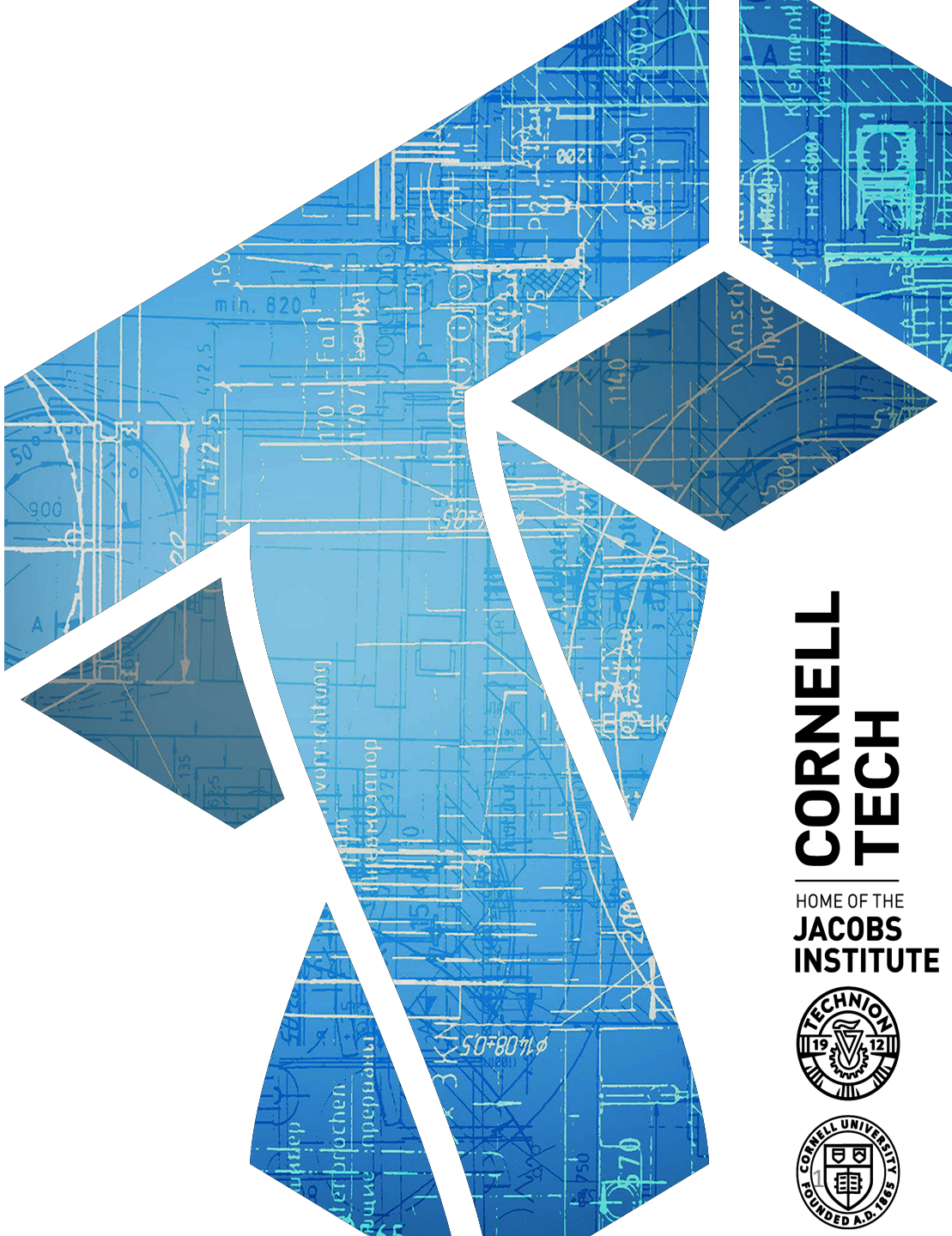
CS 5830
Cryptography

Instructor: Tom Ristenpart
TAs: Yan Ji, Sanketh Menda

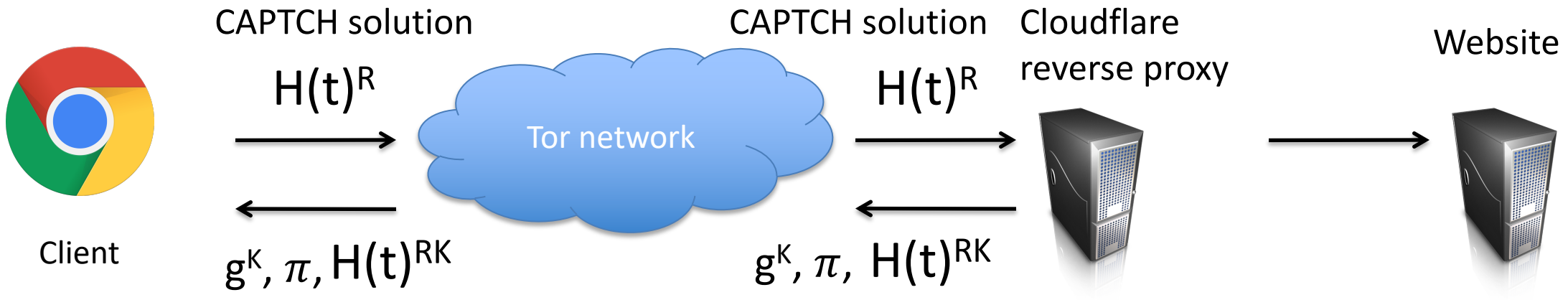


CORNELL TECH
HOME OF THE JACOBS INSTITUTE

Last time: ZKPs and Schnorr

- Recall we wanted a way to build ***verifiable*** OPRF
  - Prove to clients that server is using secret key consistently
- ZKPs provide this ability
  - Sigma protocols are standard 3-round interactive ZKP for discrete log type proofs
  - Fiat-Shamir heuristic allows making Sigma protocol ***non-interactive***
  - Interesting formal relationship with identification protocols

Privacy Pass



$t \leftarrow\$ \{0,1\}^n$
$R \leftarrow\$ \mathbf{Z}_p$
$T = H(H(t)^K)$

Verify π
- If fails, then abort
- Otherwise can use tokens

This is called a ***verifiable OPRF***. In words, it allows ensuring consistency (relative to public key g^K) of server behavior

How do prove that $dlog_g(g^K) = dlog_Y(Y^K)$ where $Y = H(t)^R$?

