Future Internet Architectures: Re-thinking the Fundamentals

Adrian Perrig
Network Security Group
Spring 2020
Future Internet Architecture is becoming a hot Topic

- EU has launched a major initiative on Next-Generation Internet (NGI): https://www.ngi.eu
Outline

In this lecture

• Overview of 4 FIA architectures, funded by NSF 2010-2016 (Slides 4-47 largely by Dave Clark)
  – XIA
  – Nebula
  – Mobility First
  – NDN
  – Plus EU system: PSIRP/Pursuit

• Brief SCION overview (starting from Slide 48)
Four schemes from FIA Slides by Dave Clark

- NSF overview: http://www.nets-fia.net
- eXpressive Internet Architecture (XIA)
- Nebula
- Mobility First
- Named Data Networking (NDN)
  - Previously known as Content Centric Networking or CCN.
... plus one scheme from EU

- PSIRP/Pursuit.
  - Another example of Information Centric Networking (ICN).

- Several others, but I just don’t understand them well enough to dare explaining them.

- Notable scheme missing: RINA (am still trying to understand it)
  - Much current activity in EU
High-level goals:

- Trustworthy: security, broadly defined.
- Evolvable usage model: hosts, services, content, ???
- Supports technology evolution.
- Effective operation of all actors.
  - Even with different goals and incentives.

http://xia.cs.cmu.edu
XIA forwarding

- XIA has classes of “locators”, which they call “identifiers”, or XIDs.
  - CID: identifier for content C
    - Hash of the content.
  - SID: identifier for service S
    - Hash of public key of the service.
  - HID: identifier for host H
    - Hash of public key of the host.
  - AD: autonomous domain identifier
    - AS number in today’s Internet
XIA: core novelty

- Routers have a distinct Per-Hop Behavior (PHB) for each kind of ID.

- Put more than one kind of XID in the packet.
  - Put them in an ordered list (actually a DAG).
  - Each router starts with the first one, and if it does not have a matching PHB (or the PHB fails), “falls back” to the next one in the DAG.
    - They call the first one the “intent”.

\[ \text{Diagram: } AD_1 \rightarrow AD_2 \rightarrow CID_1 \]
Some examples

- **CID, AD**
  - If the router knows where the content is, send it there, otherwise route toward the AD.
    - CID must be known within the scope of AD.

- **CID, SID, HID**
  - Forward toward the content, else to a service that can find the content, else to a host where the service is located.

- **HID, AD:**
  - Forward to the network where the host is to be found.
    - Think of the AD as like an AS number in today’s Internet.
XIA: Benefits

- Fallback allows the incremental deployment of new classes of PHB.
  - Achieves goal of evolvable usage model.

- Hash of public keys allows “self-certification” of communication.
  - Did I get where I meant to get: my “clean failure” approach.

- ADs provide routing at level of AS, similar to today’s Internet.
  - Since flat identifiers make routing hard.
Secondary specification

- Name services to map from user-meaningful names to XIDs.
  - These *must be* trustworthy.
  - More than one: avoid "root of trust" problems?
XIA: Location and identity

- XIDs function as identifiers, but also as locators.
  - Use nesting of IDs to make routing efficient.
- Getting an ID from the name service is an excellent idea.
  - We could do that with a modified DNS today.
  - But why use the ID as the locator?
What is missing?

- Fault isolation and user-driven re-routing.
  - They have a scheme called SCION to do this.

- Privacy of intent.
  - Lots of discussion about this.

- Routing:
  - They take this as solved.

