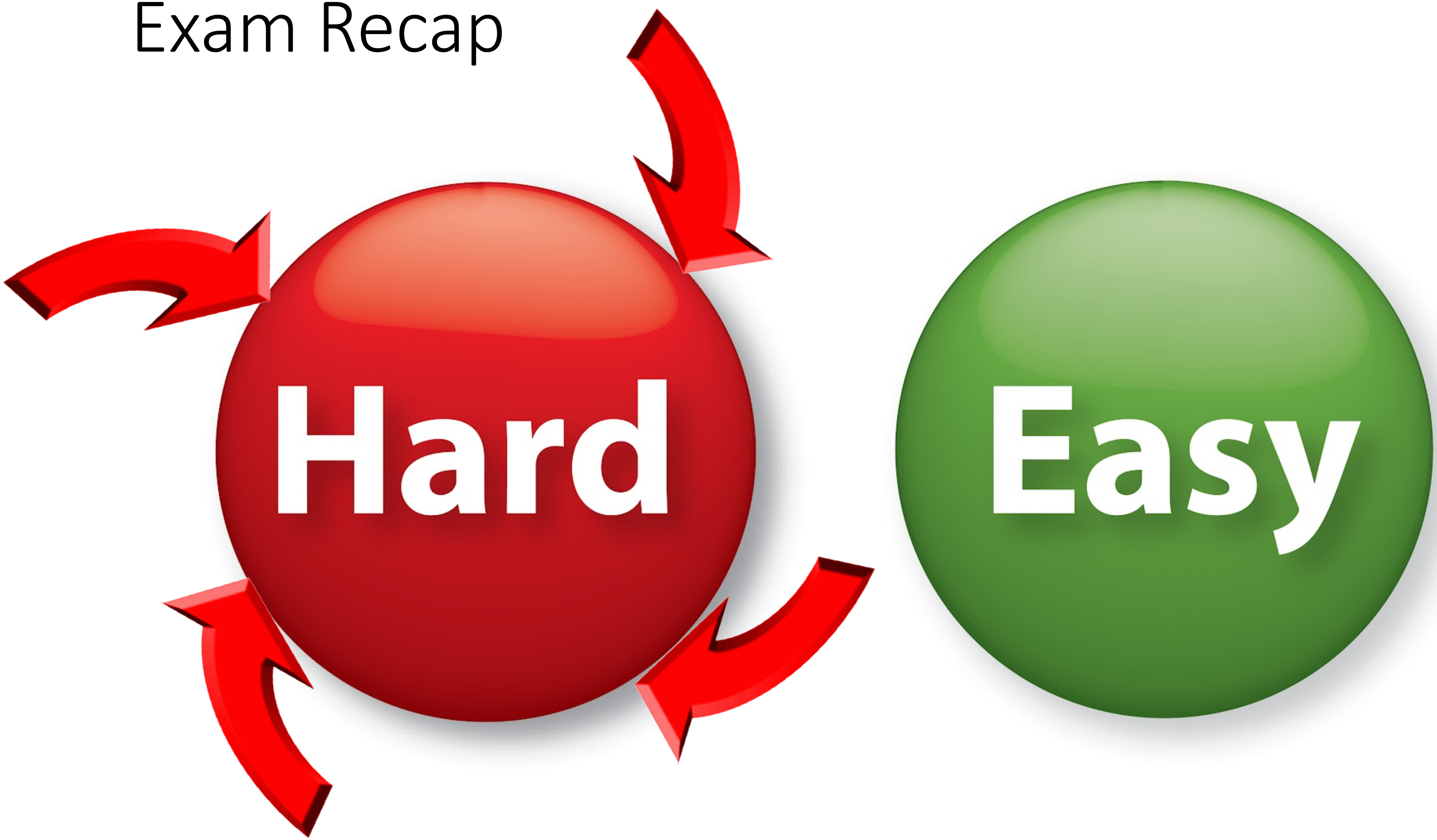
Cryptography
CS 555

Topic 22: Number Theory/Public Key-Cryptography

Exam Recap



Exam Recap

- **Highest Average Score on Question**
  - **Question 4:** (Feistel Network with round function $f(x) = 0^n$)