Zero-knowledge proofs

- Prove something about secret x without revealing any further information about x
- Security properties:
  - Zero-knowledge: can simulate proof without secret
  - Soundness: no malicious prover, without secret, can trick verifier

Schnorr protocol

$r \leftarrow\$ \mathbb{Z}_q$

$R = g^r$

Prover

Knows sk = x

R

c

z = cx+r

Verifier

Knows pk = $X = g^x$

$c \leftarrow\$ \mathbb{Z}_q$

$z = cx + r$

Check that $g^z = RX^c$

Chaum-Pedersen Sigma protocol (DL equality!)

$r \leftarrow_{\$} \mathbb{Z}_p$

$R_1 \leftarrow g^r$

$R_2 \leftarrow Y^r$

Prover

Knows x s.t. $X = g^x$

R_1, R_2

c

z

Verifier

Knows X, Y, W, g

c <-$ $\mathbb{Z}_q$

$z \leftarrow r + xc$

Check both that:

$$g^z = R_1 \cdot X^c$$

$$Y^z = R_2 \cdot W^c$$

Verifier wants to check that $\text{dlog}_g(X) = \text{dlog}_Y(W)$

$X = g^x$ $Y = g^y$ $W = g^{yx}$

$$Y^z = Y^{r+xc} = Y^r \cdot Y^{xc} = R_2 \cdot W^c$$

Chaum-Pedersen Sigma protocol (DL equality!)

