Stronger Public Key Encryption Schemes

Withstanding RAM Scraper Like Attacks

Prof. C.Pandu Rangan

Professor,
Indian Institute of Technology - Madras,
Chennai, India-600036.

Adaptive Chosen Ciphertext Attack (CCA2)

- **Setup** - Challenger $\mathcal{C}$ runs $(sk, pk) \leftarrow \texttt{KeyGen}(\kappa)$.
- **Query Phase I** - Adversary $\mathcal{A}$ is given access to $\mathcal{O}_{\texttt{Enc}_{pk}(.)}$ and $\mathcal{O}_{\texttt{Dec}_{sk}(.)}$.
- **Challenge Phase** - $\mathcal{A}$ produces two messages m_0 and m_1 to $\mathcal{C}$. $\mathcal{C}$ chooses $b \in_R \{0, 1\}$ and returns the challenge ciphertext $c^* = \texttt{Enc}_{pk}(m_b)$.
- **Query Phase II** - Same as Query Phase I, except that $\mathcal{A}$ cannot query the decryption of c^*.
- **Guess** - $\mathcal{A}$ outputs b'.

We define the advantage of an adversary in the IND-CCA2 security game to be

$$Adv_{Adversary} = |2Pr[b' = b] - 1|$$

We say that an encryption scheme is IND-CCA2 secure if for any polynomial time adversary,

$$Adv_{Adversary} = negl(\kappa)$$

Motivation for the NEW Security Model

RAM Scrapers

- *RAM Scraper* is a piece of malware.
- It grabs data residing in a systems volatile memory.
- Added to the list of **Top Data Breach Attacks by Verizon Business**.
- In one instance the RAM scraper dumped the card data to a .dll in a Windows system subdirectory.
- It waited for retrieval by the scraper's owners. [From InfoSec News - Attack of the RAM Scrapers, By Keith Ferrell]

Hybrid Computing Environment Using TPM

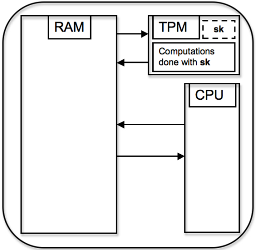

Figure: System with a TPM

- The private key of a user will be stored in TPM.
- The computations involving private keys will be carried out in TPM.
- The private key values will not be *moved to* the RAM.
- Some of the values generated by TPM may be sent to RAM
- All values in the RAM are available to the Adversary. (Values generated in untrusted environment as well as the values sent by TPM to RAM)
- This scenario can be modelled exactly with **Glass Box** decryption.

The NEW Security Model

CCA2 Security Under Glass box Decryption

- **Setup** - Challenger $\mathcal{C}$ runs $(sk, pk) \leftarrow \texttt{KeyGen}(\kappa)$.
- **Query Phase I** - Adversary $\mathcal{A}$ is given access to $\mathcal{O}_{\texttt{Enc}_{pk}(.)}$ and $\mathcal{O}_{\texttt{GlassBoxDec}_{sk}(.)}$.
- **Challenge Phase** - $\mathcal{A}$ produces two messages m_0 and m_1 to $\mathcal{C}$. $\mathcal{C}$ chooses $b \in_R \{0, 1\}$ and returns the challenge ciphertext $c^* = \texttt{Enc}_{pk}(m_b)$.
- **Query Phase II** - Same as Query Phase I, except that $\mathcal{A}$ cannot query the Glass Box Decryption of c^*.
- **Guess** - $\mathcal{A}$ outputs b'.

We define the advantage of an adversary in the IND-CCA2 security game to be

$$Adv_{\mathcal{A}} = |2Pr[b' = b] - 1|$$

We say that an encryption scheme is IND-CCA2 secure under glass box decryption, if for any polynomial time adversary,

$$Adv_{\mathcal{A}} = negl(\kappa)$$

Intuition Behind Glass Box Decryption Scheme

Usual flow in Decryption:

- Use the private key to retrieve some values from the ciphertext.
- Verify the validity of the constructed plaintext.
- The decryption oracle returns either the constructed value or NULL.

Decryption supporting Glass Box:

- Verify the validity of ciphertext.
- If valid, retrieve the potential plaintext, else "ABORT".
- If the potential plaintext passes some validity test, return the same, else "ABORT".

Remark

If we do this way, it allows a convenient partitioning of computations between trusted and untrusted parts of the system

Keeping this in mind we design a new scheme.

