International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Impact Factor (2012): 3.358

Review on Cipher X Ray Technique to Analyze and Implement Cryptographic Operation

Punam Hiwase[1], Sapna Khapre[2]

[1, 2]Department of Computer Science and Engineering
G H Raisoni Institute of Engineering and Technology for Womens, Nagpur, Maharashtra, India

Abstract: ***Identifying a given binary program implements a specific cryptographic algorithm and finding out more information about the cryptographic code is an important problem. In this paper, we present several methods to identify cryptographic primitives within a given binary programming an automated way. CipherXRay is a novel binary analysis framework that can automatically identify and recover the cryptographic operations. CipherXRay able to pinpoint the boundaries between the multiple rounds of cryptographic operation. It can further identifies certain operation modes(e.g., ECB, CBC, CFB) of the identified block cipher and tell whether the identified block cipher operation is encryption or decryption in certain cases. This results that current software implementations of cryptographic algorithms hardly achieve any secrecy if their execution can be monitored.***

Keywords: Binary analysis, Cryptographic Operations, Key Recovery, Transient secrete, Cryptographic Algorithm

1. Introduction

Analyzing a given binary program is a difficult task: an analyst typically needs to understand the assembly code and interpret it to draw meaningful conclusions from it. An analyst needs to manually identify the cryptographic algorithms and their usage to understand the malicious actions, which is typically time-consuming. If this task can be automated, a faster analysis of malware is possible, thus enabling security teams to respond quickly to emerging Internet threats [5]. To prevent in memory cryptographic secrets (e.g., key, IV) from being recovered by key searching tools (e.g., rsakeyfind), sophisticated malware can make the cryptographic secrets truly transient in memory by encrypting or destroying the secrets right after using them at runtime. The use of cryptographic algorithms and truly transient cryptographic secrets inside he malware binary executable imposes a key obstacle to effective malware analysis and defense [1].

In this paper, we present CipherXRay technique to analyze and implement cryptographic operation. CipherXRay technique can accurately pinpoint the boundaries of individual cryptographic operation from multiple rounds of cryptographic operations and recover the truly transient secrete. It can further identify certain modes of operation of block cipher.

CipherXRay is designed upon the avalanche effect, which refers to the desirable property of all cryptographic algorithms (e.g., public key cryptographic algorithms, symmetric Cryptographic algorithms, hash functions) such that a slight change (e.g., flipping a single bit) in the input would cause significant changes (e.g., half the output bits flip) in the output [1].

2. Literature Survey

An exhaustive literature review has been carried out related to the titled work to find out the current research. Abstracts of some of the most relevant research works are reported the following paragraph-

A. Dynamic Taint Analysis for Automatic Detection, Analysis, and Signature Generation of Exploits on Commodity Software

In this paper, James Newsom and Dawn Song propose dynamic taint analysis for automatic detection of overwrite attacks. This approach does not need source code or special compilation for the monitored program. To demonstrate this idea, TaintCheck mechanism is implemented. TaintCheck mechanism could improve automatic signature generation in several ways. TaintCheck produce no false positives for any of the many different programs that are tested. It monitors the execution of a program at a fine-grained level, TaintCheck can be used to provide additional information about the attack. TaintCheck is particularly useful in an automatic signature generation system, it can be used to enable semantic analysis based signature generation, enhance content pattern extraction based signature generation, and verify the quality of generated signatures[2].

B. Automatic Reverse Engineering of Data Structures from Binary Execution

In this paper, Zhiqiang Lin, Xiangyu Zhang, Dongyan Xu proposed a reverse engineering technique to automatically reveal program data structures from binaries. In this schema, REWARD technique is used which is based on dynamic analysis. REWARDS executes the binary, monitors the execution, aggregates and analyzes runtime information, and finally recovers both the syntax and semantics of data structures observed in the execution. REWARD involves backward type propagation and resolution procedure. Type sink is used which identify system calls, standard library calls, and type-revealing instructions. Reward can be applied many application like memory image forensic and binary vulnerability fuzz [4]. There were some limitations of this technique as follows:

a) REWARDS cannot achieve full coverage of data structures defined in a program.
b) A REWARD does not support the reverse engineering of kernel-level data structures.
c) A REWARD does not work with obfuscated code.

