

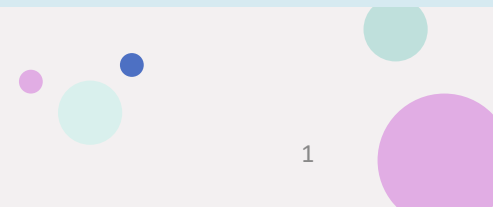


Lecture 13

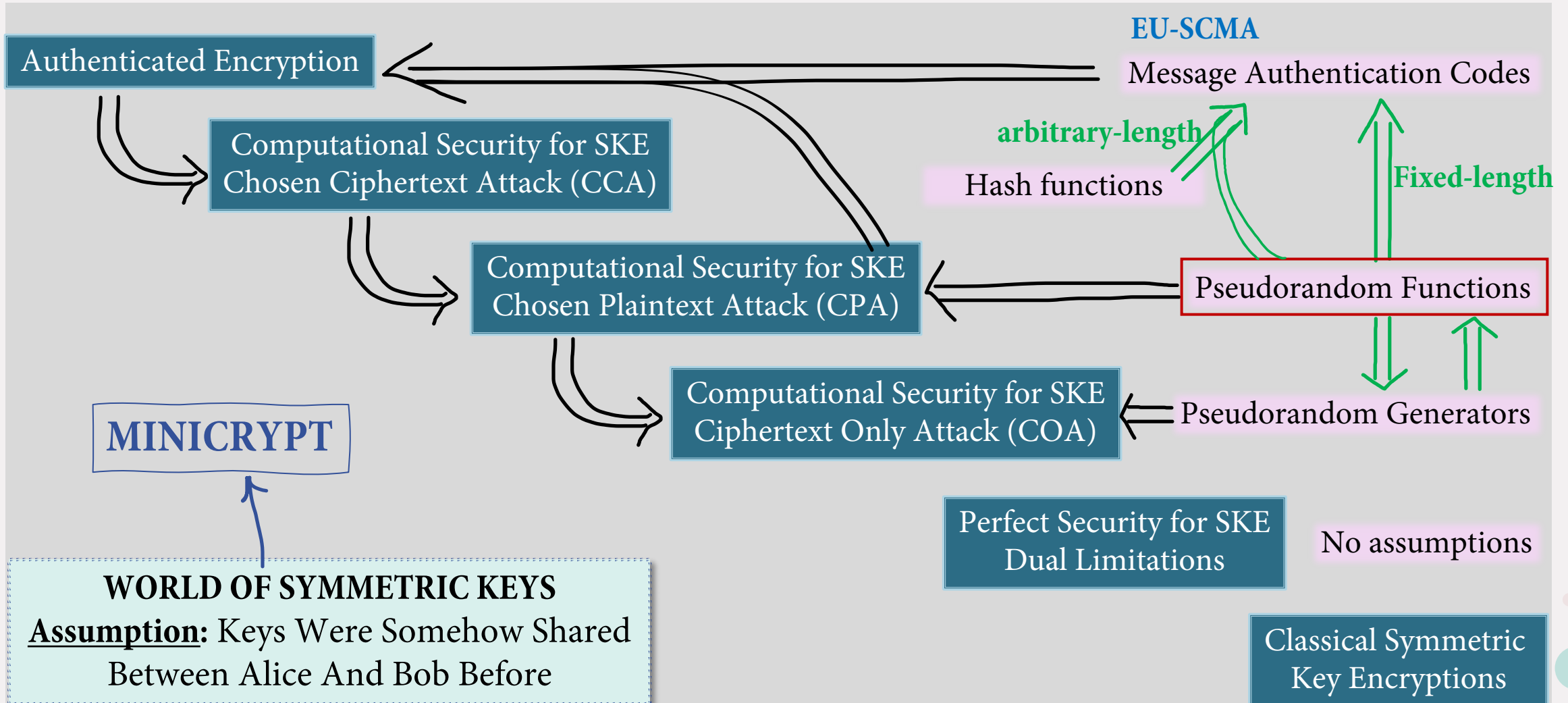
One Way Functions, Hardness Assumptions

CS409/6011: Introduction to Cryptography

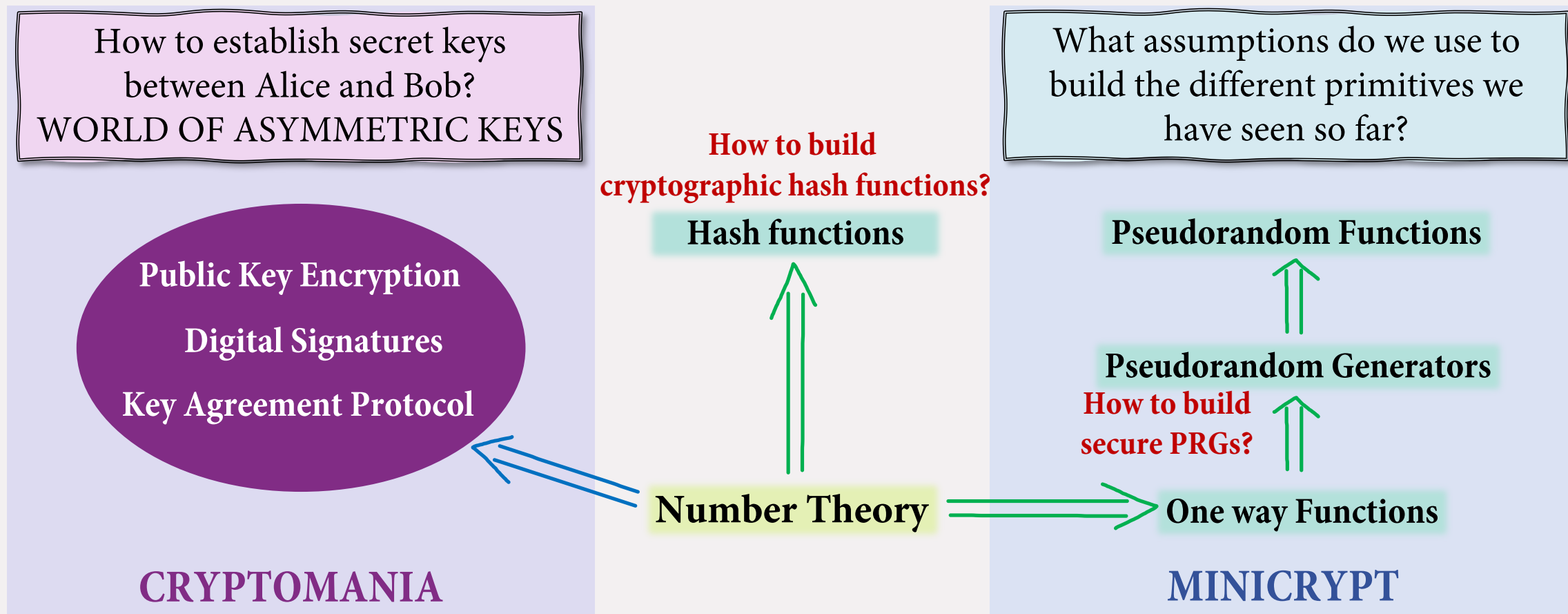
Sruthi Sekar



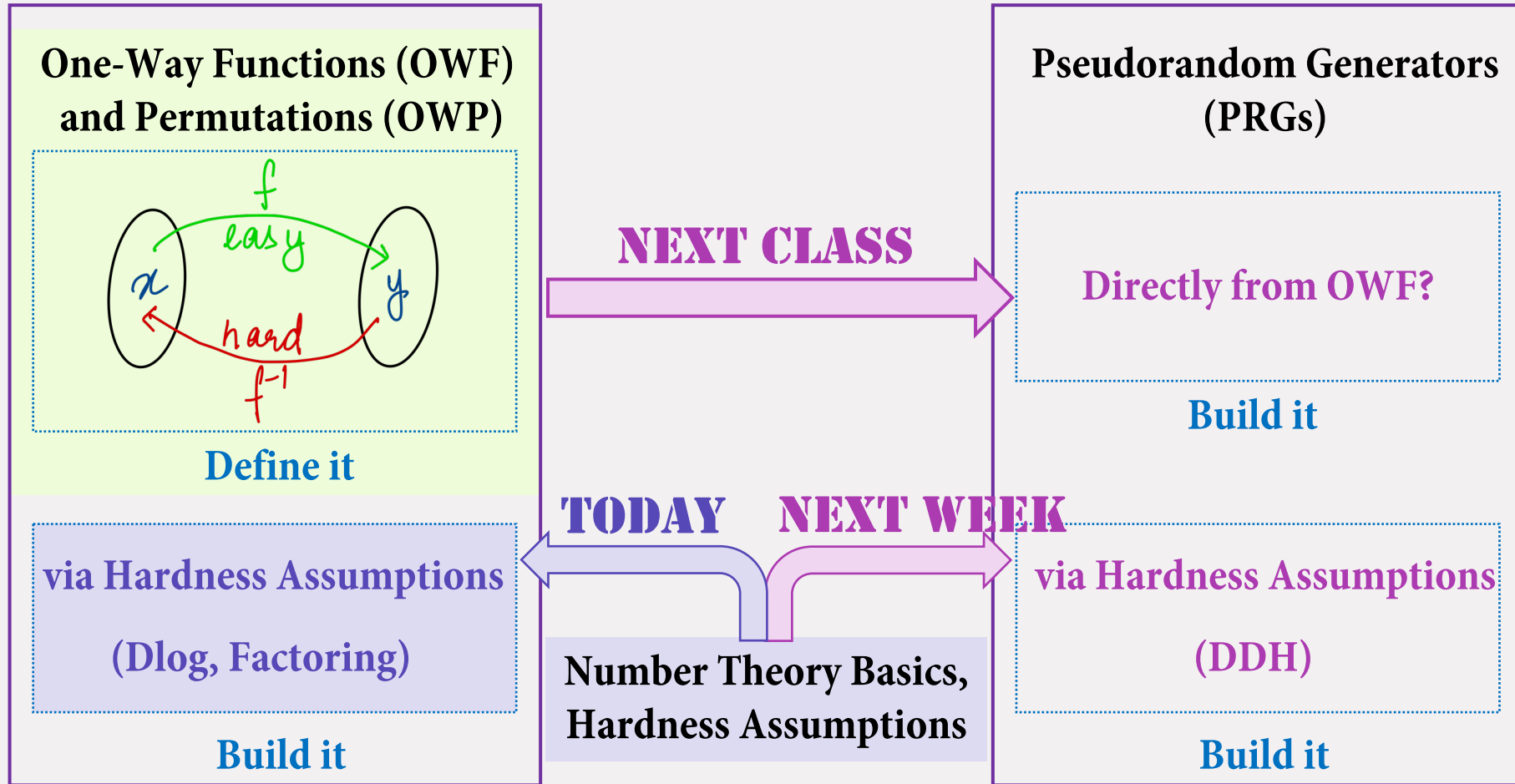
Journey so far



Journey Ahead



Coming Up..



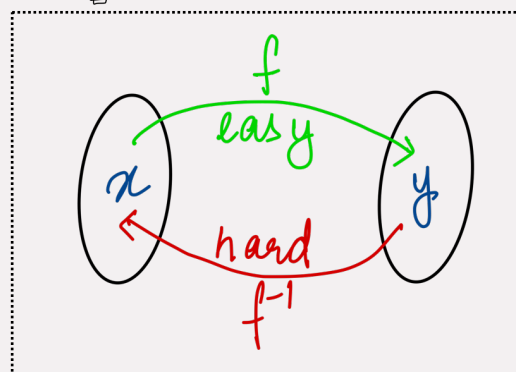
One-way Functions

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

Sudoku solution



INTUITIVELY



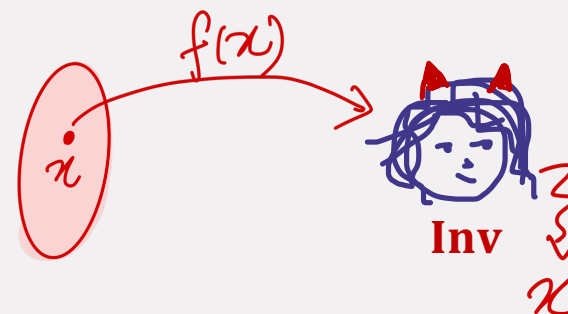
5	3		7			
6			1	9	5	
	9	8				6
8			6			3
4			8	3		1
7			2			6
	6				2	8
			4	1	9	5
			8		7	9

Sudoku puzzle

Attempt I at formally defining this:

$\forall$ PPT **Inv** and $\forall x, \exists \text{negl s.t.}$

$$\Pr[\mathbf{Inv}(f(x)) = x] \leq \text{negl}$$



Problem: Too much asking (*everywhere hardness*)

