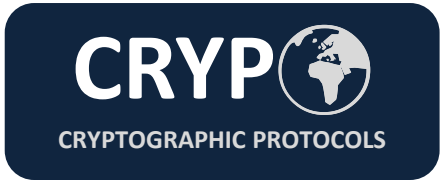
UPBA: User-Authenticated Property-Based Attestation

Mark Manulis and Marion Steiner

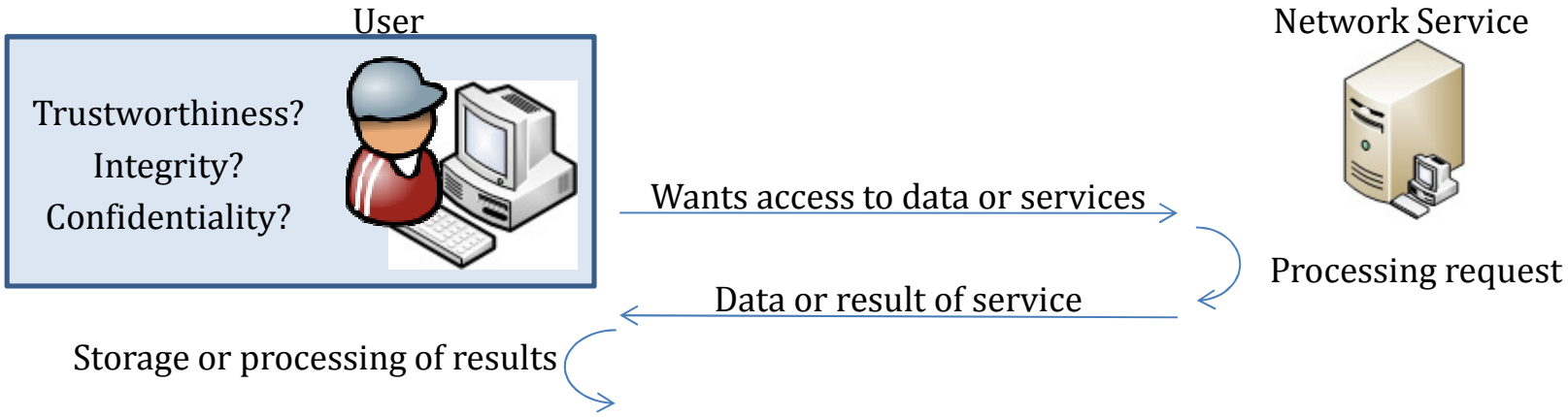
TU Darmstadt & CASED



Outline

- Why do we do this?
- Security Model
- Our UPBA Scheme
- Conclusion

Secure Interaction in Networks

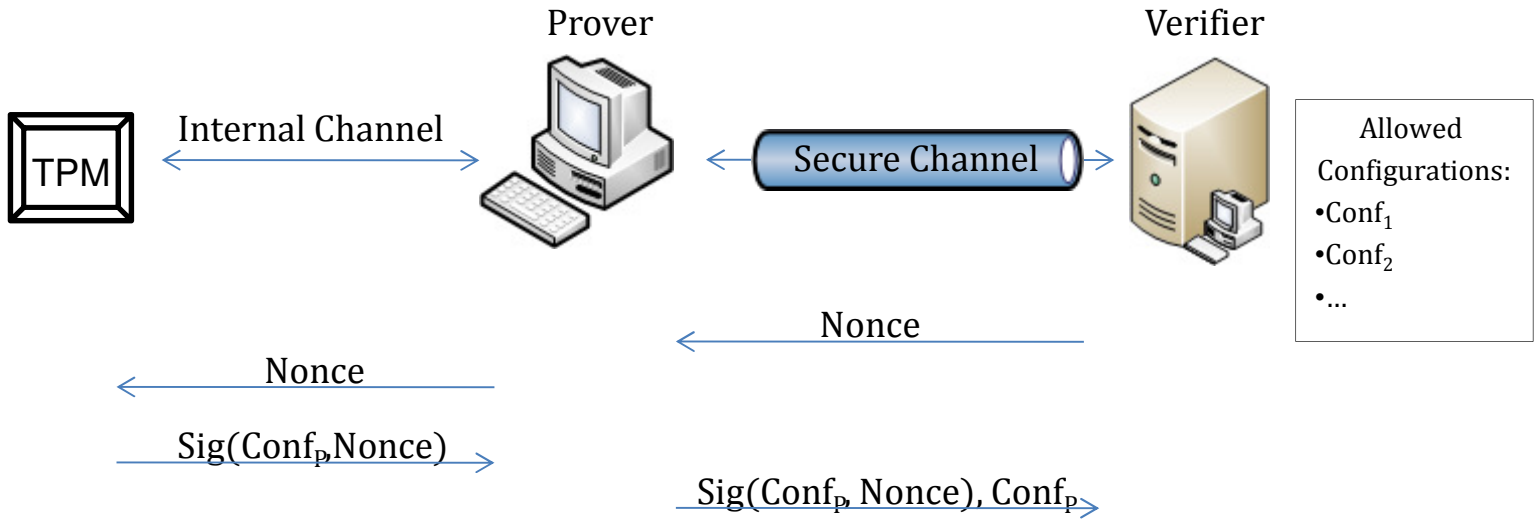


• Server Security?	√ (Responsibility of Data Owner)
• Connection Security?	√ (Secure Channel)