$r \leftarrow_{\$} \mathbb{Z}_p$

$R_1 \leftarrow g^r$

$R_2 \leftarrow Y^r$

Prover

Knows x s.t. $X = g^x$

R_1, R_2

c

z

Verifier

Knows X, Y, W, g

c <-$ $\mathbb{Z}_q$

$z \leftarrow r + xc$

Check both that:

$$g^z = R_1 \cdot X^c$$

$$Y^z = R_2 \cdot W^c$$

Verifier wants to check that $\text{dlog}_g(X) = \text{dlog}_Y(W)$

$X = g^x$ $Y = g^y$ $W = g^{yx}$

In our verifiable OPRF application: $X = g^K$, $Y = H(t)^R$, $W = H(t)^{RK}$

Why not Schnorr protocol twice?

$r1 \leftarrow\$ \mathbf{Z}_q$ $r2 \leftarrow\$ \mathbf{Z}_q$

$R_1 = g^{r1}$ $R_2 = Y^{r2}$

Prover

Knows sk = x

R_1, R_2

c1, c2

z1,z2

Verifier

Knows X, Y, W, g

$c1,c2 \leftarrow\$ \mathbf{Z}_q$

$z1 = c1\ x + r1$

$z2 = c1\ x + r2$

Check that:

$g^{z1} = R_1\ X^{c1}$

$Y^{z2} = R_2\ W^{c2}$

Correctness:

$g^{z1} = g^{c1x+r1} = R_1\ X^{c1}$ $Y^{z2} = g^{y(c1x + r2)} = g^{y*r2 + y*x*c1} = R_2\ W^{c2}$

Security?

Why not Schnorr protocol twice?

$r1 \leftarrow\$ \mathbf{Z}_q$ $r2 \leftarrow\$ \mathbf{Z}_q$

$R_1 = g^{r1}$ $R_2 = Y^{r2}$

Prover

Knows sk = x

R_1, R_2

c1, c2

z1,z2

Verifier

Knows X, Y, W, g

$c1,c2 \leftarrow\$ \mathbf{Z}_q$

$z1 = c1\ x + r1$

$z2 = c1\ x + r2$

Check that:

$g^{z1} = R_1\ X^{c1}$

$Y^{z2} = R_2\ W^{c2}$

Correctness:

$g^{z1} = g^{c1x+r1} = R_1\ X^{c1}$ $Y^{z2} = g^{y(c1x\ +\ r2)} = g^{y*r2\ +\ y*x*c1} = R_2\ W^{c2}$

Security? No guarantee that $dlog_g(X) = dlog_Y(W)$!!!

Set $X = g^x$, $Y = g^y$, $W = g^{y\ w}$. Can build satisfying proof above

R_1, R_2 as before. $z1 = c1\ x + r1$ and $z2 = c2\ w + r2$

Chaum-Pedersen Sigma protocol (DL equality!)

$r \leftarrow_{\$} \mathbb{Z}_p$

$R_1 \leftarrow g^r$

$R_2 \leftarrow Y^r$

Prover

Knows x s.t. X = g^x

R_1, R_2

c

z

Verifier

Knows X, Y, W, g

c <-$ Z_q

$z \leftarrow r + xc$

Check both that:

$$g^z = R_1 \cdot X^c$$

$$Y^z = R_2 \cdot W^c$$

Verifier wants to check that $dlog_g(X) = dlog_Y(W)$

$X = g^x$ $Y = g^y$ $W = g^{yx}$

In our verifiable OPRF application: $X = g^K$, $Y = H(t)^R$, $W = H(t)^{RK}$

Security of Chaum-Pedersen

$r \leftarrow_{\$} \mathbb{Z}_p$

$R_1 \leftarrow g^r$

$R_2 \leftarrow Y^r$

Prover

Knows x s.t. X = g^x

R_1, R_2

c

z

Verifier

Knows X, Y, W, g

c <-$ $\mathbb{Z}_q$

$z \leftarrow r + xc$

Check both that:

$$g^z = R_1 \cdot X^c$$

$$Y^z = R_2 \cdot W^c$$

Zero-knowledge:

