Secure Naming in Information-centric Networks

Walter Wong
University of Campinas
Campinas, Brazil
wong@dca.fee.unicamp.br

Pekka Nikander
Ericsson Research NomadicLab
Finland
pekka.nikander@nomadiclab.com

ABSTRACT

In this paper, we present a secure naming system to locate resources in information-centric networks. The main goal is to allow secure content retrieval from multiple unknown or untrusted sources. The proposal uses a new, flexible naming scheme that is backwards compatible with the current URL naming scheme and allows for independent content identification regardless of the routing, forwarding, and storage mechanisms by separating the source and location identification rules in the URI/URL authority fields. Some benefits of the new naming system include the opportunity to securely retrieve content from any source in the network, content mobility, content validation with the original source, and full backwards compatibility with the current naming system.

Categories and Subject Descriptors

C2.1 [**Computer-Communication Networks**]: Network Architecture Design

General Terms

Architecture, Security

Keywords

Information networking, naming system

1. INTRODUCTION

The Domain Name System (DNS) was introduced in the Internet to overcome the administrative burden caused by the growing number of hosts. As more and more computers were deployed around the world, the size of the `hosts.txt` file grew to unmanageable sizes, demanding a new resolution system that was at the same time scalable, distributed, and administratively easy to manage. The original functionality required was the resolution of a hostname into an IP address, easing the access to remote computers through human-readable names.

ACM ReArch 2010, November 30, 2010, Philadelphia, USA.

Despite the success of the naming system over the years, the current DNS faces technical limitations to attend new security requirements, for instance, resilience against security threats such as denial-of-service attacks and DNS cache poisoning [3]. In addition, the lack of awareness of the DNS about the location of the clients prevents it to efficiently return the location of the closest available sources [6]. We argue that the root cause of these problems lies in the underpinnings of the architecture: the lack of secure content identifiers that are simultaneously independent of routing, forwarding, and storage mechanisms.

The motivation of our work is to provide a more flexible naming system to enable content retrieval from multiple sources with a security mechanism embedded in the name, in the same line as proposed in the IETF DECADE working group [4]. We aim at a mechanism that allows for clients to receive and verify pieces of content from multiple sources that may not be trusted or that are not even known beforehand. In this scenario, we want to leverage security mechanisms based on the information that each data chunk carries instead of using transient security parameters from the current security protocols, such as SSL and IPSec. In those protocols, we are able to authenticate and verify the data, but we are not able to validate it[1].

In this paper, we propose a new secure naming system that separates the roles of authority, content identification and location, leveraging security regardless of the network location. The main idea is to separate the security functionality in the names (or URLs) from the routing, forwarding and storage primitives, allowing the content to be verified regardless of its storage location from where it was retrieved. Hence, instead of resolving a name into an IP address, the proposed resolution mechanism resolves it into a metadata structure containing a set of permanent data chunk identifiers that can be used to retrieve the data from the network. The benefits of the proposed approach include the possibility to retrieve content pieces from multiple unknown/untrusted sources, opportunistic content retrieval from local caches, content authentication with the original provider, and backwards compatibility with the current naming system based on URLs.

The organization of this paper is as follows. Section 2 presents the secure naming proposal, starting with the motivation of our work, and discussing the naming system. Section 3 describes the implementation architecture and presents the ongoing implementation. In Section 4 we provide a brief

[1]By *validity* we mean that the received data is what the user really looked for and not just unmodified data.

security analysis of the designed system. Section 5 describes the related work and compares with our proposal, summarizing the main differences. Section 6 discusses deployment issues. Finally, Section 7 concludes this paper and presents the future work.

2. NAMING SYSTEM DESIGN

The naming system was introduced in the Internet to ease the access to resources by using names in the Internet. At the same time, as the Internet grew in scale and the bandwidth usage migrated from remote login and email to bandwidth-hungry user generated content (e.g. YouTube), the naming system started to show its limitations. One critical limitation is the *lack of location awareness* of the DNS [6], which hinders the possibility to return the closest server based on the client's location. As a consequence, client requests need to traverse the entire path towards the IP address indicated by the DNS despite any other closer source in the network. Content Delivery Networks (CDN) [10] came as an alternative to increase content availability by redirecting the DNS requests to their closest servers where the requested content could be retrieved. However, CDNs are restricted to a group of clients that pay this service that the DNS can not provide natively.

