Challenges for Naming in Information Centric Networks

Börje Ohlman
Ericsson Research
& colleagues from the EU FP7 projects 4WARD and SAIL
THE SAIL CONSORTIUM

A strong industry-led consortium of leading operators, vendors and research institutions

- 24 Partners
- 12.4 MEUR EU funding
- 2.5 year project
Networking of Information

Today’s Internet

Focus on nodes

Evolution

Web
CDN
P2P

Future Information Centric Network

Focus on information objects and real world objects

In today’s Internet, accessing information is the dominating use case!

SCALABLE & ADAPTIVE INTERNET SOLUTIONS
Traditional node centric networking

Connect to Server X and get object B

Trusted Server

Secure Connection
Information centric networking
Object Lock-in per Application

No seamless communication between objects of different types

Object lock-in per application
Comparable to host lock-in per network before Internet
Application Development on a Common Naming and Reachability Infrastructure

Seamless communication between objects of all types

NRS  Name Resolution System
RVS  Rendezvous System

SCALABLE & ADAPTIVE INTERNET SOLUTIONS
An information centric Waist

Applications
application-specific names

Name Layer
Name Resolution

Example: NetInf name layer

The waist should neither put restrictions on, nor make assumptions about the other layers. Open to global and local routing schemes. C.f. TCP/IP
# Information Centric Naming Schemes

- Proposed information centric naming schemes includes:
  - CCN
    - /parc/van/calendar
  - PSIRP
    - Rendezvous and scope identifiers
  - DONA
    - P:L
  - 4WARD-NetInf
    - T:P:L

<table>
<thead>
<tr>
<th>Type</th>
<th>A=Hash(PK_{10})</th>
<th>L={attributes}</th>
</tr>
</thead>
</table>
Naming Requirements

• NetInf prioritized:
  – Self-certification and self-generation
    • Reduce the need for trust in the infrastructure
    • Data integrity
    • No need for a new naming authority
  – Persistent names, inert to:
    • **Owner change**
    • **Content change**
    • **Algorithm changes** (hash or crypto)
  – Support for all types of information objects:
    • Real world, Services, Streams, Static files, Dynamic data
  – Globally unique names

• More on the wish list:
  – Owner/Publisher authentication
  – Variable length
  – Human readable
  – Minimize load on (Name Resolution) infrastructure
Naming tradeoffs

- Self certification vs. Dynamic objects, Real world object, Services
- Persistent naming vs. Self certification & Dynamic objects
- Flat names vs. hierarchical names
- Verifying ownership by naming vs. persistent names and allowing change of ownership
- Simplicity vs. flexibility
NetInf Naming Scheme
Overview 1

- Information Object (IO) = (ID, Data, Metadata)
- Each IO has an owner
- All equivalent copies have the same ID
  - This might include different versions
NetInf Naming Scheme
Overview 2

- **ID** = *(Type tag, Authenticator, Label)*
  - *Type tag*: mandatory, globally standardized
    - Adapt naming scheme to named entity type
  - *Authenticator A*: bind ID to $PK_{IO}$
    - Secure “ID – security metadata” binding
    - (Original) owner authentication (see owner change)
  - *Label L*: Arbitrary, ensure global uniqueness

- **Security metadata**
  - All information required for embedded NetInf security features
  - Securely bound to ID via $PK_{IO}/SK_{IO}$ pair
Self-Certification

- Prevent unauthorized changes, ensure data integrity
  - Important to support data retrieval from any available copy/source
- Static content
  - Include $hash(content)$ in ID Label field
  - Advantage: no need to retrieve metadata
  - Verification: compute $hash(retrieved\ data)$ and compare to hash in ID
- Dynamic content
  - Storing $hash(dyn\.content)$ in ID would violate ID persistence
  - Store $hash(content)$ in security metadata and sign with $SK_{IO}$
  - Verification:
    - Verify that signature is correct and corresponds to $PK_{IO}$
    - Compute $hash(retrieved\ data)$ and compare to hash in security metadata
Name Persistence

• Location change
  – Based on ID/locator split
  – ID dynamically bound to network location(s) via name resolution service

• Content change
  – See self-certification

• Owner change
  – $PK_{IO}/SK_{IO}$ pair conceptually bound to IO, not owner
  – Basic approach: $PK_{IO}/SK_{IO}$ pair securely passed on to new owner
    • Disadvantage: not robust with respect to SK disclosure
  – Adv. approach: new owner uses new $PK’/SK’$ pair
    • Sign metadata using the new $PK’/SK’$ pair
    • Securely bind $PK’/SK’$ pair to ID via certificate chain

• Owner’s organizational change
  – IDs are flat and do not reflect organizational structures
Evaluation: Proof of concept prototype

