Engineering Code Obfuscation

Christian Collberg

University of Arizona, Tucson, AZ, USA,
collberg@gmail.com
http://cs.arizona.edu/~collberg

In the *Man-at-the-end* (MATE) security scenario [2] a user (and potential adversary) has physical access to a device and can gain some advantage by extracting or tampering with an asset within that device. Assets can be data (cryptographic keys and media streams) as well as code (security checks and intellectual property). As defenders, our goal is to protect the confidentiality and integrity of these assets.

MATE scenarios are ubiquitous. Consider, for example, the *Advanced Metering Infrastructure* where *smart meters* are installed at house-holds to allow utility companies to connect and disconnect users and to monitor usage. A malicious consumer can tamper with their meter to avoid payment and may even be able to send disconnect commands to other meters [5]. In the mobile *Snapchat* application pictures exchanged between teenagers must be deleted a few seconds after reaching a friend's device. A malicious user can tamper with the application code to save the pictures and use them later for cyber bullying. Finally, in a massive multiplayer online game a malicious player can tamper with the game client to get an unfair advantage over other players [3].

Adversarial model. In a realistic MATE scenario we must assume that, since the adversary has physical control over his device, in time all assets will be compromised, and, at best, any defenses will be time-limited [4]. Even protection techniques based on tamper-resistant hardware have shown themselves susceptible to attack [1]. In particular, in analogy with *Kerckhoffs's principles*, we must assume an adversary who has complete understanding of our system, including its source code, and who can achieve in-depth understanding of the system through static and dynamic analyses using advanced reverse engineering tools.

Protection mechanisms and strategies. MATE protection mechanisms are typically based on the application of obfuscating code transformations that add complexity (for confidentiality) and/or the insertion of tamper-detecting guards (for integrity). Given that individual mechanisms provide limited protection, strategies have to be put in place to extend the in-the-wild survival time.

An important such strategy is *diversity*. *Spatial diversity* (or *defense-in-depth*) means compounding multiple layers of interchangeable primitive protective transformations. *Temporal diversity* (or *renewability*) means to deliver, over time, an infinite and non-repeating sequence of code variants to the user. The basic principle is that every program should be protected with a different combination of transformations, that every user/potential adversary should get a uniquely protected program, and that we will provide an ever-changing attack target to the adversary. In other words, we hope to provide long-term security by overwhelming the adversary's analytical abilities with randomized, unique, and varying code variants.

Evaluation and Benchmarking. MATE protection systems are evaluated on their resilience to attack and their performance, i.e. the increase in size and speed of a protected program over the original. Many real-world applications are interactive (such as the Snapchat and game examples above), and many are running on highly constrained devices (smart meters); performance is thus always of paramount concern.

Finding the combination of primitive transformations and spatial and temporal diversity strategies that achieve the highest level of protection while staying within strict performance bounds is an unsolved engineering problem. Part of the problem is a lack of behavioral models that express the capabilities and limitations of a human adversary, and part of the problem is a lack of universally accepted benchmarks.

Summary. We present an overview of the engineering challenges in providing long-term protection of applications that run under complete control of an adversary. In particular, we discuss the principle of diversity and the need for adversarial modeling and benchmarking.

Acknowledgments: This work was funded in part by NSF grants CNF-1145913 and CNS-1525820 and BSF grant 2008362.

References

1. Anderson, R., Kuhn, M.: Low cost attacks on tamper resistant devices. In: IWSP: International Workshop on Security Protocols (1997)
2. Collberg, C., Nagra, J.: Surreptitious Software: Obfuscation, Watermarking, and Tamperproofing for Software Protection. Addison-Wesley (Jul 2009)
3. Hoglund, G., McGraw, G.: Exploiting Online Games: Cheating Massively Distributed Systems. Addison-Wesley (2007)
4. Hohl, F.: Time limited blackbox security: Protecting mobile agents from malicious hosts. In: Mobile Agents and Security. pp. 92–113 (1998)
5. R.C., P.: Sandia report: Advanced metering infrastructure security considerations (Nov 2007)