Individual Submission	C. Dannewitz
Internet-Draft	University of Paderborn
Intended status: Informational	T. Rautio
Expires: September 15, 2011	VTT Technical Research Centre of Finland
	O. Strandberg
	Nokia Siemens Networks
	B. Ohlman
	Ericsson
	March 14, 2011

Secure naming structure and p2p application interaction
draft-dannewitz-ppsp-secure-naming-02

Abstract

Today, each application typically uses its own way to identify data. The lack of a common naming scheme prevents applications from benefiting from available copies of the same data distributed via different P2P and CDN systems. The main proposal presented in this draft is idea that there should be a secure and application independent way of naming information objects that are transported over the Internet. The draft defines a set of requirements for such a naming structure. It also presents a proposal for such a naming structure that could relevant for a number of work groups (existing and potential), e.g. PPSP, DECADE and CDNI. In addition, today's P2P naming schemes lack important security aspects that would allow the user to check the data integrity and build trust in data and data publishers. This is especially important in P2P applications as data is received from untrusted peers. Providing a generic naming scheme for P2P systems so that multiple P2P systems can use the same data regardless of data location and P2P system increases the efficiency and data availability of the overall data dissemination process. The secure naming scheme is providing self-certification such that the receiver can verify the data integrity, i.e., that the correct data has been received, without requiring a trusted third party. It also enables owner authentication to build up trust in (potentially anonymous) data publishers. The secure naming structure should be beneficial as potential design principle in defining the two protocols identified as objectives in the PPSP charter. This document enumerates a number of design considerations to impact the design and implementation of the tracker-peer signaling and peer-peer streaming signaling protocols.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

Today's dominating naming schemes in the Internet, i.e., IP addresses and URLs, are rather host-centric with respect to the fact that they are bound to a location. This kind of naming scheme is not optimal for many of the predominant users of todays Internet like P2P and CDN systems as they are based on an information-centric thinking, i.e., putting the information itself in focus. In these system the source of the information is secondary and can constantly change, e.g. new caches or P2P peers becomes available. It also common to retrieve information from more than one source at once.
For any type of caching solution (network based or P2P) and network based storage, e.g. DECADE, a common application independent naming scheme is essential to be able to identify cached copies of information/data objects.
Many applications, in particular P2P applications, use their own data model and protocol for keeping track of data and locations. This poses a challenge for use of the same information for several applications. A common naming scheme for information objects is important to enable interconnectivity between different application systems, such as P2P and CDN. To be able to build a common P2P infrastructure that can serve a multitude of applications there is a need for a common application independent naming scheme. With such a naming scheme different applications can use and refer to the same information/data objects.
It is possible to introduce false data into P2P systems, only detectable when the content is played out in the user application. The false data copies can be identified and sorted out if the P2P system can verify the reference used in the tracker protocol towards data received at the peer. One option to address this can be to secure the

naming structure i.e. make the data reference be dependent on the data and related metadata.
An additional reason to introduce a common naming scheme for information objects is caching. When data are named in a host-centric way, as is done today, it is not always identify that copies of the same information object are available in multiple hosts. With location independent identifiers for information objects this becomes much easier.
This document enumerates and explains the rationale for why a common naming structure for information/data objects should be defined and used by a wide range of applications and network protocols. Examples of WGs (and potential WGs) where we think a new standard for naming of information/data objects should be valuable includes PPS, DECADE and CDNI. For P2P systems the main advantage is probably in the definition of a protocol for signaling and control between trackers and peers (the PPSP "tracker protocol") but also a signaling and control protocol for communication among the peers (the PPSP "peer protocol") might have benefits from a common and secure naming scheme. In DECADE one key feature would be that different applications can easily share the same cache entries. It should also be valuable for cooperative caching, e.g. CDNI.

2. Naming requirements

In the following, we discuss the requirements that a common naming scheme has to fulfill.
To enable efficient, large scale data dissemination that can make use of any available data copy, identifiers (IDs) have to be location-independent. Thereby, identical data can be identified by the same ID independently of its storage location and improved data dissemination can then benefit from all available copies. This should be possible without compromising trust in data regardless of its network source.
Security in an information-centric network (including P2P, caching, and network-based solutions) needs to be implemented differently than in host-centric networks. In the latter, most security mechanisms are based on host authentication and then trusting the data that the host delivers. In e.g. a P2P system, host authentication cannot be relied upon, or one of the main advantages of a P2P system, i.e., benefiting from any available copy, is defeated. Host authentication of a random, untrusted host that happens to have a copy does not establish the needed trust. Instead, the security has to be directly attached to the data which can be done via the scheme used to name the data.
Therefore, self-certification is a main requirement for the naming scheme. Self-certification ensures the integrity of data and securely binds this data to its ID. More precisely, this property means that any unauthorized change of data with a given ID is detectable without requiring a third party for verification. Beforehand, secure retrieval of IDs (e.g., via search, embedded in a Web page as link, etc.) is required to ensure that the user has the right ID in the first place.