- Java-based NetInf prototype
- Naming scheme proved easy to implement
  - Based on established security mechanisms (encryption, digital sign.)
- Easy to integrate and use naming scheme in applications
  - Built applications from scratch
  - Extended existing applications (e.g., Firefox, Thunderbird)
- Example: Firefox plugin
  - Interprets links containing NetInf IDs instead of URLs
  - User advantages: automatic content integrity check, reduce broken links
  - Publishers advantages: simplify content management via persistent IDs
- Load and overhead not an issue
  - Implementation also smoothly running on Android cell phones

Prototyping done by University of Paderborn
Network storage –
IETF DECADE WG

• Primarily aimed at supporting P2P application
• Could also be beneficial for other application, especially large scale distribution services like Internet TV
• The offline and badly connected peers problems are mitigated by in-network storage
  – Minimizes the use of week uplink networks
• Content caches are easily migrated towards flash-crowds
Preliminary Conclusion

• Don’t integrate a specific routing scheme in the architecture. This will make the architecture rigid, and it will break under the pressure of future technology development.

• The name resolution system should be open to:
  – a variety of schemes at the transport and forwarding layers
  – extension to functions such as:
    • Policy routing
    • Object, host, and network mobility
    • Object, host, and network multihoming
    • Policy control and scoping
    • Caching
Summary and Conclusion

• Design of a new network architecture based on information-centric paradigm
  – Rather than based on a host-centric paradigm
• Naming scheme for information objects needed.
  – NetInf proposal

| Type | A=Hash(PK_{10}) | L={attributes} |

• Feasibility of secure naming demonstrated via prototyping:
  – http://www.4ward-project.eu/
  – http://www.sail-project.eu/
  – http://www.netinf.org (Open source site)
Backup slides
Identifiers and Information Modeling

- Persistently identify information
  - Location-independent identifiers
  - Represent multiple copies

- Representation of information via Information Objects (IOs)
  - Another level of indirection
  - Represent information independent of a specific copy
    - E.g. a text, a song
  - Consists of a set of attributes
    - Including media components, GPS location, access rights, encoding

- Information Objects can also represent:
  - Streams
  - Services
  - Real-world objects (e.g., a physical copy of a book, a person)

- IOs can be used to organize information
Organize Information – IO, DO and BO definitions

- **Information Object (IO)**: An Information Object is a set of attributes defining the semantics of a data object. An IO may refer to a piece of music, a film or a webpage. Can be static, dynamic or real-world objects, including streams and services.

- **Data Object (DO)**: Sub-class of IO holding attributes for bit-level objects and pointer(s) to the actual data.

- **Bit-level Object (BO)**: A specific sequence of bits, independent of any semantic meaning, also independent of where they exist, like in a file, on the wire, in the air or in a primary memory.

---

- **Song1.mp3**
- **Song1.wav**

---

- **Song1**
Secure naming & P2P application interaction

• With self-certifying names, the data received is the data requested in P2P system
• In today’s P2P system, no guarantee that the downloaded content actually matches the expected/correct content
  – Like forged torrent file and/or data file can be inserted
• Additions to P2P
  – Extend torrent file with additional security metadata
  – Generate torrent name along draft method (draft-dannewitz-ppsp-secure-naming-00.txt)
**Metadata**

- Secure naming structure supports additional metadata
  - Needed for instance for PK_D and signing purposes, persistent naming
  - Additional metadata can be data attributes:
    - Classification
    - Meaning of data
    - Data status

- **Search**
  - Metadata can be used for attribute based search
  - Potentially more accurate search than full text search
Owner Authentication and Identification

- Owner authentication separated from data self-certification
  - By allowing the corresponding PK/SK pairs to be different
  - Owner authentication is possible even if multiple owners use the same PK/SK pair for data self-certification
  - More freedom in the choice of PK/SK pairs for data self-certification
- **Owner authentication** binds self-certified data to owner’s PK
  - Include hashed owner’s PK in self-certified data and sign this data with the corresponding SK (anonymous)
  - Build up trust in (anonymous) owner by reusing PK for different IOs
- **Owner identification**: in addition, bind self-certified data to owner’s real world identity
  - Achieved like owner authentication, where owner’s PK and identity data are included in self-certified data
  - Owner’s PK and identity are bound by PK certificate issued by TTP
Networking of Information Vision

- Take information-centric networking to the next level
  - General-purpose information-centric architecture
  - Generalize CDN and P2P benefits to be integral part of network services
  - Commoditize application level content distribution
  - …for a broad range of applications

SCALABLE & ADAPTIVE INTERNET SOLUTIONS
Internetworking Layer for all Types of Objects

Seamless communication between objects of all types

NRS  Name Resolution System
RVS  Rendezvous System