Security of Identity Based Encryption - A Different Perspective

Priyanka Bose and Dipanjan Das
priyanka@cs.ucsb.edu,dipanjan@cs.ucsb.edu
Department of Computer Science
University of California Santa Barbara

June 11, 2017

Abstract

The de-facto standard for many encryption systems are considered to be chosen-ciphertext (CCA2) security. The case of Identity-Based encryption is no different from this. But, a closer look into these will reveal that most implementation of such encryption schemes are still vulnerable to malware attacks. Here, we investigate the relevance of such attacks in the context of Identity-Based cryptosystems. Precisely, we demonstrate an attack on Boney-Franklin CCA2 secure IBE and later on suggest a hybrid system that will be capable of preventing such kind of attacks.

1 Introduction

It is well accepted that an encryption algorithm is considered to be highly secure if it achieves CCA2 security. CCA2 security allows the adversary to obtain the decryption of chosen ciphertexts that may in turn aid him breaking the security of such a scheme. In recent times, the emergence of sophisticated and well crafted attacks show that even though the encryption schemes achieves CCA2 security, the adversary can still decrypt ciphertexts with the help of some intermediate computation or some bits of secret keys available to him. Though the attacks seem daunting enough from the perspective of a cryptographer, but they are entirely not new. This types of attacks reminded us about side channel analysis which typically targets the implementation aspects of a scheme. Also, a branch of cryptography named as Leakage resilient cryptography deals with such kind of attacks.

One immediate solution could be to keep the secret key in a tamper resistant hardware module. This may seem a plausible solution initially, but the the malwares *aka* Memory Scrapers has taken this thread one step further. It is a piece of data-harvesting malware as mentioned by VISA in [4] and it attacks the volatile memory, collects memory snapshots matching some specific patterns e.g it hooks the payments

processing system and collects the regular expression matching credit card format from a specific memory region. It even becomes more severe when the intermediate values leaked during the execution of an algorithm enable such malwares to decrypt the ciphertexts previously encrypted. Here, we demonstrate one such attack against Boney Franklin CCA2 secure IBE by taking into account the extra piece of information an adversary can obtain from memory scrapping. In this model, an adversary that mimics the behavior of memory scrapers is equipped with another oracle which we name as $\mathcal{O}_I$. $\mathcal{O}_I$ will provide him the intermediate values during the computation of an algorithm.

As a solution to circumvent such attacks we propose a 'Hybrid System' having a tamper-resistant 'Trusted Platform Module' (TPM) installed. A TPM is a piece of hardware which is capable of doing several cryptographic computations. A TPM is considered to be very secure and Secret key or its components are safeguarded inside TPM and the values computed in it are not available to the attacker. But, a TPM has limited memory and processing power. Therefore, we assume that our protocol is minimally executed within the TPM and still secure against such devastating attacks.

2 Chosen Ciphertext Security (CCA2)

CCA2 is the standard acceptable notion of security for public key encryption (PKE) schemes [1][5]. Therefore, the same has been adapted in IBE [2] to incorporate this stronger notion of security. In an IBE scheme, CCA2 security game (IND-ID-CCA) is first defined in [2]. An identity-based encryption scheme is said to be semantically secure against chosen ciphertext attack (IND-ID-CCA) if an PPT adversary $\mathcal{A}$ has an negligible advantage in the following game:

- **Setup**: The challenger $\mathcal{C}$ takes security parameter κ as input and runs the `Setup` algorithm. It provides $\mathcal{A}$ the system parameters `params` and keeps the master secret key (MSK) to itself.

- **Phase 1**: $\mathcal{A}$ makes queries $q_1, ..., q_n$ where query q_i is either of the following:

  - Extraction query $\langle \mathtt{ID}_i \rangle$: $\mathcal{C}$ runs `Extract` and generates the private key d_i corresponding to the public key $\mathtt{ID}_i$. It gives d_i to $\mathcal{A}$.
  - Decryption query $\langle C_i, \mathtt{ID}_i \rangle$: $\mathcal{C}$ first runs `Extract` to generate the private key d_i for $\langle \mathtt{ID}_i \rangle$ and using d_i, it runs the algorithm `Decrypt` to decrypt the ciphertext C_i. It sends the message to $\mathcal{A}$.