Simulator can generate transcript (R_1,R_2,c,z) consistent with X

Choose c, z at random

Compute $R_1 = g^z X^{-c}$ and $R_2 = Y^z W^{-c}$

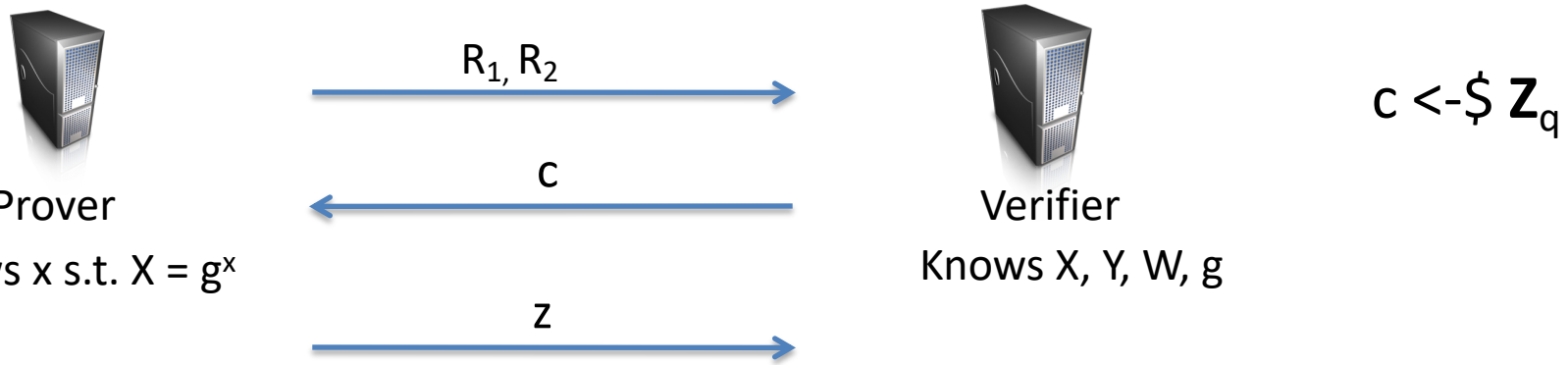
Result is indistinguishable from real transcript

Security of Chaum-Pedersen

$r \leftarrow_{\$} \mathbb{Z}_p$

$R_1 \leftarrow g^r$

$R_2 \leftarrow Y^r$



c <-$ $\mathbb{Z}_q$

$z \leftarrow r + xc$

Check both that:

$$g^z = R_1 \cdot X^c$$

$$Y^z = R_2 \cdot W^c$$

Soundness:

Suppose adversarial prover A can generate valid transcript, given just X

We can build adversary that finds x = dlog_g X

Run A twice, giving it c in first run and c' ≠ c in second

Extract x from z = r + xc and z' = r + xc'

IDPASS Security of Chaum-Pedersen

Recall ID protocol formalization as P.com, P.resp, V.ver and challenge space

Chaum-Pedersen: P.com = $R_i = (R_{1,i}, R_{2,i})$

Intuition for security:

1. Can simulate R_i , c_i , z_i due to zero-knowledge
2. Build DL-adversary *B* that runs *A* twice to get two transcripts that allow extracting sk = x

Good treatment of this in Boneh-Shoup textbook (Part III)

$\underline{\mathrm{IDPASS}^{\mathcal{A}}_{\mathsf{ID},q_{id}}}$

$(pk, sk) \leftarrow\$ \mathsf{IDkg}$

$\mathsf{queried} \leftarrow \mathsf{false}$

For $i = 1$ to q_{id} do

  $R_i \leftarrow\$ P.com(pk)$

  $c_i \leftarrow\$ \mathcal{C}$

  $z_i \leftarrow\$ P.resp(sk, R_i, c_i)$

$z^* \leftarrow\$ \mathcal{A}^{\mathrm{Ver}}(pk, (R_1, c_1, z_1), \ldots, (R_{q_{id}}, c_{q_{id}}, z_{q_{id}}))$

