CloudProtect – A Cloud-based Software Protection Service

Andreas Schaad
Univ. of Appl. Sciences Offenburg
Badstrasse 24, 77652 Offenburg
Germany
andreas.schaad@hs-offenburg.de

Björn Grohmann
Wibu-Systems AG
Rüppurer Str. 52, 76137 Karlsruhe
Germany
bjoern.grohmann@wibu.com

Oliver Winzenried
Wibu-Systems AG
Rüppurer Str. 52, 76137 Karlsruhe
Germany
oliver.winzenried@wibu.com

ABSTRACT

Protecting software from illegal access, intentional modification or reverse engineering is an inherently difficult practical problem involving code obfuscation techniques and real-time cryptographic protection of code. In traditional systems a secure element (the "dongle") is used to protect software. However, this approach suffers from several technical and economical drawbacks such as the dongle being lost or broken.

We present a system that provides such dongles as a cloud service, and more importantly, provides the required cryptographic material to control access to software functionality in real-time.

This system is developed as part of an ongoing nationally funded research project and is now entering a first trial stage with stakeholders from different industrial sectors.

CCS CONCEPTS

• CCS → Security and privacy → **Cryptography** • CCS → Security and privacy → Security services → **Access control** • CCS → Networks → Network services → **Cloud computing**

KEYWORDS

Software protection, cryptographic access control, cloud service

ACM Reference format:
Andreas Schaad, Björn Grohmann and Oliver Winzenried. 2019. CloudProtect – A cloud-based software protection service. In *Proceedings of the 24th ACM Symposium on Access Control Models and Technologies (SACMAT'19), June 3–6, 2019, Toronto, ON, Canada.* ACM, New York, NY, ACM. 3 pages. https://doi.org/10.1145/3322431.3326447

SACMAT '19, June 3–6, 2019, Toronto, ON, Canada

ACM ISBN 978-1-4503-6753-0/19/06…$15.00
https://doi.org/10.1145/3322431.3326447

1 INTRODUCTION

The protection of software, digital artefacts and intellectual property becomes continuously more challenging with the increasing interconnection of industrial components. Traditional approaches to protect software are the combination of encryption, code obfuscation and locally attached hardware trust anchors (often called "dongle") [1]. This dongle provides required cryptographic material to en- and decrypt application code and data at runtime of a system. At the same time access to protected functionality of the software can be controlled to implement commercial licensing models. This can be done as fine-grained as granting or denying access at method level of a program.

However, this approach does not necessarily scale (technically and economically) in physically and logically distributed systems. The question is whether it is possible to provide cryptographic dongle functionality as a set of cloud-based services.

2 Traditional approach

Cryptographic access to software can be controlled as fine-grained as to the individual method or function level (e.g. in Java, .Net, C++) by an independent software vendor (ISV). The ISV thus defines the commercial licenses for end users which are in turn reflected in a cryptographic keychain rooted at the ISV. Once an end user uses the software he bought, the locally attached (USB) dongle and its tamperproof internal cryptographic processor will provide the key material to access software (i.e. decrypt a required subset of code).

Communication between the software and the dongle is mediated through a daemon (the CodeMeter.dll) running in the background. The process to initially set up the dongle does require an activation process over a web portal where the end user enters an activation code. If this is valid a so-called license container (i.e. containing certificates and keys) is transferred to the locally attached dongle. Figure 3 shows such a locally attached dongle (CmStick/C) when opening the CodeMeter management console.

However, if this dongle is lost or damaged, the end user cannot perform his work until a new dongle has been provided. For that reason, most vendors of such solutions (including Wibu) provide "soft" dongles that store keys in hidden local memory regions. This, of course, significantly increases the attack surface.

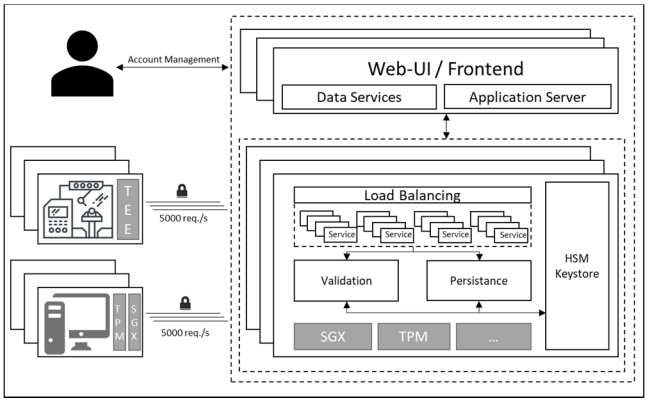

Figure 1: CloudProtect Architecture

The research question was thus how a license container could be offered as a real-time cloud service, feasible from a technical but also commercial perspective (i.e. regarding operational costs). Specifically, this last point is often neglected when offering a prior "on-premise" service as a cloud service. As an analogy, consider that in the traditional approach the customer provided the operational costs (e.g. electricity) when using an individual dongle. Now such costs are shifted towards the cloud operator. Although they are negligible for a single dongle, they add up when offering a service for thousands of end users in parallel.

