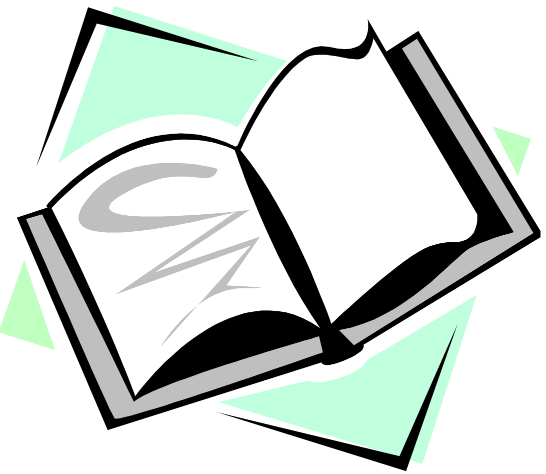
Cryptography
CS 555

Topic 23: Zero-Knowledge Proof and Cryptographic Commitment

Outline and Readings

- Outline
  - Zero-knowledge proof
  - Fiat-Shamir protocol
  - Schnorr protocol
  - Commitment schemes
  - Pedersen commitment schemes
  - Oblivious commitment based envelope



- Readings:
  - Barak’s notes on ZK

Interactive Proof Systems

- Traditionally, a proof for a statement is a static string such that one can verify for its correctness
  - Follows axioms and deduction rules.

- Generalizing proof systems to be interactive
  - A proof system involves an algorithm for a prover and a verifier.
  - A proof system can be probabilistic in ensuring correctness of the statement being proved

Zero Knowledge Proofs

- A protocol involving a prover and a verifier that enables the prover to prove to a verifier without revealing any other information
  - E.g., proving that a number n is of the form of the product of two prime number
  - Proving that one knows p,q such that n=pq
  - Proving that one knows x such $g^x \bmod p = y$

Two Kinds of Zero-Knowledge Proofs

- ZK proof of a statement
  - convincing the verifier that a statement is true without yielding any other information
  - example of a statement, a propositional formula is satisfiable
- ZK proof of knowledge
  - convincing the verifier that one knows a secret, e.g., one knows the discrete logarithm $\log_g(y)$

Fiat-Shamir Protocol for Proving Quadratic Residues

- Statement: x is QR modulo n
- Prover knows w such that $w^2=x \pmod n$
- Repeat the following one-round protocol t times
- One-round Protocol:
  – P to V: $y = r^2 \bmod n$, where r randomly chosen
  – V to P: $b \leftarrow \{0,1\}$, randomly chosen
  – P to V: $z=rw^b$, i.e., z=r if b=0, z=rw if b=1
  – V verifies: $z^2=yx^b$, i.e., $z^2=y$ if b=0, $z^2=yx$ if b=0

Observations on the Protocol

- Multiple rounds
- Each round consists of 3 steps
  - Commit; challenge; respond
- If challenge can be predicted, then cheating is possible.
  - Cannot convince a third party (even if the party is online)
  - Essense why it is ZK
- If respond to more than one challenge with one commit, then the secret is revealed.
  - Essence that this proves knowledge of the secret

Properties of Interactive Zero-Knowledge Proofs of Knowledge

- Completeness
  - Given honest prover and honest verifier, the protocol succeeds with overwhelming probability
- Soundness
  - no one who doesn't know the secret can convince the verifier with nonnegligible probability
- Zero knowledge
  - the proof does not leak any additional information

Analysis of the Fair-Shamir protocol

- Completeness, when proven is given $w^2=x$ and both party follows protocol, the verification succeeds

- Soundness: if x is not QR, verifier will not be fooled.
  - Needs to show that no matter what the prover does, the verifier's verification fails with some prob. (1/2 in this protocol)
  - Assumes that x is not QR, V receives y
    - Case 1: y is QR, then when b=1, checking $z^2=yx$ will fail.
    - Case 2: y is QNR, then when b=0, checking $z^2=y$ will fail.
    - Proof will be rejected with probability ½.

Formalizing ZK property

- A protocol is ZK if a simulator exists
  - Taking what the verifier knows before the proof, can generate a communication transcript that is indistinguishable from one generated during ZK proofs
    - Intuition: One observes the communication transcript. If what one sees can be generated oneself, one has not learned anything new knowledge in the process.