Glass box Vulnerability in an Implementation of Cramer Shoup (CS) Cryptosystem

Vulnerability in an Implementation of CS

The Cramer-Shoup encryption scheme

- $\mathcal{CS}.\texttt{Gen}$: The private key and public key of a user are $sk = (x_1, x_2, y_1, y_2, z_1, z_2)$ and public key $pk = (g_1, g_2, c, d, h)$, where $c = g_1^{x_1} g_2^{x_2}$, $d = g_1^{y_1} g_2^{y_2}$ and $h = g_1^{z_1} g_2^{z_2}$.
- $\mathcal{CS}.\texttt{Enc}$: Compute $u_1 = g_1^r$, $u_2 = g_2^r$, $e = h^r m$, $\alpha = H(u_1, u_2, e)$ and $v = c^r d^{r\alpha}$. $C = \langle u_1, u_2, e, v \rangle$.
- $\mathcal{CS}.\texttt{Dec}$: We do not perform any computation which involves the secret key outside the TPM in the implementaion. Still we are able to mount glass box attack on the implementation. On receiving a ciphertext $C = \langle u_1, u_2, e, v \rangle$ decryption is done as follows:

Vulnerability in an Implementation of CS

Conventional System:

- Compute $\alpha = H(u_1, u_2, e)$.
- Compute $V = u_1^{x_1} u_2^{x_2} (u_1^{y_1} u_2^{y_2})^{\alpha}$.
- If $(v = V)$ then,
  - Compute $Z = u_1^{z_1} u_2^{z_2}$.
  - Compute $m = e/Z$
  - Return m.

  Else *ABORT*

Hybrid System:

- NC: Compute $\alpha = H(u_1, u_2, e)$.
- RAM→TPM: $\langle \alpha, u_1, u_2 \rangle$
- SC: Compute $V = u_1^{x_1} u_2^{x_2} (u_1^{y_1} u_2^{y_2})^{\alpha}$.
- TPM→RAM: V
- NC: If $(v = V)$ then,
  - SC: Compute $Z = u_1^{z_1} u_2^{z_2}$.
  - TPM→RAM: Z
  - NC: Compute $m = e/Z$ and return m.

  Else *ABORT*

Vulnerability in an Implementation of CS

Consider the glass box execution of Decryption oracle on a ciphertext (u_1, u_2, e, v),

(a) Since all these are inputs, they are visible/available to the adversary.

(b) In the evaluation of the expression $\alpha = H(u_1, u_2, e)$ all values will be available to the adversary.

(c) The expression $V = u_1^{x_1} u_2^{x_2} (u_1^{y_1} u_2^{y_2})^{\alpha}$ is evaluated using the TPM because this involves secret keys x_1, x_2, y_1, y_2.

(d) Thus, u_1, u_2 and α are sent to the TPM and $V = u_1^{x_1} u_2^{x_2} (u_1^{y_1} u_2^{y_2})^{\alpha}$ is sent to the normal world. Thus the adversary gets V.

(e) The check $(v \stackrel{?}{=} V)$ is done outside the TPM. If this fails the adversary gets no further values. If $(v = V)$ is true, then $Z = u_1^{z_1} u_2^{z_2}$ is computed in TPM and Z is sent out. Now, the adversary obtains the values Z and $m = e/Z$ as well.

(f) Therefore, the set $\mathcal{I}$ of values returned by decryption oracle is given by $\mathcal{I} = \langle \alpha, V, -, - \rangle$ if the test fails and $\mathcal{I} = \langle \alpha, V, Z, m \rangle$ when the test succeeds.

Vulnerability in an Implementation of CS

The idea behind the attack is:

- Use the training in Phase II of CCA2 game to obtain the values $\langle u_1^{*x_1}, u_2^{*x_2}, u_1^{*y_1}, u_2^{*y_2} \rangle$.
- Use the above values to construct a valid ciphertext for $\hat{m}m_\delta$, where $\hat{m}$ is chosen by the adversary.
- Pass this to decryption oracle, obtain $\hat{m}m_\delta$, from which obtain m_δ.

Vulnerability in an Implementation of CS

We will show how an adversary distinguishes the challenge ciphertext.