- **Tougher Questions**
  - **Question 9:** Let $K=Gen(1^{1000})$ be a key for an authenticated encryption scheme…
  - Correct Answer: M (More information needed) in both cases
  - CCA-Security statement is an asymptotic statement
    - For all PPT A there exists a negligible function **negl**(n).
    - We could have **negl**(n) = $2^{100000000-n}$, which would imply that A can win if $n < 10000000$
    - It could also be the case that for all A running in time $2^{1000000}$ the attacker succeeds with probability $2^{-100000000-n}$
  - Partial Credit for False answers

Exam Recap

- **Tougher Questions**
  - **Question 10.c-e: Is it a One-Way-Function?**
    - **Correct Answers:** More information needed in each case.
      - Grading: Generous partial/full credit for “mostly correct” answers
    - **Question 10.c:** $\mathrm{f}(\mathrm{x}, \mathrm{k}) = \mathrm{EncK}(\mathrm{x}) \| x$ with |x| > |k|
      - $\mathrm{f}(\mathrm{x}, \mathrm{k}) = \mathrm{EncK}(\mathrm{x}) \| x$ is example one-way function from slides + textbook*
      - * proof uses 2|k|<|x| not |k|<|x|
      - Correct Answer is M, but full credit for answer T

    - **Question 10.d:** $\mathrm{f}(\mathrm{x}, \mathrm{k}) = \mathrm{EncK}(\mathrm{x})$
      - Counter example uses canonical eavesdropping secure encryption scheme
        $$\mathrm{Enc}_{\mathrm{K}}(\mathrm{x}) = \mathrm{G(K)} \oplus x$$
      - Can fully control output of $\mathrm{f}(\mathrm{x}, \mathrm{k})$ by changing x

Exam Recap

- **Tougher Questions**
  - **Question 10.c-e: Is it a One-Way-Function?**
    - **Correct Answers:** More information needed in each case.
      - Grading: Generous partial/full credit for "mostly correct" answers
    - **Question 10.e:** $\mathrm{f}(\mathrm{x}, \mathrm{k}) = \mathrm{Enc}_{\mathrm{K}}(\mathrm{x}) \| x$ with $|\mathrm{x}| \leq |\mathrm{k}|$
      - Counter Example (One-Time-Pad)
$$\mathrm{Enc}_{\mathrm{K}}(\mathrm{x}) = \mathrm{K} \oplus x$$
      - Can fully control output of $\mathrm{f}(\mathrm{x}, \mathrm{k})$ by altering x (as before)
      - Correct Answer is M, but full credit for answer F

Exam Recap

- **Tougher Questions**
  - **Question 11: (AKA the most popular choice for bonus question)**
    - **Part A.** Is this CPA-secure in the random oracle model?
      $$\mathrm{Enc}_{\mathrm{K}}(\mathrm{m}) = \langle F_K(m), H\big(K \oplus F_K(m)\big) \oplus m \rangle$$
    - Looks fancy, but on closer examination $\mathrm{Enc}_{\mathrm{K}}(\mathrm{m})$ is stateless/deterministic…
    - Correct Answer: False

    - **Part B.** $\mathrm{Mac}_{\mathrm{K}}(\mathrm{m}) = H(m)$
      - The secret key K is not involved at all!
      - Trivial to forge messages

Exam Recap

- **Tougher Questions**
  - **Question 11: (AKA the most popular choice for bonus question)**
    - **Part C.** Attacker has $\sqrt{n}$ queries to H(.) and we use K = H(i*) (for a uniformly random $i^* \leq n$) as the secret key in an authenticated encryption scheme. Claim: The attacker wins CCA-Security game with probability $1/\sqrt{n} + negl(n)$ at best.
      - **Case 1:** Attacker Queries H(j) at $j = i^*$
        - Attacker might win, but we only reach this case with probability $1/\sqrt{n}$
      - **Case 2:** Attacker does not query H(i*)
        - Secret Key K is uniformly random in this case
        - This *is* the standard CCA-security game.
        - Odds of PPT attacker winning are negligible.

