Secure Naming for a Network of Information (NetInf)

Christian Dannewitz

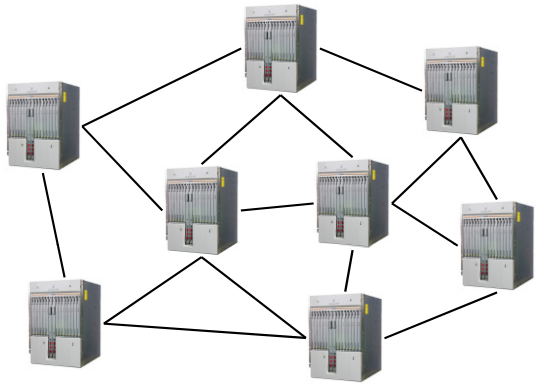
Universität Paderborn

Motivation: Network of Information (NetInf)

Today's Internet

Conversations between Hosts

Host-centric abstraction

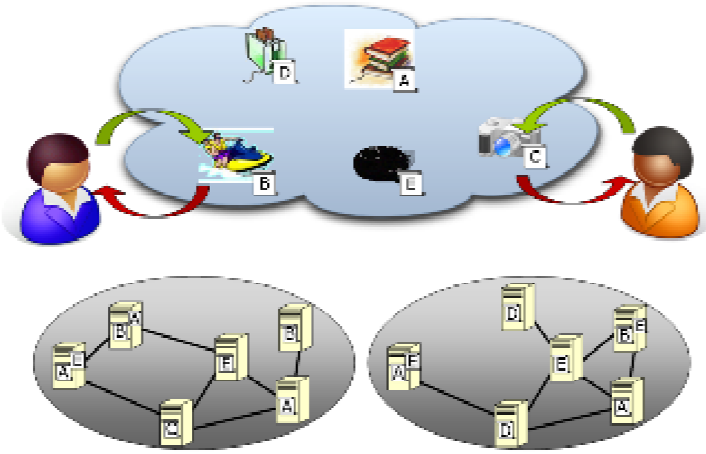


Evolution

Future
Information-centric Network

Dissemination of Information Objects (IOs)

Information-centric abstraction



- No common *persistent naming scheme* for IOs
- Host-centric security
  - *Securing channels* and *trusting servers*
  - No trust: copy from *untrusted server* -> *Caching?*

Outline

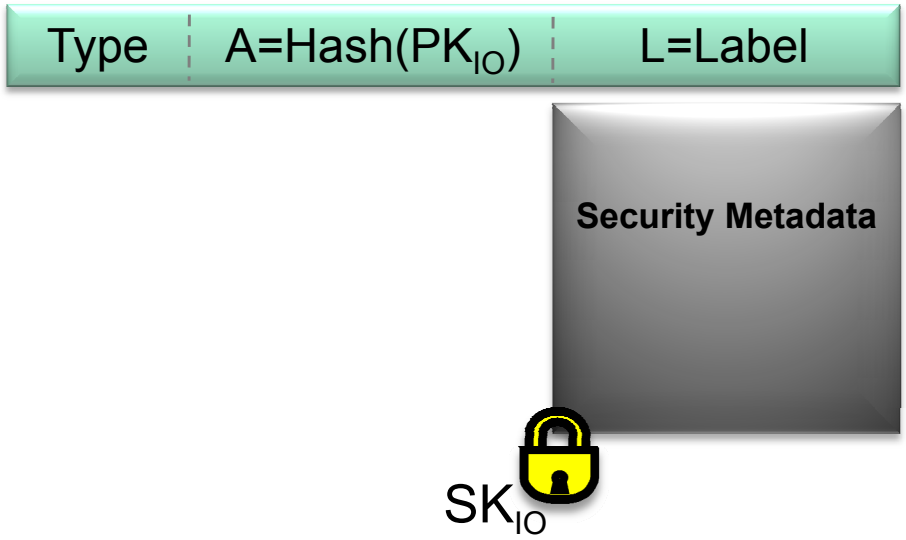
- Requirements / features
- Naming scheme overview
- Security aspects
  - Self-certification
  - Name persistence
  - Owner pseudonymity & owner identification
- Evaluation
- Summary and conclusion