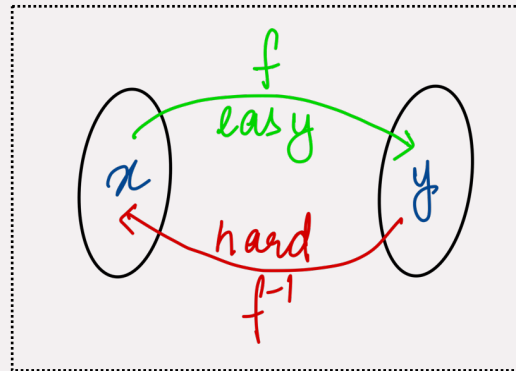
One-way Functions

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

Sudoku solution



INTUITIVELY



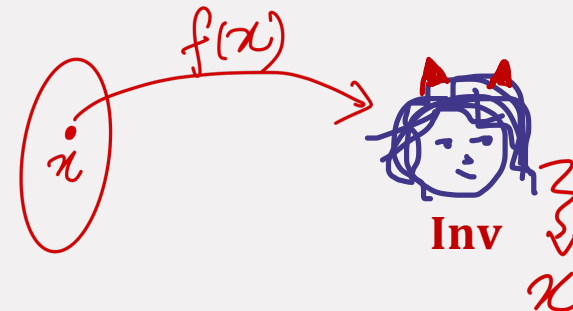
5	3			7				
6				1	9	5		
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

Sudoku puzzle

Attempt II at formally defining this:

$\forall$ PPT **Inv** and $\exists x, \exists \text{negl}$ s.t.

$$\Pr[\mathbf{Inv}(f(x)) = x] \leq \text{negl}$$



Problem: Too little asking (*worst-case hardness*)

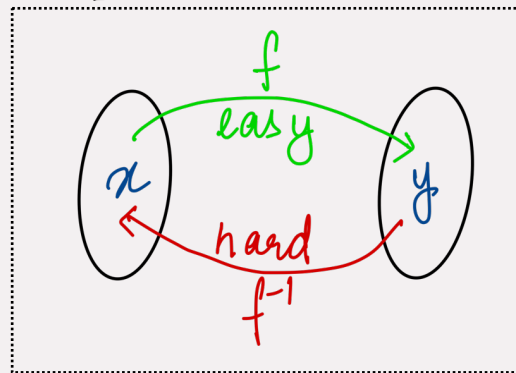
One-way Functions



INTUITIVELY

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

Sudoku solution



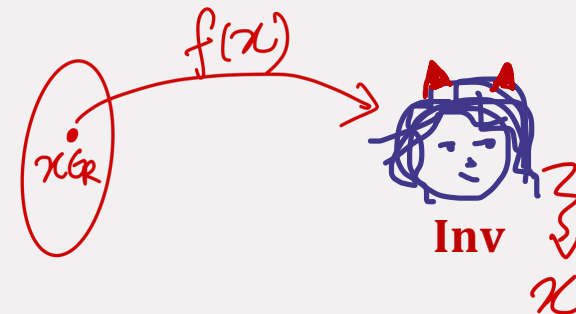
5	3			7				
6				1	9	5		
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

Sudoku puzzle

Attempt III at formally defining this:

$\forall$ PPT **Inv**, $\exists$ negl s.t.

$$\Pr_{x \in_R \{0,1\}^n} [\mathbf{Inv}(f(x)) = x] \leq \text{negl}(n)$$



Problem: Consider $f(x) = 0^{|x|}$ (what is the problem?)

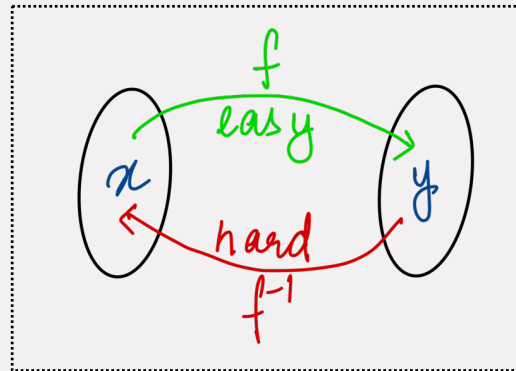
One-way Functions

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

Sudoku solution



INTUITIVELY



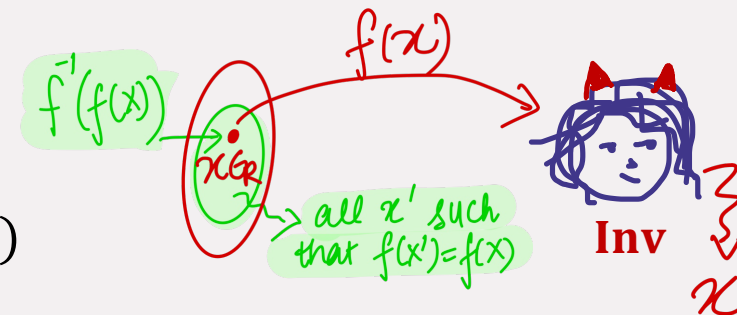
5	3		7			
6			1	9	5	
	9	8				6
8			6			3
4			8	3		1
7			2			6
	6				2	8
			4	1	9	5
			8		7	9

Sudoku puzzle

Attempt IV at formally defining this:

$\forall$ PPT **Inv**, $\exists$ negl s.t.

$$\Pr_{x \in_R \{0,1\}^n} [\mathbf{Inv}(f(x)) \in f^{-1}(f(x))] \leq \text{negl}(n)$$



Single function vs family of functions: Easier to build parametrized family of OWFs!

One-way Function Family

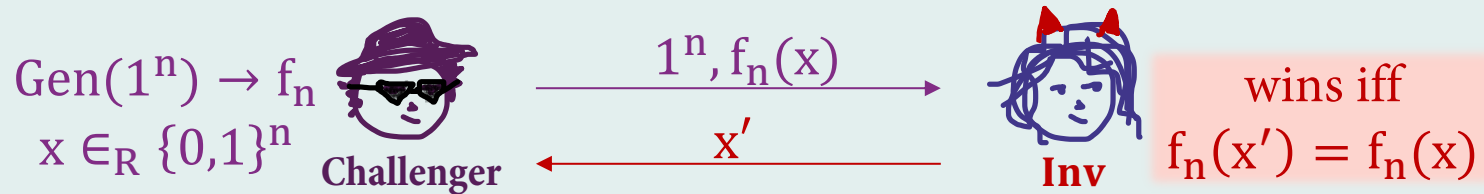
Definition 1 (One-way Function Family)

$$\{f_n: \{0,1\}^n \rightarrow \{0,1\}^{m(n)}\}_{n \in \mathbb{N}}$$

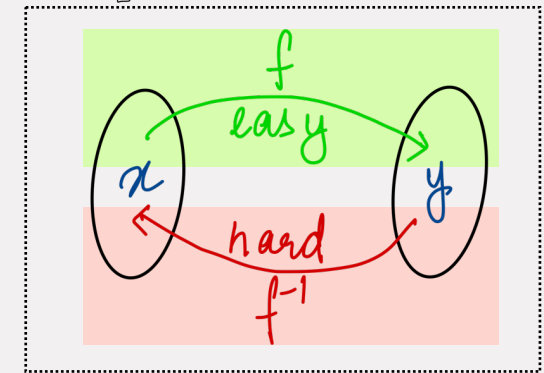
$$\text{Gen}(1^n) \rightarrow f_n$$

- $\exists$ an efficient deterministic algorithm f such that $\forall x, f(n, x) = f_n(x)$
- $\forall$ PPT **Inv**, $\exists$ negl s.t. $\forall n$

$$\Pr_{x \in_R \{0,1\}^n} [\text{Inv}(1^n, f_n(x)) \in f_n^{-1}(f_n(x))] \leq \text{negl}(n)$$



INTUITIVELY



- **Length-preserving OWF**: $m(n) = n$
- **One-way permutation Family**: for each $f_n \leftarrow \text{Gen}(1^n)$, f_n is length-preserving and bijective.