From the security point of view, DNS is prone to a number of attacks, such as cache poisoning and denial-of-service attacks. DNSSec [5] was proposed to tackle the security issues in the DNS, mainly leveraging the *provenance* of the resolution mechanism. Clients resolving a name would receive both a signed response together with the signer's public key, allowing the verification of the response. However, DNSSec requires the establishment of a trust relationship between clients and the resolution infrastructure. We will show in the security analysis section that this trust is not necessary.

2.1 Design goals

In order to address the limitations described above, we established a set of design goals with *simplicity, scalability, robustness* and *security* in mind:

- Backwards compatibility: The new naming system should be backwards compatible and also be incrementally deployable in the current Internet architecture;
- Content authentication: the separation of the data authentication, identification and location allows for data retrieval from multiple locations, content replication and mobility;
- High-level content identification: persistent content identifiers with embedded security properties allows for location-independent labels, resulting in identifiers that can be used to identify data regardless of the storage type, e.g., network caches;
- Provenance: the naming system should provide mechanisms to *verify* and *validate* the content pieces with the original content provider. The main idea is to transfer the trust placed in the storage location (mirror) to the content itself and to the authority over the content.

2.2 Initial design

In order to design a secure naming scheme, we introduce three definitions that represent the main components of the naming scheme: *authority, content* and *location*.

- Authority. Authority is any entity who, in the first place, has the direct control over the data stored and handled by the system. It can be either a content provider who generated the content itself, i.e., the content owner, or an entity appointed to act as a representative of the content owner, e.g., a proxy.
- Content. Content is any resource or piece of information that is generated by an authority, and can be stored in a location.
- Location. Location is a place in the memory, hard disk or network where a piece of content can be stored and located.

Large pieces of content can also be partitioned into smaller data chunks of fixed size to satisfy external requirements, for instance, the maximum transmission unit in the network. In addition, pieces of content can also be fragmented due to system policies, e.g., peer-to-peer networks require content to be divided and exchanged in smaller chunks to increase the overall availability of the system. Therefore, data chunks are smaller components of a piece of content and the sum of all parts forms the entire content.

2.3 Identifiers

The mapping of these concepts on the network level is represented by the *authority, content* and *location identifiers*. The benefits of using these *security tokens* in the network level are twofold: first, we decouple the *authority* from specific locations in the network, leveraging a broader concept of *authority* instead of a single administrative domain; second, we add security semantics within an identifier, easing the authority authentication in the network. As a consequence, entities are able to recognize a security token that has been confirmed (authenticated) before, regardless of the location, and are able to recreate an indicative relationship between a message and an already known *authority*.

Content identifiers must also be free of location semantics in order to allow content mobility in the network. In order to provide a strong binding between contents and their identifiers, we use cryptographic identifiers (cryptoID) [12] to identify each piece of content. These identifiers are unique and permanent, and result from a strong cryptographic hash function over a piece of content, making them to be only dependent on the content itself and providing mechanisms to self-verify the content integrity.

Large pieces of content that require fragmentation into smaller data chunks to be transferred in the network use *algorithmic IDs* (algIDs) [11] to leverage provenance with the original provider. Algorithmic IDs are a class of cryptoIDs that are algorithmically generated and provide strong binding between a content identifier and a set of chunk identifiers. Some examples of algIDs include hash chains [16] and Merkle Trees [9]. Hash chains allow for a chain of messages to be verified in sequence, as the chain anchor is usually digitally signed and the trust can be sequentially transferred from one chain to another due to the strong properties of cryptographic hash functions. In this scenario, a client first retrieves a signed hash chain anchor, which contains the hash of the first data chunk, and verifies the digital signature on it. Upon retrieving the first data chunk, it contains the hash value of the second data chunk and so on. As the first hash value (hash chain anchor) was digitally signed by the content provider, the trust on the data chunks is transferred to

the following data chunk. Any unauthorized modification in any data chunk will result in a different hash value.