- During the challenge phase $\mathcal{A}$ selects two messages $\{m_0, m_1\}$ and sends them to $\mathcal{C}$.
- Now, $\mathcal{C}$ constructs the challenge ciphertext C^* as $C^* = \langle u_1^*, u_2^*, e^*, v^* \rangle = \langle u_1, u_2, (u_1)^{z_1}(u_2)^{z_2} m_\delta, (u_1)^{x_1}(u_2)^{x_2}$ $((u_1)^{y_1}(u_2)^{y_2})^\alpha \rangle$, where δ is a random bit $\in \{0, 1\}$ and $\alpha = H(u_1^*, u_2^*, e^*)$.
- The challenger sends C^* to $\mathcal{A}$ and asks him to find the m_δ hidden in C^*.

Vulnerability in an Implementation of CS

In the *second phase* of the training $\mathcal{C}$ must respond to all legal queries raised by $\mathcal{A}$. This is what $\mathcal{A}$ asks to find m_δ.

- $\mathcal{A}$ chooses $s_1 \in_R \mathbb{Z}_q^*$ and constructs a ciphertext $C' = \langle u_1', u_2', e', v' \rangle = \langle (u_1^*)^{s_1}, (u_2^*)^{s_1}, e^*, v^* \rangle$, where u_1^* and u_2^* are the first two components of C^*. In other words C' is nothing but C^* with the first two components, namely u_1^* and u_2^* exponentiated with s_1.
- Now, $\mathcal{A}$ queries `Glass-Box-Dec`(C'). Note that it is legal to ask the decryption of C'.
- As $\mathcal{C}$ knows all the private keys, it would faithfully execute the $\mathcal{CS}$.`Dec` on C'.
- $\mathcal{C}$ will reject the ciphertext C' because $v' \neq (u_1')^{x_1}(u_2')^{x_2}((u_1')^{y_1}(u_2')^{y_2})^{\alpha_1}$.

Vulnerability in an Implementation of CS

- Now, $\mathcal{I}{=}\langle \alpha_1, V_1, Z, m\rangle$
$$= \langle H(u'_1, u'_2, e'), (u_1^*)^{s_1 x_1}(u_2^*)^{s_1 x_2}((u_1^*)^{s_1 y_1}(u_2^*)^{s_1 y_2})^{\alpha_1}, -, -\rangle$$
- Similarly, $\mathcal{A}$ constructs another ciphertext C'' by choosing $s_2 \in_R \mathbb{Z}_q^*$, computing $u''_1 = (u_1^*)^{s_2}$, $u''_2 = (u_2^*)^{s_2}$, $e'' = e^*$ and $v'' = v^*$. The newly formed ciphertext is $C'' = \langle u''_1, u''_2, e'', v''\rangle$ $\mathcal{A}$ queries `Glass-Box-Dec`(C'').
- $\mathcal{C}$ will reject C'' because it is invalid.
- Here, $\mathcal{I}{=}\langle \alpha_2, V_2, Z, m\rangle$
$$= \langle H(u''_1, u''_2, e''), (u_1^*)^{s_2 x_1}(u_2^*)^{s_2 x_2}((u_1^*)^{s_2 y_1}(u_2^*)^{s_2 y_2})^{\alpha_2}, -, -\rangle$$

Vulnerability in an Implementation of CS

We will now show that with the values V_1 and V_2, $\mathcal{A}$ performs the following and obtains m_δ:

- Computes $X_1 = V_1^{s_1^{-1}} = (u_1^*)^{x_1}(u_2^*)^{x_2}((u_1^*)^{y_1}(u_2^*)^{y_2})^{\alpha_1}$ and $X_2 = V_2^{s_2^{-1}} = (u_1^*)^{x_1}(u_2^*)^{x_2}((u_1^*)^{y_1}(u_2^*)^{y_2})^{\alpha_2}$.
- Computes $Y = \frac{X_1}{X_2} = ((u_1^*)^{y_1}(u_2^*)^{y_2})^{\alpha_1 - \alpha_2}$.
- Computes $Z_2 = Y^{(\alpha_1 - \alpha_2)^{-1}} = (u_1^*)^{y_1}(u_2^*)^{y_2}$.
- Computes $Z_1 = \frac{X_1}{Z_2^{\alpha_1}} = (u_1^*)^{x_1}(u_2^*)^{x_2}$.
- Generates a fresh ciphertext by computing $\hat{u}_1 = u_1^*$, $\hat{u}_2 = u_2^*$, $e = e^*\hat{m}$ and $\hat{v} = Z_1 Z_2^{\hat{\alpha}}$, where $\hat{m}$ is an arbitrary message chosen by $\mathcal{A}$ and $\hat{\alpha} = H(\hat{u}_1, \hat{u}_2, e)$.