Ret $V.ver(pk, R^*, c^*, z^*)$

$\underline{\mathrm{Ver}(R^*)}$

If $\mathsf{queried} = \mathsf{true}$ then Ret $\perp$

$\mathsf{queried} \leftarrow \mathsf{true}$

$c^* \leftarrow\$ \mathcal{C}$

Ret c^*

Fiat-Shamir transform to make NIZK

$r \leftarrow_{\$} \mathbb{Z}_p$

$R_1 \leftarrow g^r$

$R_2 \leftarrow Y^r$

Prover

Knows x s.t. $X = g^x$

R_1, R_2

c

z

Verifier

Knows X, Y, W, g

c <-$ $\mathbf{Z}_q$

$z \leftarrow r + xc$

Check both that:

$$g^z = R_1 \cdot X^c$$

$$Y^z = R_2 \cdot W^c$$

Signature is commitment and response. Challenge generated non-interactively with hash

Works on other Sigma protocols as well

<u>Prove(x,(g,X,Y,W))</u>

r <-$ $\mathbf{Z}_q$; R_1 <- g^r ; R_1 <- Y^r

c = H(R_1 || R_2) ; z = r + cx mod q

π = (R_1,R_2,z)

Return π

<u>Ver((g,X,Y,W), π)</u>

c = H(R_1 || R_2)

If $g^z = R_1 \cdot X^c$ and $Y^z = R_2 \cdot W^c$ then

Return 1

Return 0

NIZKs: defining security

NIZK zero-knowledge security ("plain model")
Provide simulator Sim such that

$$\mathrm{Adv}^{\mathrm{nizk}}_{\mathrm{CP},g,Y}(\mathcal{A},\mathrm{Sim}) = 2 \cdot \Pr\left[\,\mathrm{NIZK}^{\mathcal{A}}_{\mathrm{CP},g,Y} \Rightarrow \mathit{true}\,\right] - 1$$

is tiny (e.g., 2^{-80})

Question:
Can ***any*** scheme achieve this notion?

No! Giving such a Sim would directly contradict soundness

$\underline{\mathrm{NIZK}^{\mathcal{A}}_{\mathrm{CP},g,Y}}$
$b \leftarrow\!\!\$\ \{0,1\}$
$x \leftarrow\!\!\$\ \mathbb{Z}_p$
$X \leftarrow g^x \ \ ; \ \ W \leftarrow Y^x$
$\pi_1 \leftarrow\!\!\$\ \mathsf{Prove}(x,(g,X,Y,W))$
$\pi_0 \leftarrow\!\!\$\ \mathsf{Sim}((g,X,Y,W))$
$b' \leftarrow\!\!\$\ \mathcal{A}((g,X,Y,W),\pi_b)$
$\mathrm{Ret}\ b'$

NIZKs: defining security

NIZK zero-knowledge security ("plain model")
Provide simulator Sim such that

$$\mathrm{Adv}^{\mathrm{nizk}}_{\mathrm{CP},g,Y}(\mathcal{A},\mathrm{Sim}) = 2 \cdot \Pr\left[\,\mathrm{NIZK}^{\mathcal{A}}_{\mathrm{CP},g,Y} \Rightarrow \mathit{true}\,\right] - 1$$

is tiny (e.g., 2^{-80})

Question:
Can ***any*** scheme achieve this notion?

No! Giving such a Sim would directly contradict soundness

$\underline{\mathrm{NIZK}^{\mathcal{A}}_{\mathrm{CP},g,Y}}$
$b \leftarrow\!\!\$\ \{0,1\}$
$x \leftarrow\!\!\$\ \mathbb{Z}_p$
$X \leftarrow g^x \ ;\ W \leftarrow Y^x$
$\pi_1 \leftarrow\!\!\$\ \mathsf{Prove}(x,(g,X,Y,W))$
$\pi_0 \leftarrow\!\!\$\ \mathsf{Sim}((g,X,Y,W))$
$b' \leftarrow\!\!\$\ \mathcal{A}((g,X,Y,W),\pi_b)$
$\mathrm{Ret}\ b'$

$\underline{\mathrm{SOUND}^{\mathcal{B}}_{\mathrm{CP},g,Y}}$
$x \leftarrow\!\!\$\ \mathbb{Z}_p$
$X \leftarrow g^x \ ;\ W \leftarrow Y^x$
$\pi^* \leftarrow\!\!\$\ \mathcal{B}(g,X,Y,W)$
$\mathrm{Ret}\ \mathrm{Verify}((g,X,Y,W),\pi^*)$