- Vertical interfaces.
  - Not their priority—someone can sort this out.
Mobility First (MF)

- http://mobilityfirst.winlab.rutgers.edu
- High-level goals:
  - Host and network mobility at scale.
  - Exploit intrinsic properties of wireless.
  - Secure and trustworthy operation.
  - Support requirement of mobile apps.
    - Context, locality, etc.
MF: forwarding

- MF has classes of identifiers, like XIA.
  - Content, service, host, network.
  - Called Global unique ids, or GUIDs
- Routers have separate PHB for each class.
- Fixed format address:
  - GUID:NID
  - The GUID must be routable within the network N (e.g., two-level flat addresses).
MF: novelty

- Sender gets GUID from a name service.
  - Like XID in XIA.
- Global name resolution service (GNRS) maps GUID to current NID.
  - Receiver must keep this mapping current.
- As the packet is forwarded, if network N does not know about the GUID, it can re-query the GNRS to get updated location.
MF: the challenge

- Build a global system that can map perhaps 100 Billion GUIDs to NAs in 100 ms.
  - 100 Billion is their number—I think it is much too low, since they claim to deal with services and content.
  - Massively scaled “late binding”.

- The recurring vision of routing on flat names.
  - XIA assumed the correct NID was in the higher-level naming system.
MF: other features

- Routers at boundaries of wireless regions can hold content if connectivity is poor.
  - Storage-aware routing.
- Support for other sorts of PHBs (caching, computation).
  - Multicast, anycast, multi-homing.
- Region-by-region transport.
  - Tricky...
- Self-certification (similar to XIA).
Digression: flat name spaces

- A recurring idea: can we route on flat identifiers?

- One approach: distributed hash tables (DHTs).
  - Mobility First is exploring this approach.

- Another approach: nested addresses
  - XIA uses this approach.
  - How many levels? Fixed or variable?
Why do this?

- IP addresses have structure to make forwarding (and route computation) simple.
  - Seems like a good idea.
  - But using address as identifier was *bad* idea.
    - But is a “poor man’s” security.

- Advantages of flat addresses.
  - Better support for mobility.
    - I am not convinced.
  - Eliminates a lookup (round trip delay)
    - I am not convinced.
High-level goals:

- **High reliability router for cloud operation.**
  - Not part of what I am going to discuss.

- **Fine-grained control of forwarding path**
  - By all actors.
  - Availability (end-nodes can avoid regions).
  - Extensibility of forwarding policy (forwarding supports *any* policy).
  - Policy enforcement.
Nebula: defining policy

- Each potential actor is represented by a domain.
- Each domain has a behavior: Per-Domain Behavior (PDB)
  - Forwarding, computation, validation, etc.
- A policy is a sequence of domains through which the packet must flow, such that all the actors agree that the sequence of PDBs is acceptable.
  - Must specify what that PDB is.
Nebula: two layer architecture

- Nebula data plane (NDP) forwards packets according to a pre-set policy.
- Nebula Virtual And Extensible Network Techniques (NVENT) computes policies.
  - A global, distributed multi-agent system in which all relevant actors can specify service offerings and service constraints, to result in an acceptable sequence of PDBs.
Before a sender can send a packet, it consults the NVENT system.

NVENT computes and returns to the sender a source route through the domains called a “proof of consent”, (PoC).

As the packet is forwarded by the NDP, each domain computes a proof that it has performed its service, which goes into the packet as a “proof of path” (PoP).

Through the use of cunning crypto, each domain can inspect the PoP to that point and confirm that the previous steps actually happened.
NDP in a nutshell

- Use cryptography for:
  - Proof of consent (PoC) – route *authorized*?
  - Proof of path (PoP) – route *followed*?
The data plane scheme in Nebula is based on a scheme called *Icing*.


Take a look if you want to see how the crypto works, and a traditional, performance-oriented discussion.
Nebula: what is tricky?

- What distributed algorithm can be used to compute arbitrary policies?
- Are there algorithms to compute classes of policies
  - For example, routing protocols?
- What is the overhead of computing a policy to send a packet?
  - Is it feasible to pre-compute policies?
  - Like routes, for example?
Named Data Network

- https://named-data.net/project

High-level goal:
  - Rethink the fundamental paradigm of what networks do.

- Old idea: communicate among nodes.

