Internet Engineering Task Force   P. Hallam-Baker
Internet-Draft   Comodo Group Inc.
Intended status: Standards Track   R. Stradling
Expires: October 13, 2012   Comodo CA Ltd.
                                                            S. Farrell
                                                   Trinity College Dublin
                                                           D. Kutscher
                                                                   NEC
                                                            B. Ohlman
                                                              Ericsson
                                                        April 11, 2012

The Named Information (ni) URI Scheme: Parameters
draft-hallambaker-decade-ni-params-01

Abstract

   This document specifies an additional hash algorithm and parameters
   that may be used in the query string of ni URIs.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

The ni URI scheme [nischeme] supports extensibility in terms of the
algorithm used to derive a value (normally, but not always a strong
digest algorithm) and via support for a query-string thay can contain
a list of key=value pairs. This document defines some uses for both
of these extensibility points and creates IANA registries that can be
used to register additional algorithms and key strings for use in the
query part of an ni name.

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 RFC 2119 [RFC2119].

[[Comments are included in double-square brackets, like this.]]

[[Note that the features here are less mature than the specification
in the [nischeme] document. The intent is to develop these as
required for the various use-cases as we go. If something from here
appears to be as widely useful as the core ni scheme, then the
authors are willing to move features from this document to the core
document. We are also happy to incoroporate features to handle
additional use-cases here if those arise.]]

2. Additional Algorithm

This section specifies an additional algorithm that MAY be used to
handle hashes calculated over dynamically changing objects.

2.1. Hashed Dynamic Content

The ni scheme involves calculating digest values over content
objects. That works fine with static objects but is problematic for
objects whose value is dynamically generated. In this section we
define an algorithm that supports the same core "name-data integrity"
service for dynamic objects. The basic idea is simply to include a
hash of a public key in the ni name, and then for the dynamic object
to be digitally signed with the corresponding private key. With a
little work to handle the various useful formats, this allows a
client that is presented with the ni name and the signed object to
verify the binding between the name and the object data.

Note that the signature scheme used might or might not provide
additional information, e.g. a name for the signer. Applications
might benefit from that, but it is not required in order to provide
the core "name-data integrity" service for dynamically generated
objects.

Since there are a number of digital signing schemes that might be
used, our approach is to define a new algorithm for the ni scheme
that takes as input a specific public key encoded in a specific way,
and runs that through a digest function. That is, the ni algorithm
fields will specify both a public key algorithm and a digest
algorithm, just as is done with digital signature algorithm
identifiers.

Since it is possible that an ni algorithm might also be defined where
the value contains an actual digital signature we need to be careful
to ensure there is no ambiguity. However, since the lengths of
signatures and hash outputs are (with current algorithms) always
different, we could use that fact to disambigute between rsa-with-
sha256 meaning the value is a sha256 hash of an rsa public key and
the alternative meaning the the value is an rsa-with-sha256
signature. However, we prefer to use a new algorithm (see Section 5
to disambiguate these.

We define one such algorithm, "pk-rsa-with-sha256" that takes an RSA
public key as input, with the input bits formatted as a
SubjectPublicKeyInfo as defined by [nischeme] Note that this does not
mean that one cannot use e.g. PGP to sign the actual object. It
means that if you do use PGP then in order to verify the name-data
integrity, the client needs to extract the signer's PGP public key,
then reformat that as a SubjectPublicKeyInfo and then run that
through the sha-256 algorithm and make the relevant comparison.

3. **Query String Paramaeters**

This section defines query string parameters that MAY be used to
indicate the type of content hashed or to specify additional
locations from which the named content can be retrieved. We also
define a way to specify how an encryption key MAY be included in an
ni URI that allows for decryption of object content.

3.1. **Digest with Content Type**

The semantics of a digest being used to establish a secure reference
from an authenticated source to an external source may be a function
of associated meta data such as the content type. This data MAY be
specified by means of a parameter:

ni:sha-256;B_K97zTtFuOhug27fke4_Zgc4Myz4b_lZNgsQjy6fkc?ct=text/plain

The Content Type "ct" parameter specifies the MIME Content Type of
the associated data as defined in [RFC4288]

3.2. Additional Locators

   In addition to the algorithm for mapping an ni URI to an HTTP(S) URL
   specified in the ni scheme definition [nischeme], an ni name MAY
   provide information about additional locations from which the
   referenced content might be available. This is done via the
   inclusion of an "alt" or "alts" key in the query string that can
   supply more values for the authority field when constructing the HTTP
   or HTTPS URL. For example:

   ni://example.com/
   sha-
   256;B_K97zTtFuOhug27fke4_Zgc4Myz4b_lZNgsQjy6fkc?alt=ni.example.net

   The corresponding content might then also be retrieved from the URL:

   http://ni.example.net/.well-known/ni/sha-256/
   B_K97zTtFuOhug27fke4_Zgc4Myz4b_lZNgsQjy6fkc

   A ni name MAY specify multiple locations from which the content MAY
   be obtained:

   ni:///
   sha-
   256;B_K97zTtFuOhug27fke4_Zgc4Myz4b_lZNgsQjy6fkc?alt=one.example.com&
   alt=two.example.com

   The above example asserts that the content might be retrieved from
   either of the following URIs:

   http://one.example.com/.well-known/ni/sha-256/
   B_K97zTtFuOhug27fke4_Zgc4Myz4b_lZNgsQjy6fkc

   http://two.example.com/.well-known/ni/sha-256/
   B_K97zTtFuOhug27fke4_Zgc4Myz4b_lZNgsQjy6fkc

   The "alt" parameter means "use HTTP" and the "alts" parameter means
   use "HTTPS".

