# Limited use cryptographic tokens in securing cloud servers

Gautam Kumar, Brent Lagesse
Computing and Software Systems
University of Washington Bothell
{gautamk,lagesse}@uw.edu

## Abstract

## Introduction

Leslie Lamport [5] was first to propose the use of hash chains in his paper on a method for secure password authentication over an insecure medium. In this paper we try to use the limited use property of hash chains to secure configuration information on ephemeral cloud servers.

## Background

### *Cryptographic hash function [7]*

A cryptographic hash function is any one way function which meets the following requirements

- Preimage resistance
- Collision resistance
- Second Preimage resistance

A hash function has preimage resistance if given a hash value $h$ it is computationally infeasible to find any message $m$ such that $h = hash(k, m)$ where $k$ is the hash key.

A hash function is collision resistant if, given two messages $m_1$ and $m_2$ it is hard to find a hash $h$ such that $h = hash(k, m_1) = hash(k, m_2)$ where $k$ is the hash key.

A hash function has second pre-image resistance if given a message $m_1$ it is computationally infeasible to find a different message $m_2$ such that $hash(k, m_1) = hash(k, m_2)$ where $k$ is the hash key. The second pre-image resistance is a much harder property to achieve for hash functions. This property is closely related to the birthday problem [6].

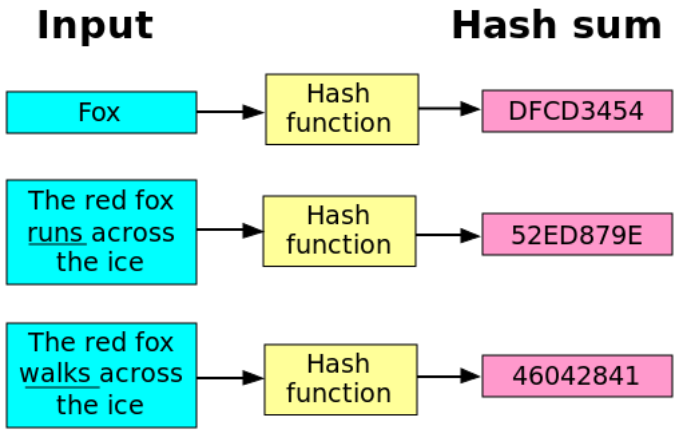


Fig. 1. A simplified view of a hash function which represents its input and potential result. The length of the hash sum always remains the same regardless of the input size. Any small change in the input drastically changes the output, Figure source: [1]

### *Hash Chains [4]*

"A hash chain is a sequence of values derived via consecutive applications of a cryptographic hash function to an initial input. Due to the properties of the hash function, it is relatively easy to calculate successive values in the chain but given a particular value,it is infeasible to determine the previous value"

As an example, Let $x$ be the initial password a hash chain of length 2 would be $H^2(x) = H(H(x))$. So a hash chain of $n$ values is denoted as $H^n(x)$ and the $i^{th}$ value in the chain would be computed as $x_i = H(x_{i-1})$.

For a given value in the chain $x_i$ its computationally infeasible to determine the previous value in the chain $x_{i-1}$.

## Proposed Architecture

### *Assumptions*

-Ephemeral client facing servers -reasoning for ephemeral servers -netflix [2]

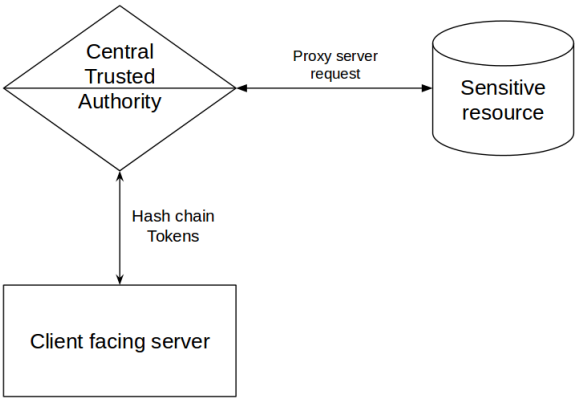


Fig. 2. The high level architectural overview of the system describes the three primary modules involved. The client facing server, The central trusted authority and the sensitive resource.

## Performance testing

## Future work

- Timed release [3] -

## Related work

## References


[1] Cryptographic hash function. Wikipedia:Cryptographic_hash_function.

[2] A. Basiri, N. Behnam, R. de Rooij, L. Hochstein, L. Kosewski, J. Reynolds, and C. Rosenthal. Chaos engineering. 33(3):35–41.

[3] Konstantinos Chalkias and George Stephanides. Timed release cryptography from bilinear pairings using hash chains. In *Communications and Multimedia Security*, pages 130–140. Springer.

[4] Dwight Horne. Hash chain. In Henk C. A. van Tilborg and Sushil Jajodia, editors, *Encyclopedia of Cryptography and Security*, pages 542–543. Springer US. DOI: 10.1007/978-1-4419-5906-5_780.

[5] Leslie Lamport. Password authentication with insecure communication. 24(11):770–772.

[6] Lawrence M. Lesser. Exploring the birthday problem with spreadsheets. 92(5):407–411.

[7] Phillip Rogaway and Thomas Shrimpton. Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. In Bimal Roy and Willi Meier, editors, *Fast Software Encryption*, number 3017 in Lecture Notes in Computer Science, pages 371–388. Springer Berlin Heidelberg. DOI: 10.1007/978-3-540-25937-4_24.