NIZKs: security in the ROM

NIZK zero-knowledge security in the ROM
Provide simulator Sim such that

$$\mathrm{Adv}^{\text{nizk-ro}}_{\mathrm{CP},g,Y}(\mathcal{A},\mathrm{Sim}) = 2 \cdot \Pr\left[\,\text{NIZK-RO}^{\mathcal{A}}_{\mathrm{CP},g,Y} \Rightarrow \mathit{true}\,\right] - 1$$

is tiny (e.g., 2^{-80})

Hash function H modeled as ***random oracle***:

- Random function
- Adversary has oracle access to it
- In ideal world, Sim gets to define returned values (called "programming" the RO)

$\underline{\text{NIZK-RO}^{\mathcal{A}}_{\mathrm{CP},g,Y}}$
$b \leftarrow\!\!\$\ \{0,1\}$
$x \leftarrow\!\!\$\ \mathbb{Z}_p$
$X \leftarrow g^x \ ;\ W \leftarrow Y^x$
$\pi_1 \leftarrow\!\!\$\ \mathsf{Prove}^{\mathrm{Hash}}(x,(g,X,Y,W))$
$\pi_0 \leftarrow\!\!\$\ \mathsf{Sim}(g,X,Y,W)$
$b' \leftarrow\!\!\$\ \mathcal{A}^{\mathrm{Hash}}((g,X,Y,W),\pi_b)$
Ret b'

$\underline{\mathrm{Hash}(M)}$
If $\mathsf{H}[M] = \perp$ then
  If $b = 1$ then $\mathsf{H}[M] \leftarrow\!\!\$\ \mathbb{Z}_p$
  $\mathsf{H}[M] \leftarrow\!\!\$\ \mathsf{Sim}(M)$
Return $\mathsf{H}[M]$

NIZKs: security in the ROM

Chaum-Pedersen NIZK zero-knowledge in ROM

Sim(g,X,Y,W):

- Choose c and z randomly
- Set $R_1 = g^z X^{-c}$ and $R_1 = Y^z W^{-c}$
- Output $\pi = (R_1, R_2, z)$
- Set $H(R_1 || R_2) = c$ when Hash() called on $M = R_1 || R_2$, otherwise set to randomly chosen point

For adversary A, identical view in b=0 and b=1 worlds

$\underline{\text{NIZK-RO}^{\mathcal{A}}_{\text{CP},g,Y}}$

$b \leftarrow\$ \{0,1\}$
$x \leftarrow\$ \mathbb{Z}_p$
$X \leftarrow g^x \;\; ; \;\; W \leftarrow Y^x$
$\pi_1 \leftarrow\$ \text{Prove}^{\text{Hash}}(x, (g, X, Y, W))$
$\pi_0 \leftarrow\$ \text{Sim}(g, X, Y, W)$
$b' \leftarrow\$ \mathcal{A}^{\text{Hash}}((g, X, Y, W), \pi_b)$
Ret b'

$\underline{\text{Hash}(M)}$
If $\text{H}[M] = \perp$ then
  If $b = 1$ then $\text{H}[M] \leftarrow\$ \mathbb{Z}_p$
  $\text{H}[M] \leftarrow\$ \text{Sim}(M)$
Return $\text{H}[M]$

<u>Prove(x,(g,X,Y,W))</u>

r <-$ $\mathbf{Z}_q$; R_1 <- g^r ; R_1 <- Y^r
c = H(R_1 || R_2) ; z = r + cx mod q
$\pi = (R_1, R_2, z)$
Return π

<u>Ver((g,X,Y,W), π)</u>

c = H(R_1 || R_2)
If $g^z = R_1 \cdot X^c$ and $Y^z = R_2 \cdot W^c$ then
  Return 1
Return 0

NIZKs: security in the ROM

Chaum-Pedersen NIZK soundness in ROM

Show reduction to DL problem

- Guess that j[th] Hash query is associated to B's attempt at proof π^*
- Run adversary B twice, feeding different reply c and c'
- If B succeeds, extract secret x from two proofs

Works if guess is right (happens with probability 1 / q_{ro})

