Security and Trusted Hardware Applications
Week #2 Tutorial

Pedro Antunes - up201507254

May 30, 2022

Instructor: Bernardo Portela

1

a)
- keys stored per user: N-1
- keys stored globally : $\frac{N(N-1)}{2}$
- Crypto technique example: AES-CBC, AES-CTR, RC4

b)
- keys stored per user: (N-1)+2
  N-1 == public-keys of others users,
  2 == public and private key of one user.
- keys stored globally : 2*N
- Crypto technique example: RSA-OAEP

c)
- keys stored per user: 1+2
  1 == public-key of trusted entity,
  2 == public and private key of one user
- keys stored globally : 1+2*N
- Crypto technique example: Diffie-Hellman

.

2

In this scenario, each person has to go to the physical space where the library exists and has to identify him/herself in order to perform book lending. Because of the portability of Java Cards, and the fact that we can associate a Java Card to a person securely, we can easily identify people on the spot using this technology. Then managing book lending would just be a stored user and book information in one Java Card.

This tecnhology also is independent from scanner hardware and its manufacturing has a low cost.

3

Both a Trusted Platform Module and a Hardware Security Module are used to perform cryptographic operations in an isolated way on a certain system and both are tamper-resistance.

However, a TPM is a microship and guarantees a root of trust for an operating system, i.e. a secure boot. Through a trusted setup, it calculates a hash of the entire OS (trusted hash) and when the system loads, it again calculates the hash and checks it against the trusted hash. This way we can be sure that the operating system was booted in a secure way.

In contrast, an HSM is an external hardware device and guarantees protection over cryptographic keys, i.e the keys are not dumped outside the context of the HSM. This technology is tamper-resistance, since it self-destructs the keys if it is under physical attacks of circuit breaking, etc.

4

In this scenario, using technologies like Intel SGX or ARM TrustZone is more fitting than using something like Hardware Security Modules because of feasibility. The application users will not carry around something like an HSM to perform authentication actions every single time that they want to do some tasks on application.

To choose between SGX or ARM we would have to know which architecture would be used by the users of the application, since SGX uses an Intel-based architecture and ARM TrustZone uses an Arm-based architecture.

5

Speculative execution is a technique created to improve and optimize CPU processing and performance by reducing memory delays. When the CPU is out of work, it starts executing instructions that it may or may not execute in the future. For example, an if-then-else instruction, both decision paths will be processed and when the path decision is ready, the right path instructions are already performed and the wrong path instructions are discarded.

Since the CPU is computing instruction from the wrong path, an attacker can access data from this wrong path because that data related to wrong path can go into cache and through this entry the attacker can know what kind of data is in the cache registers.

Eliminating dead code does not prevent these types of attacks, since this type of code is unreachable, and therefore eliminating this will not have real protective impact. These attacks target blocks of code that may be hit and processed, depending on a certain application state.

6

The Heartbleed Bug is a vulnerability in OpenSSL cryptographic software library that allows stealing the information protected on communication channel by the SSL/TLS encryption. This bug allows anyone to read the memory of the systems protected by the vulnerable versions of the OpenSSL software. The secret keys used to identify the service providers and to encrypt the traffic, the names and passwords of the users are crucial information compromised.

a) The OpenSSL library makes use of the Heartbeat Protocol and develop their own TLS Heartbeat Extension. A heartbeat message is a way of checking if a peer is still available without renegotiating the connection each time.

   The TLS heartbeat protocol works a bit like an echo command. To check if the connection peer is still active, you send an arbitrary test message and if the connection is alive, the message should be echoed back immediately and exactly. The actual heartbeat message is an SSL3 record that includes not just the message payload but also its length. To keep the connection alive, the test payload returned by the peer (typically a server) must have exactly

the same content and length as in the heartbeat request. This is where OpenSSL software failed to implement one of the checks specified in RFC 6520, Section 4:

- If the ***payload_length*** of a received ***HeartbeatMessage*** is too large, the received ***HeartbeatMessage*** ***MUST*** be discarded silently.

In the heartbeat extension as implemented in OpenSSL 1.0.1 through 1.0.1f, the server receives the heartbeat message and echoes back the requested payload with the requested length. However, the code does not check if the payload really is the same length as requested in the message. In other words, the server simply sends back a block of data from the start of the payload to the requested length. If the heartbeat message is well-formed, this is no problem, as the payload is always the same length as specified. But if the requested length exceeds the payload size, the server returns whatever is in memory after the local payload variable, thus leaking information. This is how the name Heartbleed came about – servers running vulnerable versions of OpenSSL could "bleed" sensitive data in heartbeat messages.

b) Since there is a wide range of software services using the OpenSSL library, such as web servers, email servers or VPNs, this vulnerability has been present in many systems, thus compromising their security. Ironically, this version was soon widely deployed on servers worldwide to increase security, as it added support for TLS 1.1 and 1.2 to eliminate vulnerabilities such as BEAST. The servers that have not upgraded their OpenSSL patches and have kept using the 0.9.8 through 1.0.0 branch are not vulnerable to this attack. Versions 1.0.1 (released in March 2012) through 1.0.1f(inclusive) are vulnerable. The bug was fixed on OpenSSL 1.0.1g released on 7th of April 2014.

   As of 20 May 2014, 1.5% of the 800,000 most popular TLS-enabled websites were still vulnerable to Heartbleed. As of 21 June 2014, 309,197 public web servers remained vulnerable. As of 23 January 2017, according to a report from Shodan, nearly 180,000 internet-connected devices were still vulnerable.

c) This answer is explained in the answer 6a). Since heartbeat message includes not just the message payload but also its length, there is no verification to check if the payload length is exactly the length defined in heartbeat message.

d) There is no design solution that can prevent systems from being compromised with this vulnerability or any others new ones that may arise. However, we should be aware that we should update our systems AND MUST proceed and perform tests that can detect anomalies in new versions of any library used in the system. Of course, before the updates are exposed in the production context.