• User Known and Trusted?	√ (Responsibility of Data Owner)
• Client (Device) Security?	? (Possible: Responsibility of User?)

Point-of-View : Service and Data Owner

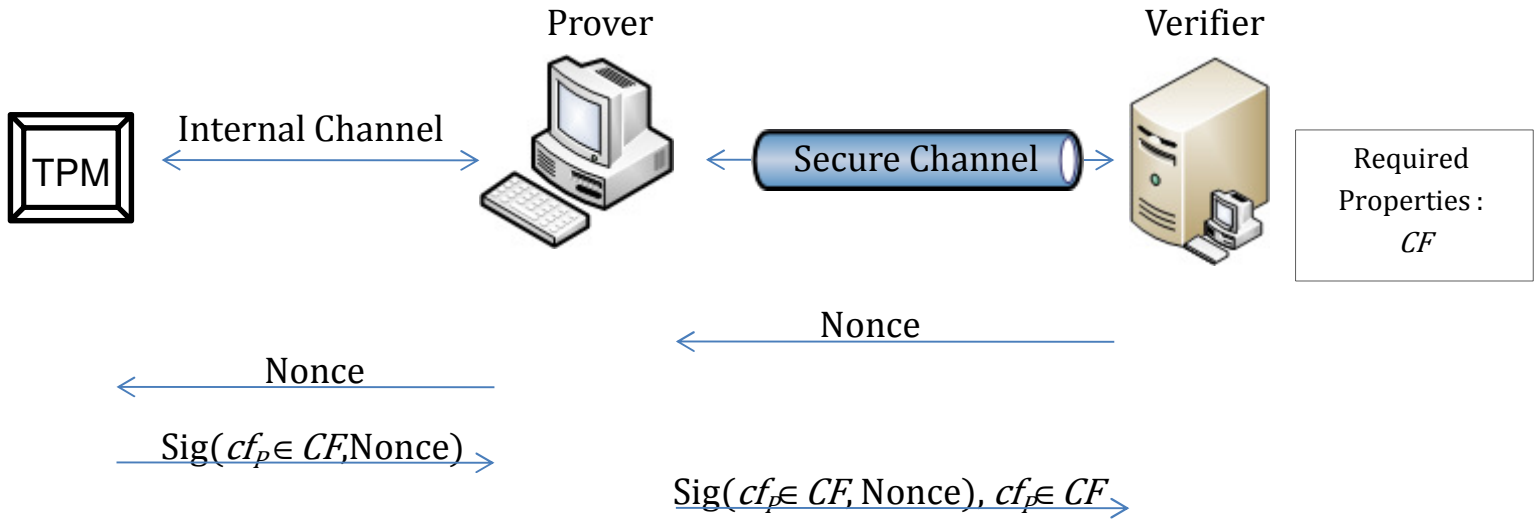
Solution: Remote Attestation - verification of client configuration

(Binary) Remote Attestation



- Problems:
  - Linking different attestation sessions to a platform
  - Possible discrimination of configurations
  - Relay attacks

