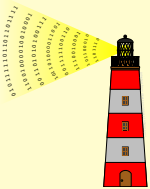
Notes about the NIST Reference for Randomness Beacons

Cryptographic Technology Group

National Institute of Standards and Technology (USA)



Presentation at The Randomness Summit 2023

March 30th, 2023 @ Tokyo, Japan

Suggested reading: NISTIR 8213 Draft

A Reference for Randomness Beacons:

Format and Protocol Version 2

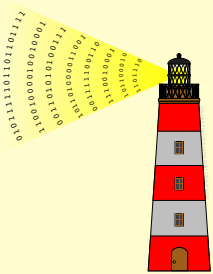
* Luís Brandão: At NIST as a Foreign Guest Researcher (non-employee), Contractor from Strativia. Expressed opinions are from the speaker and should not be construed as official NIST views. Joint work with Harold Booth, John Kelsey, and René Peralta.

Outline

Outline

A Randomness Beacon

A service that produces timed outputs of fresh public randomness.

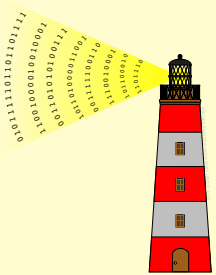


A Randomness Beacon

A service that produces timed outputs of fresh public randomness.

High-level description:

- Periodically *pulsates* randomness (e.g., 1 per min)
- Each pulse has a fresh 512-bit random string
- Each pulse is indexed, time-stamped and signed
- Any past pulse is publicly accessible
- The sequence of pulses forms a hash-chain

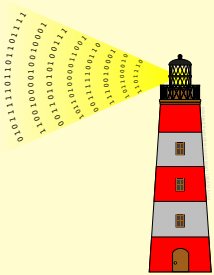


A Randomness Beacon

A service that produces timed outputs of fresh public randomness.

High-level description:

- Periodically *pulsates* randomness (e.g., 1 per min)
- Each pulse has a fresh 512-bit random string
- Each pulse is indexed, time-stamped and signed
- Any past pulse is publicly accessible
- The sequence of pulses forms a hash-chain



Can be useful for:

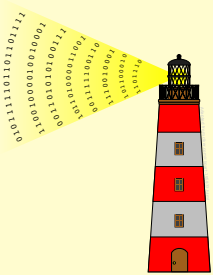
Not good for:

A Randomness Beacon

A service that produces timed outputs of fresh public randomness.

High-level description:

- Periodically *pulsates* randomness (e.g., 1 per min)
- Each pulse has a fresh 512-bit random string
- Each pulse is indexed, time-stamped and signed
- Any past pulse is publicly accessible
- The sequence of pulses forms a hash-chain



Can be useful for: (i) public auditability of randomized processes; (ii) coordination between many parties; (iii) prove something happened after a certain time.

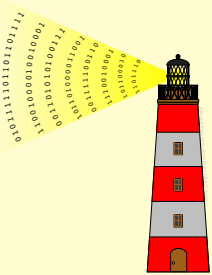
Not good for:

A Randomness Beacon

A service that produces timed outputs of fresh public randomness.

High-level description:

- Periodically *pulsates* randomness (e.g., 1 per min)
- Each pulse has a fresh 512-bit random string
- Each pulse is indexed, time-stamped and signed
- Any past pulse is publicly accessible
- The sequence of pulses forms a hash-chain



Can be useful for: (i) public auditability of randomized processes; (ii) coordination between many parties; (iii) prove something happened after a certain time.

Not good for: selecting your secret keys

Conceived applications

Clinical trials.
The public can check the trial was properly randomized.

Legal metrology.
Ensure fresh proofs of software possession by measuring instruments.

Financial audits.
Select public official for audit, without risk of political biasing.

Judge selection.
Defenders and prosecutors verify unbiased choice of judge to court case.

Quality control.
Build audit trail for later verification of the selected sample.

Historical notes

- **2011:** Start of Beacon project @ NIST, to promote development of randomness Beacons.
- **2012:** Internal grant: research QRNG, and implement a NIST Rand. Beacon.
- **2013-Sep till 2018-Dec:** NIST Beacon service version 1.0 online
- **2015:** Experimental validation of Bell loophole-free inequalities.