Merkle Trees are another class of algIDs where a *Root Hash* is generated from the cryptographic hash over the data chunks, resulting in a strong binding between data chunks and the *root* algID. Merkle Trees allow for independent block verification because each data chunk carries its own authentication information, allowing verification of data chunks by intermediate devices, for instance, network routers. Merkle Trees can be used to detect and prevent corruption of large pieces of content, where a single corrupted data chunk will result in an unusable piece of content [14]. Another interesting property is that Merkle tree chunk identifiers are location-independent, allowing for parallel and out-of-sequence chunk retrieval [15].

3. IMPLEMENTATION

3.1 Implementation architecture

The implementation of our naming system requires two main components: a name resolver and a content manager. The name resolver can be implemented as a plug-in for a Web browser, in order to support content retrieval from multiple sources. Given a URL, the plug-in will check whether it has the certificates in its internal trusted key store to authenticate the *authority*. Otherwise, the plug-in will initiate a name scheme resolution to retrieve the *authority's* certificate together with the *content* identifier, authenticate and store them in the local keychain.

Later on, the resolver requests a name resolution to the DNS to map the *authority's* name to the metadata and the Web-server's network location. The DNS returns a type TXT record containing the metadata with the chunk IDs and a type A record containing the IP address of the server. Finally, the plug-in opens multiple HTTP connections containing the chunk request header towards the server's network identifier. Fig. 1 shows the header used in the content pieces retrieval.

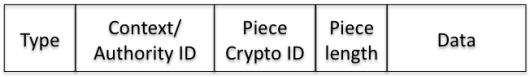

Figure 1: Content piece header. Each piece contains the context/authority identifier, its own content identifier and the piece length.

The *type* field defines the type of the message, for instance, if it is a data chunk request or a signaling message. The *context/authority ID* defines either the context of the requested data or the content provider identifier. In the first case, the *content ID* is used to identify a group that may have special access rights, e.g., a multicast group among a set of users. In the latter case, the *authority ID* is used to identify the content owner or the proxy identifier that is responsible for the requested piece of content, providing privacy for the provider. The *piece crypto ID* is the cryptographic ID of the content, protecting against unauthorized modifications in the data that the packet carries. Finally, the *piece length* contains the length of the carried data.

The proposed naming mechanism supports clean-slate architectures based on flat routing mechanism, such as DONA [8], since these architectures support routing on flat identifiers. However, for an IP network, the component will use the *server* IP address returned together with the metadata to retrieve the content. In this case, the application will open multiple HTTP connections towards the server, and routers in the path are able to parse the content header and divert the requests to them, working similarly as transparent Web-proxies. In case a cache does not have the requested piece, it forwards the request towards the server. The proposed caching mechanism is possible due to the independent piece identification mechanism proposed in the naming scheme.

3.2 Naming Scheme

We use Universal Resource Identifiers (URIs) [13] as the foundation for our naming scheme. The URIs introduce a set of names and addresses used to identify resources in the Internet, leveraging global search and retrieval of documents across different operating systems and protocols. An URI is mainly composed of three components: a *scheme*, an *authority* and a *resource path*, as illustrated in Fig. 2 (additional components are described in [13]).

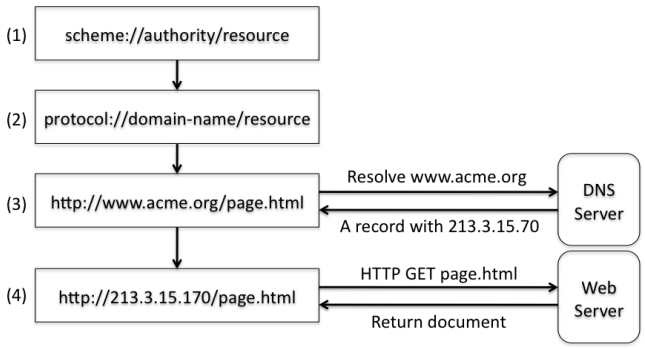

Figure 2: (1) Original URI proposal. (2) URI-to-URL mapping. (3) Example of URL resolution in the DNS. The domain name is resolved into an IP address of the server storing the document. (4) Upon having the IP address, the client goes to the server location to retrieve the resource.

