Announcements

HW6 Due SUNDAY
HW7 Due April 30th

Project 3 will be combined with HW 8, due on Dean’s date
Project 2 Debrief

Motivation: Cryptocurrencies

IOTA cryptocurrency used P-CURL hash function
• Sponge construction with SPN network
• S-box had bad differentials
• Let to collision-finding attacks
Project 2 Debrief

128 digits base 4

0 $\oplus$ f

384 digits

0 $\oplus$ f

Absorbing

Squeezing

h

m$_1$

m$_2$

m$_3$
The Function $f$

Each wire is a base 4 number
Good Differentials for $f$?

If $(\Delta x, 0), (0, \Delta z)$ is a differential for $f$, then $(\Delta x, 0)$ is a differential for $H$. 
Constructing Good Differentials

S-box differential has only 1 non-zero digit in both inputs and outputs
• Called “weight 1” differential

String together to get differential for overall SPN

Don’t care so much about exact differential, any sequence of weight 1 differentials will do
Attack Sketch:

Choose two random messages that differ in a single digit, hope that they are collision

Probability of collision $\geq \frac{3}{4} \times 2^{-20}$

- Prob $\geq 2^{-20}$ input differential gives weight 1 output differential
- Prob $\frac{3}{4}$ differing digit will be among first 128 digits
Previously on COS 433...
Identification Protocols
Identification
Identification
Types of Attacks

Direct Attack:
Types of Attacks

Eavesdropping/passive:

sk

vk
Types of Attacks

Man-in-the-Middle/Active:

```
sk
```

```
vk
```
Basic Password Protocol

Never ever (ever ever...) use

$sk = pwd$

$vk = pwd$

$sk == vk$?
Salting

Let $H$ be a hash function

$s_i$ random

<table>
<thead>
<tr>
<th>User</th>
<th>Salt</th>
<th>Pwd</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>$s_A$</td>
<td>$H(s_A, pwd_A)$</td>
</tr>
<tr>
<td>Bob</td>
<td>$s_B$</td>
<td>$H(s_B, pwd_B)$</td>
</tr>
<tr>
<td>Charlie</td>
<td>$s_C$</td>
<td>$H(s_C, pwd_C)$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Security Against Eavesdropping

$\text{sk} = \text{pwd}$

$\text{sk} = H(s_A, \text{pwd})$

$H(s_A, \text{sk}) \equiv \text{vk}$?
One-time Passwords

Let $F$ be a PRF

$s_{k_0} = F(k,0)$

$v_{k} = (k,0)$

$s_{k_0} \equiv F(k,0)$?
One-time Passwords

Let $F$ be a PRF

$sk = (k, 1)$

$sk_1 = F(k, 1)$

$vk = (k, 1)$

$sk_1 = F(k, 1)$?
One-time Passwords

Let $F$ be a PRF

$$sk_0 = F(k,0)$$
Let $F$ be a PRF

$sk_0 = F(k, 0)$

$sk = (k, 1)$

$vk = (k, 1)$
One-time Passwords

Advancing state:
• Time based (e.g. every minute, day, etc)
• User Action (button press)

Must allow for small variation in counter value
• Clocks may be off, user may accidentally press button
Stateless Schemes?

So far, all schemes secure against eavesdropping are stateful.

Easy theorem: any one-message stateless ID protocol is insecure if the adversary can eavesdrop:

• Simply replay message

If want stateless scheme, instead want at least two messages.
Today

Challenge-Response authentication

Zero Knowledge
Challenge-Response
C-R Using Encryption

\[ sk = k \]

\[ ch = Enc(k, r) \]

\[ res = Dec(k, ch) \]

\[ res \equiv r? \]

Random \( r \)

\[ vk = k \]

\[ ch = Enc(k, r') \]

\[ res? \]

\[ res == r? \]
**Theorem:** If $(\text{Enc, Dec})$ is a CPA-secure secure SKE/PKE scheme, then the C-R protocol is a secret key/public key identification protocol secure against eavesdropping attacks.
C-R Using MACs/Signatures

$sk = k$

$ch = r$

$res = MAC(k, ch)$

Random $r$
or $r = Time$

$vk = k$

$ch = r'$

$Res$?