Historical notes

- **2011:** Start of Beacon project @ NIST, to promote development of randomness Beacons.
- **2012:** Internal grant: research QRNG, and implement a NIST Rand. Beacon.
- **2013-Sep till 2018-Dec:** NIST Beacon service version 1.0 online
- **2015:** Experimental validation of Bell loophole-free inequalities.
- **2018:** QRNG based on photon detection (by Physics Measurement Lab).
- **2018-Jul till present:** NIST Beacon service version 2.0 online
- **2019-May:** "Draft NISTIR 8213: *A Reference for Randomness Beacons* (version 2)
- **2023 (upcoming):** Final version of NISTIR 8213 and open-source code.

NIST Project: Interoperable Randomness Beacons

In 2019: project renamed to differentiate various tracks.

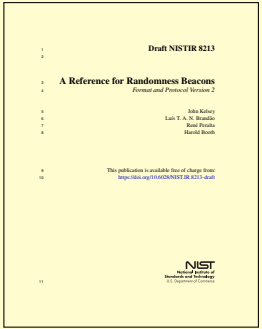
- **Track A: promote a reference for randomness beacons**
- **Track B: maintain a NIST Randomness Beacon implementatation**
- **Track C: promote the deployment of multiple independent Beacons**
- **Track D: promote applicaitons of beacon-issued randomness**
- **Track E: complementary initiatives about trusted randomness**

https://csrc.nist.gov/projects/interoperable-randomness-beacons

Outline

NISTIR 8213 (draft): A Reference for Randomness Beacons

Subtitle: Format and Protocol Version 2. https://doi.org/10.6028/NIST.IR.8213-draft (May 2019)



Draft NISTIR 8213

A Reference for Randomness Beacons

Format and Protocol Version 2

John Kelsey
Luís T. A. N. Brandão
René Peralta
Harold Booth

This publication is available free of charge from:
https://doi.org/10.6028/NIST.IR.8213-draft

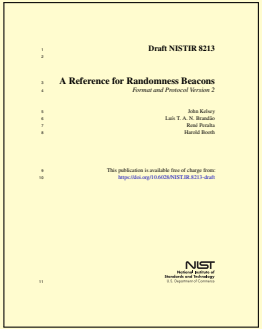
NIST
National Institute of Standards and Technology
U.S. Department of Commerce

NISTIR 8213 (draft): A Reference for Randomness Beacons

Subtitle: Format and Protocol Version 2. https://doi.org/10.6028/NIST.IR.8213-draft (May 2019)

Some topics in the report:

- format for pulses
- protocol for beacon operations
- using Beacon randomness
- security considerations



Draft NISTIR 8213

A Reference for Randomness Beacons
Format and Protocol Version 2

John Kelsey
Luís T. A. N. Brandão
René Peralta
Harold Booth

This publication is available free of charge from:
https://doi.org/10.6028/NIST.IR.8213-draft

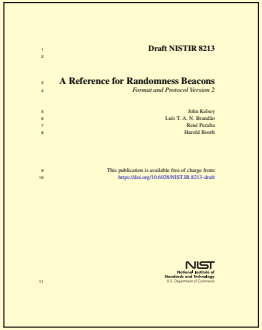
NIST
National Institute of Standards and Technology
U.S. Department of Commerce

NISTIR 8213 (draft): A Reference for Randomness Beacons

Subtitle: Format and Protocol Version 2. https://doi.org/10.6028/NIST.IR.8213-draft (May 2019)

Some topics in the report:

- format for pulses
- protocol for beacon operations
- using Beacon randomness
- security considerations



Draft NISTIR 8213

A Reference for Randomness Beacons
Format and Protocol Version 2

John Kelsey
Luís T. A. N. Brandão
René Peralta
Harold Booth

This publication is available free of charge from:
https://doi.org/10.6028/NIST.IR.8213-draft

NIST
National Institute of Standards and Technology
U.S. Department of Commerce

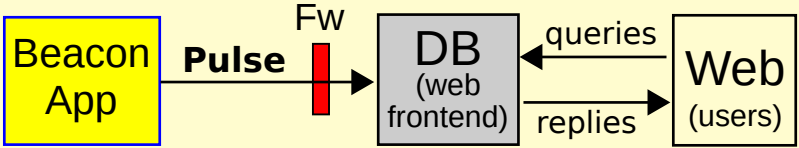
It has been a draft for a long time.

Final version will be published in 2023, along with open-source code.