Vulnerability in an Implementation of CS

- Now, $\hat{C} = \langle \hat{u}_1, \hat{u}_2, e, \hat{v} \rangle$ is a valid encryption on message $m_\delta \hat{m}$ and different from C^*. Thus $\mathcal{A}$ can legally query $\texttt{Glass-Box-Dec}(\hat{C})$.
- $\mathcal{C}$ returns $(u_1^*)^{x_1}(u_2^*)^{x_2}((u_1^*)^{y_1}(u_2^*)^{y_2})^{\hat{\alpha}}$ and $m_\delta \hat{m}$ as the output.
- Since $\mathcal{A}$ knows the value $\hat{m}$, $\mathcal{A}$ can easily obtain the message m_δ from $(m_\delta \hat{m})$.
- Thus, $\mathcal{A}$ identifies the bit δ almost always.

Vulnerability in an Implementation of CS

Lemma

The ciphertext $\hat{C} = \langle \hat{u}_1, \hat{u}_2, e, \hat{v} \rangle$ *is a valid ciphertext and the glass box decryption returns* $\mathcal{I}=\langle \hat{\alpha}, V, Z, m \rangle = \langle \hat{\alpha}, (u_1^*)^{x_1}(u_2^*)^{x_2}((u_1^*)^{y_1}(u_2^*)^{y_2})^{\hat{\alpha}}, \hat{u}_1^{z_1}\hat{u}_2^{z_2}, m_\delta \hat{m} \rangle$ *as the output.*

Proof: The ciphertext $\hat{C} = \langle \hat{u}_1, \hat{u}_2, e, \hat{v} \rangle = \langle u_1^*, u_2^*, e^*\hat{m}, Z_1 Z_2^{\hat{\alpha}} \rangle$. $\mathcal{C}$ checks whether $\hat{C}$ is valid by performing the check $\hat{v} \stackrel{?}{=} (\hat{u}_1)^{x_1}(\hat{u}_2)^{x_2} ((\hat{u}_1)^{y_1}(\hat{u}_2)^{y_2})^{\hat{\alpha}}$, where $\hat{\alpha} = H(\hat{u}_1, \hat{u}_2, e)$. Below we show that $\hat{C}$ passes this verification:

$$\begin{aligned} RHS &= (\hat{u}_1)^{x_1}(\hat{u}_2)^{x_2}((\hat{u}_1)^{y_1}(\hat{u}_2)^{y_2})^{\hat{\alpha}} \\ &= (u_1^*)^{x_1}(u_2^*)^{x_2}((u_1^*)^{y_1}(u_2^*)^{y_2})^{\hat{\alpha}} \\ &= Z_1(Z_2)^{\hat{\alpha}} \\ &= \hat{v} = LHS \end{aligned}$$

Since the above check returns true, $\mathcal{C}$ performs the decryption by computing $e/(\hat{u}_1)^{z_1}(\hat{u}_2)^{z_2})$. We show that this computation outputs $\hat{m}m_\delta$:

Vulnerability in an Implementation of CS

$$\begin{aligned} RHS &= (\hat{u}_1)^{x_1}(\hat{u}_2)^{x_2}((\hat{u}_1)^{y_1}(\hat{u}_2)^{y_2})^{\hat{\alpha}} \\ &= (u_1^*)^{x_1}(u_2^*)^{x_2}((u_1^*)^{y_1}(u_2^*)^{y_2})^{\hat{\alpha}} \\ &= Z_1(Z_2)^{\hat{\alpha}} \\ &= \hat{v} = LHS \end{aligned}$$

Since the above check returns true, $\mathcal{C}$ performs the decryption by computing $e/(\hat{u}_1)^{z_1}(\hat{u}_2)^{z_2})$. We show that this computation outputs $\hat{m}m_\delta$:

$$\frac{e}{(\hat{u}_1)^{z_1}(\hat{u}_2)^{z_2}} = \frac{e^*\hat{m}}{(\hat{u}_1)^{z_1}(\hat{u}_2)^{z_2}} = \frac{(u_1)^{z_1}(u_2)^{z_2}m_\delta\hat{m}}{(\hat{u}_1)^{z_1}(\hat{u}_2)^{z_2}} = \frac{(u_1^*)^{z_1}(u_2^*)^{z_2}m_\delta\hat{m}}{(u_1^*)^{z_1}(u_2^*)^{z_2}} = m_\delta\hat{m}$$

Since $u_1^* = \hat{u}_1 = u_1$ and $u_2^* = \hat{u}_2 = u_2$ □

Remark:

Notice that only one step is computed outside TPM but the value exposed due to that is sufficient for the adversary to break the system.

