An Upcycling Tokenization Method for Credit Card Numbers

Cyrius Nugier, **Diane Leblanc-Albarel**, Agathe Blaise, Simon Masson, Paul Huynh and Yris Brice Wandji Piugie

GDR Sécurité

July 2nd, 2021
**REDOCS : On the paper**

### Organisation

- 3 companies propose a challenge to solve.
- The doctoral students form 3 teams, one per company, depending on the subject that interests them.
- Each team is supervised by one or more members of the company.
- Each team then has four and a half days to work on the problem.
- 40 hours of work in the week.
Work

- Discovery of new topics.
- Learn to use academic skills for solving real problems.
- Learning new things.
- Work in sprint mode.
- Much more than 40 hours of work.
- 40 hours of scientific courses for the doctoral school.
Real life

Meet new people.
Learning new ways of thinking.
Very pleasant environment.
Seriousness but also a lot of fun.
Social event.
A productive week.
Outline

1. **Background**
   - Credit Card overview
   - Tokenization System
   - Specifications

2. **Related work**
   - Static pre-computed table

3. **Our work**
   - Overview
   - Functionalities
   - Performances

4. **Conclusion**
Challenges of online payments by credit card

Latest attacks

- Davinci breach, February 2019 (2.15 M stolen credit cards number).
- The Bigbadaboom-II\(^1\), March 2018.
- The Bigbadaboom-III, January 2020 (30 M stolen credit cards number).

\(^1\) Compromised details released by FIN7 threat group.
Credit Card Numbers format.

Possible token format.
Tokenisation System

Life Cycle of a Token.
Specifications

### Functional
- **Unicity**
- Uniformity
- Unlinkability
- Unforgeability

### Technical
- Expiry
- Formatting
- Timeframe
- Reusability
- Auditability
Specifications

Functional
- Unicity
- **Uniformity**
- Unlinkability
- Unforgeability

Technical
- Expiry
- Formatting
- Timeframe
- Reusability
- Auditability
Specifications

**Functional**
- Unicity
- Uniformity
- **Unlinkability**
- Unforgeability

**Technical**
- Expiry
- Formatting
- Timeframe
- Reusability
- Auditability
## Specifications

### Functional
- Unicity
- Uniformity
- Unlinkability
- **Unforgeability**

### Technical
- Expiry
- Formatting
- Timeframe
- Reusability
- Auditability
## Specifications

### Functional
- Unicity
- Uniformity
- Unlinkability
- Unforgeability

### Technical
- **Expiry**
- Formatting
- Timeframe
- Reusability
- Auditability

1. Max. # of uses
2. Expiry time
### Specifications

#### Functional
- Unicity
- Uniformity
- Unlinkability
- Unforgeability

#### Technical
- Expiry
- **Formatting**
- Timeframe
- Reusability
- Auditability

---

Diane Leblanc-Albarel  
An Upcycling Tokenization Method for Credit Card Numbers  
July 2nd, 2021
Specifications

**Functional**
- Unicity
- Uniformity
- Unlinkability
- Unforgeability

**Technical**
- Expiry
- Formatting
- **Timeframe**
- Reusability
- Auditability
Specifications

**Functional**
- Unicity
- Uniformity
- Unlinkability
- Unforgeability

**Technical**
- Expiry
- Formatting
- Timeframe
- Reusability
- Auditability
Specifications

Functional
- Unicity
- Uniformity
- Unlinkability
- Unforgeability

Technical
- Expiry
- Formatting
- Timeframe
- Reusability
- Auditability
Static pre-computed table

**Definition**
Table of all possible token values computed in advance.

**Problems**
- No mechanism avoids the saturation of the table.
- Obligation to create a new table when the previous one is saturated.
- Lower performances over time.
- More and more memory needed.
- No encryption of the table.
Our Solution: Upcycling Token Table

**Fixed size table**
- Cleaning mechanism.
- Index by token number: Very fast (constant time lookup).
- Reusable token.

**Lifespan and maximum number of use**
- Maximum number of use for each token (clean once the maximum is reached).
- Lifespan (useful in cases of forgetfulness for example).
- Second database for a trace of all operations.

**Encrypt**
Each row encrypted with AES 256.
<table>
<thead>
<tr>
<th>CCN</th>
<th># uses</th>
<th>Balance</th>
<th>Expiry</th>
<th>Auth</th>
<th>Rand</th>
</tr>
</thead>
</table>

Content of a row.
Tokenisation.
Tokenisation.
Tokenisation.

- CCN
- Expiry
- Auth
- Rand

64b → 32/64b → 32b

SHA2

$\text{hash}_{32}$

$> 2^{32} - (2^{32} \mod 10^n)$

mod $10^8$

tok

@tok not free
Tokenisation

Overview

Functionalities

Performances

Tokenisation.

Diane Leblanc-Albarel
An Upcycling Tokenization Method for Credit Card Numbers
July 2nd, 2021
Tokenisation

Tokenisation.

Overview
Functionalities
Performances
Tokenisation.

**Overview**
- **Functionalities**
- **Performances**

**Background**
- Related work

**Our work**
- Tokenisation

Diane Leblanc-Albarel
An Upcycling Tokenization Method for Credit Card Numbers
July 2nd, 2021 12/18

Tokenisation.

CCN | Expiry | Auth | Rand

> $2^{32} - (2^{32} \mod 10^n)$

$\text{hash}_{32}$

mod $10^8$

@tok

@tok not free

CCN | # uses | Balance | Expiry | Auth | Rand

encrypted with sk

tok

@tok

Diagram:
- Input: CCN, Expiry, Auth, Rand
- Output: Tokenised data
- SHA2 hash
- $\text{mod } 10^8$
- Token (tok)

Note: Tokenisation process involves transforming credit card numbers into a more secure format.
Detokenisation
Detokenisation.
Detokenisation
Detokenisation.

Detokenisation.
Detokenisation.
Clean table.
Probability of failure

Number of token according to $\lambda$

- $T$  Number of tries per timeframe.
- $n_{max}$  Number of available tokens ($10^8$).
- $n$  Number of token inserted.

Probability of failure given a fixed threshold:

$$\left(\frac{n}{n_{max}}\right)^T < \frac{1}{2^\lambda} \iff n < 2^{\log_2(n_{max}) - \frac{\lambda}{T}}.$$  

Probability of failure lower than $\frac{1}{2^{128}}$

With $T = 70,000$, maximum table fill rate: $99.8733\%$

→ The limit is the 8-digit model, not the implementation.
Experiments

Environment

- AMD EPYC 7742 Processor.
- 3240.029MHz.

Results

- table fill rate: > 99.987%.
- Tries before first failure: ≈ 70 250.
- Detokenisation time: 6µ per token.
- RAM used: 25 GB.
Take away

Significant improvement

- Average of 1 billion credit card transactions per day worldwide (i.e., 11 574 transactions per second, 7M per 10 min).
- Our construction covers 6.5 times the current number of transactions.
- With a 10-minute token lifespan, at maximum token creation speed: maximum of 45 million valid tokens can be in the table at any given time.
Do you have any questions?

Contact authors: cnugier@laas.fr  diane.leblanc-albarel@irisa.fr