3 Cloud Architecture

A CloudProtect server instance consists of three distinct layers:

- Load Balancing Layer
- Services Layer
- Date Storage Layer

The load balancer (for example, based on NGINX) distributes license validation requests that are received in high (parallel) frequency (Figure 1). At the moment, we assume a request every 10-15s per end user process. An average cloud server should serve 20 ISVs with each having 5000 active customers (end users). On average, we predict such a server to handle approx. 10.000 license (cryptographic access control) requests per second.

The service layer handles license management and corresponding generation and distribution of cryptographic keys.

The data layer uses a NoSQL DBMS for final persistency as well as an in-memory database for supporting the service layer.

An end user could also be a machine or service representing an IoT device. A client (end user) running protected software requires a proprietary daemon. At the moment, this is a separate program / daemon (CodeMeter.dll) that mediates the requests to the cloud instead of to the traditional hardware-based dongle.

The servers themselves are maintained in a professionally data center near a central internet exchange point. For larger corporate customers a server can be offered to run in an in-house cloud.

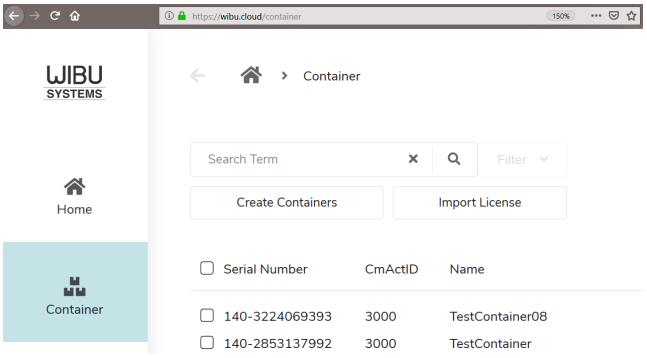

Figure 2: License administration portal for ISV

3.1 ISV perspective

The ISV will continue to use code obfuscation and encryption tools as part of his local software development process. The ISV will then use the cloud administration portal (Figure 2) under https://wibu.cloud to create so called license containers. A container does hold the actual licenses (cryptographic key material) which have also been created by the ISV. A local client process (the customer) will use credentials he received separately to access a cloud container over a secure channel and thus obtain the license material for evaluating an access request. This portal is based on an Angular [2] and Web Assembly [3] stack.

3.2 Customer perspective

Figure 3 shows that a customer now has access to a cloud license container ("TestConstainer"). More precisely, any access request to parts of the protected software is mediated by the local daemon which in turn requests the needed cryptographic keys from the cloud.

