Cryptography 4/13

Reagan Shirk

April 13, 2020

- He's been talking about the project since class started (about 10 minutes) and I'm not doing the project so my notes will be slim until he starts talking about things I have a chance of understanding
  - Let's be real... I probably don't have a chance of understanding anything he talks about

Central Lift

- I don't really know what this is, and I can't tell if it pertains to the project or to the whole class, but in case it's the whole class...

$$\left(\frac{-q}{2}, \frac{q}{2}\right)$$

  has something to do with it.
- Somehow if you center lift 52, you get -50...? I'm confused
- Here we have some Sage stuff, I can at least type exactly what he's typing in Sage.

```
sage: R=Integers(101)
sage: R(3).lift()
3
sage:R(-3).lift()
98
sage: 3^(R(3))
27
sage: 3^(R(-3))
really big number
sage: 3^(-3)
1/27
sage: R(105).lift()
4
```

- What we're seeing here is that Sage does the lift for you
- Apparently the lift is actually really simple to calculate, and similar to remainder? I'm gonna have to look it up because I'm confused
- Question: what's the difference between R(105) and R(105).lift()
  - The difference if that you get an integer...?

```
sage: 1/R(105)
76
sage: 1/(R(105).lift())
1/4
sage: Rx.<x>=R[]
sage: Ra.<a>=QuotientRing(Rx, x^10-1)
sage: g = a^9 + a^7 + a^4 + a^3 + a^2
sage: f = 1 + 2 * (a^8 + a^6 + a^5 + a)
sage: "Important: f and g secure keys small coefficients"
'Important: f and g secure keys small coefficients'
sage: f + g
not a big polynomial
sage: f * g
a little bigger but still not very big, degree will never get bigger than 9
```

```
sage: g/f
much larger coefficients
sage: h = g/f
sage: "h is the public key"
'h is the public key'
sage: "Given a polynomial h, finding f and g is hard"
'Given a polynomial h, finding f and g is hard'
sage: "The message is binary 1011010010"
'The message is binary 1011010010'
sage: m = a^9 + a^7 + a^6 + a^4 + a
sage: "To encrypt, you need another small coefficient polynomial"
'To encrypt, you need another small coefficient polynomial that does not depend on the message'
sage: e = a^7 + a^6 + a^4 + a + 1
sage: cipher = 2 * h * e + m
sage: chiper
very large polynomial coefficients
sage: cipher * f
smaller polynomial coefficients
sage: decryption = cipher * f
sage: "Parity of decription polynomial coefficients: odd = 1, even = 0"
'Parity of decription polynomial coefficients: odd = 1, even = 0'
sage: "11 = 1, 8 = 0, 11 = 1, 7 = 1, 10 = 0, 11 = 1, 10 = 0, 10 = 0, 7 = 1, 10 = 0"
'11 = 1, 8 = 0, 11 = 1, 7 = 1, 10 = 0, 11 = 1, 10 = 0, 10 = 0, 7 = 1, 10 = 0'
sage: "Decrypted message: 1011010010"
'Decrypted message: 1011010010'
sage: "chiper * f = (2he+m)*f = 2hef + mf = 2ge + mf"
```

- We know that this is correct, but how do we know that it is secure? That's what we'll talk about on Wednesday
  - Hint: it's v safe