  These queries may be adaptive, *i.e.* each query q_i may depend on the previous replies to $q_1, ..., q_{i-1}$

- **Challenge**: Once $\mathcal{A}$ decides that Phase 1 training is over, it outputs two messages $M_0, M_1 \in \mathcal{M}$ of equal length and an identity $\mathtt{ID}^*$ which it wishes to be challenged on. The only constraint is that $\mathtt{ID}^*$ did not appear in any of the private key extraction queries in Phase 1. Then $\mathcal{C}$ randomly chooses $\gamma \in_R \{0, 1\}$

and sets $C^* = \texttt{Encrypt}(\texttt{params}, \texttt{ID}^*, M_\gamma)$. It sends the challenge ciphertext C^* to $\mathcal{A}$.

- **Phase 2**: $\mathcal{A}$ asks more queries $q_{n+1}, ...q_m$ where q_i is one of the following:
  - Extraction query $\langle \texttt{ID}_i \rangle$: This queries are same as Phase 1 queries and they may be asked adaptively. Only constraint is that $\texttt{ID}^* \neq \texttt{ID}_i$.
  - Decryption query $\langle C_i, \texttt{ID}_i \rangle$: This queries are similar to Phase 1 queries except $\langle C_i, \texttt{ID}_i \rangle \neq \langle C^*, \texttt{ID}^* \rangle$.
- **Guess**: At the end, $\mathcal{A}$ outputs $\gamma' \in \{0, 1\}$. $\mathcal{C}$ outputs 1 if $\gamma = \gamma'$ ($\mathcal{A}$ wins the game), else outputs 0.

We define adversary $\mathcal{A}$'s advantage against the security of an IBE scheme as:

$$Adv_{\mathcal{A}}(\kappa) = |Pr[\gamma = \gamma'] - \frac{1}{2}|$$

3 Our Proposed Security Model

Adapting [6] to the IBE framework we replace the blackbox decryption oracle as present in CCA2 game by the intermediate value oracle $\mathcal{O}_I$.

- **Setup**: This is similar to the regular CCA game.
- **Phase 1**:
  - **Extraction query** is same as that of CCA2 game.
  - **Decryption query** The adversary will get access to the $\mathcal{O}_I$ oracle instead of regular decryption oracle.
- **Challenge**: $\mathcal{C} \xleftarrow[|M_0|=|M_1|, \texttt{ID}^* \neq \texttt{ID}_i]{M_0, M_1, \texttt{ID}^*} \mathcal{A}$

  $\mathcal{C} \xrightarrow[\gamma \in_R \{0,1\}]{C^* = \texttt{Encrypt}(M_\gamma, \texttt{ID}^*)} \mathcal{A}$
- **Phase 2**:
  - **Extraction query** $\langle \texttt{ID}_i \rangle$: $\mathcal{C} \xrightarrow[\texttt{ID}^* \neq \texttt{ID}_i]{d_i = \texttt{Extract}(\texttt{ID}_i)} \mathcal{A}$
  - **Decryption query** $\langle C_i, \texttt{ID}_i \rangle$: $\mathcal{C} \xrightarrow[\langle C_i, \texttt{ID}_i \rangle \neq \langle C^*, \texttt{ID}^* \rangle]{\mathcal{I}_i = \texttt{Decrypt}^{\mathcal{O}_I}(C_i, d_i)} \mathcal{A}$
- **Guess**: $\mathcal{C} \xleftarrow[\gamma' \in_R \{0,1\}]{\gamma \stackrel{?}{=} \gamma'} \mathcal{A}$

4 Attack on Boneh-Franklin IBE

In this section we first look into Boney-Franklin's [2] CCA2 secure IBE and mount an attack by an adversary equipped with $\mathcal{O}_I$ oracle, despite all secret key involving computations being done in a tamper resistant hardware module.