The *scheme* defines the protocol handler for the *authority*; the *authority* is the owner or the entity responsible for the *resource* and the *resource path* points to the resource in the *authority* namespace. In the current naming system, a URI is actually mapped to an URL and the *authority* becomes the Fully Qualified Domain Name (FQDN) of the network where the resource is located[2]. The domain name is resolved into an IP address through the DNS and the client is able to request the *resource* to the returned IP address. However, the main drawback is the binding between content and a specific, relatively stable set of locations.

In this paper, we take the stance that the *authority* field defines the *content owner/provider*, or the entity responsible for the content, instead of defining just a namespace authority. Hence, instead of mapping an FQDN to a location-dependent identifier to an IP address with DNS, the resolution system maps the authority field to the public key (or

[2]Of course, there may be additional fields within the authority field. However, as they are not commonly used and not essential for the current discussion, we ignore them in the rest of this paper.

other secure identifier) of the authority over the data, providing a cryptographic mapping between a user-level name into cryptographic identifiers.

We map the *resource path* to a content identifier that is also location-independent, resembling the DONA [8] style of naming based on *Principals* and *Labels* (P:L). However, in our proposal we define a more general and flexible structure to describe a resource called *metadata*. The *metadata* holds all information describing the resource, such as *type*, *total length*, *number of pieces*, *piece ID list* (cryptoIDs), *piece length*, *content anchor* and *digital signature*. Fig. 3 illustrates an example of the proposed naming scheme (step 1).

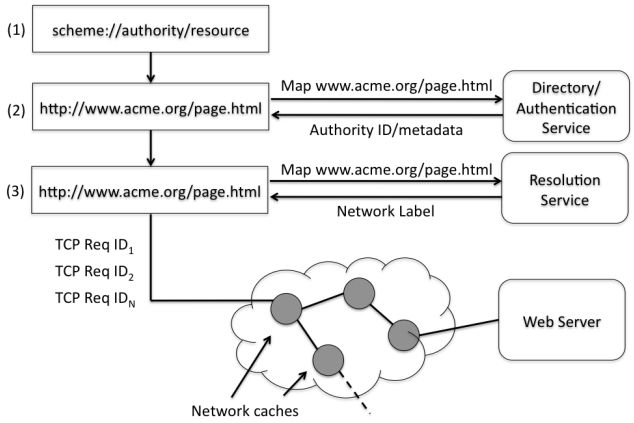

Figure 3: Naming scheme and resolution to authority and metadata.

3.3 Authority Mapping

The mapping of an authority to a security token can be performed in different ways, for example, local configuration based on administrative rules, integrated with legacy resolution system (DNS) or can be a totally external entity, such as a Distributed Hash Table (DHT).

In the first scenario, the bootstrapping procedure is done by an external administrator, so the local host has the key embedded in the system, for instance, a list of root certificates in a Web-browser. The benefit of this approach is the pre-establishment of a trust relationship between the software vendor (the entity that embedded the certificates) and the entities that may provide security services (Web-sites that are trusted).

In the second scenario, clients rely on a trusted resolution infrastructure, for instance, DNSSec. In this approach, a user queries a distributed system to reach the authoritative server hosting an authority-to-identifier mapping. The benefit of this approach is that the key management is easier from the infrastructure point of view. However, the drawback is the unnecessary trust placed in the resolution infrastructure to resolve queries correctly. For example, if a client trusts her bank (and its digital certificate), then she does not need to trust in the infrastructure (DNSSec) to resolve the mapping to the bank. The client can spot any forgery since she has the bank's identifier and can check the identity against the bank's certificate.

Finally, the third approach uses a distributed database to store the mappings, such as a DHT. The DHT works as a distributed directory where entities post and request certificates in the Internet. The benefit of this approach is that clients do not need to trust the infrastructure since it is a mere placeholder for the certificates. The trust in the infrastructure is minimum because clients just need to trust that the DHT will store and return the certificates whenever queries, and clients are able to verify whether a returned certificate is valid or not by checking its digital signature within the returned certificate. Hence, clients do not need to trust the infrastructure to return the correcting mapping, but rather to just return the content provider's digital certificate.

In our proposal, we choose to use the last approach since we need to place the minimum level of trust on the name resolution mechanism, reducing the number of possible vulnerabilities against an eventual attacker.