Quiz Time

OWF or not?

✓ $f_1(x) := f(x) || 0^{|x|}$, where f is a OWF

✓ $f_3(x) := G(x)$, where G is a PRG

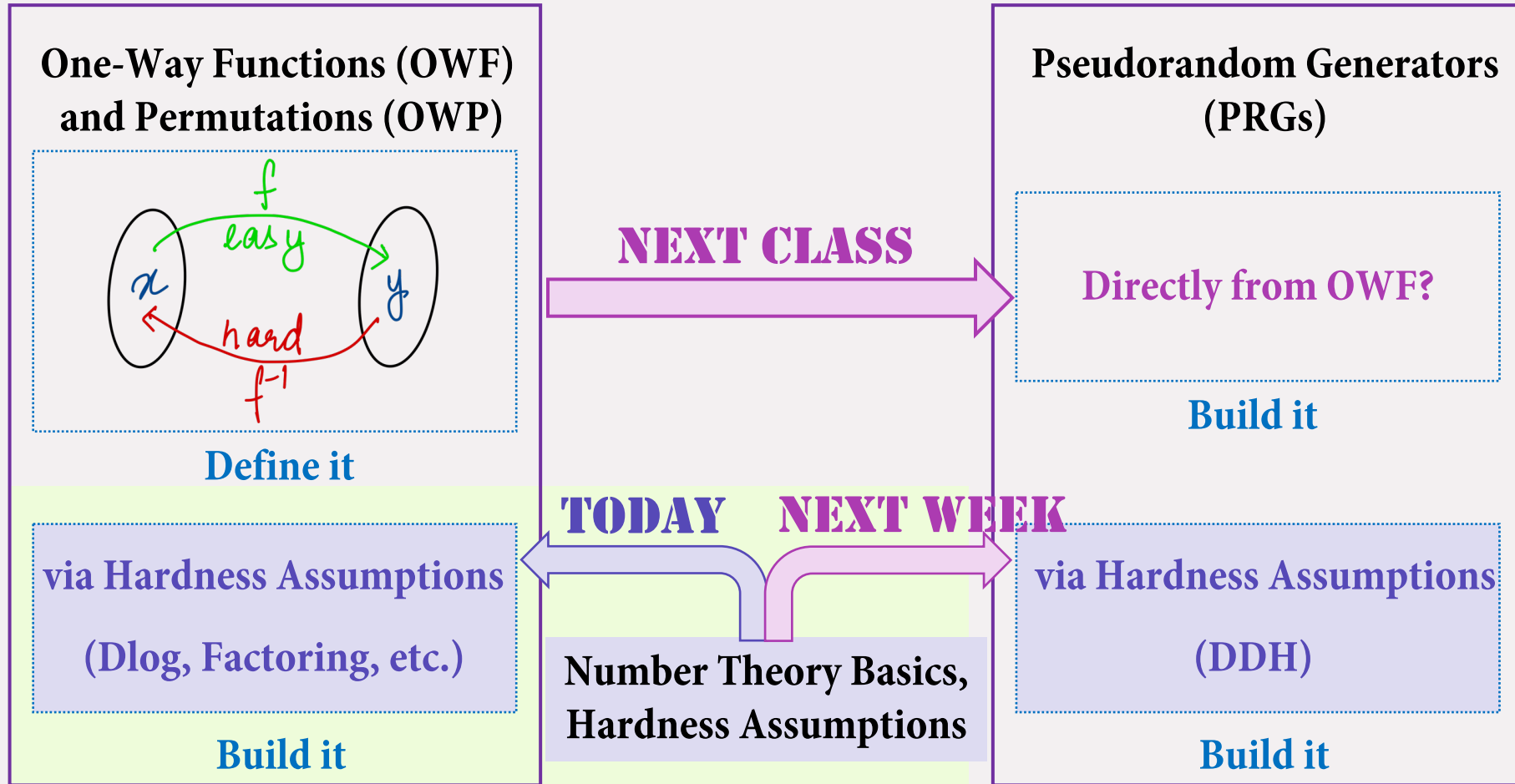
? $f_2(x_1 || x_2) := x_1 || f(x_1 || x_2)$, where f is a OWF

HW 2: Prove security of f_1 and f_3 via a reduction!

HW 1:
Come up with an example of f such that
1. f_2 is one-way and 2. f_2 is invertible.

HOW DO I BUILD OWF FAMILY NOW?

Coming Up..



Coming Up..

Number Theory Basics, Hardness Assumptions

```
graph TD; A[Number Theory Basics, Hardness Assumptions] -.-> B[Groups]; A -.-> C[Cryptographic Hardness Assumptions]; A -.-> D[Computations on Groups];
```

Groups

- Definition and Properties
- Examples

Cryptographic Hardness Assumptions

- Discrete Log Assumption
- Factoring Assumption

Computations on Groups

- Primality testing, generating random primes and random group elements
- Efficient computation on groups
- Chinese Remainder Theorem

Groups: A Visual Example



Suppose I give you a solved Rubik's Cube. Imagine every possible way you can scramble it just by turning the faces — no peeling off stickers!

What is the mathematical object that consists of all these possible moves?



Set: all possible moves

Operation: Composition of different moves

Defining Groups

- Group $(\mathbb{G}, *)$ consists of **set** $\mathbb{G}$ and **operation** $* : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}$ that satisfies:
 - Closure Property**: For all $a, b \in \mathbb{G}$, $a * b \in \mathbb{G}$ (denote $*(a, b)$ as $a * b$)
 - Associativity**: For all $a, b, c \in \mathbb{G}$, $a * (b * c) = (a * b) * c$
 - Identity**: There exists an $e \in \mathbb{G}$ such that $a * e = a = e * a, \forall a \in \mathbb{G}$
 - Inverse**: For every $a \in \mathbb{G}$, there exists $a^{-1} \in \mathbb{G}$ such that $a * a^{-1} = e = a^{-1} * a$
 - Commutativity (additional property)**: For all $a, b \in \mathbb{G}$, $a * b = b * a$
- Order** of a group $|\mathbb{G}|$: number of elements in $\mathbb{G}$.