4.1 Boneh-Franklin CCA Secure IBE

The construction of FULLIDENT achieves CCA security by applying a transformation due to Fujisaki-Okamoto [3] on Boneh-Franklin's CPA secure IBE scheme (BASICIDENT) [2]. We split up the algorithm in a hybrid system such a way that all the computation involving secret key is performed in TPM and rest of them in RAM.

- **Setup**(κ): PKG run this algorithm to set up system parameters and the `master secret key` (`MSK`). It takes the security parameter κ as input and works as follows:
  - $\langle e, \mathbb{G}_1, \mathbb{G}_2, p\rangle \leftarrow \mathcal{G}(\kappa)$ where both the groups $\mathbb{G}_1$ and $\mathbb{G}_2$ are of prime order p and the bilinear map is defined as $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$
  - Pick $P_1 \in \langle \mathbb{G}_1 \rangle$
  - Choose `MSK` $s \in_R \mathbb{Z}_p^*$
  - Set`master public key` $P_{pub} = sP_1$
  - Select the following:
    - $H_1 : \{0,1\}^* \rightarrow \mathbb{G}_1^*$
    - $H_2 : \mathbb{G}_2 \rightarrow \{0,1\}^m$ for some $m \in \mathbb{Z}^+$
    - $H_3 : \{0,1\}^m \times \{0,1\}^m \rightarrow \mathbb{Z}_p^*$
    - $H_4 : \{0,1\}^m \rightarrow \{0,1\}^m$

    The message space $\mathcal{M} = \{0,1\}^m$ and ciphertext space $\mathcal{C} = \langle \mathbb{G}_1 \times \{0,1\}^m \times \mathbb{G}_1 \times \mathbb{G}_1 \rangle$
  - Publish `param` $= \langle e, \mathbb{G}_1, \mathbb{G}_2, p, P_1, P_{pub}, H_1, H_2, H_3, H_4\rangle$
- **KeyGen**(`ID`, `param`): For any arbitrary identity `ID` $\in \{0,1\}^*$ associated to an user, the PKG generates the private key for that identity in the following way:
  - $Q_{\texttt{ID}} = H_1(\texttt{ID})$
  - private key $d_{\texttt{ID}} = sQ_{\texttt{ID}}$
- **Encrypt**(m, `ID`, `param`): The sender encrypts a message $m \in \mathcal{M}$ under receiver's public key `ID`:
  - Compute $Q_{\texttt{ID}} = H_1(\texttt{ID})$
  - Choose $\sigma \in_R \{0,1\}^m$

- Set $r = H_3(\sigma, m)$
- Compute $g_{\text{ID}} = e(Q_{\text{ID}}, P_{pub})$
- Set $C_1 = rP_1$
- Set $C_2 = \sigma \oplus H_2(g_{\text{ID}}^r)$
- Set $C_3 = m \oplus H_4(\sigma)$
- Ciphertext $C = \langle C_1, C_2, C_3 \rangle$

- **Decrypt**$(C, d_{\text{ID}}, \texttt{param})$: To decrypt the ciphertext $C = \langle C_1, C_2, C_3 \rangle$ the receiver runs this algorithm in the hybrid platform (TPM/RAM) and recovers message m. Here, private key d_{ID} resides in TPM. Computations, as done in RAM/TPM, are marked accordingly.
  - RAM: If $C_1 \notin \mathbb{G}_1^*$, then **ABORT**
  - TPM: Compute $\alpha = e(d_{\text{ID}}, C_1)$
  - Send TPM $\xrightarrow{\alpha}$ RAM
  - RAM: Compute $\sigma = C_2 \oplus H_2(\alpha)$
  - RAM: Compute $m = C_3 \oplus H_4(\sigma)$
  - RAM: Compute $r = H_3(\sigma, m)$
  - RAM: If $C_1 \neq rP_1$, then **ABORT**
  - Output m as decryption of C

4.2 Attack on BF IBE Scheme

We let the adversary $\mathcal{A}$ play CCA2 security game with the challenger $\mathcal{C}$ and show that how leaking intermediate values enable him to win the CCA2 security game against the challenger. We assume here that the secret key is inside the TPM.