A Scheme in the Standard Model $\texttt{EncryptI}^{\texttt{GB}}$

$\texttt{EncryptI}^{\texttt{GB}}$

- $\texttt{Gen}^{\texttt{GB}}$: Key Generation Algorithm

  Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be groups with prime order q. Let $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ be an admissible bilinear pairing.

  Hash functions:

  - $H_1 : \mathbb{G}_2 \rightarrow \{0,1\}^{l_m}$
  - $H_2 : \mathbb{G}_1 \times \{0,1\}^{l_m} \rightarrow \mathbb{Z}_q^*$, where l_m is the size of the message
  - $H_3 : \mathbb{G}_1 \rightarrow \mathbb{Z}_q^*$

  User Keys:

  - Choose $x, s \in_R \mathbb{Z}_q$ and $P, Q, Y, Z \in_R \mathbb{G}_1$.
  - Compute $X = xP \in \mathbb{G}_1$.
  - Compute $\alpha = \hat{e}(P, Q)^s \in \mathbb{G}_2$.

  The private key $\mathbf{sk} = \langle x, s \rangle \in \mathbb{Z}_q^2$.

  The public key $\mathbf{pk} = \langle P, Q, X, Y, Z, \alpha \rangle \in \mathbb{G}_1^5 \times \mathbb{G}_2$.

$\texttt{EncryptI}^{\texttt{GB}}$

$\texttt{Enc}^{\texttt{GB}}$: Encryption Algorithm

- Choose $r, t \in_R \mathbb{Z}_q$
- Compute $C_1 = rP$
- Compute $C_2 = m \oplus H_1(\alpha^r)$
- Compute $\hat{h} = H_2(C_1, C_2)$
- Compute $h = H_3(r(\hat{h}P + tX))$
- Compute $C_3 = r(hY + Z)$.
- Set $C_4 = t$.
- The ciphertext is $C = \langle C_1, C_2, C_3, C_4 \rangle$.

$\texttt{Dec}^{\texttt{GB}}$: Decryption Algorithm

Decryption of $C = \langle C_1, C_2, C_3, C_4 \rangle$ in Conventional Environment:

- Compute $\hat{h} = H_2(C_1, C_2)$
- Compute $U = \hat{h}C_1$
- Compute $V = C_4 x C_1$
- Compute $h = H_3(U + V)$
- If $\hat{e}(C_3, P) \stackrel{?}{=} \hat{e}(hY + Z, C_1)$
  - Compute $W = \hat{e}(C_1, Q)^s)$
  - Compute $m = C_2 \oplus H_1(W)$

  Else
  - *ABORT*

EncryptI$^{\text{GB}}$

Dec$^{\text{GB}}$ Decryption of $C = \langle C_1, C_2, C_3, C_4 \rangle$ in Hybrid Environment:

- **NC**: Compute $\hat{h} = H_2(C_1, C_2)$ and $U = \hat{h}C_1$
- RAM$\rightarrow$TPM: $\langle C_1, C_4 \rangle$
- **SC**: Compute $V = C_4 x C_1$
- TPM$\rightarrow$RAM: V
- **NC**: Calculate $h = H_3(U + V)$.
- **NC**: Check if $e(C_3, P) \stackrel{?}{=} e(hY + Z, C_1)$

  If *true* then
  - **NC**: Compute $e(C_1, Q)$
  - RAM$\rightarrow$TPM: $e(C_1, Q)$
  - **SC**: Compute $e(C_1, Q)^s$
  - TPM$\rightarrow$RAM: $e(C_1, Q)^s$
  - **NC**: Compute $H_1(e(C_1, Q)^s)$
  - **NC**: Compute $m = C_2 \oplus H_1(e(C_1, Q)^s)$

  else *ABORT*.

A glass box decryption oracle exposes all the values computed and used in the NC, $\mathcal{I} = \langle \hat{h}, U, V, h, e(C_1, Q)), e(C_1, Q))^s, H_1(e(C_1, Q)^s), m \rangle$ to the adversary.