Secure ID retrieval can be achieved by using recommendations, past experience, and specialized ID authentication services and mechanisms that are out of the scope of this discussion.
Another important requirement is name persistence, not only with respect to storage location changes as discussed above, but also with respect to changes of owner and/or owner's organizational structure, and content changes producing a new version of the information. Information should always be identifiable with the same ID as long as it remains essentially equivalent. Spreading of persistent naming schemes like the Digital Object Identifier (DOI) [Paskin2010] also emphasizes the need for a persistent naming scheme. However, name persistence and self-certification are partly contradictory and achieving both simultaneously for dynamic content is not trivial. From a user's perspective, persistent IDs ensure that links and bookmarks remain valid as long as the respective information exists somewhere in the network, reducing today's problem of "404 - file not found" errors triggered by renamed or moved content. From a content provider's perspective, name persistence simplifies data management as content can, e.g., be moved between folders and different servers as desired. Name persistence with respect to content changes makes it possible to identify different versions of the same information by the same consistent ID. If it is important to differentiate between multiple versions, a dedicated versioning mechanism is required, and version numbers may be included as a special part of the ID.
The requirement of building trust in an information-centric system combined with the desire for anonymous publication as well as accountability (at least for some content) can be translated into two related naming requirements. The first is owner authentication, where the owner is recognized as the same entity, which repeatedly acts as the object owner, but may remain anonymous. The second is owner identification, where the owner is also identified by a physically verifiable identifier, such as a personal name. This separation is important to allow for anonymous publication of content, e.g., to support free speech, while at the same time building up trust in a (potentially anonymous) owner.
In general, the naming scheme should be able to adapt to future needs. Therefore, the naming scheme should be extensible, i.e., it should be able to add new information (e.g., a chunk number for BitTorrent-like protocols) to the naming scheme. The need for such extensions is stressed by today's variety of naming schemes (e.g., DOI or PermaLink) added on top of the original Internet architecture that fulfill specialized needs which cannot be met by the common Internet naming schemes, i.e., IP addresses and URLs.

3. Basic Concepts for an Application-independent Naming Scheme

In this section, we introduce an exemplary naming scheme that illustrates a possible way to fulfill the requirements posed upon an application-independent naming scheme for information-centric networks.

The naming scheme integrates security deeply into the system architecture. Trust is based on the data's ID in combination with additional security metadata. Section 3.1 gives an overview of the naming scheme in general with details about the ID structure, and Section 3.2 describes the security metadata in more detail.

3.1. Overview

Building on an identifier/locator split, each data element, e.g., file, is given a unique ID with cryptographic properties. Together with the additional security metadata, the ID can be used to verify data integrity, owner authentication, and owner identification. The security metadata contains information needed for the security functions of the naming scheme, e.g., public keys, content hashes, certificates, and a data signature authenticating the content. In comparison with the security model in today's host-centric networks, this approach minimizes the need for trust in the infrastructure, especially in the host(s) providing the data.

In an information-centric network, multiple copies of the same data element typically exist at different locations. Thanks to the ID/locator split and the application-independent naming scheme, those identical copies have the same ID and, hence, each application can benefit from all available copies.

Data elements are manipulated (e.g., generated, modified, registered, and retrieved) by physical entities such as nodes (clients or hosts), persons, and companies. Physical entities able of generating, i.e., creating or modifying data elements are called owners here. Several security properties of this naming scheme are based on the fact that each ID contains the hash of a public key that is part of a public/secret key pair PK/SK. This PK/SK pair is conceptually bound to the data element itself and not directly to the owner as in other systems like DONA [Koponen]. If desired, the PK/SK pair can be bound to the owner only indirectly, via a certificate chain. This is important to note because it enables owner change while keeping persistent IDs. The key pair bound to the data is thus denoted as PK_D/SK_D.