Fetching pulses

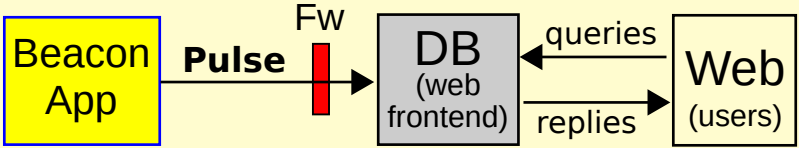
Fetching pulses

Beacon App: a *pulse release* means *sending the pulse to the database*



Fetching pulses

Beacon App: a *pulse release* means *sending the pulse to the database*

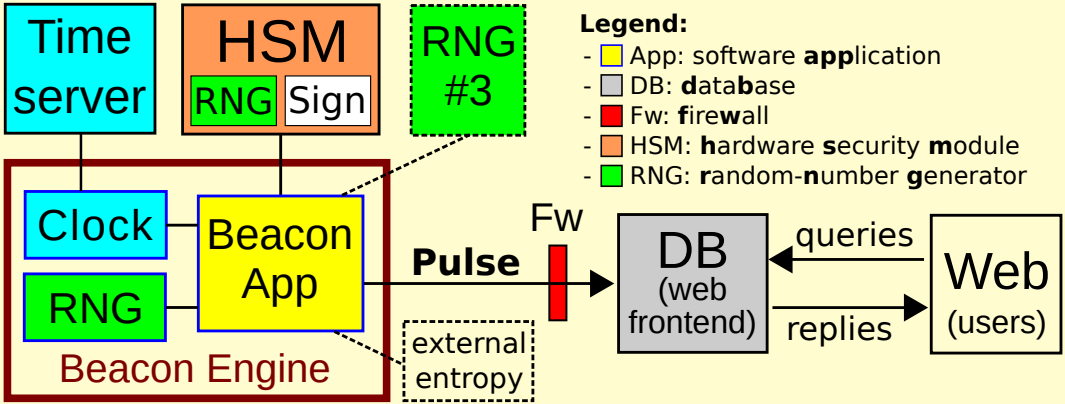


Users request pulses from the database using a URI/URL.

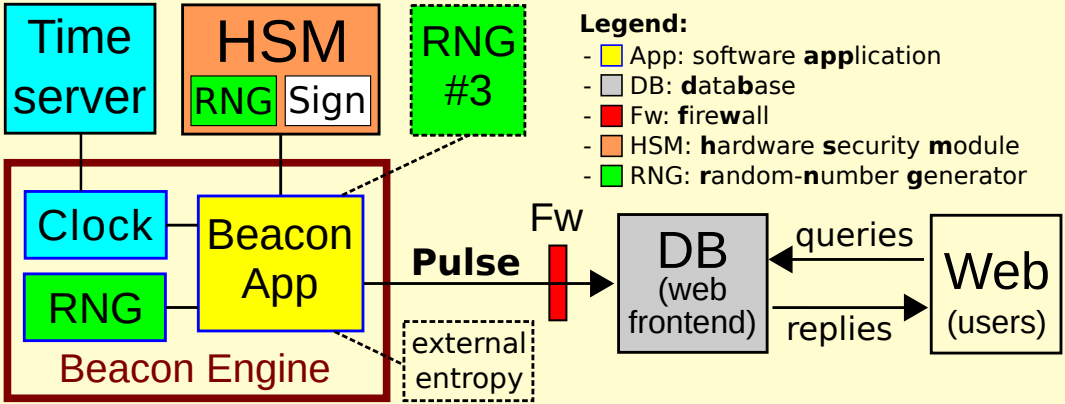
https://beacon.nist.gov/beacon/2.0/chain/last/pulse/last

Example: URI for the latest pulse in chain 1 of the NIST randomness Beacon (version 2)

Components of the Beacon service, at a high level



Components of the Beacon service, at a high level



But what exactly is a *pulse*, what is its randomness, ...?

Outline

The two “rands” in a pulse

randLocal (a.k.a. local random value):

randOut (a.k.a. output value):

The two "rands" in a pulse

randLocal (a.k.a. local random value):

- Hash (SHA512) of randomness output by ≥ 2 RNGs
- Pre-committed 1 minute in advance of release
- Useful for combining beacons

randOut (a.k.a. output value):

The two “rands” in a pulse

randLocal (a.k.a. local random value):

- Hash (SHA512) of randomness output by ≥ 2 RNGs
- Pre-committed 1 minute in advance of release
- Useful for combining beacons

randOut (a.k.a. output value):

- Hash of all other fields
- **Fresh** at the time of release
- The actual randomness to be used by applications

The two “rands” in a pulse

Pulse i
T_i=2019-05-17T16:1**3**:00.000Z
...
out.Prev: R_{i-1}=0110...
...
rand**Local**: r_i = 1001...
preCom: C_i = 0101...
...
sig: S_i = 1010...
rand**Out**: R_i = 1110...