EncryptI$^{\mathrm{GB}}$

Proof of Correctness: To show that the decryption works properly, we have to show that:

1. $U + V = r(\hat{h}P + tX)$.
2. If $C = \langle C_1, C_2, C_3, C_4 \rangle$ is properly constructed, then $\hat{e}(C_3, P) \stackrel{?}{=} \hat{e}(hY + Z, C_1)$.
3. $\hat{e}(C_1, Q)^s = \alpha^r$, where $C_1 = rP$.

Proof: Assume that for some $r \in \mathbb{Z}_q$,

$$C_1 = rP \tag{1}$$

With respect to the same r,

$$C_3 = r(hY + Z) \tag{2}$$

Hence it should be true that,

$$\hat{e}(C_3, P) \stackrel{?}{=} \hat{e}(hY + Z, C_1) \tag{3}$$

This proves the second assertion. Now,

Proof of Correctness Contd...:

$U + V = \hat{h}C_1 + C_4 x C_1 = \hat{h}rP + txrP = r(\hat{h}P + txP) = r(\hat{h}P + tX)$

Thus,

$$U + V = r(\hat{h}P + tX) \tag{4}$$

- This shows that $h = H_3(U + V)$ correctly recovers the h computed in the encryption algorithm.
- This proves the first claim.

For the third claim, we note that
$\hat{e}(C_1, Q)^s = \hat{e}(rP, Q)^s = [\hat{e}(P, Q)^s]^r = \alpha^r$, Therefore,

$$\hat{e}(C_1, Q)^s = \alpha^r \tag{5}$$

This completes the proof that the decryption correctly recovers the message.

Proof for the security of $\texttt{EncryptI}^{\texttt{GB}}$

Theorem

The encryption scheme $\texttt{EncryptI}^{\texttt{GB}}$ is adaptive chosen ciphertext secure under glass box decryption if the DBDH Problem is hard to solve in polynomial time.

Definition

Decisional Bilinear Diffie Hellman Problem - DBDHP: Given $(R, aR, bR, cR) \in_R \mathbb{G}_1^4, \gamma \in_R \mathbb{G}_2$, the DBDHP in $\langle \mathbb{G}_1, \mathbb{G}_2 \rangle$ is to decide whether $\gamma \stackrel{?}{=} \hat{e}(R, R)^{abc}$.

The advantage of an adversary $\mathcal{A}$ in solving the DBDH problem is.

$$Adv_{\mathcal{A}}^{DBDH} = |Pr[\mathcal{A}(R, aR, bR, cR, \hat{e}(R, R)^{abc}) = 1] - Pr[\mathcal{A}(R, aR, bR, cR, \gamma) = 1]|$$

The DBDH Assumption is that, for any probabilistic polynomial time algorithm $\mathcal{A}$, the advantage $Adv_{\mathcal{A}}^{DBDH}$ is negligibly small.

Proof for the security of $\mathtt{EncryptI}^{\mathtt{GB}}$

Setup: $\mathcal{C}$ sets up a system as follows:

- Set
$$P = R \tag{6}$$
- Set
$$Q = bR \tag{7}$$
- Set
$$\alpha = \hat{e}(aR, bR) \tag{8}$$

Therefore, $\alpha = \hat{e}(aR, bR) = \hat{e}(R, bR)^a = \hat{e}(P, Q)^a$

Thus, the second component of the private key denoted as s, is in fact a (implicitly). $\mathcal{C}$ does not know the value of a. Now, choose $x \in_R \mathbb{Z}_q$ and set

$$X = xP \tag{9}$$

This fixes the first component of the private key. Thus the private keys are $\langle x, s = a \rangle$ and $\mathcal{C}$ knows x but does not know s.