Abelian
groups

IS THIS A GROUP? WHAT IS THE ORDER?

✓ $\mathbb{Z} = (\text{Integers}, +)$

$\mathbb{Z}_n = (\text{Integers modulo } n, + \bmod n)$ ✓

✗ $\mathbb{R} = (\text{Real Numbers}, \times)$

✗ $\mathbb{Z}_n = (\text{Integers modulo } n, \times \bmod n)$

Properties of Groups

Lagrange's Theorem

For any $g \in \mathbb{G}$, $g^{|\mathbb{G}|} = g * g * \dots * g$ ($|\mathbb{G}|$ times) = identity, i.e., $|g|$ divides $|\mathbb{G}|$

Notation: g^i is used to denote $g * g * \dots * g$ (i times), $|g|$, the order of $g \in \mathbb{G}$ is $\min_i (g^i = \text{identity})$

Finite Cyclic Group

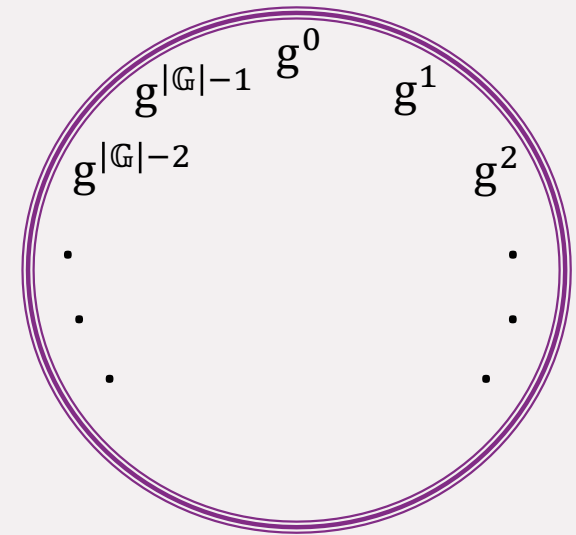
$\exists g \in \mathbb{G}$ such that $\mathbb{G} = \{g^0, g^1, \dots, g^{|\mathbb{G}|-1}\}$. g is called **generator** of the cyclic group.

EXAMPLE:

$\mathbb{Z}_n, + \text{mod } n$: generator?

Theorem: Prime order groups are cyclic

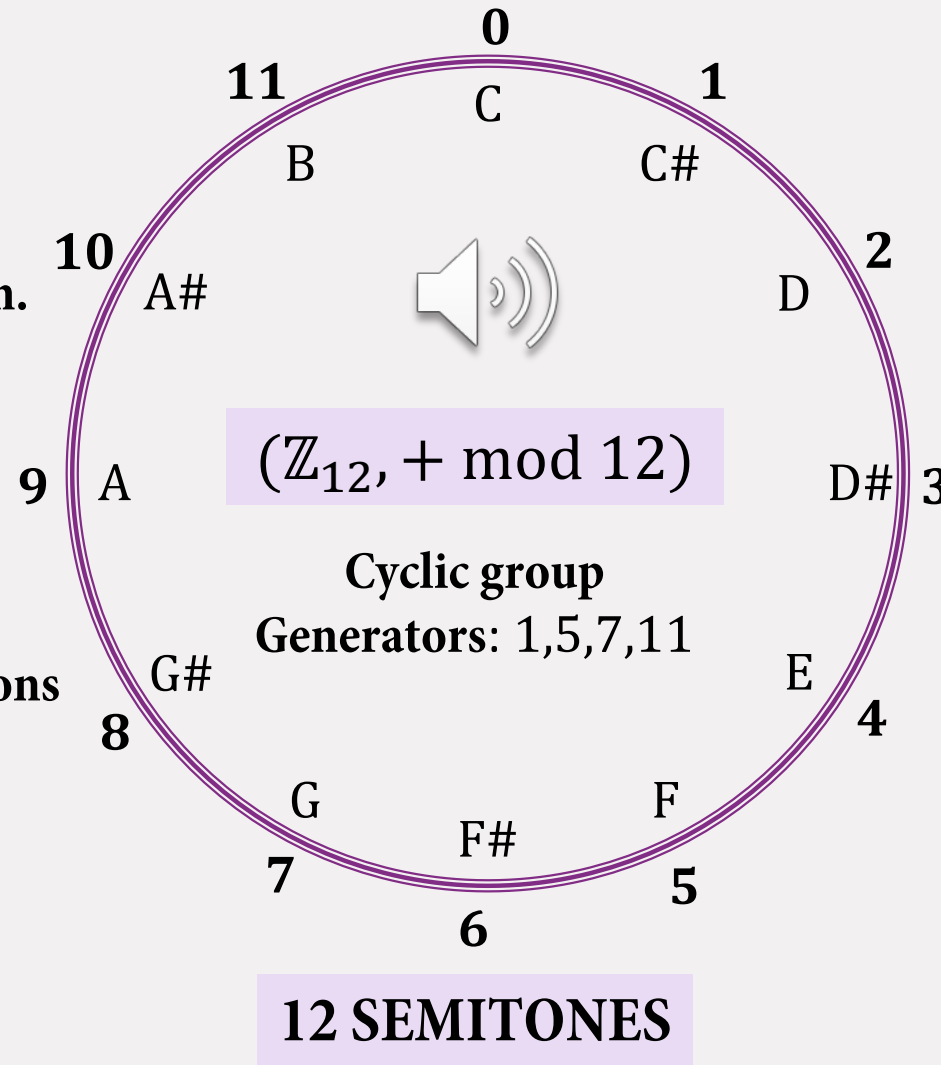
Every group $\mathbb{G}$ with $|\mathbb{G}| = p$, where p is a prime, is cyclic.
Every element other than the identity is a generator of $\mathbb{G}$.



Groups: An Audible Example

Eg.: +7 semitone
(often used in music
compositions to signify heroism.
E.g. Star wars main theme)
CGDAEBF#C#G#D#A#FC

Eg.: +5 semitone
(often used in music compositions
to capture anticipation.
E.g. Hedwig's Theme)
CFA#D#G#C#F#BEADGC



Multiplicative Group $\mathbb{Z}_n^*$

Greatest common divisor

$\gcd(x, n)$: largest positive integer d that divides both x and n

Multiplicative group of integers mod n

$\mathbb{Z}_n^*$, $\times \bmod n$: set consists of integers $1 \leq a < n$ such that $\gcd(a, n) = 1$

Why only integers with **$\gcd 1$** ?



For an integer a , there exists an integer b such that $a \times b = 1 \bmod n$ if and only if $\gcd(a, n) = 1$

Inverse of $a \in \mathbb{Z}_n^*$: Extended Euclidean Algorithm

The algorithm finds two integers X and Y such that $Xa + Yn = \gcd(a, n)$, in $\text{poly}(\|a\|, \|n\|)$ -time.