Making the (hash of the) public key part of ID enables self-certification of dynamic content while keeping persistent IDs. Self-certification of static content can be achieved by simply including the hash of content in the ID, but this would obviously result in non-persistent IDs for dynamic content. For dynamic content, the public key in the ID can be used to securely bind the hash of content to the ID, by signing it with the corresponding secret key, while not making it part of ID.

The owner's PK as part of the ID inherently provides owner authentication. If the public key is bound to the owner's identity (i.e., to its real-world name) via a trusted third party certificate, this also allows owner identification. Without this additional certificate, the owner can remain anonymous.

To support the potentially diverse requirements of certain groups of applications and adapt to future changes, the naming scheme can enable flexibility and extensibility by supporting different name structures, differentiated via a Type field in the ID.

3.2. ID Structure

The naming scheme uses flat IDs to support self-certification and name persistence. In addition, flat IDs are advantageous when it comes to mobility and they can be allocated without an administrative authority by relying on statistical uniqueness in a large namespace, with the rare case of ID collisions being handled by the applications. Although IDs are not hierarchical, they have a specified basic ID structure. The ID structure given as ID = (Type field | A = hash(PK) | L) is described subsequently.
The Authenticator field A=Hash(PK_D) binds the ID to a public key PK_D. The hash function Hash is a cryptographic hash function, which is required to be one-way and collision-resistant. The hash function serves only to reduce the bit length of PK_D. PK_D is generated in accordance with a chosen public-key cryptosystem. The corresponding secret key SK_D should only be known to a legitimate owner. In consequence, an owner of the data is defined as any entity who (legitimately) knows SK_D.
The pair (A, L) has to be globally unique. Hence, the Label field L provides global uniqueness if PK_D is repeatedly used for different data.
To build a flexible and extensible naming scheme, e.g., to adapt the naming scheme to future changes, different types of IDs are supported by the naming scheme and differentiated via a mandatory and globally standardized Type field in each ID. For example, the Type field specifies the hash functions used to generate the ID. If a used hash function becomes insecure, the Type field can be exploited by the P2P system in order to automatically mark the IDs using this hash function as invalid.

3.3. Security Metadata Structure

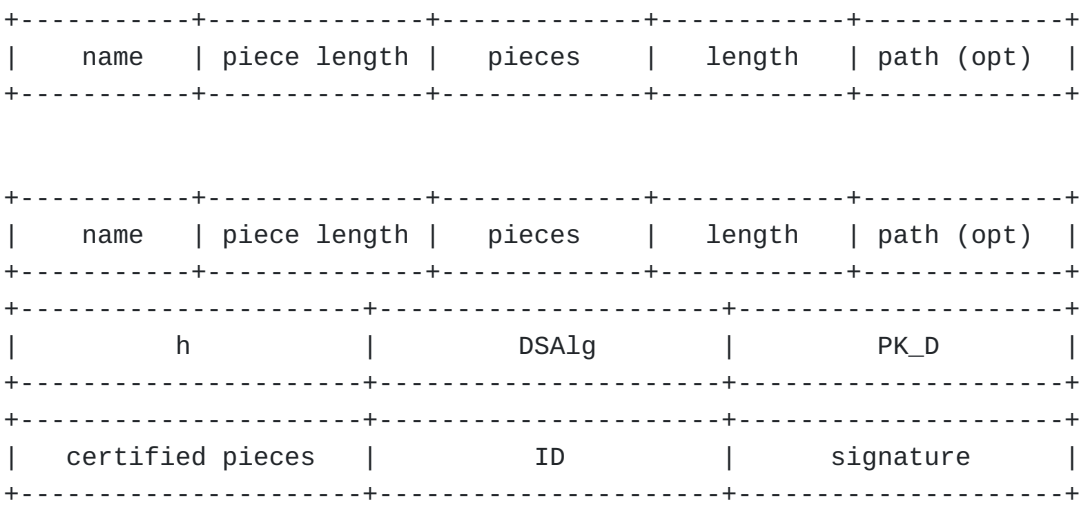
The security metadata is extensible and contains all information required to perform the security functions embedded in the naming scheme. The metadata (or selected parts of it) will be signed by SK_D corresponding to PK_D. This securely binds the metadata to the ID, i.e., to the Hash(PK_D) which is part of the ID. For example, the security metadata may include:
A detailed description and security analysis of this naming scheme and its security properties, especially self-certification, name persistence, owner authentication, and owner identification can be found in the GIS paper Secure Naming for a Network of Information [Dannewitz_10].

