Gauge Hartwell

9 January, 2021

CS427

Homework 2

1. Our KeyGen function is as follows:

$$\begin{array}{|l|}\hline \underline{\text{KeyGen:}} \\ \quad \text{do} \\ \qquad k \leftarrow \{0,1\}^\lambda \\ \quad \text{until } k \neq 0^\lambda \\ \quad \text{return } k \\ \hline \end{array}$$

We will use these EAVESDROP functions:

$$\begin{array}{|l|}\hline \qquad \mathcal{L}^{\Sigma}_{\text{ots-L}} \\ \hline \underline{\text{EAVESDROP}(m_L, m_R):} \\ \quad k \leftarrow \Sigma.\text{KeyGen} \\ \quad c := k \oplus m_L \\ \quad \text{return } c \\ \hline \end{array} \quad \begin{array}{|l|}\hline \qquad \mathcal{L}^{\Sigma}_{\text{ots-R}} \\ \hline \underline{\text{EAVESDROP}(m_L, m_R):} \\ \quad k \leftarrow \Sigma.\text{KeyGen} \\ \quad c := k \oplus m_R \\ \quad \text{return } c \\ \hline \end{array}$$

This calling function will call an EAVESDROP function with the values 0^λ and 1^λ.

$$\begin{array}{|l|}\hline \qquad\qquad A \\ \hline \underline{\text{EAVESDROP}(m_L, m_R):} \\ \quad c := EAVESDROP(0^\lambda, 1^\lambda) \\ \quad \text{return } c \stackrel{?}{=} 1^\lambda \\ \hline \end{array}$$

A ◊ ots-L produces c as an encryption of m_L. This produces a uniformly random result out of the 15 possibilities ($1/(2^\lambda-1)$). Those 15 possibilities will never generate 0^λ.

A ◊ ots-R produces c as an encryption of m_R. This produces a different result, because you will never get 1^λ back from ots-R.

Every possible input to EAVESDROP will produce a different set of possible outputs with the same probability. Incidentally, every possible input to EAVESDROP will never produce that input in the set of possible outputs.

Pr[A ◊ ots-L => true] = $1/(2^\lambda-1)$.
Pr[A ◊ ots-R => true] = 0.

These two probabilities are different, so the modified KeyGen does not satisfy one-time secrecy.

2. Encryption Scheme:

$\mathcal{K} = (\Sigma.\mathcal{K})^2$
$\mathcal{M} = \Sigma.\mathcal{M}$
$\mathcal{C} = \Sigma.\mathcal{C}$

$\underline{\text{KeyGen:}}$
$\quad k_1 \leftarrow \Sigma.\mathcal{K}$
$\quad k_2 \leftarrow \Sigma.\mathcal{K}$
$\quad \text{return } (k_1, k_2)$

$\underline{\text{Enc}((k_1, k_2), m)\text{:}}$
$\quad c_1 := \Sigma.\text{Enc}(k_1, m)$
$\quad c_2 := \Sigma.\text{Enc}(k_2, m)$
$\quad \text{return } (c_1, c_2)$

$\underline{\text{Dec}((k_1, k_2), (c_1, c_2))\text{:}}$
$\quad m_1 := \Sigma.\text{Dec}(k_1, c_1)$
$\quad m_2 := \Sigma.\text{Dec}(k_2, c_2)$
$\quad \text{if } m_1 \neq m_2 \text{ return err}$
$\quad \text{return } m_1$

It is given that:

$\mathcal{L}^{\Sigma}_{\text{ots-L}}$
$\underline{\text{EAVESDROP}(m_L, m_R)\text{:}}$
$\quad k \leftarrow \Sigma.\text{KeyGen}$
$\quad c := k \oplus m_L$
$\quad \text{return } c$

$\equiv$

$\mathcal{L}^{\Sigma}_{\text{ots-R}}$
$\underline{\text{EAVESDROP}(m_L, m_R)\text{:}}$
$\quad k \leftarrow \Sigma.\text{KeyGen}$
$\quad c := k \oplus m_R$
$\quad \text{return } c$

We will start with ots-L encrypt twice (ots-L2):

$\mathcal{L}^{\Sigma}_{\text{ots-L2}}$
$\underline{\text{EAVESDROP}(m_L, m_R)\text{:}}$
$\quad k_1 \leftarrow \Sigma.\text{KeyGen}$
$\quad k_2 \leftarrow \Sigma.\text{KeyGen}$
$\quad c_1 := k_1 \oplus m_L$
$\quad c_2 := k_2 \oplus m_L$
$\quad \text{return } c_1, c_2$

Now we will factor out two separate sets of statements into one subroutine, changing nothing about the behavior of the scheme:

$\underline{EAVESDROP(m_L, m_R)\text{:}}$
$\quad c_1 := CTXT(m_L)$
$\quad c_2 := CTXT(m_L)$
$\quad \text{return } c_1, c_2$

$\diamond$

$\mathcal{L}^{\Sigma}_{\text{ots\$-real}}$
$\underline{\text{CTXT}(m)\text{:}}$
$\quad k \leftarrow \Sigma.\text{KeyGen}$
$\quad c := k \oplus m$
$\quad \text{return } c$

Next, we will replace ots$-real with ots$-rand, changing nothing about the behavior of the scheme:

$\underline{EAVESDROP(m_L, m_R)\text{:}}$
$\quad c_1 := CTXT(m_L)$
$\quad c_2 := CTXT(m_L)$
$\quad \text{return } c_1, c_2$

$\diamond$

$\mathcal{L}^{\Sigma}_{\text{ots\$-rand}}$
$\underline{\text{CTXT}(m)\text{:}}$
$\quad c \leftarrow \Sigma.\mathcal{C}$
$\quad \text{return } c$

Next, we will change the statements in EAVESDROP to use m_R instead of m_L, again changing nothing about the behavior of the scheme:

$EAVESDROP(m_L, m_R)$:		$\mathcal{L}^{\Sigma}_{\text{ots\$-rand}}$
$c_1 := CTXT(m_R)$	$\diamond$	CTXT(m):
$c_2 := CTXT(m_R)$		$c \leftarrow \Sigma.\mathcal{C}$
return c_1, c_2		return c

Now, we can make a series of changes to get back to our original scheme, but using m_R instead of m_L:

$EAVESDROP(m_L, m_R)$:		$\mathcal{L}^{\Sigma}_{\text{ots\$-real}}$
$c_1 := CTXT(m_R)$	$\diamond$	CTXT(m):
$c_2 := CTXT(m_R)$		$k \leftarrow \Sigma.\mathsf{KeyGen}$
return c_1, c_2		$c := k \oplus m$
		return c

Finally, we can inline our ots\$-real subroutine to make ots-R2:

$\mathcal{L}^{\Sigma}_{\text{ots-R2}}$
EAVESDROP(m_L, m_R):
$k_1 \leftarrow \Sigma.\mathsf{KeyGen}$
$k_2 \leftarrow \Sigma.\mathsf{KeyGen}$
$c_1 := k_1 \oplus m_R$
$c_2 := k_2 \oplus m_R$
return c_1, c_2