   The alt and alts parameters are used to specify a possible means of
   resolving the referenced content. Multiple locator parameters MAY be
   used to specify alternative sources for accessing the content.

   The alt and alts parameters take a single argument, the authority to
   be used for resolution. To permit the use of ni URIs in ASCII-only
   environments, the ASCII encoding (aka punycode) of the domain name
   MUST be used. [[Note sure if this is needed/correct.]]

3.3. Digest with Decryption Key

   An ni name MAY provide a key for decrypting the referenced data. The
   use-case here is where the referenced data has be distributed
   (somehow) in ciphertext form, probably with little or no access
   control required (since the data is strongly encrypted) and where a
   client wishing to decrypt that data subsequently acquires an ni name
   for that data that provides the required decryption key.

   Clearly, to be of any benefit, access to the ni name that includes
   the decryption key MUST be controlled so that only the appropriate
   clients get access to the ni name and of course this ni name MUST be
   strongly protected via some (probably mutual) authentication and
   confidentiality service such as can be provided by TLS. [RFC5246]

   ni///:sha-256;B_K97zTtFuOhug27fke4_Zgc4Myz4b_lZNgsQjy6fkc?enc=aes-
   cbc:Fw3x20nEKfq6FDGzq7ttIQ

   The "enc" specifier is used when the encrypted object consists of the
   ciphertext alone. The "menc" spcifier is used when the encrypted
   object consists of a MIME header containing metadata followed by the
   binary object encoding. [[Note: there may be more needed here.]]

   The encryption specifiers both take an agrument of the form:

   algorithm ":" base64url (key) [":" base64url (iv)]

   Where

   algorithm Is the algorithm used to encrypt the associated content

   key Is the value of the cryptographic key

   iv (optional) Is the value of the cryptographic Initialization
      Vector.

      If the IV is not spcified for a block cipher mode that requires
      one, the IV MUST be prepended to the encrypted content.

      [[Note: Actually the IV does not provide any additional security
      for this application but explaining the reason might be more
      effort than it is worth and what we really care about is saving
      bytes in the identifier, not the resulting data package.]

3.4. **Wrapped URL**

   The "ni" URI form can usefully "wrap" an HTTP URL in order to provide
   a way for an HTTP client that has securely received an "ni" URI (e.g.
   embedded within some HTML received via a TLS session) to validate the
   referred-to content, at the same level of security as the embedding
   page. A good use for this might be to provide data integrity for
   javascript or other files referred to by an HTML page.

   To achieve the above, we define the "url" parameter which allows for
   the inclusion of any URL within the query string. The intent is that
   the content accessed via that URL match the hash in the ni name.

   [[TBD: say how to encode the URL]]

4. **Security Considerations**

   [[TBD for sure.]]

5. **IANA Considerations**

5.1. **Additional algorithm in RFCXXXX registry**

   We request IANA to add a new entry to the hash algorithm registry
   created in [nischeme], Section 9.3, as follows:

   ID: 7
   Hash string name: pk-rsa-with-sha256
   Value length: 256
   Reference: [RFC-THIS]

5.2. **Creation of ni parameter registry**

   This specification creates a new IANA registry entitled "Named
   Information URI Parameter Definitions".

   The policy for future assignments to the registry is "RFC Required".

   The initial contents of the registry are:

```
   Parameter     Meaning                                     Reference
   -----------   ---------------------------------------     ---------
   ct            Content Type                                [RFC-THIS]
   alt           Additional HTTP Locator                     [RFC-THIS]
   alts          Additional HTTPS Locator                    [RFC-THIS]
   enc           Encryption Key                              [RFC-THIS]
   menc          Encryption Key                              [RFC-THIS]
   url           Wrapped URL                                 [RFC-THIS]
```

6. Normative References

   [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
              Requirement Levels", BCP 14, RFC 2119, March 1997.

   [RFC4288] Freed, N. and J. Klensin, "Media Type Specifications and
              Registration Procedures", BCP 13, RFC 4288, December 2005.

   [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security
              (TLS) Protocol Version 1.2", RFC 5246, August 2008.

   [nischeme]
              Farrell, S., Kutscher, D., Ohlman, B., Keranen, A., and P.
              Hallam-Baker, "Naming things with hashes",
              draft-farrell-decade-ni-03 (work in progress), April 2012.

Authors' Addresses

   Phillip Hallam-Baker
   Comodo Group Inc.

   Email: philliph@comodo.com

   Rob Stradling
   Comodo CA Ltd.

   Email: rob.stradling@comodo.com

Stephen Farrell
Trinity College Dublin
Dublin,   2
Ireland

Phone: +353-1-896-2354
Email: stephen.farrell@cs.tcd.ie

Dirk Kutscher
NEC
Kurfuersten-Anlage 36
Heidelberg,
Germany

Phone:
Email: kutscher@neclab.eu

Boerje Ohlman
Ericsson
Stockholm S-16480
Sweden

Email: Borje.Ohlman@ericsson.com