4. Examples of application use of secure naming structure

This section contains a number of examples how a secure naming system, as outlined in this draft, could be used by different types of applications.

4.1. Secure naming for P2P applications

From an P2P application perspective the main advantage of a secure naming structure for a P2P infrastructure is that multiple P2P applications can have common access to the same data elements. Another benefit of application-independent naming is that locally available and cached copies can easily be located. The secure naming also enables that data can be verified even if it is received from an untrusted host.
For example, when an application like BitTorrent [WWWbittorrent] uses self-certifying names, the user is guaranteed that the data received is actually the data that has been requested, without having to trust any servers in the network (e.g., the tracker) or the peers that provide the data.
This means that BitTorrent's validation of the data integrity can be improved significantly using the presented secure naming structure. Currently, a standard BitTorrent system has no means to verify the integrity of the torrent file and consequently of the data. The torrent file (see Figure 1) contains the SHA1 hashes of the content pieces (pieces in Figure 2). However, anyone can modify a torrent file to bind different content to this file. If the torrent file gets modified, the user has no means any more to verify the integrity of the data.
Modification of the torrent affects only to info_hash value, which is SHA1 hash calculated from the torrent's info field (see figure). The info_hash is respectively used for torrent session identification in different software entities (e.g. in trackers). After changes in the torrent's info field, the torrent is referring to different torrent session that is carrying a forged content. Additionally if, the tracker allows insertion of several torrents with the same name - delivers forged data (consistent with the forged torrent file), a user could effectively be tricked into downloading forged content which would falsely be identified as being correct by the BitTorrent client. On the other hand, the torrent referring to a forged content can be also modified to point to, another, "convenient" tracker by modifying the announce field in the torrent, and the outcome would be the same from user perspective. I.e., in the current BitTorrent system, a user has no guarantee that the downloaded content actually matches the expected/ correct content.



The secure naming structure presented in this draft can provide a simple solution for this problem by securely binding the content of the torrent file to the name/ID of the torrent file. This can be done by extending the torrent file to include the above described security metadata information, as it is seen in Figure 3. In practice, during the torrent file creation, an object owner would store information about utilized algorithms (h - hash function and DSAlg - digital signature algorithm), the public key (PK_D), specification of signed data and ID into the torrent's info field, and will sign the combination of the secure metadata and the piece hash values (pieces in the torrent's info field) with the private key (SK_D). The generated signature will also be included in the extension part of the info field (signature).

After the content of the extended torrent is created, the respective torrent file ID would be generated according to the rules described in Section 3. As defined in that section, ID contains three different fields, namely Type, A and L. In the case of BitTorrent, Type field would carry on information about used hash function to generate field A from PK_D, and also structure of the field L. If, for example, L has name and version of the distributed file, Type field should tell that by including strings "Name" and "Version" in it. The next one, field A, includes hash values of the used PK_D (method defined in Type). And finally the proposed BitTorrents ID field L, can take in name and version of the distributed file. According to the description and by using separators - (within one field) and _ (between fields) the torrent file name could look, for example, like: HashMethod-Name-Version_HashofPK_Filename-Fileversion.torrent.

Consequently, whenever a user knows the ID of the content/torrent file and retrieves the torrent file, she/he can now open the torrent with the secure naming supported BitTorrent client. The client verifies the integrity of the torrent file by comparing PK_D in secure metadata and field A in the ID, in addition, conformance of ID in the torrent name and ID in the metadata is verified. With respect to the secure metadata

the signature and actual data is compared also. Once these three are verified, the client can download the data pieces, and can use the BitTorrent's included (and now secured) hash(es) to verify the integrity of the received data. As a result, the user can be sure that the correct content was retrieved.