- **Phase 1**: This phase is similar to standard CCA2 game i.e $\mathcal{A}$ makes key extraction and decryption queries.
- **Challenge**: $\mathcal{A}$ sends two messages $\langle m_0, m_1 \rangle$ and an identity $\texttt{ID}^*$ to the challenger $\mathcal{C}$. $\mathcal{C}$ chooses $\delta \in_R \{0, 1\}$, encrypts m_δ under $\texttt{ID}^*$ and sends the ciphertext $C^* = \langle C_1^*, C_2^*, C_3^* = m_\delta \oplus H_4(\sigma^*) \rangle$ back to $\mathcal{A}$.
- **Phase 2**: $\mathcal{A}$ queries the intermediate value oracle $\mathcal{O}_I$ for the decryption of $C' = \langle C_1' = C_1^*, C_2' = C_2^*, C_3' \in_R \{0, 1\}^m \rangle$ under the challenge identity $\texttt{ID}^*$ itself. $\mathcal{C}$ hands over intermediate values $\mathcal{I} = \{\alpha', H_2(\alpha'), \sigma', H_4(\sigma'), m', r'\}$ for some arbitrary message m'. Since, $C_1' = C_1^*$ and $C_2' = C_2^*$, evidently $\sigma' = \sigma^*$. $\mathcal{A}$ can trivially recover the challenge message by computing $m_\delta = C_3^* \oplus H_4(\sigma^*)$, thus identifying bit δ always. Thereby, winning the CCA2 game.

5 Our Solution - Attack Resilient IBE Scheme

- **Setup**(κ):
  - Pick $P_1 \in \langle G_1 \rangle$, $Y, Z \leftarrow \mathbb{G}_1$, $s \leftarrow \mathbb{Z}_p$;
  - `master public key` $P_{pub} := sP_1$
  - `param` $:= \langle e, \mathbb{G}_1, \mathbb{G}_2, p, P_1, P_{pub}, H_1, H_2, H_3, H_4, Y, Z \rangle$
  - master secret key $:= s$
  - Select cryptographic hash functions:
    - $H_1 : \{0,1\}^* \rightarrow \mathbb{G}_1$
    - $H_2 : \mathbb{G}_2 \rightarrow \{0,1\}^m$ for some $m \in \mathbb{Z}^+$
    - $H_3 : \mathbb{G}_1 \times \{0,1\}^m \times \mathbb{G}_1 \rightarrow \mathbb{Z}_p$
    - $H_4 : \mathbb{G}_2 \times \mathbb{G}_1 \rightarrow \mathbb{Z}_p$
  - The message space $\mathcal{M} \in \{0,1\}^m$ and ciphertext space $\mathcal{C} \in \langle \mathbb{G}_1 \times \{0,1\}^m \times \mathbb{G}_1 \times \mathbb{G}_1 \rangle$
  - *return* (`param`, master secret key)

- **KeyGen**(`ID`, `param`, master secret key) :: For any arbitrary identity `ID` $\in \{0,1\}^*$ associated to an user, below are the steps that PKG performs to generate the private key for that identity:
  - $Q_{\texttt{ID}} := H_1(\texttt{ID})$
  - $r \leftarrow \mathbb{Z}_p$
  - $d_{\texttt{ID1}} := rsQ_{\texttt{ID}}, d_{\texttt{ID2}} := r^{-1}$
  - $d_{\texttt{ID}} := \langle d_{\texttt{ID1}}, d_{\texttt{ID2}} \rangle$
  - return $d_{\texttt{ID}}$