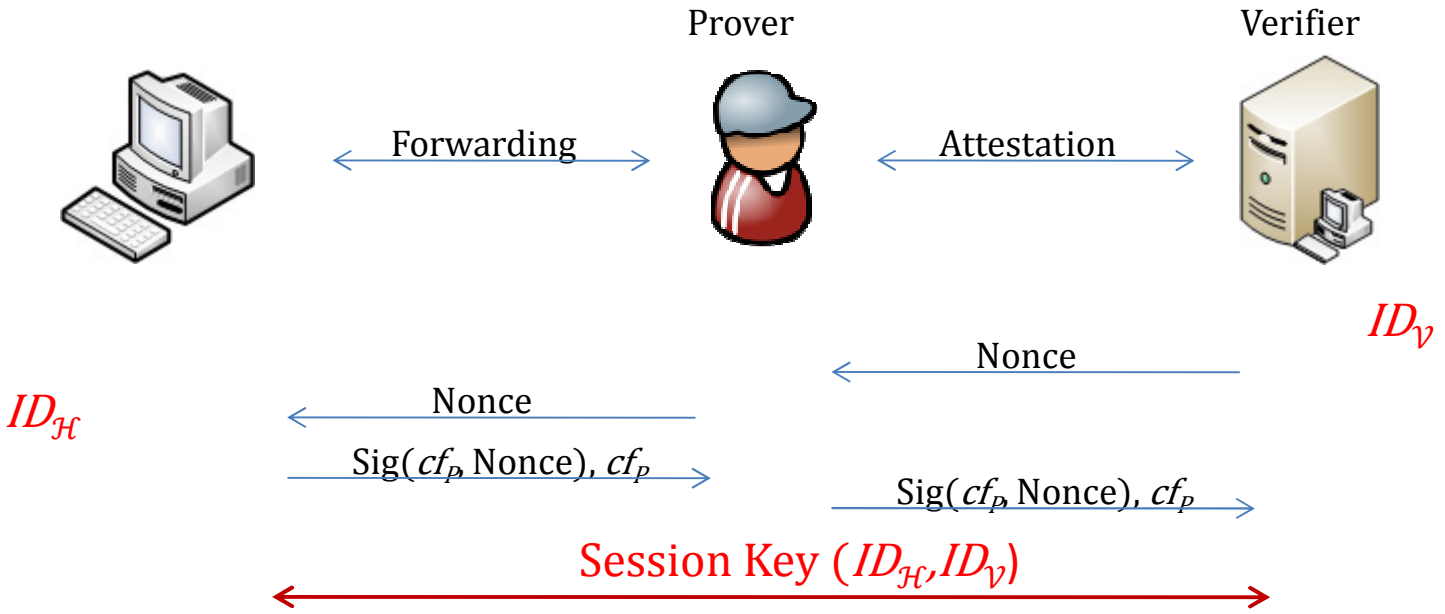
Preventing Privacy Problems



Solution:

- Property-Based Attestation: Attestation over prove "$cf_P \in CF$"
- Can address privacy of configuration and unlinkability of attestation sessions