**** $\|a\|$: number of bits to represent a**



How does this give you the inverse?

Order of $\mathbb{Z}_p^*$ for prime p

$p - 1$

Order of $\mathbb{Z}_n^*$

called **Euler totient function of n : $\varphi(n)$**



Why? For $n = \prod_i p_i^{\alpha_i}$ the prime factorization of n , $\varphi(n) = \prod_i (p_i^{\alpha_i} - p_i^{\alpha_i - 1})$

Multiplicative Group $\mathbb{Z}_n^*$

Greatest common divisor

$\gcd(x, n)$: largest positive integer d that divides both a and b

Multiplicative group of integers mod n

$\mathbb{Z}_n^*$, $\times \bmod n$: set consists of integers $1 \leq a < n$ such that $\gcd(a, n) = 1$

Order of $\mathbb{Z}_n^*$

called **Euler totient function of n : $\varphi(n)$**

For $n = \prod_i p_i^{\alpha_i}$ the prime factorization of n , $\varphi(n) = \prod_i (p_i^{\alpha_i} - p_i^{\alpha_i-1})$



Is $\mathbb{Z}_n^*$ cyclic?

Yes, when $n = p$, for some prime p . Why? Order of the group?
Otherwise, need not be cyclic! Example?

Integer Operations and Modular Arithmetic

Group Operation	Time Complexity	Reference/Algorithm
$a + b$	$O(t)$	High-school algorithm
ab	$O(t \log t)$	Harvey and Van Der Hoeven
$\gcd(a, b)$	$O(t^2)$	Euclidean Algorithm
$a \bmod n$	$O(t^2)$	Division algorithm for integers
$a + b \bmod n, a - b \bmod n$	$O(t^2)$	
$ab \bmod n$	$O(t^2)$	
$a^{-1} \bmod n$	$O(t^2)$	Extended Euclidean algorithm
$a^b \bmod n$	$O(t^3)$	Repeated squaring algorithm
Check if an integer p is prime or not?	$O(t^2 \log 1/\varepsilon)$ $O(t^6)$	Miller-Rabin test (error ε) Agrawal et al. (deterministic)

**** a, b, p be integers of t -bits each.**

Efficient Computation on Groups

Group exponentiation

Given a $g \in \mathbb{G}$ and a non-negative integer b , g^b can be computed using $\text{poly}(\|b\|)$ operations.

Generating a random group element

- Choosing a group element uniformly at random $g \in_R \mathbb{Z}_n$ (randomized algorithm)
 - in time $\text{poly}(\|n\|)$
 - failure probability (not outputting a random value) is negligible $\text{negl}(\|n\|)$(similar algorithm exists for $\mathbb{Z}_n^*$)
- For a general cyclic group $\mathbb{G} = \langle g \rangle$ of order q , to choose uniformly random $h \in_R \mathbb{G}$ do:
 - Pick $x \in_R \mathbb{Z}_q$
 - Compute $h = g^x$(for any finite group $\mathbb{G}$, uniform random elements can be picked for certain representation of $\mathbb{G}$)

Efficient Computation on Groups

Testing and finding generators

Given a cyclic group $\mathbb{G}$ of order q , there exists a $\text{poly}(\|q\|)$ -time algorithm that given q and prime factorization of q

- Given $g \in \mathbb{G}$, decides whether g is a generator or not.
- Outputs a generator of $\mathbb{G}$ except with probability $\text{negl}(\|q\|)$.
(uniformly random from all generators)



How many generators does $\mathbb{G}$ have?

Chinese Remainder Theorem

Group Isomorphism denoted by $\mathbb{G} \equiv \mathbb{H}$

There exists a **bijective map** $\phi: \mathbb{G} \rightarrow \mathbb{H}$ such that $\forall a, b \in \mathbb{G}, \phi(a *_{\mathbb{G}} b) = \phi(a) *_{\mathbb{H}} \phi(b)$

$\mathbb{G} \times \mathbb{H}$ consists of all ordered pairs $(g, h), g \in \mathbb{G}, h \in \mathbb{H}$ with the operation: $(g, h) * (g', h') := (g *_{\mathbb{G}} g', h *_{\mathbb{H}} h')$

Chinese Remainder Theorem (CRT):

Let $N = pq$ where $p, q > 1$ are relatively prime (i.e., $\gcd(p, q) = 1$), then

(additive group) $\mathbb{Z}_N \equiv \mathbb{Z}_p \times \mathbb{Z}_q$ and $\mathbb{Z}_N^* \equiv \mathbb{Z}_p^* \times \mathbb{Z}_q^*$ (multiplicative group)

With the same bijection map

$$\phi(x) := ([x \bmod p], [x \bmod q])$$

In the birthday demo we started with: we needed to efficiently find the inverse of this map.

$$\phi^{-1}((x_1, x_2)) := (c_1 x_1 + c_2 x_2) \bmod N, \text{ where } c_1 = q \cdot (q^{-1} \bmod p), c_2 = p \cdot (p^{-1} \bmod q)$$

**** ϕ^{-1} or ϕ usually need not be efficiently computable, CRT is an example where both are!**

Chinese Remainder Theorem

Where does isomorphism help?

Given $\mathbb{G} \equiv \mathbb{H}$ with $\cdot$, one can choose to do group computations in the group where it is more efficient!

EXAMPLES

$N = pq$ for distinct primes p and q

CRT tells you that addition, multiplication and group exponentiation **mod N** can be **transformed to analogous operations mod p and mod q** .

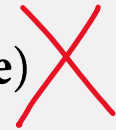
1. **Want to compute:** $14 \times 13 \bmod 15$ (in $\mathbb{Z}_{15}^*$) $\phi_{\text{CRT}}(14) = (14 \bmod 5, 14 \bmod 3) = (4, 2)$
 $\phi_{\text{CRT}}(13) = (13 \bmod 5, 13 \bmod 3) = (3, 1)$

$$(4, 2) * (3, 1) = ((4 \times 3 \bmod 5), (2 \times 1 \bmod 3)) = (2, 2) \quad \phi_{\text{CRT}}^{-1}((2, 2)) = 2 = 14 \times 13 \bmod 15$$

2. **Want to compute:** $29^{100} \bmod 35$ $\phi_{\text{CRT}}(29) = (29 \bmod 5, 29 \bmod 7) = (-1 \bmod 5, 1)$

$$(-1 \bmod 5, 1)^{100} = ((-1)^{100} \bmod 5, 1) = (1, 1) \quad \phi_{\text{CRT}}^{-1}((1, 1)) = 1 = 29^{100} \bmod 35$$

One Way Function Family

$f_{a,p}(x) := ax \bmod p$, where $a, x \in \mathbb{Z}_p^*$ and p is a prime. (Multiplication mod prime) 