- Three kinds of indistinguishability
  - Perfect (information theoretic)
  - Statistical
  - Computational

Honest Verifier ZK vs. Standard ZK

- Honest Verifier ZK means that a simulator exists for the Verifier algorithm V given in the protocol.

- Standard ZK requires that a simulator exists for any algorithm V* that can play the role of the verifier in the protocol.

Fiat-Shamir is honest-verifier ZK

- The transcript of one round consists of
  - (n, x, y, b, z) satisfying $z^2=yx^b$
  - The bit b is generated by honest Verifier V is uniform independent of other values
- Construct a simulator for one-round as follows
  - Given (x,n)
  - Pick at uniform random $b \leftarrow \{0,1\}$,
  - If b=0, pick random z and sets $y=z^2 \bmod n$
  - If b=1, pick random z, and sets $y=z^2x^{-1} \bmod n$
  - Output (n,x,y,b,z)
- The transcript generated by the simulator is from the same prob. distribution as the protocol run

Fiat-Shamir is ZK

- Given any possible verifier V*, A simulator works as follows:
  1. Given (x,n) where x is QR; let T=(x,n)
  2. Repeat steps 3 to 7 for
  3. Randomly chooses $b \leftarrow \{0,1\}$,
  4. When b=0, choose random z, set $y=z^2 \bmod n$
  5. When b=1, choose random z, set $y=z^2x^{-1} \bmod n$
  6. Invoke let b'=V*(T,y), if $b' \neq b$, go to step 3
  7. Output (n,x,y,b,z); T.append((n,x,y,b,z));

- Observe that both z^2 and z^2x^{-1} are a random QR; they have the same prob. distribution, thus the success prob. of one round is at least ½

Zero Knowledge Proof of Knowledge

- A ZKP protocol is a proof of knowledge if it satisfies a stronger soundness property:
  - The prover must know the witness of the statement
- Soundness property: If a prover A can convince a verifier, then a knowledge exactor exists
  - a polynomial algorithm that given A can output the secret
- The Fiat-Shamir protocol is also a proof of knowledge:

Knowledge Extractor for the QR Protocol

- If A can convince V that x is QR with probability significanly over ½, then after A outputs y, then A can pass when challenged with both 0 and 1.

- Knowledge extractor
  – Given an algorithm A that can convince a verifier,
  – After A has sent y, first challenge it with 0, and receives z_1 such that $z_1^2=y$
  – Then reset A to the state after sending y, challenge it with 1 and receives z_2 such that $z_2^2=xy$, then compute $s=z_1^{-1}z_2$, we have $s^2=x$

Running in Parallel

- All rounds in Fiat-Shamir can be run in parallel
  1. Prover: picks random $r_1,r_2,\ldots,r_t$, sends $y_1=r_1^2$, $y_2=r_2^2$, …, $y_t=r_t^2$
  2. Verifier checks the y's are not 0 and sends t random bits $b_1,\ldots b_t$
  3. Prover sends $z_1,z_2,\ldots,z_k$,
  4. Verifier accept if $z_j^2 \equiv y_j x^{b\_j}$ mod n
- This protocol still a proof of knowledge.
- This protocol still honest verifier ZK.
- This protocol is no longer ZK!
  - Consider the V* such that V* chooses $b_1,\ldots b_t$ to be the first t bits of $H(y_1,y_2,\ldots,y_t)$, where H is a cryptographic hash function.
  - One can no longer generate an indistinguishable transcript.

Schnorr Id protocol (ZK Proof of Discrete Log)

- System parameter: p, g generator of Z_p^*
- Public identity: v
- Private authenticator: s $v = g^s \bmod p$
- Protocol (proving knowledge of discrete log of v with base g)
  1. A: picks random r in [1..p-1], sends $x = g^r \bmod p$,
  2. B: sends random challenge e in $[1..2^t]$
  3. A: sends y=r-se mod (p-1)
  4. B: accepts if $x = (g^y v^e \bmod p)$

Security of Schnorr Id protocol

- Completeness: straightforward.
- Soundness (proof of knowledge):
  - if A can successfully answer two challenges e_1 and e_2, i.e., A can output y_1 and y_2 such that $x=g^{y1}v^{e1}=g^{y2}v^{e2}$ (mod p) then $g^{(y1-y2)}=v^{(e2-e1)}$ and
    $g^{(y1-y2)\ (e2-e1)\text{^}\{-1\} \bmod (p-1)}=v$ thus the secret
    $s=(y_1-y_2)(e_2-e_1)^{-1} \bmod (p-1)$
- ZK property
  - Is honest verifier ZK, how does the simulate works?
  - Is not ZK if the range of challenge e is chosen from a range that is too large (2^t>log n). Why?