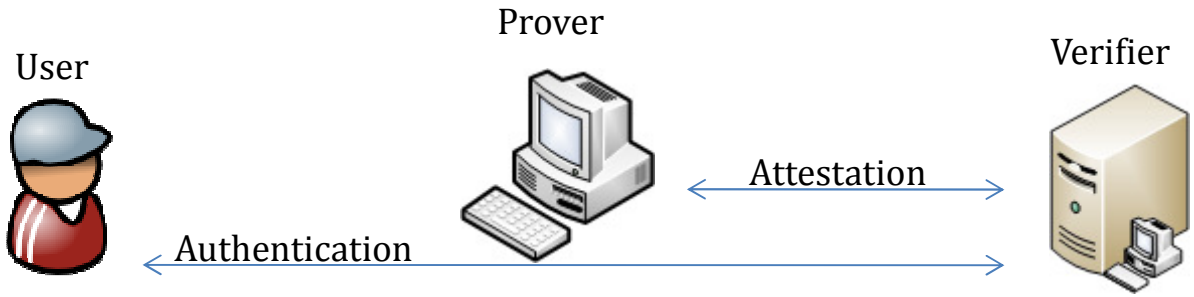
Preventing Relay Attacks



Problem: Attestation can be relayed

Solution: Establishment of session keys

Idea of UPBA



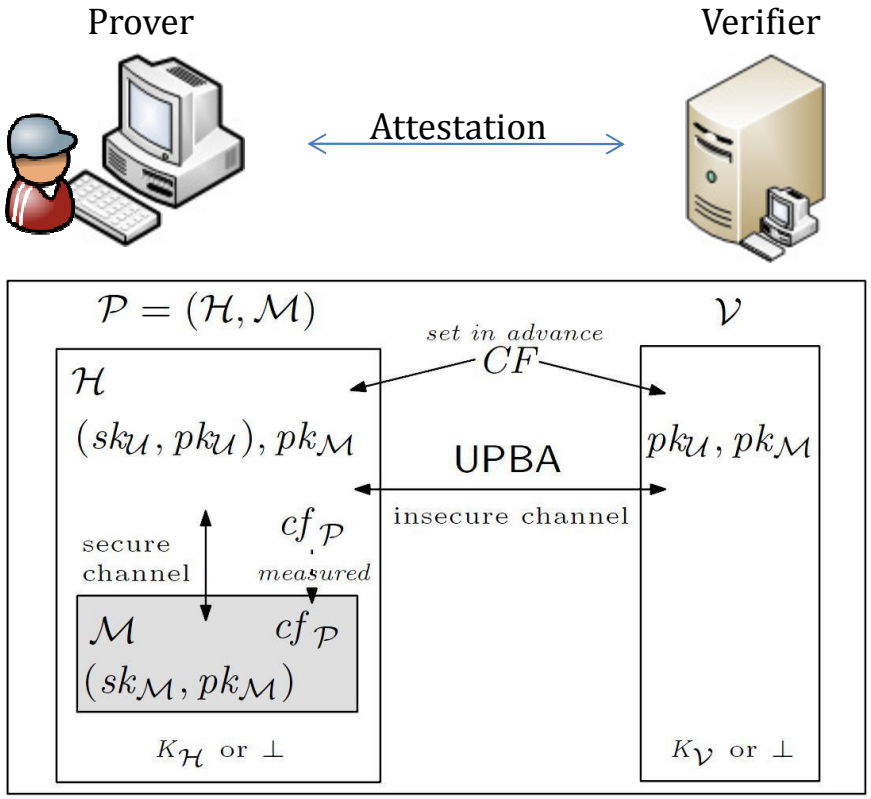
- **Still:** User authentication can be relayed on higher protocol level!
- **Our Idea:** Add user authentication to verify user is using the attested device!

- Build a protocol robust to problems described:
  - Privacy of configuration
  - Unlinkability of attestation sessions
  - Robust to replay attacks
  - User authentication is part of attestation protocol

Building Blocks

- **TPM and Signatures**
- **Commitment Scheme**
  - Allows commitment to some property or message
  - Hides actual property
- **Ring-Signatures**
  - Signature with respect to a set of public keys
  - Allows verification, that one key of the set was used
  - No disclosure of specific Key
- **Key Derivation Function**