Proof for the security of $\texttt{EncryptI}^{\texttt{GB}}$

Setup - Contd...:

$\mathcal{C}$ chooses $\tilde{h}, y, \tilde{z} \in_R \mathbb{Z}_q$ and computes

$$\beta = \tilde{h}(cP) \tag{10}$$

$$h^* = H_3(\beta) \tag{11}$$

$$Y = \frac{1}{h^*}(Q + yP) \tag{12}$$

$$Z = -Q + \tilde{z}P \tag{13}$$

The public keys are $\langle P, Q, X, Y, Z, \alpha \rangle$ and the private keys are $\langle x, s = a \rangle$

Proof for the security of $\texttt{EncryptI}^{\texttt{GB}}$

Phase I:

$\mathcal{O}_{\texttt{Glass-Box-Dec}}$ *Oracle:* $\mathcal{C}$ decrypts the ciphertext $C = \langle C_1, C_2, C_3, C_4 \rangle$ in the following way:

- Computes

$$\hat{h} = H_2(C_1, C_2) \quad (14)$$

$$U = \hat{h} C_1 \quad (15)$$

- Since, $\mathcal{C}$ knows the private key x, $\mathcal{C}$ can also compute

$$V = C_4 x C_1 \quad (16)$$

- Since the values of U and V are correct, $\mathcal{C}$ computes correctly

$$h = H_3(U + V) \quad (17)$$

- Note that H_3 is a target collision resistant hash function and if $(h = h^*)$, abort. Since the Y and Z values are public $\mathcal{C}$ computes correctly the value.

$$hY + Z \quad (18)$$

Proof for the security of $\texttt{EncryptI}^{\texttt{GB}}$

Phase I - Contd...:

- So far, $\mathcal{C}$ could compute and return to $\mathcal{A}$ the values $\langle \hat{h}, U, V, h, hY + Z \rangle$.
- If $\hat{e}(C_3, P) \stackrel{?}{=} \hat{e}(hY + Z, C_1)$ passes, $\mathcal{C}$ must return the value $\hat{e}(C_1, Q)^s$ as well to $\mathcal{A}$,
- $\mathcal{C}$ does not know the value of s.
- $\mathcal{C}$ has to simulate this value. Since P is a generator,

$$C_1 = rP, \textit{ for some } r \in \mathbb{Z}_q \tag{19}$$

- Since $\hat{e}(C_3, P) = \hat{e}(hY + Z, C_1)$ it follows that

$$C_3 = r(hY + Z) \tag{20}$$

  For the same r defined in equation (19).
- Now,

$$\begin{aligned} \hat{e}(C_1, Q)^s &= \hat{e}(rP, Q)^s = \hat{e}(P, Q)^{rs} \\ &= \hat{e}(sP, Q)^r = \hat{e}(aP, rQ), \text{ Since } (s = a) \end{aligned}$$

Proof for the security of $\texttt{EncryptI}^{\texttt{GB}}$

Phase I - Contd...:

- $\mathcal{C}$ knows the value of $aP = aR$ and value of Q.
- $\mathcal{C}$ does not know the value of r.
- Hence, $\mathcal{C}$ will compute the value of rQ indirectly. From equations (12), (13) and (20),

$$\begin{aligned} C_3 &= r(hY + Z) \\ &= r\left(\frac{h}{h^*}(Q + yP) - Q + \tilde{z}P\right) \\ &= \left(\frac{h}{h^*} - 1\right) rQ + \left(\frac{h}{h^*}y + \tilde{z}\right) rP \text{ (Since } h \neq h^*) \end{aligned}$$

Rearranging, we obtain

$$rQ = \left(\frac{h}{h^*} - 1\right)^{-1} \left[C_3 - \left(\frac{h}{h^*}y + \tilde{z}\right) C_1\right] \tag{21}$$

Proof for the security of $\texttt{EncryptI}^{\texttt{GB}}$

Phase I - Contd...:

- Observe that all values in the RHS of equation (21) is available to $\mathcal{C}$.
- Hence rQ can be computed using equation (21).
- Thus, $\hat{e}(C_1, Q)^s = \hat{e}(aP, rQ)$ can be computed even without knowing s.
- Hence, the glass box decryption queries can be perfectly answered by $\mathcal{C}$.
- That is $\mathcal{C}$ can give perfect training to $\mathcal{A}$.

Proof for the security of $\texttt{EncryptI}^{\texttt{GB}}$

Challenge Ciphertext Generation: $\mathcal{A}$ gives $\mathcal{C}$ two messages m_0, m_1 of equal length. C^* is computed as follows:

- Set
$$C_1^* = cR = cP \tag{22}$$
Where, cR is the input to the hard problem.
- Compute
$$C_2^* = m_\delta \oplus H_1(\gamma) \tag{23}$$
Here, $\delta \in \{0,1\}$ is a random bit and γ is an input to the hard problem
- Compute
$$C_3^* = yC_1^* + \tilde{z}C_1^* \tag{24}$$
- Compute
$$C_4^* = (\tilde{h} - \hat{h})x^{-1} \tag{25}$$
Where, $\hat{h} = H_2(C_1^*, C_2^*)$ and $\tilde{h}$ was chosen by $\mathcal{C}$ at setup time. x is one of the private keys known to $\mathcal{C}$.
- The challenge ciphertext $C^* = \langle C_1^*, C_2^*, C_3^*, C_4^* \rangle$ is send to $\mathcal{A}$.