Exam Recap

- **Tougher Questions**
  - **Question 11: (AKA the most popular choice for bonus question)**
    - **Part D.** Attacker has n queries to H(.) and we use K = $H^n(i^*)$ (for a uniformly random $i^* \leq n$) as the secret key in an authenticated encryption scheme. Claim: The attacker wins CCA-Security game with probability $1/\sqrt{n} + negl(n)$ at best.
      - **Case 1**: Attacker Queries H (.) at $H^{n-1}(i^*)$
        - Attacker might win, but we only reach this case with probability $1/n + negl(n)$
        - **Intuition**, it should take n-1 queries to compute $H^{n-1}(i^*)$ and one more to check
      - **Case 2**: Attacker does not query H(.) at $H^{n-1}(i^*)$
        - Secret Key K is uniformly random in this case
        - This *is* the standard CCA-security game.
        - Odds of PPT attacker winning are negligible.

Mid-Semester Recap

- We built an authenticated encryption scheme
  - **Theory**: From one-way functions
    - Encrypt then MAC
  - **Practice**: AES-GCM

- Authenticated Encryption guarantees
  - **Secrecy** (attacker cannot decrypt message)
  - **Integrity** (attacker cannot modify ciphertext)

- What else is there to do?

Public Key Cryptography

- **Key-Exchange Problem:**
  - Obi-Wan and Yoda want to communicate securely
  - Suppose that
    - Obi-Wan and Yoda don’t have time to meet privately and generate one
    - Obi-Wan and Yoda share an asymmetric key with Anakin
    - Can they use Anakin to exchange a secret key?

Public Key Cryptography

- Key-Exchange Problem:
  - Obi-Wan and Yoda want to communicate securely
  - Suppose that
    - Obi-Wan and Yoda don’t have time to meet privately and generate one
    - Obi-Wan and Yoda share an asymmetric key with Anakin
    - Can they use Anakin to exchange a secret key?
    - **Remark**: Obi-Wan and Yoda both trust Anakin, but would prefer to keep the key private just in case.

Public Key Cryptography

- Key-Exchange Problem:
  - Obi-Wan and Yoda want to communicate securely
  - Suppose that
    - Obi-Wan and Yoda don’t have time to meet privately and generate one
    - Obi-Wan and Yoda share an asymmetric key with Anakin
    - Can they use Anakin to exchange a secret key?
    - **Remark**: Obi-Wan and Yoda both trust Anakin, but would prefer to keep the key private just in case.
- Need for Public-Key Crypto
  - We can solve the key-exchange problem using public-key cryptography.
  - No solution is known using symmetric key cryptography alone

Public Key Cryptography

- Suppose we have n people and each pair of people want to be able to maintain a secure communication channel.
  - How many private keys per person?
  - **Answer**: n-1

- Key Explosion Problem
  - n can get very big if you are Google or Amazon!

Number Theory

- Key tool behind public key-crypto
  - RSA, El-Gamal, Diffie-Hellman Key Exchange

- Aside: don’t worry we will still use symmetric key crypto
  - It is more efficient in practice
  - First step in many public key-crypto protocols is to generate symmetric key
    - Then communicate using authenticated encryption

Polynomial Time Factoring Algorithm?

FindPrimeFactor

Input: N

For i=1,…,N

  if N/i is an integer then

    Output I

Running time: O(N) steps

Correctness: Always returns a factor

Did we just break RSA?

Polynomial Time Factoring Algorithm?

FindPrimeFactor

Input: N

For i=1,…,N

  if N/i is an integer then

    Output I

Running time: O(N) steps

Correctness: Always returns a factor

We measure running time of an arithmetic algorithm (multiply, divide, GCD, remainder) in terms of the number of bits necessary to encode the inputs.

How many bits $||N||$ to encode N?
Answer: $||N|| = \log_2(N)$

Polynomial Time Operations on Integers

Polynomial time in $||a||$and $||b||$

- Addition
- Multiplication
- Division with Remainder
  - **Input: a** and **b**
  - **Output**: quotient q and remainder r < **b** such that

$$\boldsymbol{a} = q\boldsymbol{b} + r$$

Convenient Notation: r = **a** mod **b**

- Greatest Common Divisor
  - **Example:** gcd(9,15) = 3
- Extended GCD(**a**,**b**)
  - Output integers X,Y such that

$$X\boldsymbol{a} + Y\boldsymbol{b} = \gcd(\boldsymbol{a}, \boldsymbol{b})$$

Polynomial Time Operations on Integers

- Division with Remainder
  - **Input:** a and b
  - **Output**: quotient q and remainder r < b such that

$$\boldsymbol{a} = q\boldsymbol{b} + r$$

- Greatest Common Divisor
  - **Key Observation:** if $\boldsymbol{a} = q\boldsymbol{b} + r$

  Then gcd(**a,b**) = gcd(r, **b**)=gcd(**a** mod **b**, **b**)

  Proof:

  - Let d = gcd(**a**,**b**). Then d divides both a and b. Thus, d also divides r=a-qb.