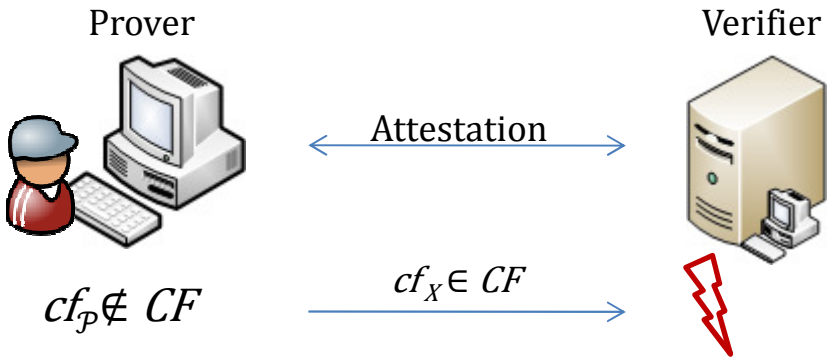
System Model for UPBA



Assumption: Single Ownership Model: User's Secret Keys can be stored on Host

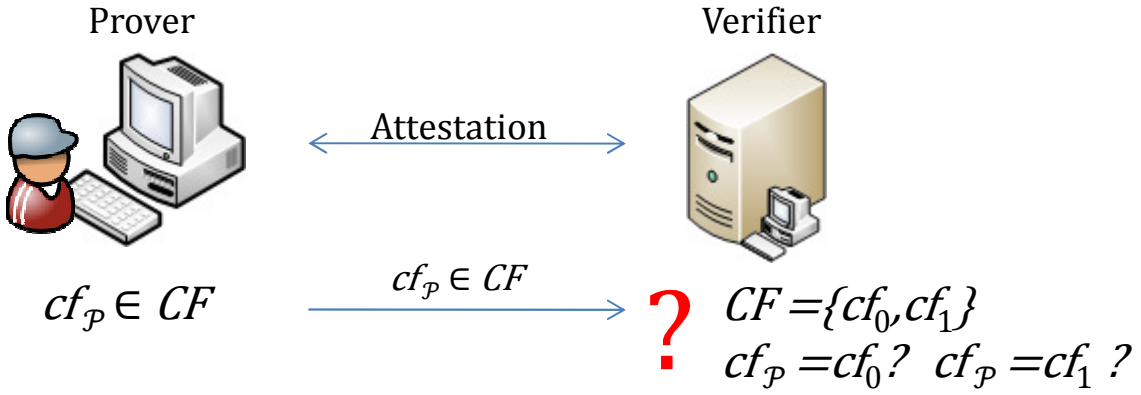
Security Goal: Property Attestation

- **Goal:** Configuration $cf_{\mathcal{P}}$ of prover $\mathcal{P} = (\mathcal{H};\mathcal{M})$ satisfies some predefined property
- **Requirement:** Adversary $\mathcal{A}$ should not be able to convince verifier $\mathcal{V}$ that
  - $cf_{\mathcal{P}}$ in admissible set CF when in fact $cf_{\mathcal{P}} \notin CF$.
- **Setting:**
  - $\mathcal{A}$ can corrupt the host and change its configuration, which captures potential malware attacks as well as malicious user activities



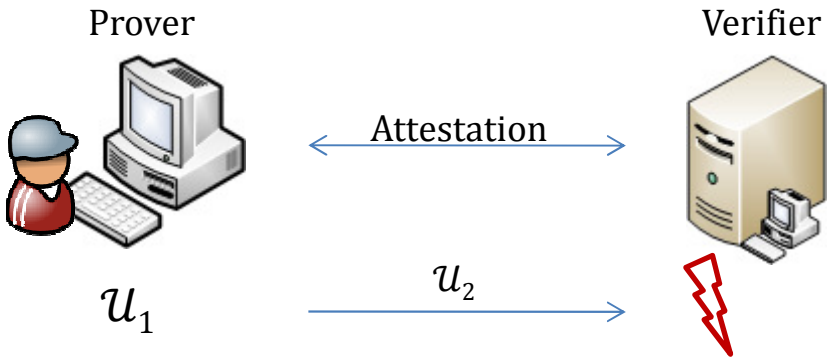
Security Goal: Configuration Privacy

- **Goal:** Hiding actual configuration $cf_{\mathcal{P}}$ of prover $\mathcal{P}$ from malicious verifiers
- **Requirement:** Prevent adversary $\mathcal{A}$ from being able to decide, which configuration $cf_{\mathcal{P}} \in CF$ has been used by $\mathcal{P}$ in a successful execution of UPBA
- **Setting:**
  - $\mathcal{A}$ chooses two different configurations cf_0 ; $cf_1 \in CF$
  - $\mathcal{A}$ as malicious verifier must decide, which configuration has been used in an execution of the UPBA protocol
  - $cf_{\mathcal{P}}$ is implicitly known to TPM $\mathcal{M}$ and host $\mathcal{H}$, $\mathcal{A}$ may not corrupt $\mathcal{H}$