$Ver(k, ch, res)?$
Theorem: If $(\text{MAC, Ver})$ is a CMA-secure secure MAC/Signature scheme, then the C-R protocol is a secret key/public key identification protocol secure against eavesdropping attacks.
Active Attacks

\[ \text{sk} \quad \rightarrow \quad \text{vk} \]
Active Attacks

For enc-based C-R, CPA-secure is insufficient
  • Instead need CCA-security (lunch-time sufficient)

For MAC/Sig-based C-R, CMA-security is sufficient
Non-Repudiation

Consider signature-based C-R

\[ vk = pk \]
\[ ch = r \]
\[ res = \text{Sig}(vk, ch) \]

\[ r = \text{Time} \]

Bob can prove to police that Alice passed identification
Zero Knowledge

What if Bob could have come up with a valid transcript, without ever interacting with Alice?
• Then Bob cannot prove to police that Alice authenticated

Seems impossible:
• If (public) vk is sufficient to come up with valid transcript, why can’t an adversary do the same?
Zero Knowledge

Adversary CAN come up with valid transcripts, but Bob doesn’t accept transcripts
• Instead, accepts interactions

Ex: public key Enc-based C-R
• Valid transcript: \((c, r)\) where \(c\) encrypts \(r\)
• Anyone can come up with a valid transcript
• However, only Alice can generate the transcript for a given \(c\)

Takeaway: order of messages matters
Zero Knowledge Proofs
Mathematical Proof
Mathematical Proof

Statement $\mathbf{x}$

Witness $\mathbf{w}$

$\mathbf{w}$

$\mathbf{Ver(x,w)}$
Interactive Proof

Statement \( \times \)

Witness \( \mathbf{w} \)
Properties of Interactive Proofs

Let \((P, V)\) be a pair of probabilistic interactive algorithms for the proof system

**Completeness:** If \(w\) is a valid witness for \(x\), then \(V\) should always accept

**Soundness:** If \(x\) is false, then no cheating prover can cause \(V\) to accept

- Perfect: accept with probability 0
- Statistical: accept with negligible probability
- Computational: cheating prover is comp. bounded
Zero Knowledge

Intuition: verifier doesn’t learn anything by engaging in the protocol (other than the truthfulness of $x$)

How to characterize what adversary “knows”?
• Only outputs a bit
• May “know” witness, but hidden inside the programs state
Zero Knowledge

First Attempt:

\[ \exists \text{"simulator"} \quad \text{s.t. for every true statement } x, \text{ valid witness } w, \]

\( P(x,w) \approx_{c} V(x) \)
Zero Knowledge

First Attempt:

Assumes Bob obeys protocol
• “Honest Verifier”

But what if Bob deviates from specified prover algorithm to try and learn more about the witness?
Zero Knowledge

For every malicious verifier $V^*$, $\exists$ “simulator” s.t. for every true statement $x$, valid witness $w$,

$\approx_c$
QR Protocol

Statements: $x$ is a Q.R. mod $N$
Witness: $w$ s.t. $w^2 \mod N = x$

Protocol:

$u \leftarrow Z_N^*$

$y \leftarrow u^2 \mod N$

$w$

$y \leftarrow u^2 \mod N$

$b \leftarrow \{0,1\}$

$z = w^b u \mod N$

$z^2 = x^b y \mod N$?
QR Protocol

Zero Knowledge:

What does Bob see?
- A random QR $y$,
- A random bit $b$,
- A random root of $x^by$

Idea: simulator chooses $b$, then $y$,
- Can choose $y$ s.t. it always knows a square root of $x^by$
QR Protocol

Honest Verifier Zero Knowledge:

\[ (x): \]
- Choose a random bit \( b \)
- Choose a random string \( z \)
- Let \( y = x^{-b}z^2 \)
- Output \( (y,b,z) \)

- If \( x \) is a QR, then \( y \) is a random QR, no matter what \( b \) is
- \( z \) is a square root of \( x^by \)

\( (y,b,z) \) is distributed identically to \( (P,V)(x) \)
QR Protocol

(Malicious Verifier) Zero Knowledge:

\[
x \quad \xrightarrow{\text{b}'} \quad \{0,1\} \quad \xleftarrow{\text{y}} \quad \frac{x-z^2}{b} \quad \text{if } b = b', \text{ else repeat}
\]

\[(y, b, z)\]
QR Protocol

(Malicious Verifier) Zero Knowledge:

Proof:
• If $x$ is a QR, then $y$ is a random QR, independent of $b'$
• Conditioned on $b'=b$, then $(y,b,z)$ is identical to random transcript seen by $V^*$
• $b'=b$ with probability $1/2$
Repetition and Zero Knowledge

(sequential) repetition also preserves ZK

Unfortunately, parallel repetition might not:

• makes guesses $b_1', b_2', ...$
• Generates valid transcript only if all guesses were correct
• Probability of correct guess: $2^{-t}$

Maybe other simulators will work?
• Known to be impossible in general, but nothing known for QR
Zero Knowledge Proofs

Known:
• Proofs for any NP statement assuming statistically-binding commitments
• Non-interactive ZK proofs for any NP statement using trapdoor permutations
Proofs of Knowledge

Sometimes, not enough to prove that statement is true, also want to prove “knowledge” of witness

Ex:
• Identification protocols: prove knowledge of key
• Discrete log: always exists, but want to prove knowledge of exponent.
Proofs of Knowledge

We won’t formally define, but here’s the intuition:

Given any (potentially malicious) PPT prover $P^*$ that causes $V$ to accept, it is possible to “extract” from $P^*$ a witness $w$
Schnorr PoK for DLog

Statement: \((g, h)\)
Witness: \(w\) s.t. \(h = g^w\)

Protocol:

\[
\begin{align*}
& r \leftarrow \mathbb{Z}_p \\
& a \leftarrow g^r \\
& a \times h^b = g^c? \\
& b \leftarrow \mathbb{Z}_p \\
& c = r + wb
\end{align*}
\]
Schnorr PoK for DLog

Completeness:
• $g^c = g^{r+wb} = a \times h^b$

Honest Verifier ZK:
• Transcript = $(a, b, c)$ where $a = g^c / h^b$ and $(b, c)$ random in $\mathbb{Z}_p$
• Can easily simulate. How?
Schnorr PoK for DLog

Proof of Knowledge?

Idea: once Alice commits to $a=g^r$, show must be able to compute $c = r + bw$ for any $b$ of Bob’s choosing
• Intuition: only way to do this is to know $w$
• Run Alice on two challenges, obtain:

$$c_0 = r_0 + b_0 \ w, \ c_1 = r_1 + b_1 \ w$$

(Can solve linear equations to find $w$)
Deniability

Zero Knowledge proofs provide deniability:
• Alice proves statement \( x \) is true to Bob
• Bob goes to Charlie, and tries to prove \( x \) by providing transcript
• Charlie not convinced, as Bob could have generated transcript himself
• Alice can later deny that she knows proof of \( x \)
$\sum$ Protocols

(fancy name for 3-round “public coin” protocols)
Fiat-Shamir Transform

Idea: set $b = H(a)$
- Since $H$ is a random oracle, $a$ is a random output

Notice: now prover can compute $b$ for themselves!
- No need to actually perform interaction

$w \rightarrow a, b = H(a), c$
Theorem: If $(P,V)$ was a secure ZKPoK for honest verifiers, and if $H$ is a random oracle, then compiled protocol is a ZKPoK

Proof idea: second message is exactly what you’d expect in original protocol

Complication: adversary can query $H$ to learn second message, and throw it out if she doesn’t like it
Signatures from $\Sigma$ Protocols

Idea: what if set $b = H(m, a)$
- Challenge $b$ is message specific
- Intuition: proves that someone who knows $sk$ engaged in protocol depending on $m$
- Can use resulting transcript as signature on $m$

Schnorr PoK $\rightarrow$ Schnorr Signatures
Applications of ZK (PoK)

Identification protocols: prove that you know the secret without revealing the secret

Signatures: prove that you know the secret in a “message dependent” way

Protocol Design:
• E.g. CCA secure PKE
  • To avoid mauling attacks, provide ZK proof that ciphertext is well formed
  • Problem: ZK proof might be malleable
  • With a bit more work, can be made CCA secure
• Example: multiparty computation
  • Prove that everyone behaved correctly
Announcements

HW6 Due SUNDAY
HW7 Due April 30th

Project 3 will be combined with HW 8, due on Dean’s date