C. Towards Revealing Attackers Intent by Automatically Decrypting Network Traffic

In this paper, Noe Lutz proposed a system based on dynamic analysis. Malware analysis encompasses two types of analysis: static and dynamic. Static analysis uses reverse engineering technique to disassemble the malware and extract its feature. The dynamic analysis approach circumvents the problem of binary obfuscation by running the malware binary and extracting information from its execution rather than from its code. Encryption and decryption can be performs with various algorithm like AES, libgcrypt, Blowfish etc. on text file. The greatest limitation of this type of analysis is that it can be detected and evaded, rendering the analysis useless. Our design uses dynamic tainting techniques to track the memory that depends upon the encrypted input of the program. This technique effectively reduces the number of candidate memory locations that may contain the program's decrypted input [4].

D. Automated Identification of Crypto-graphic Primitives in Binary Programs.

An Analysts needs to manually identify the cryptographic algorithms and there usage to understand the malicious actions, which is typically time-consuming. If this task is automated, a faster analysis is possible. In this schema identifying the cryptographic primitives used by given binary program. Execution tracing, or simply tracing, is the process of analyzing a binary executable during runtime to generate a protocol that describes the instructions executed and the data accessed by the executable. For detecting cryptographic primitives used heuristic method. This paper having certain drawbacks, Dynamic analysis has the general constraint that if code is not executed, it cannot be analyzed. DBI framework Pin cannot handle all kinds of malicious software since the malware might detect the presence of the instrumentation code[5].

E. Cryptography in the Web: The Case of Cryptographic Design Flaws in ASP.NET

This paper discussed how the cryptography is misused in the security design of a large part of the Web. Highly efficient attacks that allow attackers to steal cryptographic secret keys and forge authentication tokens to access sensitive information. The attacks combine decryption oracles, unauthenticated encryptions, and the reuse of keys for different encryption purposes. This paper is the first to describe step-by-step how to use decryption oracles and CBC-R to compromise any application using the ASP.NET framework.

3. Proposed Work

Traditional key recovery attacks assume physical access to the cryptosystem (e.g., smart card), and they can use timing (e.g., timing attack) and power (e.g., differential power analysis) to recover the cryptographic secrets. Recently, researchers have investigated how to recover secret keys from memory offline and live applications. Most of these key recovery attacks depend on specific implementations, and they are not able to pinpoint the cryptographic operations. Therefore, they are unable to recover transient keys involved in multiple rounds of nested cryptographic operations. Mixing-based approaches are not able to pinpoint the boundary between multiple rounds of cryptographic operations thus they cannot recover the transient cryptographic secrets in between nested cryptographic operations.

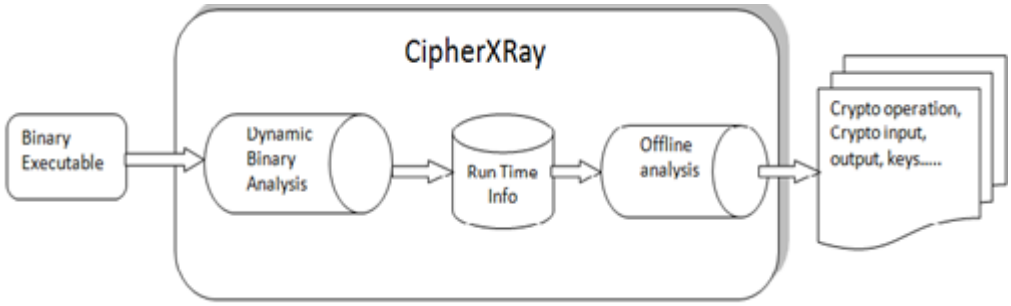

Figure a: CipherXRay Architecture

CipherXRay can accurately pinpoint the boundary of individual cryptographic operation from multiple nested cryptographic operations. This enables it to recover transient cryptographic secrets that only exist in between nested cryptographic operations.

CipherXRay is designed upon the avalanche effect, which refers to the desirable property of all cryptographic algorithms (e.g., public key cryptographic algorithms, symmetric Cryptographic algorithms, hash functions) such that a slight change (e.g., flipping a single bit) in the input would cause significant changes (e.g., half the output bits flip) in the output [1]. Another nice feature of the avalanche effect is that it allows us to accurately pinpoint the location, size and boundary of both the input and output buffer.