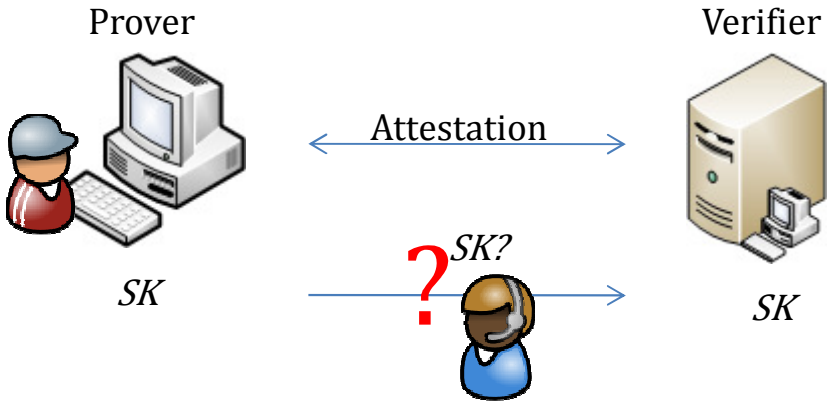
Security Goal: User Authentication

- **Goal:** $\mathcal{U}$ is involved in the attestation protocol
- **Requirement:** $\mathcal{A}$ can not convince verifier $\mathcal{V}$ that some UPBA protocol session between $\mathcal{P}$ and $\mathcal{V}$ is executed on behalf of $\mathcal{U} \neq \mathcal{A}$
  - $\mathcal{A}$ can not impersonat $\mathcal{U}$
- **Setting:**
  - secret keys stored on the host - $\mathcal{A}$ must not corrupt host
  - $\mathcal{A}$ may initiate protocol sessions with $\mathcal{V}$ using other hosts and secret authentication keys of users behind those hosts

