Friction

Moving up-level is an amplifier.

- *We shouldn't amplify mistakes.* (E.g., if you accidentally delete a file anywhere, FolderShare makes sure it's deleted everywhere.)

- *We shouldn't amplify attacks.* (Machines need a very high level of confidence in context & data integrity).

- CCN gets rid of a useless abstraction (the host and file that contain the bits) and captures:
  - *Ontology* via hierarchical names and links.
  - *Provenance* via signing the binding between the bits and their name ("Z asserts that X is his name for Y")
  - *Locality* via a "guided diffusion" dissemination model.
- CCN reduces friction by moving from a 'container' to a 'collection' model.

Why is networking & data security so bad?

- Files and network connections are containers for information.

- Containers are inherently insecure (contents can be replaced).

- Containers are *not* intrinsic (think SVN or git versioning).

- It is very hard to secure a container. It's relatively easy to secure content.

CCN gets rid of containers

- CCN Names identify an information *collection* (*not* an information container).

- Name hierarchy indicates membership.

- The same information can have many names (web-like links).

What's in a Name (user/app view)


```
/parc.com/van/cal/417.vcf/v3/s0/0x3fdc96a4...
```

Note that this binding is *immutable* (the data associated with the name can't change). This changes the *coherency problem* to a communication problem:
A remote change makes your knowledge incomplete but cannot make it wrong.

Establishing Provenance

/parc.com/van/cal/417.vcf/v3/s0/0x3fdc96a4...

signed checksum 0x1b048347

key

parc.com/van/cal/desktop public key

Signed by parc.com/van

Signed by parc.com

Metadata contains encrypted cryptographic checksum and locator for the public key of the producer. Producer's key is typically hierarchically structured.

- History and motivation
- Content Model
- Security Model
- Node Model
- Routing
- Transport

- Content is opaque to TCP/IP so security is a one-size-fits-all wrapper with no notion of context and very weak notions of provenance.

- All trust is delegated to highly-suspect ‘root’ certification authorities (263 in M$ IE).

- Because it can’t name content, fine-grain keying is almost impossible.

- Security policies are expressed in terms of *‘what can be shared’*, network enforcement is in terms of *‘who can talk to who’*. It’s hard to make these congruent.

CCN signing

- Packets in CCN are authenticated and publicly verifiable (as opposed HMACed and verifiable only by the endpoints).
  - This doesn't mean computing a digital signature on every packet. For example, Merkle hash trees or ticket-signing mechanisms can be used to amortize signing cost.
- Public verifiability enables scalable, cooperative consistency checking even with massive replication.

Trust model

- We are doing straight-up SDSI for the trust model with a lot of control over local namespace linkage.
- It is intuitive and fully distributed yet very strong (fully axiomatized and verified).

R. Rivest, *A Simple Distributed Security Architecture (SDSI)*, 1996
J. Halpern, *A logic for SDSI's linked local named spaces*, 1999

Key distribution

- CCN is an ideal communication model for fine-grained trust and privacy: signing and encryption key names can be derived from content names then retrieved via CCN.
- Everyone doesn't have to establish trust in the same key the same way.
- CCN doesn't care how you encrypt your data or encode your decryption keys; we just offer some suggestions to help.

‘Identity creates organism’

– John Maynard Smith, 1920-2004

- The only configuration a CCN node needs is its ‘identity’ (signing key).
- Signing creates a robust “sense of self” that:
  - Tells an element what configuration and coordination information it can trust.
  - Tells it who to cooperate with.
  - Lets elements recognize and help repel invaders. E.g., “defending” a name space.