$f_p(x) := x^2 \bmod p$, where $x \in \mathbb{Z}_p^*$ and p is a prime. (Squaring mod prime) 

$f_{p,g}(x) := g^x \bmod p$, where $x \in \mathbb{Z}_{p-1}$, $\mathbb{Z}_p^* = \langle g \rangle$, p is a prime. (Exponentiation mod prime) 

(If “*Dlog Assumption*” for $\mathbb{Z}_p^*$ then $\{f_{p,g}\}$ is a OWF family)

Discrete Log Assumption (intuitively)

The exponentiation function

$$f_g: \mathbb{Z}_q \rightarrow \mathbb{G}: f_g(x) = g^x$$

It's a group isomorphism.

 inverse

Discrete Log with respect to g

$$\text{Dlog}_g(h) := \text{unique } x \in \mathbb{Z}_q \text{ such that } h = g^x, \text{ for each } h \in \mathbb{G}$$

Cyclic Group $(\mathbb{G}, q, g)$

Group Order Generator



Is f_g invertible efficiently?

In “some groups”, for random $h \in_R \mathbb{G}$ it is “hard” to find $\text{Dlog}_g(h)$

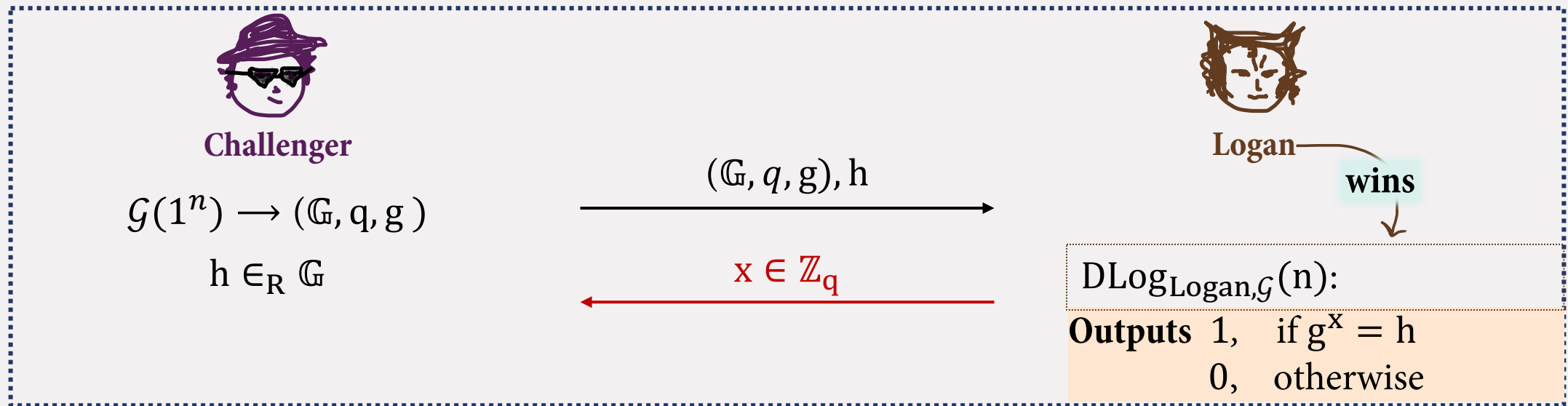
Example: Is it hard in this group?

$(\mathbb{Z}_N, + \bmod N)$

$(\mathbb{Z}_p^*, \times \bmod p)$

Discrete Log Assumption

Some generic, polytime **group generation algorithm** $\mathcal{G}(1^n) \rightarrow (\mathbb{G}, q, g)$, where $\|q\| = n$ and the description of group specifies how to represent each element as a unique bit-string, such that one can compute efficiently on the elements of $\mathbb{G}$.



Discrete Log Assumption

The **discrete log problem is hard relative to $\mathcal{G}$** if for all **PPT Logan**, there exists a negligible negl such that, $\forall n$

$$\Pr[\text{Dlog}_{\text{Logan}, \mathcal{G}}(n) = 1] \leq \text{negl}(n)$$

Discrete Log Assumption

Why do we believe this is a hard assumption?

Best known algorithm to find discrete log in an **arbitrary group** $\mathbb{G}$ runs in **time** $O(\sqrt{|\mathbb{G}|})$
(Baby-step giant-step algorithm)

Can we do better?

- For $\mathbb{Z}_p^*$, there are better algorithms:

Index Calculus Algorithm ($2^{O(\sqrt{\log y \log \log y})}$),

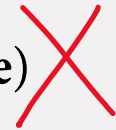
Number Field Sieve Algorithm ($2^{O((\log y)^{1/3} (\log \log y)^{2/3})}$), where $y \sim \sqrt{|\mathbb{G}|}$

- For finite abelian group with “smooth integer” order:

Pohlig-Hellman Algorithm (if $|\mathbb{G}| = N = \prod_i p_i^{\alpha_i}$, runs in time $O(\sum_i \alpha_i (\log N + \sqrt{p_i}))$)

Moral of the story: Yes, you can do better than the naïve algorithm, but still the time complexity is exponential for arbitrary groups, and sub-exponential for certain specific groups.

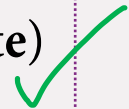
One Way Function Family

$f_{a,p}(x) := ax \bmod p$, where $a, x \in \mathbb{Z}_p^*$ and p is a prime. (Multiplication mod prime) 

$f_p(x) := x^2 \bmod p$, where $x \in \mathbb{Z}_p^*$ and p is a prime. (Squaring mod prime) 

$f_{p,g}(x) := g^x \bmod p$, where $x \in \mathbb{Z}_{p-1}$, $\mathbb{Z}_p^* = \langle g \rangle$, p is a prime. (Exponentiation mod prime) 

