Securing cryptographic assets for the Internet of Things

By Michael Mehlberg

THIS ARTICLE SURVEYS various white box cryptography techniques for protecting critical cryptographic operations and data in an environment where white-box attacks are available. We will review the need for white-box cryptography, describe the techniques and technologies behind a typical white-box cryptography implementation, review how white-box cryptography prevents attacks on critical cryptographic data and operations, and discuss important features in any white-box implementation. Finally, due to the expanding need for software cryptography combined with a rise in threats and attacks in the Internet of Things, we aim to conclude that white-box cryptography should be considered an essential technology for protecting cryptographic operations in any software systems.

The need for white box cryptography

The economic growth of the Internet of Things is unlike any other in recorded history. With estimates of over 200 billion connected devices by 2020, Internet-connected devices are influencing nearly every facet of modern life. The Internet of Things is impacting a multitude of markets from robotics to point-of-sale systems to mobile computing devices to 3D-printing. Embedded systems produced in these markets are helping to inform us, make autonomous decisions on our behalf, communicate with business associates and even manage our finances.

Access to data, information systems, and digital content on these systems is commonly protected by encryption. To protect encrypted information, it is imperative that the cryptography key used to encrypt such data is never revealed. Standard cryptography implementations leave both the algorithm and key vulnerable to tampering and reverse engineering: the single point of failure for any crypto system is the instance in which the key is used. This point is easily identifiable in modern systems using signature, pattern, and memory analysis. As an example, key extraction attacks against keys coded as literal data arrays in unprotected software can typically be successfully completed in a matter of hours.

White box cryptography overview

White-box cryptography is a well-described method for obfuscating a cryptographic algorithm such that the key material is sufficiently hidden from prying eyes. White-box cryptography aims to prevent critical information (such as a key) in cryptographic operations from being revealed to a would-be attacker with full access to the system.

The name "white-box cryptography" is derived from what is known as a white-box attack. As opposed to black-box attacks in which an attacker does not have access to system internals, a white-box attack is one in which the attacker has full access to the system, its memory, its software routines, etc. One can safely assume, as modern systems become more open and mobile (laptops, tablets, phones), they become more accessible and therefore vulnerable to white-box attacks without appropriate security measures.

```
White-Box AES Key:
0000000 0e18 80bc bae7 e250 708d ea28 04dd 9a18
0000010 f615 0d93 cf64 b9b6 7c5c 73be 2282 2176
0000020 d870 ade7 656b c188 48a4 cbe0 6ec6 9f1f
0000030 2c54 dd21 30f3 bdc7 d438 3b61 6850 8094
0000040 83e7 d907 57e8 db00 a39a 1ddb 59ec 7e29
0000050 6bae fd3d b2b0 604e edf7 98a3 c519 56c4
0000060 deb8 93c4 432d 4146 9fa6 5637 9d8e d7df
0000070 986e b925 d5a4 81ed c4c6 3778 9cc8 aa8b
0000080 8006 3b73 2a7e 87d9 3248 157c b5f6 f9b3
0000090 c30c 3a62 0dbe 5bb4 0cc7 f788 664e 2f69
00000a0 57d0 ad7d 0b70 1a92 b251 efb3 60c0 bdf4
00000b0 27d2 ebdf 916d dae5 c981 be66 667c c9cc
00000c0 2634 e17c 082d d0f8 338f 3e58 c9ee 3780
00000d0 2a01 9224 6d71 6344 66bb b037 5e96 2320
00000e0 13d7 d7aa 9f42 f210 5dfa 66b0 dc5b 070e
00000f0 a2dc 5fb3 7e53 bd5e 0830 e021 83cf 3764
0000100 e870 30a5 3320 8d0b aa3b f86a 3a75 e71c
0000110 5e85 84e8 1db4 6d82 0ee4 c64a 1bf7 2657

Based on the Classical 256-bit AES Key:
0000000 e502 d48a 18d7 95cd 5992 b8b0 d88b 65f1
0000010 78e8 264f 3652 bb4b fbb9 6802 c914 c4d0
```

Fig. 1: The relationship between the white-box and classical key is non-trivial making it impractical to reconstruct the classical key using the tools available to a network-based attacker.

A white-box algorithm is typically obscured such that access to or knowledge of the implementation doesn't compromise the key material, even during cryptographic operations. A typical white-box implementation of a cryptographic standard encrypts, decrypts, signs, and verifies sensitive data in the same way as a classical implementation, yet it forces the attacker to reverse engineer complex mathematical transformations to obtain the secret key.

Thus, white-box cryptography is useful wherever cryptography must be performed in a potentially vulnerable environment, where the crypto keys and/or plaintext data must be protected, or where an untrusted user could take control of the host system. Such use cases include compromise of networked systems, software delivered to business competitors, or commercially deployed software with private keys.

Preventing attacks with white box cryptography

One relevant example of a high-profile attack is the 2014 Heartbleed vulnerability that allowed an attacker to retrieve memory contents from vulnerable server-side software (namely OpenSSL) controlling security on a server. A properly constructed Heartbleed attack allows the memory contents to be retrieved

Michael Mehlberg is Vice President of Sales & Product Management at Microsemi - www.microsemi.com

which may contain portions of cryptographic key material used to secure communications between that server and the outside world. Exposing keys can lead to compromise the very (sensitive) data being protected by that secure communications channel.