Pulse i+1
T_i=2019-05-17T16:1**4**:00.000Z
...
out.Prev: R_i = 1110...
...
rand**Local**: r_{i+1} = 1101...
preCom: C_{i+1} = 0010...
...
sig: S_{i+1} = 0111...
rand**Out**: R_{i+1} = 1011...

The two "rands" in a pulse

`randLocal`: r_{i+1} $= \text{Hash}(\rho_{1,i} \mid\mid \rho_{2,i}\ [\mid\mid \rho_{3,i}])$, with random $\rho_{j,i}$ from i^{th} RNG

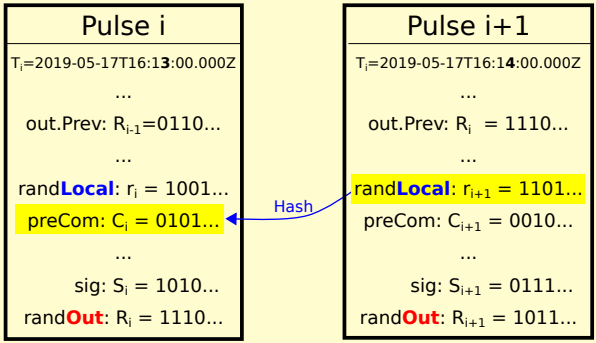
Pulse i
T_i=2019-05-17T16:1**3**:00.000Z
...
out.Prev: R_{i-1}=0110...
...
rand**Local**: r_i = 1001...
preCom: C_i = 0101...
...
sig: S_i = 1010...
rand**Out**: R_i = 1110...

Pulse i+1
T_i=2019-05-17T16:1**4**:00.000Z
...
out.Prev: R_i = 1110...
...
rand**Local**: r_{i+1} = 1101...
preCom: C_{i+1} = 0010...
...
sig: S_{i+1} = 0111...
rand**Out**: R_{i+1} = 1011...

The two “rands” in a pulse

`randLocal`: $r_{i+1} = \text{Hash}(\rho_{1,i} \mid\mid \rho_{2,i}\ [\mid\mid \rho_{3,i}])$, with random $\rho_{j,i}$ from i^{th} RNG

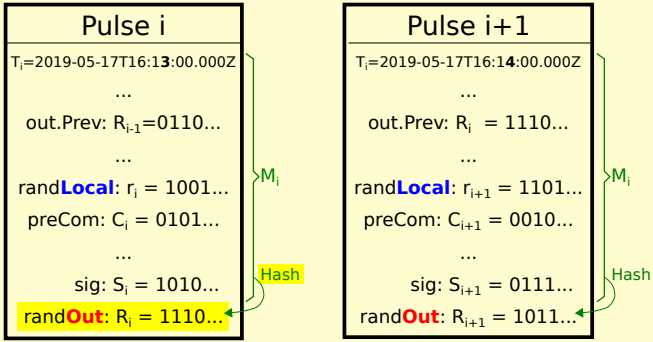
`preCom`: $C_i = \text{Hash}(r_{i+1})$, released 1 min before r_{i+1}



The two "rands" in a pulse

`randLocal`: $r_{i+1} = \text{Hash}(\rho_{1,i} \mid\mid \rho_{2,i}\ [\mid\mid \rho_{3,i}])$, with random $\rho_{j,i}$ from i^{th} RNG

`preCom`: $C_i = \text{Hash}(r_{i+1})$, released 1 min before r_{i+1}

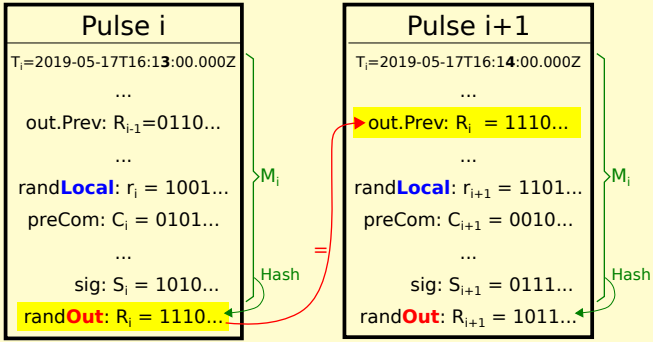


`randOut`: $R_i = \text{Hash}(M_i)$, with M_i being the serialization of all previous fields