- New idea: disseminate information.
  - Names for stuff, not endpoints.
  - Both memory and communication explicit in design.
  - Security focus on the stuff, not the connection.
Getting your head around this

- **Spoon boy**: There is no spoon.
- **Neo**: There is no spoon?
- **Spoon boy**: Then you'll see, that it is not the spoon that bends, it is only yourself.

http://www.youtube.com/watch?v=ZaJPNrf1DPY

- There are no addresses (or locators), only the identifiers of packets of *stuff*.
NDN: the basics

- Two kinds of packets: *interests* and *data*.
  - NDNI and NDND

- **NDNI**: contains a content name CN.
- **NDND**: contains the CN, the actual content, and a signature.

- CNs are hierarchical (NOT FLAT).
  - /parc/videos/WidgetA.mpg
  - /parc/videos/WidgetA.mpg/1/3
NDN forwarding

- (Ignore routing for the moment…)
- To get a data item, send a NDNI.
- As an NDNI arrives at a router:
  - Look up the NDNI to see if the content (THE NDND) is there. If so, return it.
  - In a local table, check if the NDNI already exists
    - yes: add the path it came in on. Drop NDNI.
    - no: store the path it came in on. Forward NDNI based on standard routing table.
- As an NDND arrives at a router:
  - See if there is a matching NDNI stored.
  - If so, forward along the path, if not drop.
NDN: what is tricky?

- Making routing work—sending the NDNI in the right direction to find the data.
  - Profoundly tricky if there are no addresses.
- Per packet state in every router.
  - Not per flow, per packet. Every NDNI is stored at each step.
    - That is why there are no addresses.
    - The NDNI lay down a trail, the NDND follow it.
NDN: benefits

- Highly efficient delivery of popular content.
  - Cache the NDND packets along the path.
- Most forms of DDoS eliminated.
  - Cannot flood a site with unwanted NDND.
    - Just get dropped.
  - Cannot demand content over and over again.
    - Caching will diffuse the attack.
  - Need to keep asking for new content.
- Traffic analysis degraded.
  - No addresses, remember?
Every NDN forwarder has a *strategy* component, that tries to find an effective way to forward a NDNI.

- Could be driven by the name: `/parc/videos/WidgetA.mpg/1/3`
  - But how???
- Could just flood.
  - Storage of NDNI prevents loops.
- A rich topic for research.
- In practice: NDNI also contains destination address, and traditional routing protocol used.
NDN security

- The authoritative creator of an NDND signs it with its private key.
- Public keys are widely known.
  - Forwarders remember popular public keys.
- Any node with the public key can verify:
  - The content is not corrupted.
  - The name matches the content.
  - Who owned the public key.
  - However, signature verification is too slow for router to verify all NDNDs.
- Same old problem: how to get the key reliably?
NDN: warning

- This discussion is very incomplete
  - There is another kind of packet, the sync.
- NDNI can be used to control things.
  - In which case, they must be signed.
Information Centric Networks

- ICNs were all the rage at the time.
  - Workshops, conferences, etc.

- NDN gets a lot of attention, but there are other variants.
  - XIA is (to some extent) is ICN.

- Others:
  - PSIRP/Pursuit
  - NetInf (part of 4ward)
  - Dona
PSIRP/Pursuit

- High-level goals:
  - Security
  - Mobility
  - Scalability
  - Evolvability
  - Quality of Service
  - Viable economics
  - Multi-dimensionality

- Specifics
  - Media-independent information access
  - Information governance
    - Tussle mediation
  - Privacy and accountability
  - Information scarcity
Components

- Publishers, subscribers, brokers.
- Rendezvous point (RP)
- Recursive information semantics
- Scopes
- Trust-to-trust
Information IDs

- Every information item identified by a pair:
  - Rendezvous ID (RId) and (nested) Scope (SlId)
  - RId is like CID in XIA, except...

- RId must be unique within the scope, and scopes can be nested.
  - IDs are flat (mostly) in the scope of meaning.
    - But nested, and not globally unique.

