Enhancing the security of cloud server configuration with hash chains

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

I. Executive Summary

Information security is a difficult problem, especially in cloud computing environments where threats to information security are difficulty to identify and mitigate. One such problem is managing the sensitive configuration information.

In traditional enterprise deployments developers often stored configuration within code or on config files along with code. This was considered a safe practice because the hardware and Operating System where code deployment occurred was owned and managed by the enterprise themselves. These systems were usually behind a strong firewall so threats were much easier to mitigate.

In cloud computing environments, where hardware and the underlying software hypervisor are shared amoung thousands of customers who could potentially be using the resources of a single data center, Threats to security are much more complex and we need a layered strategy to secure our systems. In this environment storing sensitive configuration information on disk could potentially be dangerous.

One of the ways developers have secured systems in this environment is to use a centralised trusted server to store and retrieve sensitive configuration information. The goal of this project is investigating ways to improve the security and allow for forward secracy using Hash Chains[8].

II. Project Description

A. Generic cloud architecture

A simple cloud architecture (figure 1) generally consists of one or more servers respoding to requests from clients[5]. These servers are usually behind a load balancer which routes request to each server based on various factors such as CPU usage, Memory usage, Request Volume and Response time. These factors also serve as the basis for scaling the number of servers. All servers read and write to the central database.

1) Ephemeral nature: To facilitate auto scaling during peak and lean load times all servers are expected to be ephemeral. Thus any server can be added or removed from the pool at any time. Cloud computing providers are able to achieve such high levels of varied demands by sharing the available hardware resources amoung many customers.

2) Networking: All cloud providers offer some solution to the problem of networking between their cloud components. This means that the loadbalancer, servers and the central database will appear to be operating on the same local area network. This is achieved through overlay networks or some other kind of Software defined network. These networks also tend to be dynamic and ephemeral in nature and cloud architects generally have the ability to redefine the network structure.

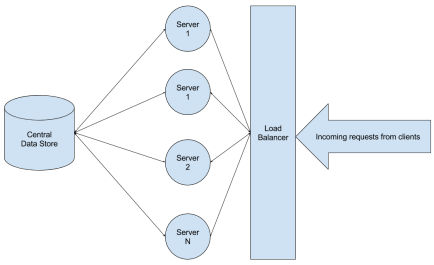

Fig. 1. A simple cloud architecture with N servers and a central database

B. Sensitive server configuration

Server configurations can consist of a huge variety of information that is meant to be easily modified to change server behaviour. The most valuable and senstive of these are usually database credentials and API keys.

Storing sensitive information such as API Keys and DB credentials along with source code on disk is commonly practiced. This practice is highly insecure and should be avoided at all costs because of the ease with which data can be recovered from disk in shared environments[3].

Full disk encryption can help mitigate the risk of data recovery from a shared disk but it comes with its own set of pitfalls such as

- Information leakage and brute force attacks on disk encryption
- Side channel attack[7]
- Vulnerability in code could lead to a directroy traversal attack[6]
- Lower server performance requiring more resources leading to higher costs, This can quickly add up on large scale cloud deployments with thousands of server instances.

Storing configuration as part of code slows down the mitigation effort incase of a breach because migrating a large

number of servers to the new configuration could take a long time.

C. Centralized Trusted Authority

A commonly deployed solution to the server configuration problem is using a Centralized Trusted Authority (CTA). The concept of trusted authority has existed for a long time [1]. All servers in a cloud deployment would fetch sensitive configuration information from the CTA server during startup and can be easily prompted to reconfigure themselves whenever needed there's any change.

The introduction of a CTA brings with it a new dimension of threats against the whole system such as

- CTA Authentication
- Client Authentication
- Client Authorization
- Attacks against the CTA
- Trustability of the CTA

The aim of the proposed research project is to investigate and propose a method for preventing or mitigating one such threat where an attacker has potentially gained command execution access to a CTA client. Our research would focus on using hash chains as single use keys to provide forwards secrecy[2] and revokability while preventing the attacker from gaining access to any sensitive information.

D. Hash Chains

A hash chain is a set of values obtained by successively hashing a value with a cryptographic Hash function. Owing to the one way nature of hash functions, hash chains are directed.

$$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow x_3 ... x_n \tag{1}$$

where $x_0 = x$, $x_1 = f(x)$ and $x_2 = f(f(x))$. Here f refers to the hashing function. The directed and one way nature of hash chains naturally lends itself to the implementation of single use tokens.

Single use tokens are key to implementing forward secrecy in this project. Such tokens work well in a cloud environment because server instances, as described earlier, are ephemeral. Each server could be allocated a hash chain of a certain length and once it runs out the server would no longer be able to access sensitive configuration information and would thus be discarded by the load balancer.

E. Activities

The first step in building a solution to the previously described problem would be to understand the attack vectors by constructing a threat model which describes the assumptions and scope of the protection mechanism. The next step would be to conduct a feasibility study to investigate the potential mechanisms for implementing hash chains as a solution to the problem of forward secrecy. The final step would be to build a prototype to demonstrate forward secrecy and re-evaluate the threat model describe earlier.

F. Broader Impact

The system of forward secrecy in stored configurations can be applied to any system operating in a potentially untrusted environment where sensitive information needs to be re-deployed / re-configured. Some examples include wireless sensor networks, ubiquitous computing devices and IoT devices.

G. Related Project

Confidant[4] is a project which offers authentication, authorization and revokability to clients using a propriatery key management system from amazon.Confidant serves as inspiration for our research. Our aim is to build upon some of the concepts implemented in confidant and offer revokability and forward secracy without the use of a propriatery Key management System.

REFERENCES

[1] System and method for providing trusted services via trusted server agents. U.S. Classification 713/178, 726/10, 380/246; International Classification H04L9/32, G06F21/00, G06F11/30, H04L9/00, G06F12/14, H04L29/06; Cooperative Classification G06F21/606, H04L63/0823, G06F2221/2151; European Classification G06F21/60C, H04L63/08C.

[2] M. Conti, R. D. Pietro, L. V. Mancini, and A. Spognardi. RIPP-FS: An RFID identification, privacy preserving protocol with forward secrecy. In *Fifth Annual IEEE International Conference on Pervasive Computing and Communications Workshops, 2007. PerCom Workshops '07*, pages 229–234.

[3] Michael Jordon. Cleaning up dirty disks in the cloud. 2012(10):12–15.

[4] lyft. Confidant: Your secret keeper.

[5] R. Moreno-Vozmediano, R. S. Montero, and I. M. Llorente. IaaS cloud architecture: From virtualized datacenters to federated cloud infrastructures. 45(12):65–72.

[6] owasp. Path traversal - OWASP.

[7] Thomas Ristenpart, Eran Tromer, Hovav Shacham, and Stefan Savage. Hey, you, get off of my cloud: Exploring information leakage in third-party compute clouds. In *Proceedings of the 16th ACM Conference on Computer and Communications Security*, CCS '09, pages 199–212. ACM.

[8] Biming Tian, Song Han, T. S. Dillon, and Sajal Das. A self-healing key distribution scheme based on vector space secret sharing and one way hash chains. In *World of Wireless, Mobile and Multimedia Networks, 2008. WoWMoM 2008. 2008 International Symposium on a*, pages 1–6.