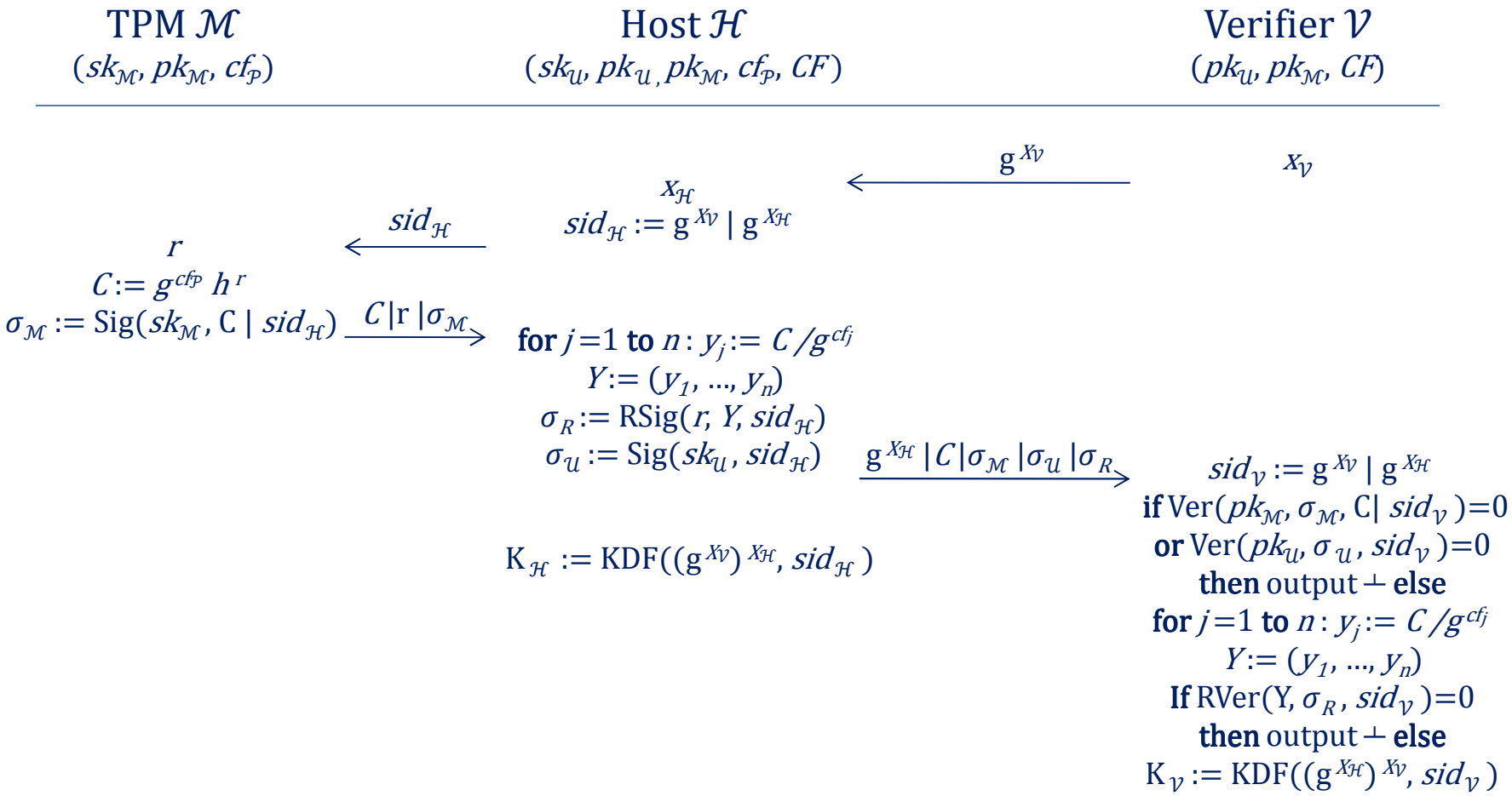


Security Goal: Session Key (SK) Security

- **Goal:** Protect ongoing communication between user and verifier
- **Requirement:** Establish secure session key between protocol participants
- **Setting:**
  - attestation and authentication only on the prover's side, not for verifier
  - Prevention of trivial impersonation attacks on unauthenticated verifiers by restricting $\mathcal{A}$ in sending messages to $\mathcal{H}$
  - $\mathcal{A}$ can send messages to $\mathcal{H}$ only after compromising $\mathcal{V}$ before



Basic UPBA Scheme



Security of UPBA

- **Fulfillment of our Security Goals:**
  - Property Attestation
    - Unforgeability of signature scheme
    - Unforgeability of ring signature scheme
    - Binding property of the commitment scheme
  - Configuration Privacy
    - Unconditional anonymity of the ring signature scheme
    - Perfect hiding property of the commitment scheme
  - User Authentication
    - Unforgeability of the signature scheme
  - Session Key Security
    - Unforgeability of the signature scheme
    - Hardness of the DDH problem in G
    - Pseudorandomness of key derivation function

Conclusion

- Shortcoming of modern attestation techniques concerning user authentication
- No binding from attestation session to user authentication
- UPBA designed to prevent these shortcomings:
  - Relay attack
  - Privacy considerations
- UPBA bases on Property-Based Attestation
- Session keys against relay attacks
- User authentication with secret keys
- Practical and efficient protocol with today TCG-hardware
- Several possible extensions

Backup

Problems Of Remote Attestation

Linking different Attestation Sessions to a Platform: DAA, Privacy CA

[Brickel-Camenisch-Li 2004, Brickell-Li 2007]

Direct Anonymous Attestation Scheme (DAA)

[TCGSpec]

Specification of Trusted Platform Modul (TPM)

Discrimination of Configurations: Property-based attestation

[Sadeghi-Stüble 2004, Nagarajan et al. 2009]

Property-Based Attestation Approaches

[Chen et al. 2008]

Property-Based Attestation without Third Party

Relay Attacks

[Stumpf et al. 2006]