- Within any scope, names must be resolved to a lower-level locator (e.g., an IP address).
  - Scopes are not necessarily topological.
  - But they are represented by lower-level locators.
To publish an item:

- Pick a scope.
- Generate a Rld unique in that scope.
- Determine a physical location to host the item.
- “Register” the item with the scope.
  - E.g., send a message to the machine representing the scope, with the Rld and the location(s) of the item.
- Decide how to advertise the name.
  - Not all scopes are globally known.
Send a message that is routed through the nested sequence of scopes until it reaches the machine in the final scope that can resolve the RId into a lower-level rendezvous location.

- The content can be in many places, not topologically bounded.

The source address and the item location are handed to a topology manager that sets up a path along which the item is returned.

- If PSIRP is an overlay on IP, could just use normal routing, but this is more general.
Compare to NDN

- Subscribe = *Interest*. No Publish in NDN.
- In PSIRP, routing of the “subscribe” is explicitly driven by the scopes.
  - Lower-level addresses are necessary and explicit.
- In NDN, the name can implicitly guide the routing, but no explicit binding.
- In PSIRP, content caching is managed.
- In NDN, all routers can be caches.
- PSIRP caches items.
- NDN caches packets.
Compare to XIA / MobilityFirst

- IDs are hash of content, so presumably globally unique.
  - XIA uses a form of scoping, but it is topologically constrained (NID).
  - Mobility First assumes the ability to map from ID to location at global scale.
    - NetInf also assumes this.
    - Updating this is sort of like a PSIRP publish.
  - XIA uses higher-level name system as the mechanism to “publish”.

Same ideas, different words.

What is the difference between PSIRP scopes and a hierarchical naming system like DNS, but with multiple roots?

- PSIRP gives names to the parts (e.g., “scopes”).
  - Names do not have human meaning.
  - But they could have those kinds of names as well.
- XIA says naming is outside the architecture, but has to resolve names to IDs.
- NDN assumes higher-level naming.
Summary thoughts

- All these schemes address the “locator-identity split”.
  - Separate what I am trying to get from where it is.

- All these schemes address the idea that most applications don’t want to get to a node, but a higher-level abstraction.
  - But should the network or the application control that?
SCION Overview

- Path-aware networking architecture, optimized for high security, efficiency, and scalability
  - No “special” purpose architecture, simply highly available end-to-end packet delivery
- Majority of network attacks are prevented by secure architecture design
  - Supported by flexible and scalable trust architecture
- Main goal: provide communication availability in a network with malicious parties
- Packets contain AS-level path
  - Per-flow stateless operation of routers, no inter-domain forwarding tables
    - Enables native multi-path communication
- Deployed and operational, in production use today
SCION in a Nutshell

Path-based network architecture

Control Plane - Routing
- Constructs and Disseminates Path Segments

Data Plane - Packet forwarding
- Combine Path Segments to Path
- Packets contain Paths
- Routers forward packets based on Path
  - Simple routers, stateless operation

Packet: F→D→B
B→K→L
L→O→S

Payload

Isolation Domains
Real-World Deployment: SCION Production Network

- Led by ETH spinoff Anapaya Systems
- Important point: BGP-free global communication
  - We need failure-independence from BGP protocol
- Discussions with domestic and international ISPs
  - Goal: First inter-continental public secure communication network
- Construction of SCION network backbone at select global locations to bootstrap adoption
- Current deployment
  - ISPs: Swisscom, Sunrise, SWITCH, +others
  - Bank deployment: 4 major Swiss banks, some in production use
  - Swiss government has SCION in production use
Summary

- With the tremendous expansion of the Internet, new use cases expose limitations
  - Amazing how well the Internet held up, knowing that BGP was designed to scale up to 1000 networks (today > 60’000 ASes)

- Numerous new use cases
  - Secure / available communication
  - Large-scale data dissemination
  - Cloud computing
  - IoT
  - Mobile devices
  - Future use cases?