The two "rands" in a pulse

`randLocal`: $r_{i+1} = \text{Hash}(\rho_{1,i} \mid\mid \rho_{2,i}\ [\mid\mid \rho_{3,i}])$, with random $\rho_{j,i}$ from i^{th} RNG

`preCom`: $C_i = \text{Hash}(r_{i+1})$, released 1 min before r_{i+1}

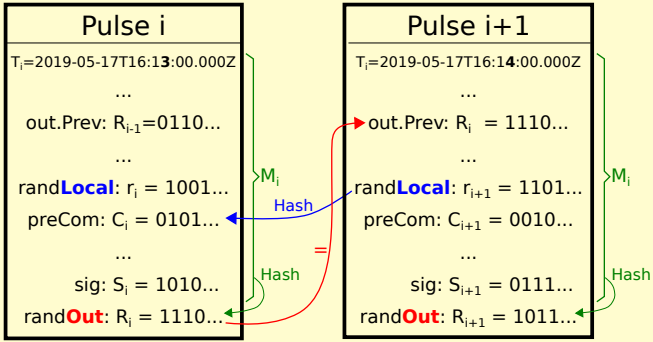


`randOut`: $R_i = \text{Hash}(M_i)$, with M_i being the serialization of all previous fields

The two "rands" in a pulse

`randLocal`: $r_{i+1} = \text{Hash}(\rho_{1,i} \mid\mid \rho_{2,i}\ [\mid\mid \rho_{3,i}])$, with random $\rho_{j,i}$ from i^{th} RNG

`preCom`: $C_i = \text{Hash}(r_{i+1})$, released 1 min before r_{i+1}



`randOut`: $R_i = \text{Hash}(M_i)$, with M_i being the serialization of all previous fields

A pulse (simplified example)

```
uri:str="https://beacon.nist.gov/beacon/2.0/chain/1/pulse/220394"
version:str="2.0"
...
period:dec="60000"
...
chainId:dec="1"
pulseId:dec="220394"
time:str="2018-12-26T16:07:00.000Z"
randLocal:hex="5FF1E0C019C42C77FA72D522...(512 bits total)"
...
out.Prev:hex="BA646CC4E7AE195D2C85E9D3...(512 bits total)"
...
preCom:hex="269908B840E79BE71CEC4EBA...(512 bits total)"
...
sig:hex="17943D886DA8C7C24B9244BE...(4096 bits total)"
randOut:hex="0A8863E03E200F6940A009B0...(512 bits total)"
```

A pulse (simplified example)

```
uri:str="https://beacon.nist.gov/beacon/2.0/chain/1/pulse/220394"
version:str="2.0"
...
period:dec="60000"
...
chainId:dec="1"
pulseId:dec="220394"
time:str="2018-12-26T16:07:00.000Z"
randLocal:hex="5FF1E0C019C42C77FA72D522...(512 bits total)"
...
out.Prev:hex="BA646CC4E7AE195D2C85E9D3...(512 bits total)"
...
preCom:hex="269908B840E79BE71CEC4EBA...(512 bits total)"
...
sig:hex="17943D886DA8C7C24B9244BE...(4096 bits total)"
randOut:hex="0A8863E03E200F6940A009B0...(512 bits total)"
```

- Each pulse is indexed.

A pulse (simplified example)

```
uri:str="https://beacon.nist.gov/beacon/2.0/chain/1/pulse/220394"
version:str="2.0"
...
period:dec="60000"
...
chainId:dec="1"
pulseId:dec="220394"
time:str="2018-12-26T16:07:00.000Z"
randLocal:hex="5FF1E0C019C42C77FA72D522...(512 bits total)"
...
out.Prev:hex="BA646CC4E7AE195D2C85E9D3...(512 bits total)"
...
preCom:hex="269908B840E79BE71CEC4EBA...(512 bits total)"
...
sig:hex="17943D886DA8C7C24B9244BE...(4096 bits total)"
randOut:hex="0A8863E03E200F6940A009B0...(512 bits total)"
```

- Each pulse is indexed.

A pulse (simplified example)

```
uri:str="https://beacon.nist.gov/beacon/2.0/chain/1/pulse/220394"
version:str="2.0"
...
period:dec="60000"
...
chainId:dec="1"
pulseId:dec="220394"
time:str="2018-12-26T16:07:00.000Z"
randLocal:hex="5FF1E0C019C42C77FA72D522...(512 bits total)"
...
out.Prev:hex="BA646CC4E7AE195D2C85E9D3...(512 bits total)"
...
preCom:hex="269908B840E79BE71CEC4EBA...(512 bits total)"
...
sig:hex="17943D886DA8C7C24B9244BE...(4096 bits total)"
randOut:hex="0A8863E03E200F6940A009B0...(512 bits total)"
```

- Each pulse is indexed.