Establishment of Session Keys for preventing Relay Attacks

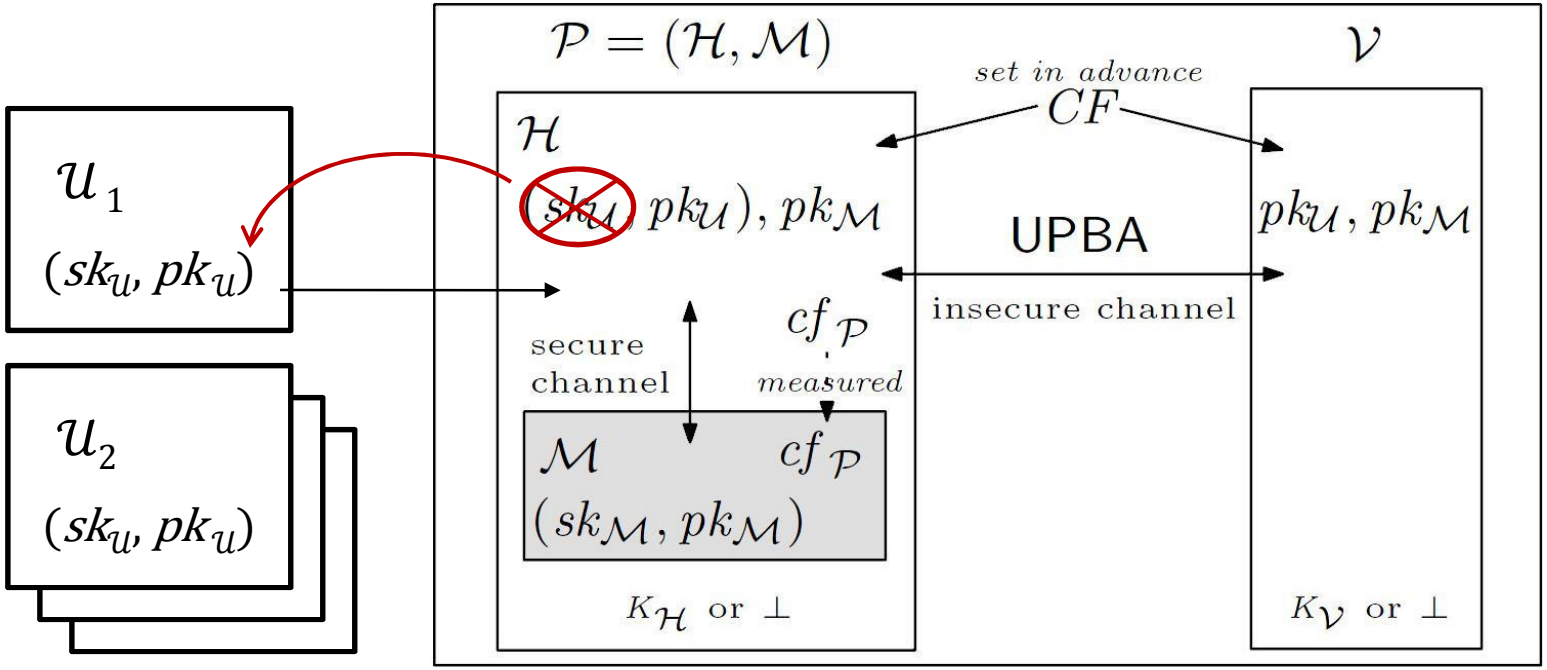
Adversary Model

We model security and privacy of UPBA using a (flexible) game-based approach.

PPT adversary $\mathcal{A}$ interacts with the parties via following set of queries:

send(E, *sid*, m)	with $E \in \{\mathcal{H}, \mathcal{V}\}$. $\mathcal{A}$ can send message m to entity E in session *sid*. Returns message generated by E (if any).
sendTPM($\mathcal{M}$, m)	$\mathcal{A}$ can send message m to TPM $\mathcal{M}$. Can only be asked after corrupt($\mathcal{H}$)
corrupt(E)	with $E \in \{\mathcal{H}, \mathcal{V}\}$. Returns all secrets stored by E and so enables $\mathcal{A}$ to impersonate E.
reveal(E, *sid*)	with $E \in \{\mathcal{H}, \mathcal{V}\}$. $\mathcal{A}$ can reveal session key K_E (if existing) computed by entity E in session *sid*.
test(E, *sid*)	Returns either K_E or an uniformly chosen string $K \in_R \{0,1\}$ depending on $b \in_R \{0,1\}$.

Multi-User Ownership Model



- sk_u may only be accessed by the corresponding user $\mathcal{U}$
- Possibilities: Storage on personal secure devices
- Alternative: Password or credentials not being stored
- Problem: possible corruption of $\mathcal{H}$