    →d=gcd(**a**,**b**) $\leq$ gcd(r, **b**)
  - Let d’ = gcd(r, **b**). Then d’ divides both b and r. Thus, d’ also divides a = qb+r.

    →gcd(**a**,**b**) $\geq$ gcd(r, **b**)=d’
  - Conclusion: d=d’.

More Polynomial Time Operations on Integers

- **(Modular Arithmetic)** The following operations are polynomial time in $\|a\|$ and $\|b\|$ and $\|N\|$.

1. Compute [**a** mod **N**]
2. Compute sum [(**a**+**b**) mod **N**], difference [(**a**-**b**) mod **N**] or product [**ab** mod **N**]
3. Determine whether **a** has an inverse $\mathbf{a}^{-1}$ such that 1=[$\mathbf{aa}^{-1}$ mod **N**]
4. Find $\mathbf{a}^{-1}$ if it exists
5. Compute the exponentiation [$\mathbf{a}^{\mathbf{b}}$ mod **N**]

More Polynomial Time Operations on Integers

- (Modular Arithmetic) The
  in |

1. Compute [**a** mod **N**]
2. Compute sum [(
   [**ab** mod **N**]
3. Determine whether **a** has an inverse $\mathbf{a}^{-1}$ such that 1=[$\mathbf{aa}^{-1}$ mod **N**]
4. Find $\mathbf{a}^{-1}$ if it exists
5. Compute the exponentiation [$\mathbf{a}^{\mathbf{b}}$ mod **N**]

Remark: Part 3 and 4 use extended GCD algorithm

More Polynomial Time Operations on Integers

- (Modular Arithmetic) The following operations are polynomial time in in $\|a\|$ and $\|b\|$ and $\|N\|$.

1. Compute the exponentiation [$\mathbf{a}^{b}$ mod $\mathbf{N}$]

Attempt 1:

X =1

For i=1,…,b

    X = X*a

What is wrong?

More Polynomial Time Operations on Integers

(Modular Arithmetic) The following operations are polynomial time in $\|a\|$, $\|b\|$ and $\|N\|$.

1. Compute the exponentiation [$\mathbf{a}^{\mathbf{b}}$ mod **N**]

Attempt 2:

If (b=0) return 1

X[0]=a;

For i=1,…,$\log_2$(b)+1

    X[i] = X[i-1]*X[i-1]   // Invariant: X[i] = $\boldsymbol{a}^{2^i}$

$$[\mathbf{a}^{\mathbf{b}} \text{ mod } \mathbf{N}] = \boldsymbol{a}^{\sum_i \boldsymbol{b}[i] 2^i} \text{mod } \mathbf{N}$$

$$= \prod_i \mathbf{b}[\mathrm{i}]\ \mathrm{X}[\mathrm{i}] \ \text{mod } \mathbf{N}$$

What is wrong?

The number of bits in $a^{2^{\|b\|+1}}$ is O($2^{\|b\|+1}$).

More Polynomial Time Operations on Integers

(Modular Arithmetic) The following operations are polynomial time in $||a||$, $||b||$ and $||N||$.