Commitment schemes

- An electronic way to temporarily hide a value that cannot be changed
  - Stage 1 (Commit)
    - Sender locks a message in a box and sends the locked box to another party called the Receiver
  - State 2 (Reveal)
    - the Sender proves to the Receiver that the message in the box is a certain message

Security properties of commitment schemes

- Hiding
  - at the end of Stage 1, no adversarial receiver learns information about the committed value
- Binding
  - at the end of State 1, no adversarial sender can successfully convince reveal two different values in Stage 2

A broken commitment scheme

- Using encryption
  - Stage 1 (Commit)
    - the Sender generates a key *k* and sends $E_k[M]$ to the Receiver
  - State 2 (Reveal)
    - the Sender sends *k* to the Receiver, the Receiver can decrypt the message

- What is wrong using the above as a commitment scheme?

Formalizing Security Properties of Commitment schemes

- Two kinds of adversaries
  - those with infinite computation power and those with limited computation power
- Unconditional hiding
  - the commitment phase does not leak any information about the committed message, in the information theoretical sense (similar to perfect secrecy)
- Computational hiding
  - an adversary with limited computation power cannot learn anything about the committed message (similar to semantic security)

Formalizing Security Properties of Commitment schemes

- Unconditional binding
  - after the commitment phase, an infinite powerful adversary sender cannot reveal two different values
- Computational binding
  - after the commitment phase, an adversary with limited computation power cannot reveal two different values
- No commitment scheme can be both unconditional hiding and unconditional binding

Another (also broken) commitment scheme

- Using a one-way function *H*
  - Stage 1 (Commit)
    - the Sender sends c=*H*(*M*) to the Receiver
  - State 2 (Reveal)
    - the Sender sends *M* to the Receiver, the Receiver verifies that c=*H*(*M*)
- What is wrong using this as a commitment scheme?
- A workable scheme (though cannot prove security)
  - Commit: choose r1, r2, sends (r1, H(r1||M||r2))
  - Reveal (open): sends M, r2.
  - Disadvantage: Cannot do much interesting things with the commitment scheme.

Pedersen Commitment Scheme

- Setup
  - The receiver chooses two large primes p and q, such that q|(p-1). Typically, p is 1024 bit, q is 160 bit. The receiver chooses an element g that has order q, she also chooses secret a randomly from Z_q={0,..,q-1}. Let h = g^a mod p. Values <p,q,g,h> are the public parameters and a is the private parameter.
    - We have $g^q = 1 \pmod p$, and we have $\langle g \rangle = \{g, g^2, g^3, \ldots, g^q = 1\}$, the subgroup of Z_p^* generated by g

- Commit
  - The domain of the committed value is Z_q. To commit an integer $x \in Z_q$, the sender chooses $r \in Z_q$, and computes $c = g^x h^r \bmod p$
- Open
  - To open a commitment, the sender reveal x and r, the receiver verifies whether $c = g^x h^r \bmod p$.

Pedersen Commitment Scheme (cont.)

- Unconditionally hiding
  - Given a commitment c, every value x is equally likely to be the value committed in c.
  - For example, given x,r, and any x', there exists r' such that $g^x h^r = g^{x'} h^{r'}$, in fact r = (x-x')a^{-1} + r mod q.
- Computationally binding
  - Suppose the sender open another value x' ≠ x. That is, the sender find x' and r' such that c = $g^{x'} h^{r'}$ mod p. Now the sender knows x,r,x',r' s.t., $g^x h^r = g^{x'} h^{r'}$ (mod p), the sender can compute $\log_g(h) = (x'-x)\cdot(r-r')^{-1}$ mod q. Assume DL is hard, the sender cannot open the commitment with another value.