A pulse (simplified example)

```
uri:str="https://beacon.nist.gov/beacon/2.0/chain/1/pulse/220394"
version:str="2.0"
...
period:dec="60000"
...
chainId:dec="1"
pulseId:dec="220394"
time:str="2018-12-26T16:07:00.000Z"
randLocal:hex="5FF1E0C019C42C77FA72D522...(512 bits total)"
...
out.Prev:hex="BA646CC4E7AE195D2C85E9D3...(512 bits total)"
...
preCom:hex="269908B840E79BE71CEC4EBA...(512 bits total)"
...
sig:hex="17943D886DA8C7C24B9244BE...(4096 bits total)"
randOut:hex="0A8863E03E200F6940A009B0...(512 bits total)"
```

- Each pulse is indexed.
- Two main random values ("rands"): `randLocal` and `randOut`.

A pulse (simplified example)

```
uri:str="https://beacon.nist.gov/beacon/2.0/chain/1/pulse/220394"
version:str="2.0"
...
period:dec="60000"
...
chainId:dec="1"
pulseId:dec="220394"
time:str="2018-12-26T16:07:00.000Z"
randLocal:hex="5FF1E0C019C42C77FA72D522...(512 bits total)"
...
out.Prev:hex="BA646CC4E7AE195D2C85E9D3...(512 bits total)"
...
preCom:hex="269908B840E79BE71CEC4EBA...(512 bits total)"
...
sig:hex="17943D886DA8C7C24B9244BE...(4096 bits total)"
randOut:hex="0A8863E03E200F6940A009B0...(512 bits total)"
```

- Each pulse is indexed.
- Two main random values ("rands"): `randLocal` and `randOut`.
- Other features: signature

A pulse (simplified example)

```
uri:str="https://beacon.nist.gov/beacon/2.0/chain/1/pulse/220394"
version:str="2.0"
...
period:dec="60000"
...
chainId:dec="1"
pulseId:dec="220394"
time:str="2018-12-26T16:07:00.000Z"
randLocal:hex="5FF1E0C019C42C77FA72D522...(512 bits total)"
...
out.Prev:hex="BA646CC4E7AE195D2C85E9D3...(512 bits total)"
...
preCom:hex="269908B840E79BE71CEC4EBA...(512 bits total)"
...
sig:hex="17943D886DA8C7C24B9244BE...(4096 bits total)"
randOut:hex="0A8863E03E200F6940A009B0...(512 bits total)"
```

- Each pulse is indexed.
- Two main random values ("rands"): `randLocal` and `randOut`.
- Other features: signature, precommit `randLocal`

A pulse (simplified example)

```
uri:str="https://beacon.nist.gov/beacon/2.0/chain/1/pulse/220394"
version:str="2.0"
...
period:dec="60000"
...
chainId:dec="1"
pulseId:dec="220394"
time:str="2018-12-26T16:07:00.000Z"
randLocal:hex="5FF1E0C019C42C77FA72D522...(512 bits total)"
...
out.Prev:hex="BA646CC4E7AE195D2C85E9D3...(512 bits total)"
...
preCom:hex="269908B840E79BE71CEC4EBA...(512 bits total)"
...
sig:hex="17943D886DA8C7C24B9244BE...(4096 bits total)"
randOut:hex="0A8863E03E200F6940A009B0...(512 bits total)"
```

- Each pulse is indexed.
- Two main random values ("rands"): `randLocal` and `randOut`.
- Other features: signature, precommit `randLocal`, chain `randOut`, ...

Outline

Other notes on Beacon operation

- **Skiplists:** for efficient verification chain verification
- **Combining beacons:** Coin-flipping based on preCom before opened localRand
- **Timing requirements:** various "promises" and recommendations
- **Beacon interface:** specifies how to get useful info from a beacon
- **External values:** to mitigate advanced computation attacks (by malicious beacon)
- **Future guidance (extra documents):** recommendations on using beacon randomness, external values, combining beacons, ...

Skiplists — efficient chain verification

How to prove that an **old** pulse is consistent with a **recent** pulse?

Example: ANCHOR = **2019-05-17 14:12** → TARGET = **2016-02-14 17:45**

Skiplists — efficient chain verification

How to prove that an **old** pulse is consistent with a **recent** pulse?