B succeeds both times

$\underline{\mathrm{SOUND}^{\mathcal{B}}_{\mathrm{CP},g,Y}}$

$x \leftarrow\$ \mathbb{Z}_p$

$X \leftarrow g^x \;\; ; \;\; W \leftarrow Y^x$

$\pi^* \leftarrow\$ \mathcal{B}^{\mathrm{Hash}}(g, X, Y, W)$

$\mathrm{Ret\ Verify}^{\mathrm{Hash}}((g, X, Y, W), \pi^*)$

$\underline{\mathrm{Hash}(M)}$

$\mathrm{If\ H}[M] = \perp \mathrm{\ then}$

$\quad \mathrm{H}[M] \leftarrow\$ \mathbb{Z}_p$

$\mathrm{Return\ H}[M]$

Prove(x,(g,X,Y,W))

r <-$ $\mathbf{Z}_q$; R_1 <- g^r ; R_1 <- Y^r

c = H(R_1 || R_2) ; z = r + cx mod q

π = (R_1,R_2,z)

Return π

Ver((g,X,Y,W), π)

c = H(R_1 || R_2)

If $g^z = R_1 \cdot X^c$ and $Y^z = R_2 \cdot W^c$ then

Return 1

Return 0

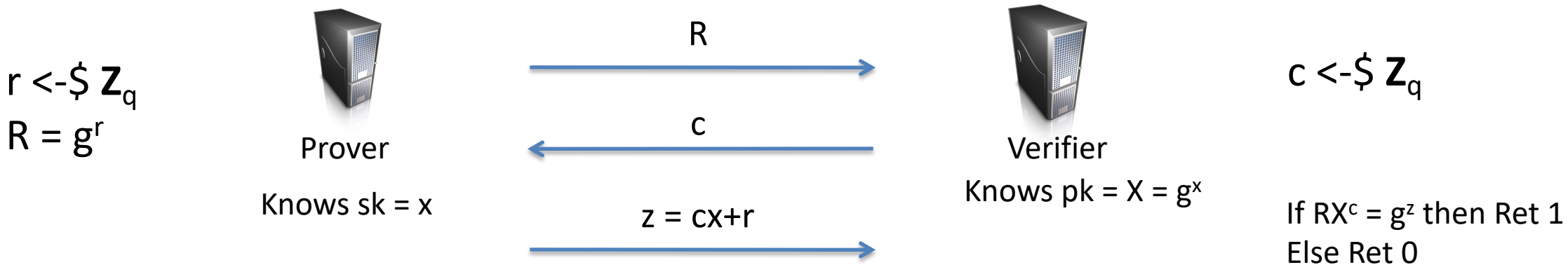
NIZK definitions

- Definitions given tailored to Chaum-Pedersen setting
  - Can derive more general ones
- Common reference string (CRS) model another route for NIZK protocols
  - Protocols make use of a CRS that (typically) encodes a public-key for some kind of encryption
  - Simulator can choose the CRS (and know a trapdoor)
  - Approach generally quite fragile, painful for practice: by design these have trapdoors that break security, and need special care
    - See the ZCash parameter selection story

Random oracle model debates

- ROM suggested as model in Bellare-Rogaway 1993 paper
  - Built off earlier ideas from Fiat-Shamir and Goldreich, Goldwasser, & Micali
- Provides only known reduction-based security analyses for a wide array of important in-use schemes
- Uninstantiability results of Canetti, Goldreich, Halevi
  - Exist schemes that are secure in ROM, but insecure for any real hash function H
  - Specially constructed counter-examples
- Pragmatic issues have arisen rarely: length-extension attacks
- ROM been a good predictor for security in practice

Fiat-Shamir transform for signatures



Signature is commitment and response. Challenge generated non-interactively with hash function
Works on other Sigma protocols as well

<u>Sign(x, M)</u>
r <-$ Z_q
R = g^r ; c = H(M || R) ; z = r + cx mod q
Return (R,z)