4.2. Secure naming use in DECADE

DECADE WG is specifying requirements for a protocol concerning accessing data in a network storage and resource control of said data. A key aspect in accessing data in a network storage is the way the data is referenced. This naming draft has outlined a naming structure that can be utilized to enhance the reference to include additional features. The secure naming structure tries to fulfill several design targets and requirements, however not all are necessarily the priority requirements for the DECADE scope.
The DECADE storage is used by individual and uncoordinated entities, thus the naming of the data must be collision free. Also when a user accesses data the name should point to the correct data. With no entity to keep track of used names for data, one potential approach is to use large enough identifier designed with statistically collision free random properties. One obvious identifier alternative is to base it on the hash of the content.
The basic requirement for naming in DECADE is that the data identifier is tied to the hash of the content and that it is taken from a large enough flat namespace. In this way, wherever the same data is stored, the same name identifier can be used. Someone accessing the data can verify that the content is correct based on relationship between data and identifier. Other requirements can be included to further enhance the meaning and capability of the data reference identifier. Additional naming requirements could be:

- self-certified name for verifying content owner (owner of Pk/Sk keys), the self-certification can be used for building trust about data publisher

- solution for persistent identifier names for dynamic (changing) data

- potentially way to identify content owner, this typically requires trusted third party certifier.

This draft specifies several requirements that would be useful for the DECADE protocol, the main requirement is hashing of data into the identifier name. Depending of data use (like enhanced security properties) other secondary requirements will be beneficial for additional functionality.

4.3. Secure naming for CDNs

The use of common naming within a CDN is not the challenge, it is the common naming between end users and the CDN or between CDNs that can be feasible. A common naming enables use of numerous caching points and CDNs as the same data can be referenced in the same way. The same data can depending of popularity be available in multiple location like caches and CDNs. The population of the resources (caches and CDNs) can be efficient if a common naming of data is used. The interaction between CDNs to 'negotiate' data population of caching resources would benefit from a common data reference model. The security features of the naming scheme also helps the CDN provider trust the data it is accessing and providing. The CDN interaction is potentially in the scope of the CDNI WG BOF.

5. Conclusion

The main proposal presented in this draft is idea that there should be a secure and application independent way of naming information objects that are transported over the Internet. The draft defines a set of requirements for such a naming structure. It also presents a proposal for such a naming structure that could relevant for a number of work groups (existing and potential), e.g. PPSP, DECADE and CDNI.
Specifically for the PPSP WG the secure naming structure is proposed for consideration as common reference ID structure. For any P2P streaming application to have fair and multitude of data access, it is essential to have a common naming structure that is suitable for many different needs. The common naming is probably best displayed in the tracker protocol case but potential benefit in the actual streaming protocol case has to still be identified. The secure binding of reference ID to the actual content is manifested in the end user peer possibility to check correct data reception in regard to the used ID.
The naming structure has been implemented in the 4WARD project prototypes and has been released as open source (www.netinf.org). The naming structure is also available through a public NetInf registration service at www.netinf.org. Three NetInf-enabled applications have also been published, the InFox (Firefox plugin), InBird (Thunderbird plugin), and a NetInf Information Object Management Tool, all available at the www.netinf.org site.
Continued work on defining a common naming structure for information objects is carried out in the SAIL project. More information is available at the www.sail-project.eu site.

6. IANA Considerations

This document has no requests to IANA.

7. Security Considerations

There are considerations about what private/public key and hash algorithms to utilize when designing the naming structure in a secure way.

8. Acknowledgements

We would like to thank all the persons participating in the Network of Information work packages in the EU FP7 projects 4WARD and SAIL and the Finnish ICT SHOK Future Internet 2 project for contributions and feedback to this document.

9. References

[RFC2119]	Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
[Dannewitz_10]	Dannewitz, C., Golic, J., Ohlman, B. and B. Ahlgren, "Secure Naming for a Network of Information", 13th IEEE Global Internet Symposium , 2010.
[Paskin2010]	Paskin, N., "Digital Object Identifier ({DOI}?) System", Encyclopedia of Library and Information Sciences , 2010.
[Koponen]	Koponen, T., Chawla, M., Chun, B.-G., Ermolinskiy, A., Kim, K. H., Shenker, S. and I. Stoica, "A Data-Oriented (and beyond) Network Architecture", Proc. ACM SIGCOMM , 2007.
[WWWbittorrent]	Cohen, B., "The BitTorrent Protocol Specification", http://www.bittorrent.org/beps/bep_0003.html , 2008.

Authors' Addresses

Christian Dannewitz Dannewitz University of Paderborn Paderborn, Germany EMail: cdannewitz@upb.de

Teemu Rautio Rautio VTT Technical Research Centre of Finland Oulu, Finland EMail: teemu.rautio@vtt.fi

Ove Strandberg Strandberg Nokia Siemens Networks Espoo, Finland EMail: ove.strandberg@nsn.com

Borje Ohlman Ohlman Ericsson Stockholm, Sweden EMail: Borje.Ohlman@ericsson.com