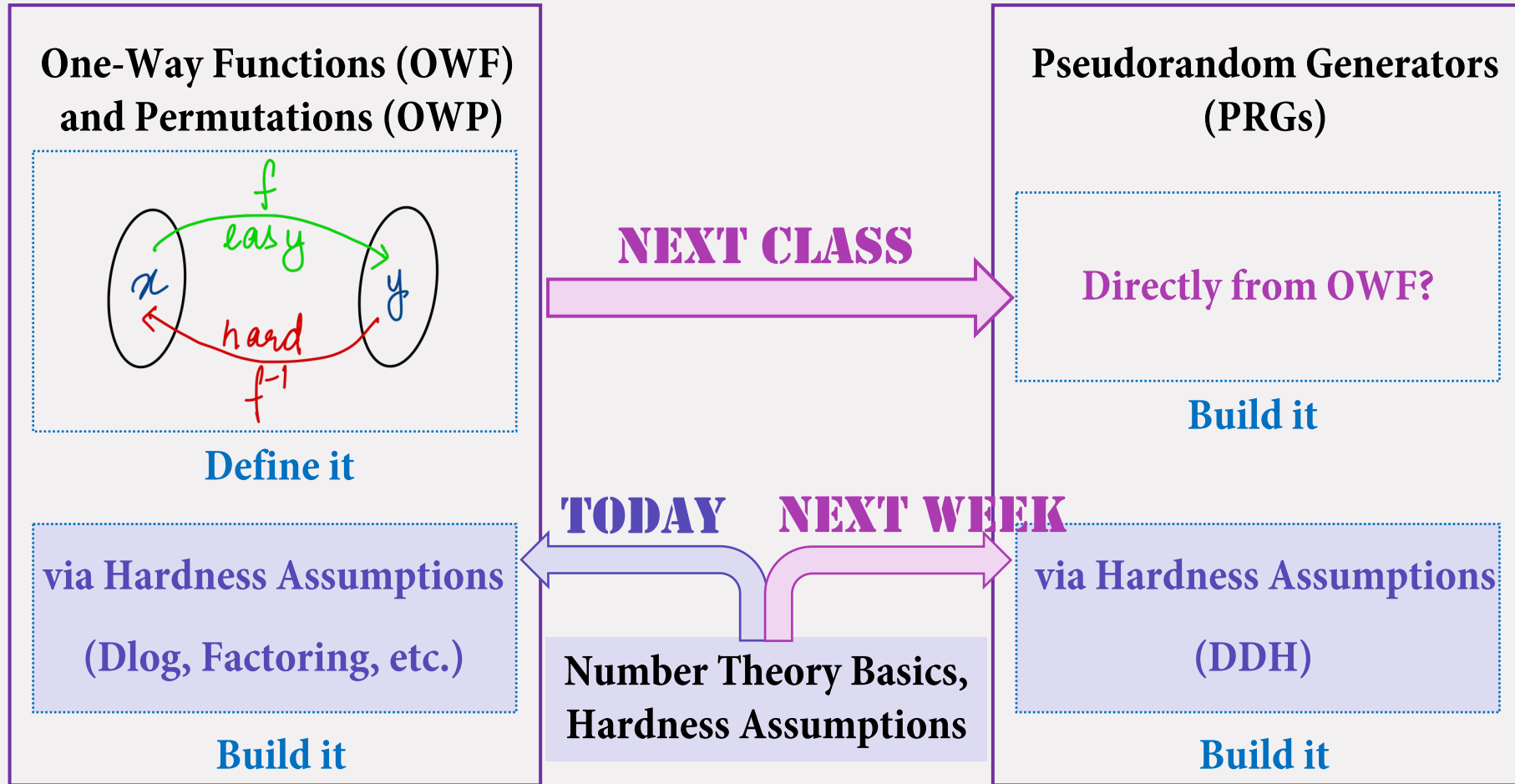
(If Dlog Assumption holds (in $\mathbb{Z}_p^*$ it is believed to hold) then $\{f_{p,g}\}$ is a OWF family)

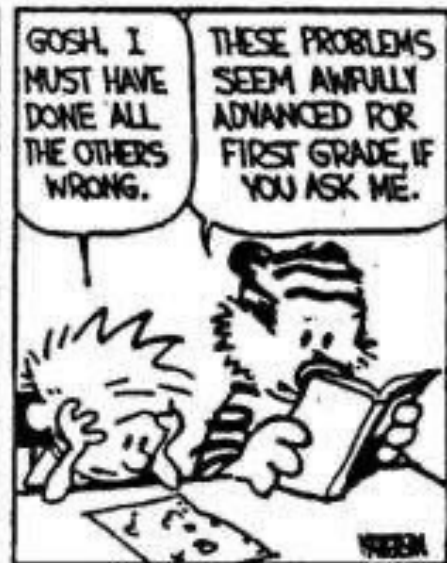
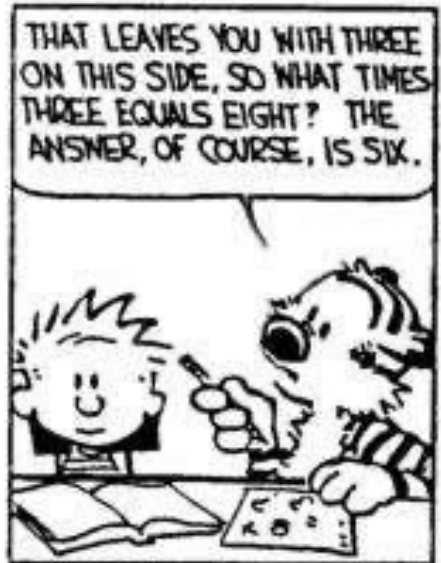
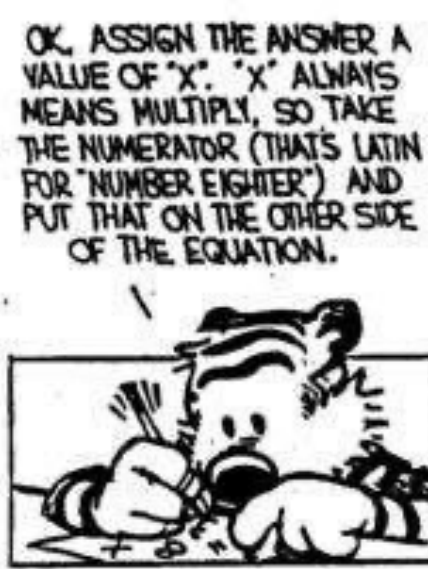
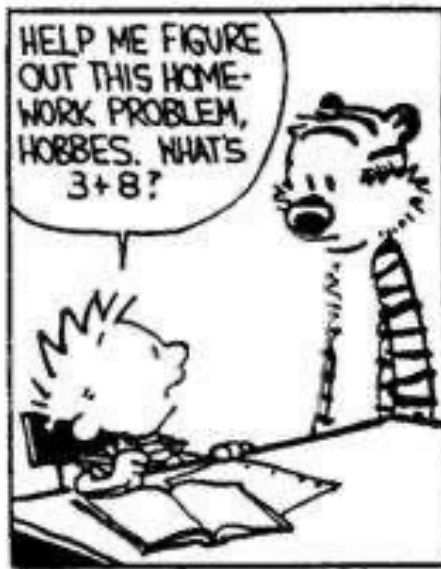
$f_N(x) := x^2 \bmod N$, where $x \in \mathbb{Z}_N^*$ and $N = pq$ (large primes p and q). (Squaring mod composite) 

(If factoring N is hard (factoring assumption), then $\{f_N\}$ is a OWF family)

HW 3: Show that taking square root mod N
is equivalent to factoring N
(Hint: $x^2 - y^2 = (x + y)(x - y) \bmod N$), use CRT!)

Coming Up..





THANK YOU