Example: ANCHOR = **2019-05-17 14:12** → TARGET = **2016-02-14 17:45**

Inefficient: check the hash-chain of (>1M) consecutive pulses

2019-05-17 14:12 → 2019-05-17 14:11 → (1 per minute) → **2016-02-14 17:45**

Skiplists — efficient chain verification

How to prove that an **old** pulse is consistent with a **recent** pulse?

Example: ANCHOR = **2019-05-17 14:12** → TARGET = **2016-02-14 17:45**

Inefficient: check the hash-chain of (>1M) consecutive pulses

2019-05-17 14:12 → 2019-05-17 14:11 → (1 per minute) → **2016-02-14 17:45**

Efficient: check a **skiplist** (< 125 pulses), using the 5 *past output* fields:

Skiplists — efficient chain verification

How to prove that an **old** pulse is consistent with a **recent** pulse?

Example: ANCHOR = **2019-05-17 14:12** → TARGET = **2016-02-14 17:45**

Inefficient: check the hash-chain of (>1M) consecutive pulses

2019-05-17 14:12 → 2019-05-17 14:11 → (1 per minute) → **2016-02-14 17:45**

Efficient: check a **skiplist** (< 125 pulses), using the 5 *past output* fields:

- `out.Prev`: the *previous* pulse
- `out.H`, `out.D`, `out.M`, `out.Y`: the first of the **hour**/**day**/**month**/**year**

Skiplists — efficient chain verification

How to prove that an **old** pulse is consistent with a **recent** pulse?

Example: ANCHOR = **2019-05-17 14:12** → TARGET = **2016-02-14 17:45**

Inefficient: check the hash-chain of (>1M) consecutive pulses

2019-05-17 14:12 → 2019-05-17 14:11 → (1 per minute) → **2016-02-14 17:45**

Efficient: check a **skiplist** (< 125 pulses), using the 5 *past output* fields:

- `out.Prev`: the *previous* pulse
- `out.H`, `out.D`, `out.M`, `out.Y`: the first of the **hour**/**day**/**month**/**year**

2019-05-17 14:12 → **2019-01**-01 00:00 → **2018-01**-01 00:00 → **2017-01**-01 00:00 →

Skiplists — efficient chain verification

How to prove that an **old** pulse is consistent with a **recent** pulse?

Example: ANCHOR = **2019-05-17 14:12** → TARGET = **2016-02-14 17:45**

Inefficient: check the hash-chain of (>1M) consecutive pulses

2019-05-17 14:12 → 2019-05-17 14:11 → (1 per minute) → **2016-02-14 17:45**

Efficient: check a **skiplist** (< 125 pulses), using the 5 *past output* fields:

- `out.Prev`: the *previous* pulse
- `out.H`, `out.D`, `out.M`, `out.Y`: the first of the **hour**/**day**/**month**/**year**

2019-05-17 14:12 → **2019-01**-01 00:00 → **2018-01**-01 00:00 → **2017-01**-01 00:00 →
2016-**12-01** 00:00 → (1 per month) → 2016-**03-01** 00:00

Skiplists — efficient chain verification

How to prove that an **old** pulse is consistent with a **recent** pulse?

Example: ANCHOR = **2019-05-17 14:12** → TARGET = **2016-02-14 17:45**

Inefficient: check the hash-chain of (>1M) consecutive pulses

2019-05-17 14:12 → 2019-05-17 14:11 → (1 per minute) → **2016-02-14 17:45**

Efficient: check a **skiplist** (< 125 pulses), using the 5 *past output* fields:

- `out.Prev`: the *previous* pulse
- `out.H`, `out.D`, `out.M`, `out.Y`: the first of the **hour**/**day**/**month**/**year**

2019-05-17 14:12 → **2019-01**-01 00:00 → **2018-01**-01 00:00 → **2017-01**-01 00:00 →
2016-**12-01** 00:00 → (1 per month) → 2016-**03-01** 00:00 → 2016-02-**29 00**:00 →
(1 per day) → 2016-02-**15 00**:00

Skiplists — efficient chain verification

How to prove that an **old** pulse is consistent with a **recent** pulse?

