Cryptography 4/24

Reagan Shirk

April 24, 2020

Sage stuff for. . . something?

- It seems like he's just been rambling about viruses for like 13 minutes

```
sage: p = ZZ.random_element(2^200).next_prime()
sage: q = ZZ.random_element(2^210).next_prime()
sage: n = p * q
sage: gcd(65537, (p-1)*(q-1))
1
sage: phin = (p-1)*(q-1)
sage: d = Integers(phin)(e)^(-1)
sage: Integers(n)(8237583)^(e*d-1)
1
sage: a = 98235842
sage: gcd(Integers(n)(a)^(e*d-1) -1, n)
0
sage: d = d.lift()
sage: t = e*d - 1
sage: t = t/2
sage: t = t/2
sage: t = t/2
sage: "keep dividing by 2 until you get an odd number"
'keep dividing by 2 until you get an odd number'
sage: gcd(Integers(n)(a)^(t) - 1, n)
```

- Guys, I'm so confused about what's going on. I glance away from the screen for 15 seconds and I'm lost. I wish he recorded these lectures.

Textbook RSA

- $c = m^e \bmod n$
  - $e = 65537$
- Textbook RSA isn't safe for a few reasons:
  - Even if you don't know the decryption key, you can encrypt any message because e and n are public. This means a chosen plaintext attack is easy
  - Say that the message is your ID number. What is the problem?
    - Plaintext space is small
    - Searching space is small - maximum is 10^9 which is easy for modern computing, but the first two digits of our ID number are fixed, so it's actually 10^7
  - How do you fix these problems? You use padding
- Good version of RSA: RSA-OAEP - this kind has padding to artificially increase your message space
  - This is the RSA that is used every day