To protect encrypted information, it is imperative that the key never reveals itself in memory or on disk. Standard crypto implementations (as were exploited in OpenSSL in the aforementioned Heartbleed attack) leave both the algorithm and key vulnerable to tampering and reverse engineering. White-box cryptography mathematically transforms the key into a complex graph of numbers and executable code. This graph has multiple valid paths randomly chosen at runtime based on a user-supplied random source.

Combining mathematical algorithms, data, and code obfuscation techniques to transform the key and related crypto operations in complex ways requires deep knowledge in multiple disciplines to attack. Importantly, the key is never present in static or runtime memory. Rather, the key becomes an inert collection of data that is useless without the uniquely generated white box algorithm. In short, replacing the standard cryptographic libraries with a white-box enabled library would never expose the keys, thereby preventing such attacks from ever being effective.

Important techniques in a white box implementation

White-box products and technologies vary from institution to institution. Naturally, certain features and techniques are stronger than others. The following techniques should be considered fundamental to any white-box implementation for use in a potentially vulnerable system:

Diversity

Rather than implementing a single white-box cryptography algorithm for all users (which would lead to break-once-run-everywhere attacks), code generators should be used to produce unique variants of the algorithms. This mitigates first pass observations of sensitive data (i.e., keys or selected plaintext). Uniquely, "tailored algorithms" also eliminate algebraic attacks that could easily unwind data protections that result from understanding a single standard implementation.

Algorithms should be implemented using alternate mathematical methods. White-box algorithms should not simply automate transformations of standard algorithms. Each algorithm/cipher should be modified in ways that leverage the specific properties of the underlying mathematics; blanket transformation should never be applied over all algorithms.

Hardware binding

Software is inherently easier to attack than hardware. By simply copying the original software system bit for bit, an attacker is guaranteed unlimited attempts to break the system. Hardware however can enforce more permanent penalties. A strong white-box cryptography implementation should take advantage of hardware when available to limit reverse engineering attempts on the obfuscated algorithm(s).

One such technique includes hardware binding. Cryptographically binding a hardware identifier to the white-box algorithm and/or data forces an attacker to reverse-engineer a complex, dynamically changing key-graph while tied to a single hardware system—a system that can enforce more permanent penalties should an intrusion be detected.

Side-channel resistance

Resistance against side-channel attacks (such as simple or differential power analysis) is paramount to protecting the key material from exposure. A solid white-box cryptography implementation should utilize numerous side-channel analysis countermeasures to resist exposing the key to such attacks.

Important obfuscations

Certain attacks against many cryptographic algorithms may yield well-known answers. Many times, standard cryptography algorithm designs result in implementations that have fundamental vulnerabilities to white-box attacks because they make an explicit assumption of executing on a secure host. A strong white-box implementation should not preserve these vulnerabilities.

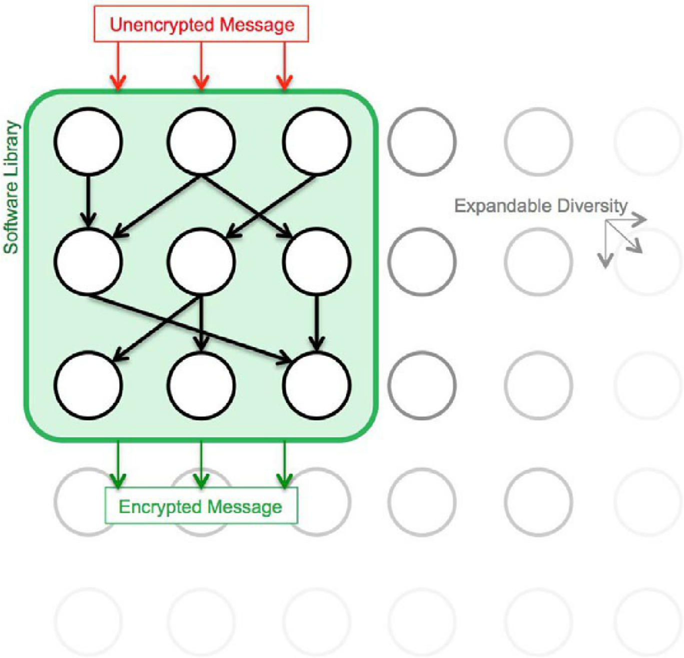

Fig. 2: A diverse white-box implementation should include randomization and multiple transformation obfuscations leveraging the underlying mathematics of the cryptography algorithm.

White-box obfuscations should prevent well-versed attackers from exposing the underlying mathematical principals of an algorithm to trick the algorithms into yielding an obfuscated version of a well-known answer. Additionally, obfuscations such as round boundary blurring should be employed to hide clear cut attack points that would compromise, as an example, an AES round.

Using white-box cryptography, keys are made unavailable to an attacker forcing them to go through the pain of reverse engineering complex and numerous combinations of obfuscation transformations with a detailed understanding of Abstract Algebra and Discrete Math.

Given the rise in mobile Internet connected devices combined with a growing need for secure operations and communications, a strong white-box cryptography implementation using (at a minimum) the techniques described above should be considered an essential component to any software system using cryptography.