Example: ANCHOR = **2019-05-17 14:12** → TARGET = **2016-02-14 17:45**

Inefficient: check the hash-chain of (>1M) consecutive pulses

2019-05-17 14:12 → 2019-05-17 14:11 → (1 per minute) → **2016-02-14 17:45**

Efficient: check a **skiplist** (< 125 pulses), using the 5 *past output* fields:

- `out.Prev`: the *previous* pulse
- `out.H`, `out.D`, `out.M`, `out.Y`: the first of the **hour**/**day**/**month**/**year**

2019-05-17 14:12 → **2019-01**-01 00:00 → **2018-01**-01 00:00 → **2017-01**-01 00:00 →
2016-**12-01** 00:00 → (1 per month) → 2016-**03-01** 00:00 → 2016-02-**29 00**:00 →
(1 per day) → 2016-02-**15 00**:00 → 2016-02-14 **23:00** → (1 per hour) →
2016-02-14 **18:00**

Skiplists — efficient chain verification

How to prove that an **old** pulse is consistent with a **recent** pulse?

Example: ANCHOR = **2019-05-17 14:12** → TARGET = **2016-02-14 17:45**

Inefficient: check the hash-chain of (>1M) consecutive pulses

2019-05-17 14:12 → 2019-05-17 14:11 → (1 per minute) → **2016-02-14 17:45**

Efficient: check a **skiplist** (< 125 pulses), using the 5 *past output* fields:

- `out.Prev`: the *previous* pulse
- `out.H`, `out.D`, `out.M`, `out.Y`: the first of the **hour**/**day**/**month**/**year**

2019-05-17 14:12	→	**2019-01**-01 00:00	→	**2018-01**-01 00:00	→	**2017-01**-01 00:00	→	
2016-**12-01** 00:00	→	(1 per month)	→	2016-**03-01** 00:00	→	2016-02-**29 00**:00	→	
(1 per day)	→	2016-02-**15 00**:00	→	2016-02-14 **23:00**	→	(1 per hour)	→	
2016-02-14 **18:00**	→	2016-02-14 17:**59**	→	(1 per minute)	→	**2016-02-14 17:45**		

Outline

Final Remarks

Final Remarks

- Randomness Beacons have a **great potential to serve as a public utility**, e.g., to promote public auditability of randomized processes

Final Remarks

- Randomness Beacons have a **great potential to serve as a public utility**, e.g., to promote public auditability of randomized processes
- The *Reference* (NISTIR 8213) for version 2 introduces new features for better **interoperability, security and efficiency**. They envision an International ecosystem of randomness beacons.

Final Remarks

- Randomness Beacons have a **great potential to serve as a public utility**, e.g., to promote public auditability of randomized processes
- The *Reference* (NISTIR 8213) for version 2 introduces new features for better **interoperability, security and efficiency**. They envision an International ecosystem of randomness beacons.
- **Further guidance and promotion is needed:** how to use beacon randomness to allow verifiability; safe integration of external values, ...

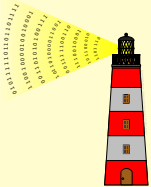
Final Remarks

- Randomness Beacons have a **great potential to serve as a public utility**, e.g., to promote public auditability of randomized processes
- The *Reference* (NISTIR 8213) for version 2 introduces new features for better **interoperability, security and efficiency**. They envision an International ecosystem of randomness beacons.
- **Further guidance and promotion is needed:** how to use beacon randomness to allow verifiability; safe integration of external values, ...
- **External values:** Not yet used in NIST Rand Beacon. It is conceivable to integrate randomness from other Rand beacons.

Final Remarks

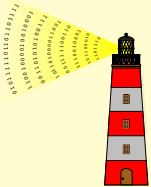
- Randomness Beacons have a **great potential to serve as a public utility**, e.g., to promote public auditability of randomized processes
- The *Reference* (NISTIR 8213) for version 2 introduces new features for better **interoperability, security and efficiency**. They envision an International ecosystem of randomness beacons.
- **Further guidance and promotion is needed:** how to use beacon randomness to allow verifiability; safe integration of external values, ...
- **External values:** Not yet used in NIST Rand Beacon. It is conceivable to integrate randomness from other Rand beacons.
- **Stay tuned:** Soon **NISTIR 8213** final; open-source code of NIST Beacon.

- Draft NISTIR 8213: https://doi.org/10.6028/NIST.IR.8213-draft
- Email for feedback on the NISTIR 8213: beacon-nistir@nist.gov
- https://csrc.nist.gov/projects/interoperable-randomness-beacons



Thank you for your attention

- Draft NISTIR 8213: https://doi.org/10.6028/NIST.IR.8213-draft
- Email for feedback on the NISTIR 8213: beacon-nistir@nist.gov
- https://csrc.nist.gov/projects/interoperable-randomness-beacons



Presentation at the The Randomness Summit 2023

March 30th, 2023 @ Tokyo, Japan

luis.brandao@nist.gov