3.4 Naming Resolution

The third step in the secure naming system is to provide a resolution service for the *authority* and *content* identifiers into network locations. We adopt the DNS hierarchical infrastructure to provide the mapping service for the proposed naming scheme. The idea is that a distributed directory provides the mapping of the *authority* and its identifier to be used in the content authentication and the DNS system to provide the mapping between the *authority* name to its location. The benefits of this approach are threefold: first, we remove the implicit trust placed in the resolution system to a direct trust with the *authority*; second; the *authority* name to its identifier resolution is agnostic about the forwarding fabric, it can be used with many forwarding technologies; third, it provides backward compatibility with the current naming scheme and resolution system.

Fig. 3 illustrates the resolution procedure (steps 2 and 3). First, an application resolves an URI into the authority and content identifiers in a DHT (step 1). Then, the application resolves the URI into the resource metadata and server IP using the hierarchical resolution in the authority name to reach the authoritative DNS server (step 2). Finally, the application opens multiple connections requesting the data chunks identified with cryptoIDs retrieved from the metadata (step 3). The request is addressed to the server's network location and routers in the path storing the content identified by the cryptoID can return the data chunk on behalf of the server.

3.5 Ongoing work

In order to evaluate our proposal, we have an early implementation of the naming scheme mechanism, implemented as a plug-in for the Firefox Web-browser to provide backward compatibility. The plug-in is composed of two main components: a protocol handler written in JavaScript, responsible for intercepting the new name scheme calls from the Web-browser, and a XPCOM[3] component responsible for parsing the header and retrieving the data chunks.

The JavaScript component registers itself in the Mozilla Framework, thus, whenever the secure naming scheme is used, the Web-browser loads out customized protocol handler. The handler parses the URI and splits it into *authority* and *content IDs*, which are passed as a parameter to the XPCOM component. The component receives the IDs and

[3]XPCOM is a cross-platform component that has multiple language bindings, easing the deployment of a new feature as a component.

subscribes to them. Currently, our implementation works integrated with the PSIRP Blackhawk prototype [11]. As future work, we will integrate it with the DNSSec system and also integrate the parsing for the metadata structure.

4. SECURITY ANALYSIS

In this section, we analyze the proposed naming system from the external and internal security perspective. From the external point of view, we analyze the roles of the distributed directory based on the DHT and the DNS infrastructure.

The distributed directory system plays a role as digital certificate storage of authority identities, allowing clients to query for a digital certificate of a content provider. The trust placed in the directory itself is limited, since a client needs to trust that it will behave accordingly with its pre-established function, i.e., store and answer digital certificates. One possible threat is the misbehaving of the directory system, e.g., not replying to a query, resulting in lack of availability due to a malfunctioning or corrupted node. However, clients can notice such a problem and try with another directory, preventing any forgery (man-in-the-middle) attack since the trust is established with the content provider and not with the directory service.

Upon retrieving the content provider's certificate, clients are able to verify the digital signature in the certificate and check against the set of trusted keys in their key chain. We assume here that there is at least a small set of trusted keys in order to bootstrap the authentication procedure. Such set of trusted keys are installed by default in some Web-browsers and others can be added with the users approval.

The second resolution step involves the resolution of content IDs to network identifiers in the DNS. The trust level is minimal since clients establish a direct chain of trust with the content provider. Therefore, the DNS infrastructure is used as a simple mapping service and the possible attacks that can be performed is related to the denial of service to the clients. Attacks such as DNS cache poisoning does not affect clients since they have already the trust relationship established with the original provider. Therefore, if a DNS returns a modified entry pointing to a network label belonging to a malicious node, clients will be able to spot that by verifying the certificate retrieved in the previous step and check against the identity of the malicious attacker. As the attacker does not own a valid certificate for that identity, he will not be able to spoof a victim's identity.

From the internal point of view, we analyze the security aspects of the binding between content identifier, algorithmic identifier, and the carried data. Content identifiers are based on either cryptoIDs or algIDs. In the former case (cryptoID), identifiers are generated from a strong cryptographic hash function, e.g. SHA-256, binding the content identifier with the data that it carries. Therefore, it is statistically impossible for an attacker to tamper with a piece of data without modifying its identifier due to the preimage attack resistance of these cryptographic hash functions. In the latter case (algIDs), identifiers are generated through the composition of cryptographic hash functions. Similarly to the cryptoIDs, it is unfeasible for an attacker to tamper with a data chunk without modifying the algID.