<u>Ver(X, M, (R,z))</u>
c = H(M || R)
If $g^z = RX^c$ then Return 1
Return 0

Putting it all together: UF-CMA to DL

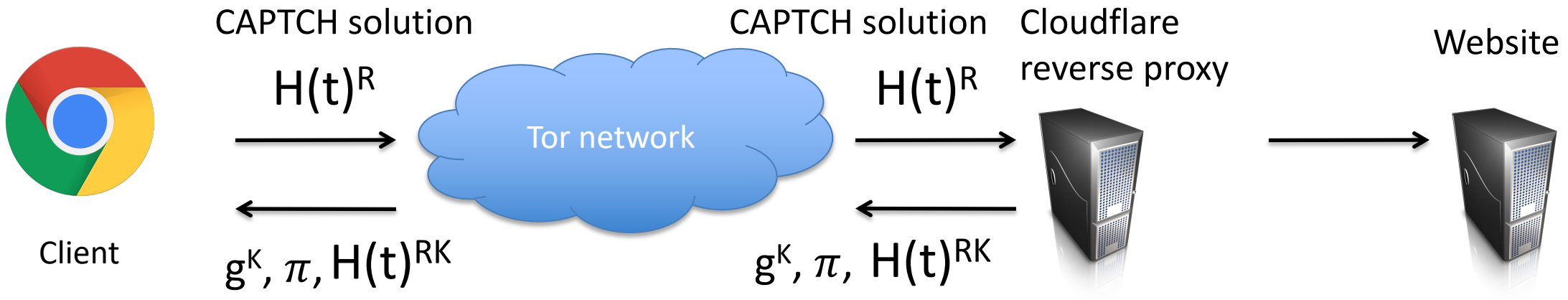
Theorem. *Let G be a cyclic group and* DS *be the Schnorr digital signature scheme using random oracle* $\mathsf{H}\colon \mathcal{M} \to \mathbb{Z}_{|G|}$. *Let $\mathcal{A}$ be an* $\text{UF-CMA}_{\mathsf{DS}}$*-adversary making at most q_h RO queries and q_s signing queries. Then we give an* DL_G*-adversary $\mathcal{B}$ such that*

$$\mathbf{Adv}^{\text{uf-cma}}_{\mathsf{DS}}(\mathcal{A}) \leq \frac{q_s(q_s+q_h+1)}{|G|} + (q_h+1)\left(\sqrt{\mathbf{Adv}^{\text{dl}}_{G}(\mathcal{B})} + \frac{1}{|G|}\right) .$$

Adversary $\mathcal{B}$ runs in time at most twice that of the sum of the running time of $\mathcal{A}$ and $\mathcal{O}(q_h + q_s)$.

Analysis uses IDPASS security of Schnorr, also in the ROM due to use of Fiat-Shamir

Privacy Pass



$t \leftarrow\$ \{0,1\}^n$
$R \leftarrow\$ \mathbf{Z}_p$
$T = H(H(t)^K)$

Verify π
- If fails, then abort
- Otherwise can use tokens

This is called a ***verifiable OPRF***. In words, it allows ensuring consistency (relative to public key g^K) of server behavior

π is a Chaum-Pedersen NIZK of DL equality
- Soundness: Client knows that same K is used, rules out tagging attacks
- Zero knowledge: Client can't use π to learn something about K

NIZKs for General NP Languages

- Can we build NIZKs for arbitrary statements?
  - Interactive proofs for NP long known [Goldreich, Micali, Wigderson 1991]
    - Protocol for proving knowledge of 3-coloring of graph
- CRS-based NIZKs [Groth, Ostrovsky Sahai 2006]
  - Convert statement to circuit C
  - Bilinear-pairing based NIZK whose proofs are size $O(|C|)$
- Groth 2010
  - Show how to do bilinear-pairing based scheme with constant size proofs.
  - Uses knowledge-of-exponent assumptions
- zkSNARKs
  - More efficient representation than circuits: quadratic span programs
  - Then use Groth-style bilinear pairing proofs
  - Pinocchio system: 288 byte proofs of arbitrary programs

Emerging topics in applied crypto

- We'll spend a couple lectures covering some cryptographic tools that have started seeing use in practice recently
  - Secure computation
  - Oblivious PRFs
  - Zero-knowledge proofs

- Theory goes back decades, but real-world applications were historically elusive