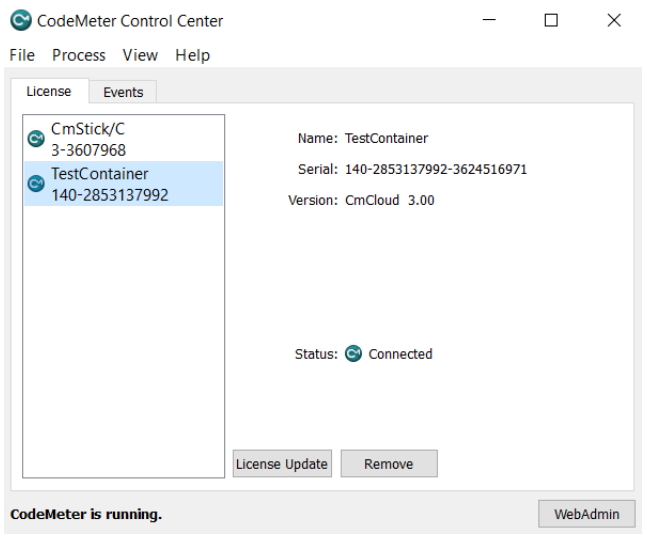

Figure 3: Local protection management

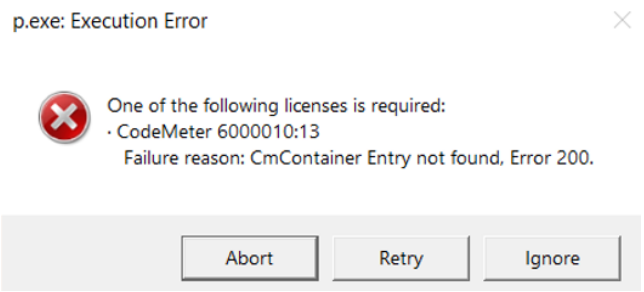

Figure 4: Protected Application

How often and at which level such requests are made can be configured as fine-grained as required (i.e. seconds). If a license is either missing or does not commercially allow to provide access to a specific function, the protected application will refuse to work (Figure 4).

4 Related Work

Software Protection has been scientifically discussed as early as [4], around the same time as Wibu-Systems offered the first commercial solutions as a printer port extension.

Oorschot later identified 4 approaches to software protection [5]: Obfuscation via automated code-transformation; white-box cryptography; Software Tamper Resistance; and Software Diversity. Attacks on obfuscated software [6] and the resulting improvements [7] are two competing disciplines and hardware supported isolated execution has been analysed extensively [8-11].

On the commercial side, there are vendors that already offer cloud-based license management [12]. Prominent services such as STEAM [13] also do, for example, offer the APIs which application developers use to enforce such access control checks. However, in both cases this is not true cryptographic software protection but rather a one-time access-control check based on a purchased license. The Steam Bind service does in fact offer cryptographic protection but has been reported to be broken [14].

5 Summary and Conclusion

We discussed the technical implementation of a cloud-based service to provide real-time cryptographic access control to software. To our knowledge we are the first to enable such real-time cryptographic access and license control as a cloud service.

This service can complement or fully replace traditional hardware-based dongles and enable new business models. We had reported on the initial requirements in [15] and now demonstrated practical feasibility. In a next step we need to prove economic feasibility of running such an access control service in a profitable fashion.

From a security perspective, we intend to address the integration of Intel's SGX [16] technology to further secure the interaction between the local protection service and the cloud.

ACKNOWLEDGMENTS

This work was partially supported by the German Ministery of Research and Education in the "CloudProtect" project under grant number (FKZ) 16KIS0850.

REFERENCES

[1] https://www.wibu.com/de/produkte/codemeter.html
[2] https://angular.io/
[3] https://webassembly.org/
[4] Kent, S. 1980 Protecting externally supplied software in small computers. Massachusetts Institute of Technology, Cambridge, MA, USA
[5] Oorschot, P. 2003 Revisiting Software Protection. 6th International Conference on Information Security.
[6] Rolles, R. 2009 Unpacking Virtualization Obfuscators. 3rd USENIX Workshop on Offensive Technologies (WOOT).
[7] Averbuch, A., Kiperberg, M., Zaidenberg, N. 2013. Truly-Protect: An Efficient VM-Based Software Protection. IEEE Systems Journal.
[8] Suh, G. E., O'Donnell, C. W., Devadas, S. 2007 Aegis: A Single-Chip Secure Processor. IEEE Design & Test of Computers.
[9] Costan, V., Lebedev, I., Devadas, S. 2016. Sanctum: Minimal Hardware Extensions for Strong Software Isolation. 25th USENIX Security Symposium (USENIX Security)
[10] Koeberl, P., Schulz, S., Sadeghi, A.-R. Varadharajan, V. 2014. TrustLite: A security architecture for tiny embedded devices. *EuroSys*.
[11] Strackx, R., Piessens, F., Preneel, B. 2010 Efficient isolation of trusted subsystems in embedded systems. Security and Privacy in Communication Networks.
[12] Flexera, 2018. https://www.flexera.com/products/software-license-optimization/flexnet-manager-cloud-infrastrucure.html
[13] Valve, 2018 https://developer.valvesoftware.com/wiki/ Steam_Web_API
[14] Steamless 2016 https://gitlab.com/atom0s/Steamless
[15] Andreas Schaad, Björn Grohmann, Oliver Winzenried, Ferdinand Brasser, Ahmad-Reza Sadeghi: Towards a Cloud-based System for Software Protection and Licensing. ICETE (2) 2018: 698-702.
[16] https://www.intel.de/content/www/de/de/architecture-and-technology/software-guard-extensions.html