5. RELATED WORK

Content Centric Network (CCNx) [7] is a receiver-driven content-oriented communication model driven by consumers' *interests*. Requests for pieces of content are identified by hierarchical names, where each name is composed of a high-level application name, concatenated together with the version, segmentation index and the cryptographic hash of the content to provide data integrity. These requests are routed directly on the names towards the server. Network caches on the path are able to identify and respond with the cached data on behalf of the server. As CCN relies on application-level identifiers to identify pieces of content in the network, content resolution and forwarding is bound to the hierarchical structure described in the URL. In our approach, we use identifiers that are simultaneously independent from the forwarding, routing and storage, allowing content to be placed in multiple locations, e.g. edge networks, to server clients.

DONA [8] proposes a clean-slate approach for the Internet naming system with a name-based anycast resolution primitive. Each piece of content is identified by its *principals* and *labels* (P:L), and data objects are handled by *find/register* primitives the architecture uses *Resolution Handlers* to resolve a name into the closest sources over the legacy IP network. However, the proposal does not address backwards compatibility with the current naming system and the content fragmentation. Our naming system leverages backward compatibility with the current naming system and uses algorithmic IDs to provide binding between content and data chunks.

NetInf [2] proposes a secure naming scheme composed of two main components, a naming scheme based on the DONA scheme (P:L) and an *information object* structure to hold the information of a piece of content. An information object contains the object identifier, the data itself and a metadata used to describe the data that an information object carries. In addition, the metadata carries all the security parameters required to verify an object, for instance, owner's public key and the content piece hashes. The metadata may not contain the owner's public key, but the public key of a proxy to preserver *anonymity* of the content publisher. Compared to our approach, Net-Inf lacks backward compatibility with the current naming system based on DNS. As they use DONA-like naming scheme, there is no mapping function to map the user-level names to the cryptographic identifiers. Moreover, the naming resolution is left as future work. In our proposal, we provide a secure mapping between human-readable names (URIs) into cryptographic identifiers and we use the legacy DNS resolution mechanism to provide the mapping of the cryptoIDs into network labels.

6. DEPLOYMENT CONSIDERATIONS

The proposed naming scheme can be gradually deployed in the current naming system. From the resolution side, it required an external infrastructure acting as a placeholder for digital certificates, which can be either a DHT, e.g. Pastry [1] or an underlay forwarding/storage mechanism, e.g. PSIRP [11]. Currently, some already deployed DHTs in the PlanetLab testbed can be used for this purpose since resolvers need to place a minimal trust in the infrastructure. The benefit of using such distributed infrastructure is the resistance to some security attacks, such as denial-of-service attacks against the storage and better resistance against node failures.

For the second resolution step, it is required to introduce

a new DNS type TXT record in the servers containing the content metadata. This record type is already supported by DNS, thus, servers that support the metadata scheme just need to insert a new type in the DNS to provide enough information about the data chunks.

The content servers should support both complete (or legacy request) and partial content retrieval, when a client issues a request with the content header. In the latter case, the server should have an internal mapping of the cryptoIDs to the data chunks in the complete file. For example, given a cryptoID, the server needs to be able to calculate which position of the complete file it should return, similar to the `HTTP Range` header. In this type of request, a client defines the byte range that she wants to receive from the file. Another possible approach is to save all data chunks in the server to prevent the mapping to the correct offset in the complete file.

In the client side, users need to install a plug-in in the Web-browsers to handle the new naming scheme and the metadata structure. The plug-in can be easily integrated with Web-browsers, providing authority and resource metadata authentication, metadata parsing and multiple connection management for data retrieval.

Finally, in-network caching feature can also be gradually deployed in the network. Some routers have built-in Web-servers that can be used as an alternate data source. On the other hand, more specialized in-network caches can also be deployed as *bumps-in-the-wire* to optimize the traffic efficiency by caching a certain amount of the in-transit traffic based on the content popularity.