1. Compute the exponentiation [$\mathbf{a}^{\mathbf{b}}$ mod **N**]

Fixed Algorithm:

If (b=0) return 1

X[0]=a;

For i=1,…,$\log_2$(b)+1

X[i] = X[i-1]*X[i-1] mod **N** // Invariant: X[i] = $\boldsymbol{a}^{2^i}$ mod **N**

[$\mathbf{a}^{\mathbf{b}}$ mod **N**]=$\boldsymbol{a}^{\sum_i \boldsymbol{b}[i] 2^i} \bmod \mathbf{N}$

$$= \prod_i \mathbf{b}[\mathrm{i}]\ \mathrm{X}[\mathrm{i}] \bmod \mathbf{N}$$

More Polynomial Time Operations on Integers

(Sampling) Let

$$\mathbb{Z}_N = \{1, \ldots, N\}$$
$$\mathbb{Z}_N^* = \{x \in \mathbb{Z}_N \mid \gcd(N, x) = 1\}$$

Examples:

$$\mathbb{Z}_6^* = \{1,5\}$$

$$\mathbb{Z}_7^* = \{1,2,3,4,5,6\}$$

More Polynomial Time Operations on Integers

(Sampling) Let

$$\mathbb{Z}_N = \{1, \dots, N\}$$
$$\mathbb{Z}_N^* = \{x \in \mathbb{Z}_N | \gcd(N, x) = 1\}$$

- There is a probabilistic polynomial time algorithm (in |N|) to sample from $\mathbb{Z}_N^*$ and $\mathbb{Z}_N$
- Algorithm to sample from $\mathbb{Z}_N^*$ is allowed to output "fail" with negligible probability in |N|.
- Conditioned on not failing sample must be uniform.

Useful Facts

$$x, y \in \mathbb{Z}^*_{\mathrm{N}} \rightarrow [xy \bmod \mathrm{N}] \ \in \mathbb{Z}^*_{\mathrm{N}}$$

Example 1: $\mathbb{Z}_8^* = \{1,3,5,7\}$

$$[3 \times 7 \bmod 8] = [21 \bmod 8] = [5 \bmod 8] \ \in \mathbb{Z}^*_{\mathrm{N}}$$

Proof: gcd(xy,N) = d

Suppose d>1 then for some prime p and integer q we have d=pq.

Now p must divide N and xy (by definition) and hence p must divide either x or y.

(WLOG) say p divides x. In this case gcd(x,N)=p > 1, which means $x \notin \mathbb{Z}^*_{\mathrm{N}}$

More Useful Facts

$$x, y \in \mathbb{Z}^*_{\mathrm{N}} \rightarrow [xy \bmod \mathrm{N}] \in \mathbb{Z}^*_{\mathrm{N}}$$

Fact 1: Let $\boldsymbol{\phi}(\boldsymbol{N}) = |\mathbb{Z}^*_{\mathrm{N}}|$ then for any $x \in \mathbb{Z}^*_{\mathrm{N}}$ we have

$$\left[x^{\boldsymbol{\phi}(\boldsymbol{N})} \bmod \mathrm{N}\right] = 1$$

Example: $\mathbb{Z}^*_8 = \{1,3,5,7\}$, $\phi(8) = 4$

$$\left[3^{\mathbf{4}} \bmod 8\right] = [9 \times 9 \bmod 8] = 1$$
$$\left[5^{\mathbf{4}} \bmod 8\right] = [25 \times 25 \bmod 8] = 1$$
$$\left[7^{\mathbf{4}} \bmod 8\right] = [49 \times 49 \bmod 8] = 1$$

More Useful Facts

$$x, y \in \mathbb{Z}_{\mathrm{N}}^{*} \rightarrow [xy \bmod \mathrm{N}] \in \mathbb{Z}_{\mathrm{N}}^{*}$$

Fact 1: Let $\boldsymbol{\phi}(\boldsymbol{N}) = |\mathbb{Z}_{\mathrm{N}}^{*}|$ then for any $x \in \mathbb{Z}_{\mathrm{N}}^{*}$ we have $\left[x^{\boldsymbol{\phi}(\boldsymbol{N})} \bmod \mathrm{N}\right] = x$

Fact 2: Let $\boldsymbol{\phi}(\boldsymbol{N}) = |\mathbb{Z}_{\mathrm{N}}^{*}|$ and let $N = \prod_{i=1}^{m} p_i^{e_i}$, where each p_i is a distinct prime number and $e_i > 0$ then

$$\boldsymbol{\phi}(\boldsymbol{N}) = \prod_{i=1}^{m} (p_i - 1) p_i^{e_i - 1} = n \prod_{i=1}^{m} \left(1 - \frac{1}{p_i}\right)$$

Next Class

- Read Katz and Lindell 8.1
    - And review number theory background in appendix (B.1 and B.2)
- More Number Theory