waiting for a response from the PayPal servers
- The proxy system introduces the highest overhead, but it is less than 100ms
- Overhead when streaming video was the same as the proxy, however testing was only done with one user on the centrally brokered system due to hardware limitations

LIMITATIONS

AUTHENTICATION CHALLENGES

- CAPTCHA authentication is supported by DelegaTEE
- Some services have contextual verification challenges; login from a new IP address or at a different time of day may trigger an additional authentication step
- Some service authentication methods are hard or cannot be overcome – personal questions, phone challenges (requiring numerical input over a phone line)

AUTHENTICATION CHALLENGES

- Production deployment of DelegaTEE will address these issues in several ways:
  - Individual service applications will include specific configuration for the APIs of a service and its authentication policies
  - Handling of two-factor authentication, which can be run in the enclave
  - Email verification, and geolocation simulation

AUTHENTICATION COLLISIONS

- Some services do not allow for multiple users to be logged in at the same time
- If a delegatee is logged in using delegated credentials, the owner may not be able to access the account
- Failure modes can address this, along with the owner setting policies that only allow access at times they are unlikely to use it, or disconnecting the delegatee

SERVICE PREVENTION

- Service providers are unlikely to support delegated usage – it may undercut profits, skew analytics, and prevent them from tracking users
- IP geofencing, pattern matching of usage, two-factor authentication, and other methods may be employed to prevent brokered delegation
- Future work involves investigation of possible improvements to mitigate service prevention

CONCLUSION

CONCLUSION

- Authors propose a new concept called brokered delegation – using TEEs to flexibly delegate access rights to internet services with fine granularity
- Two architectures: centrally brokered and peer-to-peer
- DelegaTEE can be applied to several real-world applications with low overhead
- DelegaTEE has potential to enable delegation for existing services without knowledge or support from the service