ndn-cxx - Feature #4294

ndnsec: Extend key-gen command line to allow selection of different KeyId types

09/20/2017 08:05 AM - Alex Afanasyev

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Zhiyi Zhang	**% Done:**	100%
Category:	Security	**Estimated time:**	0.00 hour
Target version:	v0.7		

Description

Right now, it is hard-coded to use the default option (random number). We have option for timestamp, digest, and user-specified component.

History

#1 - 01/05/2018 01:47 PM - Zhiyi Zhang

- Status changed from New to Code review

- % Done changed from 0 to 80

#2 - 01/07/2018 11:30 AM - Junxiao Shi

- Tracker changed from Task to Feature

I notice that https://gerrit.named-data.net/4427 patchset3 interprets user-specified KeyId with name::Component::Component(const char*) constructor that takes the string as-is. I think it is better to interpret the KeyId string as URI component with name::Component::fromEscapedString.

As specified in certificate-format spec, KeyId is an opaque name component. It is not restricted to printable characters.
Therefore, interpreting the command line argument as URI makes it easier to specify a non-printable KeyId.

#3 - 01/08/2018 12:32 PM - Junxiao Shi

Change 4427,7 test log:

```
ubuntu@m0213:~/ndn-cxx$ ndnsec key-gen /A --key_id ...
ndnsec: ../src/security/key-params.cpp:43: ndn::KeyParams::KeyParams(ndn::KeyType, const ndn::name::Component&
): Assertion `!keyId.empty()' failed.
Aborted (core dumped)
```

It's okay to reject zero-octet name component as KeyId, but it should be a graceful failure instead of hitting an assertion.

```
ubuntu@m0213:~/ndn-cxx$ ndnsec key-gen /A --key_id DD
Bv0ClwcdCAFBCANLRVkIAkRECARzZWxmCAn9AAABYNdvfRUUCRgBAhkEADbugBX9
ASYwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDMgsudIVrNx0+TUmF8
(omitted)
ubuntu@m0213:~/ndn-cxx$ ndnsec list -c
* /A
  +->* /A/KEY/DD
       +->* /A/KEY/DD/self/%FD%00%00%01%60%D7o%7D%15
(omitted)
ubuntu@m0213:~/ndn-cxx$ ndnsec key-gen /A --key_id DD
Error: Key `/A/KEY/DD` already exists
```

OK.

```
ubuntu@m0213:~/ndn-cxx$ ndnsec key-gen /A --key_id %00
Bv0ClQccCAFBCANLRVkIAQAIBHNlbGYICf0AAAFg13QmAxQJGAECGQQANu6AFf0B
JjCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBANMFThhG/Ty8YCjOhBkj
(omitted)
```

OK.

```
ubuntu@m0213:~/ndn-cxx$ ndnsec key-gen /A --key_id sha256digest=XX
Error: Cannot convert to ImplicitSha256DigestComponent(expected sha256 in hex encoding)
```

OK.

```
ubuntu@m0213:~/ndn-cxx$ ndnsec key-gen /A --key_id sha256digest=0000000000000000000000000000000000000000000000000
000000000000000000
Bv0C0wc7CAFBCANLRVkBIAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
CARzZWxmCAn9AAABYNdxa80UCRgBAhkEADbugBX9ASYwggEiMA0GCSqGSIb3DQEB
(omitted)
```

This one should be rejected. The resulting certificate name would have a ImplicitSha256DigestComponent in the middle, which is an invalid Data packet.

```
ubuntu@m0213:~/ndn-cxx$ ndnsec key-gen /A --key_id $(python -c "print '%CC'*12000") | head -2
Bv1gXwf9Lv0IAUEIA0tFWQj9LuDMzMzMzMzMzMzMzMzMzMzMzMzMzMzMzMzMzMzM
zMzMzMzMzMzMzMzMzMzMzMzMzMzMzMzMzMzMzMzMzMzMzMzMzMzMzMzMzMzMzMzM
```

This is acceptable. Although the resulting Data exceeds the "practical" 8800-octet limit, it is not a protocol violation.
ndnsec cert-dump works properly with this certificate.
The difficulty in certificate retrieval is not ndnsec's concern.

#4 - 01/08/2018 01:49 PM - Zhiyi Zhang

Junxiao Shi wrote:

```
ubuntu@m0213:~/ndn-cxx$ ndnsec key-gen /A --key_id sha256digest=0000000000000000000000000000000000000000000000000000000000000000
Bv0C0wc7CAFBCANLRVkBIAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
CARzZWxmCAn9AAABYNdxa80UCRgBAhkEADbugBX9ASYwggEiMA0GCSqGSIb3DQEB
(omitted)
```

> This one should be rejected. The resulting certificate name would have a ImplicitSha256DigestComponent in the middle, which is an invalid Data packet.

I thought the component::fromEscapedString() function should have already erased sha256digest=. Why does it still exist in the name? I paste the code here:

```
Component
Component::fromEscapedString(const char* escapedString, size_t beginOffset, size_t endOffset)
{
  std::string trimmedString(escapedString + beginOffset, escapedString + endOffset);
  boost::algorithm::trim(trimmedString);

  if (trimmedString.compare(0, getSha256DigestUriPrefix().size(),
                            getSha256DigestUriPrefix()) == 0) {
    if (trimmedString.size() != getSha256DigestUriPrefix().size() + util::Sha256::DIGEST_SIZE * 2)
      BOOST_THROW_EXCEPTION(Error("Cannot convert to ImplicitSha256DigestComponent"
                                  "(expected sha256 in hex encoding)"));

    try {
      trimmedString.erase(0, getSha256DigestUriPrefix().size());
      return fromImplicitSha256Digest(fromHex(trimmedString));
    }
    catch (const StringHelperError&) {
      BOOST_THROW_EXCEPTION(Error("Cannot convert to a ImplicitSha256DigestComponent (invalid hex "
                                  "encoding)"));
    }
...
```

#5 - 01/08/2018 02:04 PM - Alex Afanasyev

I think you're misunderstanding. What Junxiao is saying this component type should not be allowed as key id. fromEscapedString() is doing what it is suppose to do. Just add check that after decoding the component type (.type()) is not ImplicitSha256DigestComponent

#6 - 01/08/2018 02:08 PM - Junxiao Shi

I'd suggest allowing only *GenericNameComponent* and rejecting all other types. With "typed component" coming, it's best to whitelist each type individually.

#7 - 01/09/2018 12:58 AM - Junxiao Shi

> Exactly the opposite. He current way is correct and whitelisting would lead to problems

Let me elaborate why I think KeyId should whitelist *GenericNameComponent*.

1. When typed component is implemented, I anticipate there will be a "key id" component type, along with many other component types that cannot be "key id".
2. "key id" is the recommended component type for key id. GenericNameComponent is still accepted for backwards compatibility.
3. Blacklisting ImplicitSha256DigestComponent would cause ndnsec key-gen to incorrectly accept other component types.
4. Whitelisting GenericNameComponent for now would then require extending the whitelist to include "key id" component, which should be part of the typed component implementation commit.

#8 - 01/09/2018 10:05 AM - Alex Afanasyev

- Status changed from Code review to Closed

- % Done changed from 80 to 100