7. CONCLUSION

In this paper, we have presented a secure naming system that aims at decoupling the content authentication from its location in a network. The proposed mechanism allows for content authentication using cryptographic identifiers that are independent from the routing, forwarding, and storage location. As a consequence, pieces of content are authenticated with the original provider or authority and the data can be retrieved from any location, for instance, a proxy or a network cache on the path. The naming system uses a distributed directory service to store the authorities' digital certificates, allowing for the direct establishment of the trust relations between clients and content providers. Some benefits of the proposed naming system include the security mapping function between high-level names and cryptographic identifiers in the network level, content authentication with the provider regardless of the network location where it was retrieved, and migration of the trust from the resolution infrastructure to the provider, reducing the number of possible threats during the resolution procedure.

8. ACKNOWLEDGMENTS

Walter Wong is funded by the Brazilian CAPES agency and Ericsson Research. Wes also would like to thanks Teemu Koponen for the comments on the paper.

9. REFERENCES

[1] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. *ACM International Conference on Distributed Systems Platforms (Middleware)* (November 2001), 329–350.

[2] C. Dannewitz, J. Golic, B. Ohlman, B. Ahlgren. Secure Naming for a Network of Information. *INFOCOM IEEE Conference on Computer Communications Workshops* (March 2010), 1–6.

[3] Dagon, D., Antonakakis, M., Vixie, P., Jinmei, T., and Lee, W. Increased dns forgery resistance through 0x20-bit encoding: security via leet queries. In *15th ACM conference on Computer and communications security* (2008), ACM, pp. 211–222.

[4] Decoupled Application Data Enroute (DECADE). URL http://datatracker.ietf.org/wg/decade/.

[5] Friedlander, A., Mankin, A., Maughan, W. D., and Crocker, S. D. Dnssec: a protocol toward securing the internet infrastructure. *Commun. ACM 50*, 6 (2007), 44–50.

[6] J. Scott, P. Hui, J. Crowcroft and C. Diot. Haggle: A Networking Architecture Degisned around Mobile Users. *3rd Wireless On-demand Network Systems and Services (WONS)* (2006).

[7] Jacobson, V., Smetters, D. K., Thornton, J. D., Plass, M. F., Briggs, N. H., and Braynard, R. L. Networking named content. In *CoNEXT '09: Proceedings of the 5th international conference on Emerging networking experiments and technologies* (New York, NY, USA, 2009), ACM, pp. 1–12.

[8] Koponen, T., Chawla, M., Chun, B.-G., Ermolinskiy, A., Kim, K. H., Shenker, S., and Stoica, I. A data-oriented (and beyond) network architecture. *SIGCOMM Comput. Commun. Rev. 37*, 4 (2007), 181–192.

[9] Merkle, R. C. A certified digital signature. In *CRYPTO '89: Proceedings on Advances in cryptology* (New York, NY, USA, 1989), Springer-Verlag New York, Inc., pp. 218–238.

[10] Pallis, G., and Vakali, A. Insight and perspectives for content delivery networks. *Commun. ACM 49*, 1 (2006), 101–106.

[11] Publish/Subscribe Internet Routing Paradigm. Conceptual architecture of psirp including subcomponent descriptions. Deliverable d2.2, PSIRP project. (August 2008).

[12] R. Moskowitz, P. Nikander, P. Jokela, T. Henderson. RFC 5201: Host Identity Protocol, April 2008.

[13] T. Berners-Lee. RFC1630: Universal Resource Identifiers in WWW, June 1994.

[14] W. Wong, M. Magalhaes and J. Kangasharju. Piece Fingerprinting: Binding Content and Data Blocks Together in Peer-to-peer Networks. *IEEE Global Communications Conference (Globecom'10), Miami, Florida, USA* (December 6-10 2010).

[15] W. Wong, M. Magalhaes and J. Kangasharju. Towards Verifiable Parallel Content Retrieval. *6th Workshop on Secure Network Protocols (NPSec'10), Kyoto, Japan* (October 2010).

[16] Yih-Chun Hu, M. Jakobsson and A. Perrig. Efficient Constructions for One-Way Hash Chains. *Applied Cryptography and Network Security* (May 2005), 423–441.