Proof for the security of $\texttt{EncryptI}^{\texttt{GB}}$

Challenge Ciphertext Generation - Contd...:

Lemma

The challenge ciphertext $C^* = \langle C_1^*, C_2^*, C_3^*, C_4^* \rangle$ is a valid and properly formed ciphertext.

Proof: Since $C_1^* = cP$, we should show that

$$C_3^* = c(hY + Z) \tag{26}$$

Where, $h = H_3(c(\hat{h}P + tX))$ and $C_4^* = t = (\tilde{h} - \hat{h})x^{-1}$ Now,

$$\begin{aligned} c(\hat{h}P + tX) &= c(\hat{h}P + C_4^* X) \\ &= c(\hat{h}P + (\tilde{h} - \hat{h})x^{-1}xP) \text{ (From equation (25)} \\ &= c(\hat{h}P - \hat{h}P + \tilde{h}P) \\ &= \tilde{h}(cP) = \beta \text{ (From equation (10))} \end{aligned}$$

Therefore,

$$h = H_3(c(\hat{h}P + tX)) = H_3(\beta) = h^* \tag{27}$$

Proof for the security of $\mathtt{EncryptI}^{\mathtt{GB}}$

Challenge Ciphertext Generation - Contd...:

- From equations (24) and (27), we conclude that C^* is valid / consistent ciphertext, if $C_3^* = c(h^*Y + Z)$.
- C_3^* was computed as $yC_1^* + \tilde{z}C_1^*$ in equation (24).
- Thus we have to show that:

$$c(h^*Y + Z) = yC_1^* + \tilde{z}C_1^* \tag{28}$$

- In fact,

$$\begin{aligned} c(h^*Y + Z) &= c[Q + yP - Q + \tilde{z}P] \text{ (From equations (12) and (13))} \\ &= y(cP) + \tilde{z}(cP) \\ &= yC_1^* + \tilde{z}C_1^* \end{aligned}$$

- This proves that $C^* = \langle C_1^*, C_2^*, C_3^*, C_4^* \rangle$ is a valid / consistent ciphertext. $\square$

Phase II: Same as Phase I.

Proof for the security of $\texttt{EncryptI}^{\texttt{GB}}$

Solving the DBDH Problem:

- The hard problem instance is $\langle R, aR, bR, cR, \gamma \rangle$.
- $\mathcal{C}$ has set $P = R$, $Q = bR$ and $\alpha = e(a\hat{R}, bR) = \hat{e}(P, Q)^s$.
- In C^*, $C_1^* = cR = rP$ and $C_2^* = m_{\delta \oplus H_2(\gamma)}$.
- If m_δ were correctly identified by $\mathcal{A}$, then implicitly, by the collision resistant property of H_2,

$$\begin{aligned}\gamma &= \alpha^r \\ &= \alpha^c \\ &= \hat{e}(P, Q)^{ac} \\ &= \hat{e}(R, bR)^{ac} \\ &= \hat{e}(R, R)^{abc}\end{aligned}$$

Conclusion

Summary:

- We have given a new, strong security model for public key encryption.
- Designed a scheme to withstand the RAM scraper attack and proved the security of the schemes in the Standard Model respectively.

Future Work:

- Establishing the relationship between CCA2 and the new security notion.
- Investigating the security of other primitives like signature and signcryption schemes in the presence of harmful RAM scrapers.
- Constructing a generic transformation that converts CPA/CCA1/CCA2 secure schemes into a Glass Box secure schemes.

References:

Publication details: Sree Vivek S, Sharmila Deva Selvi S, Akshayaram S and Pandu Rangan C: *Stronger public key encryption system withstanding RAM scraper like attacks*. To be published in Wiley, SCN Journal.

- **RAM Scraper** - **http://securityblog.verizonbusiness.com/2009/12/11/ram-scraper-coverage/**
- POS - **http://en.wikipedia.org/wiki/Point_of_sale**

Thank you for your attention.