Requirements / Features

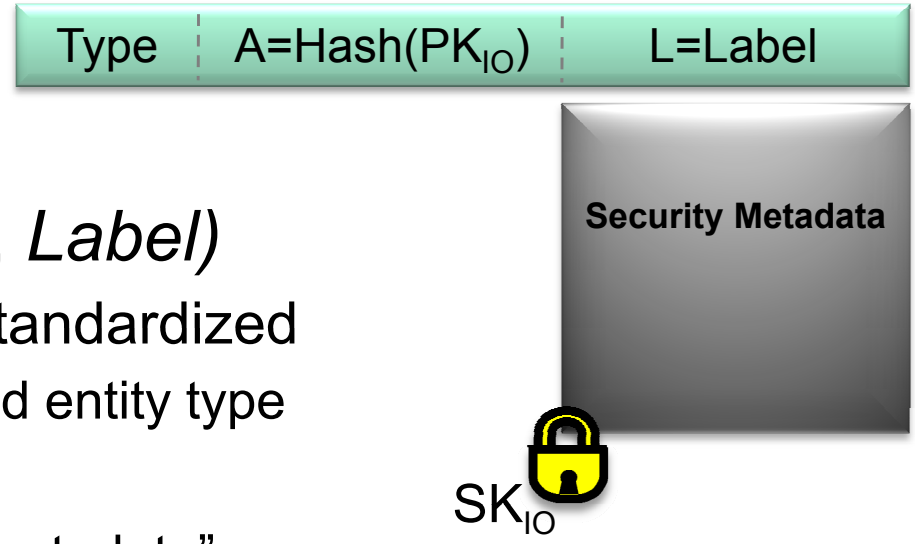
- Secure data retrieval from any available source
  - Name-Data integrity
- Owner pseudonymity
- Owner identification
  - Allow for anonymity
- Globally unique names
- Name persistence
  - Location changes
  - Content changes
  - Organizational changes
  - Owner changes
- Name variety of objects
- Extensible naming scheme

Naming Scheme Overview 1

- Information Object (IO) = (ID, Content, Metadata)
- IO has *owner*
- Identical copies = same ID
  - Different versions = version no.



Naming Scheme Overview 2



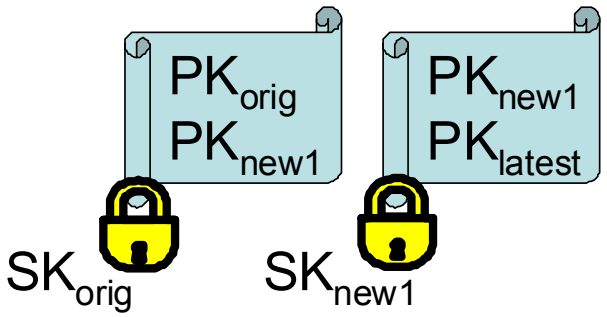
- ID = (*Type tag*, *Authenticator*, *Label)*
  - *Type* tag: mandatory, globally standardized
    - Adapt naming scheme to named entity type
  - *Authenticator A*
    - Secure binding: “ID – security metadata”
    - Owner pseudonymity
  - *Label L*: Arbitrary, ensure global uniqueness
- *Security metadata*
  - NetInf security information, e.g., *hash(content)*
  - Securely bound to ID via PK_{IO}/SK_{IO}

Self-Certification

- Prevent unauthorized changes of named IO
  - Name–data integrity
  - Use any available copy/source
- Static content
  - *Label = hash(content)*
  - Verification: compare hashes
  - No retrieval of security metadata
- Dynamic content
  - *Hash(content)* in ID: violates ID persistence
  - *Hash(content)* in *security metadata,* sign with SK_{IO}
  - Verification: (1) signature, (2) hash

Name Persistence

- Location change
  - ID/locator split
  - Dynamic binding: Name Resolution Service
- Content change
  - See self-certification
- Owner change
  - PK_{IO}/SK_{IO} bound to IO, not owner
  - Basic approach: Pass on PK_{IO}/SK_{IO}
    - Disadvantage: not robust (SK disclosure)
  - Adv. approach: Certificate chain
    - Sign metadata with new *PK'/SK'*
    - Secure chain: *PK'/SK'* – PK/SK – ID
- Owner's organizational change
  - Flat IDs: no organizational structures



Owner Pseudonymity and Identification

- *Owner pseudonymity*
  - Binds data to owner's PK
  - Trust in (anonymous) owner: reuse PK
  - Can be separated from self-certification: different PK'/SK'
    - $Sign_{SK}$(PK + hash(content))

- *Owner identification*:
  - Bind data to owner's real world identity
  - $Sign_{SK}$(PK + PK certificate + hash(content))
  - PK certificate by certificate authority needed (like today)

Evaluation

- Open source prototype: OpenNetInf
  - www.netinf.org
- Easy to implement
  - Established security mechanisms (hashing, digital sign.)
- No overhead for applications
  - Part of NetInf
  - E.g.: Firefox plugin, Thunderbird plugin, video streaming
- Example: Firefox plugin
  - Handle links: *NetInf IDs* instead of *URLs*
  - Automatic content *integrity* check, reduce *broken links*
  - NetInf: Efficient data dissemination

Summary and Conclusion

- Information-centric networking: need for secure naming
- NetInf naming: main feature set
  - Name persistence
  - Self-certification
  - Owner pseudonymity
  - Owner identification
- OpenNetInf prototype: www.netinf.org
- Standardization: IETF Internet drafts
  - “URIs for Named Information“
  - “The Named Information (ni) URI Scheme“ (core syntax, parameters)

Thank you for your attention