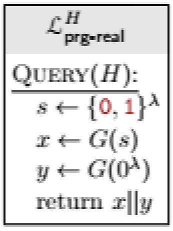
Gauge Hartwell

21 January, 2021

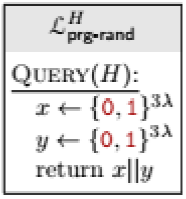
CS427

Homework 5

A) Insecure PRG. Ends in same string every time.
Libraries:

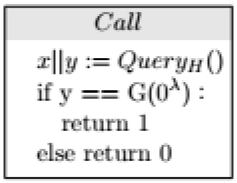


```
$\mathcal{L}^{H}_{\text{prg-real}}$
QUERY($H$):
    $s \leftarrow \{0,1\}^{\lambda}$
    $x \leftarrow G(s)$
    $y \leftarrow G(0^{\lambda})$
    return $x||y$
```



```
$\mathcal{L}^{H}_{\text{prg-rand}}$
QUERY($H$):
    $x \leftarrow \{0,1\}^{3\lambda}$
    $y \leftarrow \{0,1\}^{3\lambda}$
    return $x||y$
```

Calling Program:



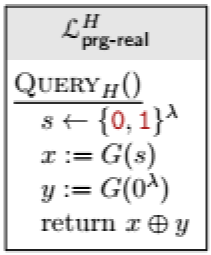
```
Call
    $x||y := Query_H()$
    if y == G($0^{\lambda}$) :
        return 1
    else return 0
```

Pr[Call ◊ prg-real = 1] = 1

Pr[Call ◊ prg-rand = 1] = $1/2^{3\lambda}$

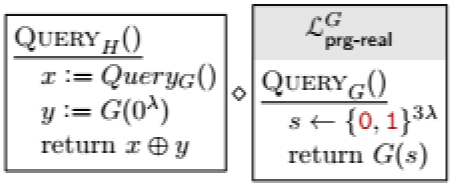
Advantage: 1 - ($1/2^{3\lambda}$), not negligible

B) secure PRG.
Starting function:



```
$\mathcal{L}^{H}_{\text{prg-real}}$
QUERY$_H$()
    $s \leftarrow \{0,1\}^{\lambda}$
    $x := G(s)$
    $y := G(0^{\lambda})$
    return $x \oplus y$
```

First, we can factor out the first call to G, changing nothing about the function of the library.

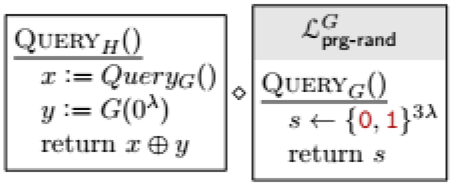


```
QUERY$_H$()
    $x := Query_G()$
    $y := G(0^{\lambda})$
    return $x \oplus y$
```

◊

```
$\mathcal{L}^{G}_{\text{prg-real}}$
QUERY$_G$()
    $s \leftarrow \{0,1\}^{3\lambda}$
    return $G(s)$
```

Second, we can swap prg-real for prg-rand, changing nothing about the operation of the library.

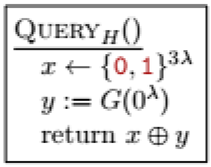


```
QUERY_H()
    $x := Query_G()$
    $y := G(0^\lambda)$
    return $x \oplus y$
```

$\diamond$

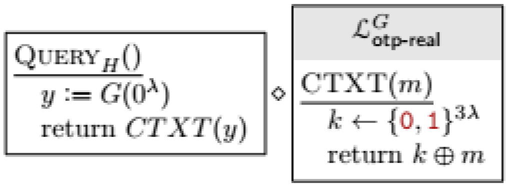
```
$\mathcal{L}^G_{\text{prg-rand}}$
QUERY_G()
    $s \leftarrow \{0,1\}^{3\lambda}$
    return $s$
```

Next, we can inline the prg-rand subroutine, changing nothing about the operation of the library.



```
QUERY_H()
    $x \leftarrow \{0,1\}^{3\lambda}$
    $y := G(0^\lambda)$
    return $x \oplus y$
```

Now, we can use OTP to show that this library is still secure. We will factor out our x value into CTXT and call CTXT() with y as our message. This does not change how the library operates. X is now k in CTXT.

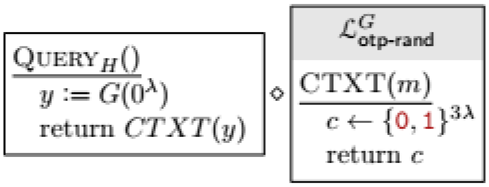


```
QUERY_H()
    $y := G(0^\lambda)$
    return $CTXT(y)$
```

$\diamond$

```
$\mathcal{L}^G_{\text{otp-real}}$
CTXT($m$)
    $k \leftarrow \{0,1\}^{3\lambda}$
    return $k \oplus m$
```

Next, we can replace otp-real with otp-rand, changing nothing about the operation of the library.

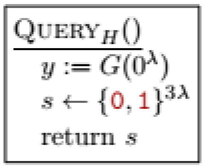


```
QUERY_H()
    $y := G(0^\lambda)$
    return $CTXT(y)$
```

$\diamond$

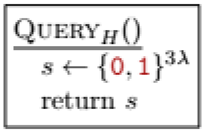
```
$\mathcal{L}^G_{\text{otp-rand}}$
CTXT($m$)
    $c \leftarrow \{0,1\}^{3\lambda}$
    return $c$
```

And then we can inline otp-rand into Query$_H$. This does not change the operation of the library.



```
QUERY_H()
    $y := G(0^\lambda)$
    $s \leftarrow \{0,1\}^{3\lambda}$
    return $s$
```

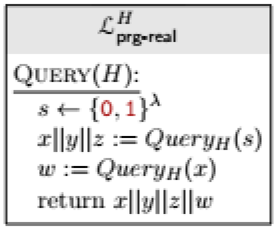
Finally, y isn't doing anything here, so it can be removed.



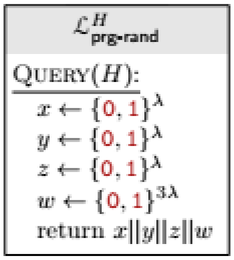
```
QUERY_H()
    $s \leftarrow \{0,1\}^{3\lambda}$
    return $s$
```

C) Insecure PRG. G(x) = W.

Libraries:

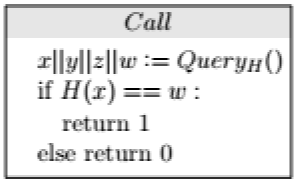


```
$\mathcal{L}^{H}_{\text{prg-real}}$
QUERY($H$):
    $s \leftarrow \{0,1\}^{\lambda}$
    $x\|y\|z := Query_H(s)$
    $w := Query_H(x)$
    return $x\|y\|z\|w$
```



```
$\mathcal{L}^{H}_{\text{prg-rand}}$
QUERY($H$):
    $x \leftarrow \{0,1\}^{\lambda}$
    $y \leftarrow \{0,1\}^{\lambda}$
    $z \leftarrow \{0,1\}^{\lambda}$
    $w \leftarrow \{0,1\}^{3\lambda}$
    return $x\|y\|z\|w$
```

Calling Program:



```
Call
    $x\|y\|z\|w := Query_H()$
    if $H(x) == w$ :
        return 1
    else return 0
```

Pr[Call ◊ prg-real = 1] = 1

Pr[Call ◊ prg-rand] = $1/2^{3\lambda}$

Advantage: 1 – ($1/2^{3\lambda}$), not negligible.