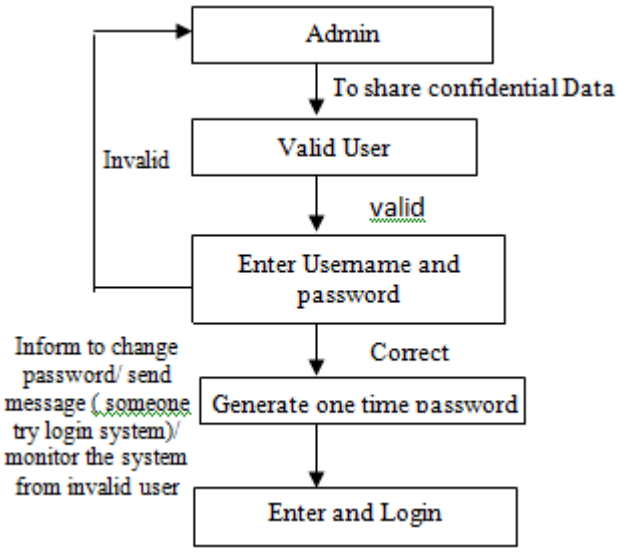

Figure b: Research plan for achieving security

CipherXRay is designed upon the avalanche effect, which refers to the desirable property of all cryptographic algorithms (e.g., public key cryptographic algorithms, symmetric Cryptographic algorithms, hash functions) such that a slight change (e.g., flipping a single bit) in the input would cause significant changes (e.g., half the output bits flip) in the output [1]. Another nice feature of the avalanche effect is that it allows us to accurately pinpoint the location, size and boundary of both the input and output buffers.

CipherXRay, a novel binary analysis framework that can automatically identify and recover the cryptographic operations and transient secrets from the execution of potentially obfuscated binary executable. There are some techniques used to achieve highly secrecy and protect confidential data from invalid user as follows-

1) **Hiding:** This technique is used to hide the data at different levels and grant access to a confidential attribute to user or group that need to read confidential data in the attribute.
2) **Hashing:** Producing hash values for accessing data or security. It is generated by a formula in such a way that it is extremely unlikely that some other text will produce the same hash value.
3) **Removal of Identity field:** Remove identity field to maintain more security to access confidential data.
4) **Anonmyzation:** It is the process of destroying tracks, or the electronic trail, on the data that would lead an eavesdropper to its origins.
5) **Permutation:** It relates to the act of permuting, or rearranging, members offset into a particular sequence or order

4. Conclusion

This approach may prove to be fast and efficient technique to analyze the characteristics of all cryptographic operations. It has been able to detect public key cryptography, block cipher, and hash operations and pinpoint exactly when and where the cryptographic input, output, and keys will be in the memory even if they exist for only a few microseconds .While this new capability helps better analyze sophisticated malwares protected by strong cryptographic algorithms.

References

[1] Xin Li, Xinyuan Wang, andWentao Chang, "CipherXRay: Exposing Cryptographic Operations and Transient Secrets from Monitored Binary Execution" IEEE transactions on dependable and secure computing, vol. 11, no. 2, march/april 2014
[2] J. Newsome and D. Song, "Dynamic Taint Analysis for Automatic Detection, Analysis, and Signature Generation of Exploits on Commodity Software," Proc. 12th Network and Distributed System Security Sump. (NDSS '05), Feb. 2005.
[3] Z. Lin, X. Zhang, and D. Xu, "Automatic Reverse Engineering of Data Structures from Binary Execution," Proc. 17th Network and Distributed System Security Symp. (NDSS 2010), Feb. 2010.
[4] N. Lutz, "Towards Revealing Attackers' Intent by Automatically Decrypting Network Traffic," master's thesis MA-2008-08, SwissFed. Inst. of Technology Zurich, 2008.
[5] F. Gro¨ bert, C. Willems, and T. Holz, "Automated Identification of Cryptographic Primitives in Binary Programs," Proc. 14th Int'l Symp.Recent Advances in Intrusion Detection (RAID '11), Sept. 2011.
[6] T. Duong and J. Rizzo, "Cryptography in the Web: The Case of Cryptographic Design Flaws in ASP.NET," Proc. IEEE Symp. Security & Privacy (S&P '11), pp. 481-489, May 2011.
[7] T. Wang, T. Wei, G. Gu, and W. Zou, "TaintScope: A Checksum- Aware Directed Fuzzing Tool for Automatic Software Vulnerability Detection," Proc. IEEE Symp. Security and Privacy (S&P '10),pp. 497-512, May 2010.