- **Encrypt**(m, `ID`, `param`): The sender executes the algorithm below to encrypt a message $m \in \mathcal{M}$ under public key `ID`:
  - $u \leftarrow \mathbb{Z}_p$, $X \leftarrow \mathbb{G}_1$
  - $C_1 := uP_1$
  - $Q_{\texttt{ID}} := H_1(\texttt{ID})$, $g_{\texttt{ID}} := e(Q_{\texttt{ID}}, P_{pub})$
  - $C_2 := m \oplus H_2(g_{\texttt{ID}}^u)$
  - $t := H_3(C_1, C_2, Q_{\texttt{ID}})$
  - $h := H_4(e(tP_1 + X, C_1))$
  - $C_3 := u(hY + Z)$
  - $C_4 := X$
  - $C := (C_1, C_2, C_3, C_4)$

– *return* C

- **Decrypt**$(C, d_{\mathtt{ID}}, \mathtt{param})$: The receiver runs this algorithm on a hybrid platform to decrypt the ciphertext $C = \langle C_1, C_2, C_3, C_4 \rangle$ to recover message m. Private key component $d_{\mathtt{ID1}}$ resides in RAM while $d_{\mathtt{ID2}}$ resides in TPM. Computations, as done in RAM/TPM, are marked accordingly.

  – parse $C = (C_1, C_2, C_3, C_4)$
  – RAM: $Q_{\mathtt{ID}} := H_1(\mathtt{ID})$
  – RAM: $t := H_3(C_1, C_2, Q_{\mathtt{ID}})$
  – RAM: $h := H_4(e(tP_1 + C_4, C_1))$
  – RAM: **if** $e(C_3, P_1) = e(hY + Z, C_1)$
  – RAM: $\alpha := e(d_{\mathtt{ID1}}, C_1)$
  – RAM: $\xrightarrow{\alpha}$ TPM
  – TPM: $\beta := (\alpha)^{d_{\mathtt{ID2}}}$
  – TPM: $\xrightarrow{\beta}$ RAM
  – RAM: $m := C_2 \oplus H_2(\beta)$
  – return m
  – **else ABORT**

6 Conclusions

Here, we have defined a new security model for identity based cryptosystems and shown an attacks on one of the CCA2 secure popular Boney-Franlin's IBE . We have also seen that the notion we introduced here, can perfectly model the security threats posed by memory scraper type of malwares. Thus several CCA2 secure protocols can be proven to be vulnurable under such kind of attacks. At the end, we propose a hybrid system consisting of minimal tamper resistant hardware. Such systems may be of immense useful to thwart, mitigate or prevent such state-of-the-art malware attacks.

References

[1] Mihir Bellare, Anand Desai, David Pointcheval, and Phillip Rogaway. Relations among notions of security for public-key encryption schemes. In *Advances in Cryptology - CRYPTO '98*, volume 1462 of *LNCS*, pages 26–45. Springer Berlin Heidelberg, 1998.

[2] Dan Boneh and Matt Franklin. Identity-based encryption from the weil pairing. In *Advances in Cryptology - CRYPTO 2001*, volume 2139 of *LNCS*, pages 213–229. Springer Berlin Heidelberg, 2001.

[3] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. *Journal of Cryptology*, 26(1):80–101, 2013.

[4] Visa Inc. Visa data security alert, debugging software memory parsing vulnerability, 2008.

[5] Charles Rackoff and DanielR Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *Advances in Cryptology - CRYPTO '91*, volume 576 of *LNCS*, pages 433–444. Springer Berlin Heidelberg, 1992.

[6] S. Sree Vivek, S. Sharmila Deva Selvi, and C. Pandu Rangan. Stronger public key encryption schemes withstanding ram scraper like attacks. *Cryptology ePrint Archive, Report*, 2012.