CS 427, Assignment 2

Cody Malick
malickc@oregonstate.edu

January 26, 2017

1

The one-time secrecy property of an encryption scheme is defined as having the ability to be changed with another library, and not chaning the resulting probability distribution. Given that information, if we can show that $\Sigma \equiv \Sigma^2$, then we can say that is satisfies one-time secrecy.

Using the notation provided, Σ is defined by its three function, keygen, encryption, and decryption. Then, to show that they are equivilant, we have to show both encryption functions are equal:

```
$Enc((k_1, k_2), m):$
        $c_1 \leftarrow \Sigma.Enc(k_1, m)$
        $c_2 \leftarrow \Sigma.Enc(k_2, m)$
        $return(c_1, c_2)$
```

First, we can replace $\Sigma.Enc()$ with an equivilent function, such as OTP:

```
$Enc((k_1, k_2), m):$
        $c_1 \leftarrow OTP.Enc(k_1, m)$
        $c_2 \leftarrow OTP.Enc(k_2, m)$
        $return(c_1, c_2)$
```

We can do this because we have shown in the past that the output of OTP is uniformly random, and has the one-time secrecy property. Assuming that Σ has this property, then the output of the calling function is unchanged.

Next, we can simply replace c_1 and c_2 with random values z_1 and z_2 as the output of OTP is uniformly random:

```
$Enc((k_1, k_2), m):$
        $z_1 \leftarrow OTP.Enc(k_1, m)$
        $z_2 \leftarrow OTP.Enc(k_2, m)$
        $return(z_1, z_2)$
```

Because the output of the function is uniformly random, regardless of input, we can substitute the message m for any value, as the output for $\mathcal{L}_{ots-L}^{\Sigma^2}$ is completely random, and the output for $\mathcal{L}_{ots-R}^{\Sigma^2}$ is completely random. Because of the random output, the two messages are interchangable, and Σ^2 has one-time secrecy

2

The definition of one-time secrecy is that a calling program, given appropriate inputs, cannot differentiate between two libraries. More specifically, the output distributions of both libraries should be identitical. We can show that a library does not have one-time secrecy by showing a situation where the resulting probability distribution of one input is not the same as the rest.

Suppose we have a calling function A:

```
$func\ boolA():$
        $mes_1\ :=\ 1$
        $k\ :=\ KeyGen()$
        $c\ :=\ Enc(k, mes_1)$
        $if(c\ ==\ k)\{$
            $return\ true$
        $\}$
        $return\ false$
```

Picking the value 1 for the message, function A will always return true. The value of the key, as we encrypt by multiplying the key by the message value. $1 * k$ mod 10 will always return k. This means that, from the ciphertext, we can deduce the value of the message as we know the value of the key. Being able to get information about the original message proves that one-time secrecy does not apply here.

3

To find the secret, we need to solve for the polynomial expression that defines the secret share. The line is defined by a polynomial of type $t-1$, where $t =$ the threshold. In this case, we need to find a poly function of $t-1=1$, so a simple linear equation. The secret we want to find is $f(0)$, the y intercept. Our equation then:

$y = mx + b$

First, we can solve for the slope:

$\frac{3-6}{7-4} = -1$

Then:

$y = -x + b$

Next, plug in a point and solve for the intercept:

$3 = -7 + b$

$10 = b$

Then we have our equation, and incidently our secret, 10
We can find the rest of our shares by plugging in the missing shares, $1 \to 10$:

Share 0: $0 + 10 = 10$

Share 1: $-1 + 10 = 9$

Share 2: $-2 + 10 = 8$

Share 3: $-3 + 10 = 7$

Share 5: $-5 + 10 = 5$

Share 6: $-6 + 10 = 4$

Share 8: $-8 + 10 = 2$

Share 9: $-9 + 10 = 1$

Share 10: $-10 + 10 = 0$

Given two of these other shares, we could have used them to find the slope of our equation.