Pedersen Commitment – ZK Prove know how to open (without actually opening)

- Public commitment $c = g^x h^r \pmod p$
- Private knowledge x,r
- Protocol:

1. P: picks random y, s in [1..q], sends
   $d = g^y h^s \bmod p$
2. V: sends random challenge e in [1..q]
3. P: sends $u=y+ex$, $v=s+er \pmod q$
4. V: accepts if $g^u h^v = dc^e \pmod p$

- Security property – similar to Schnorr protocol

Proving that the committed value is either 0 or 1

- Let <p,q,g,h> be the public parameters of the Pedersen commitment scheme. Let $x \in \{0,1\}$, $c = g^x h^r \bmod p$
- The prover proves to the verifier that x is either 0 or 1 without revealing x
  - Note that $c = h^r$ or $c = gh^r$
  - The prover proves that she knows either $\log_h(c)$ or $\log_h(c/g)$
  - Recall if the prover can predict the challenge e, she can cheat
  - The prover uses Schnorr protocol to prove the one she knows, and to cheat the other one

Bit Proof Protocol (cont.)

- Recall Schnorr Protcol of proving knowledge of discrete log of c with basis h:
  - $P \rightarrow V$: x; $V \rightarrow P$: e; $P \rightarrow V$: y;   Verifies: $x=h^y c^e$
  - To cheat, chooses e and f, compute x
  - To prove one, and cheat in another, conduct two proofs, one for challenge e_1 and the other for e_2 with $e_1+e_2=e$
    - Prover can control exactly one of e_1 and e_2, Verifier doesn't know which
- Case 1: $c=h^r$
  - $P \rightarrow V$: choose w,y_1,e_1 from Z_q, sends $x_0=h^w$, $x_1=h^{y1}(c/g)^{e1}$
  - $V \rightarrow P$: e
  - $P \rightarrow V$: $e_0= e-e_1$ mod q, $y_0= w+r \cdot e_0$ mod q sends y_0,y_1,e_0,e_1
  - V: verify $e=e_0+e_1$, $x_0=h^{y0}c^{e0}$, $x_1=h^{y1}(c/g)^{e1}$

Bit Proof Protocol (cont.)

- Case 2: $c=gh^r$
  - $P \rightarrow V$: choose w, y_0, e_0 from Z_q, computes $x_1=h^w$, $x_0=h^{z0}c^{e0}$, and sends a_0, a_1
  - $V \rightarrow P$: e
  - $P \rightarrow V$: computes $e_1 = e - e_0 \bmod q$, $y_1 = w + r \cdot e_1 \bmod q$, sends y_0, y_1, e_0, e_1
  - V: verify $e=e_0+e_1$, $x_0=h^{y0}c^{e0}$, $x_1 = h^{y1} (c/g)^{e1}$

Security of Bit Proof Protocol

- Zero-knowledge
  - The verifier cannot distinguish whether the prover committed a 0 or 1, as what the prover sends in the two cases are drawn from the same distribution.
- Soundness
  - Bit proof protocol is a proof of knowledge

An Application

- Oblivious Commitment Based Envelope and Oblivious Attribute Certificates
- Jiangtao Li, Ninghui Li: OACerts: Oblivious Attribute Certificates. ACNS 2005: 301-317

Oblivious Attribute Certificates (OACerts)

California Driver License

Expired: 04-11-06

Name: Bear Boy
DoB: 12-01-96
Address:
206 Sweet Rd

Sex: M
HT: 20”
WT: 75

Signed by PMV

X.509 Certificate

California Driver License

Expired: 04-11-06

Name: [redacted]
DoB: [redacted]
Address:
[redacted]

Sex: [redacted]
HT: [redacted]
WT: [redacted]

Signed by PMV

OACerts

Features of OACerts

- Selective show of attributes
- Zero-Knowledge proof that attributes satisfy some properties
- Compatible with existing certificate systems, e.g., X.509
- Revocation can be handled using traditional techniques, e.g., CRL

Oblivious Usage of Attributes

Receiver

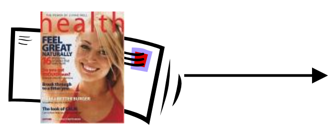
Sender

$c = \text{commit}(a)$

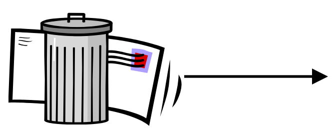
Message: 

Policy:
Pred

Case 1:
$\text{Pred}(a) = \text{true}$



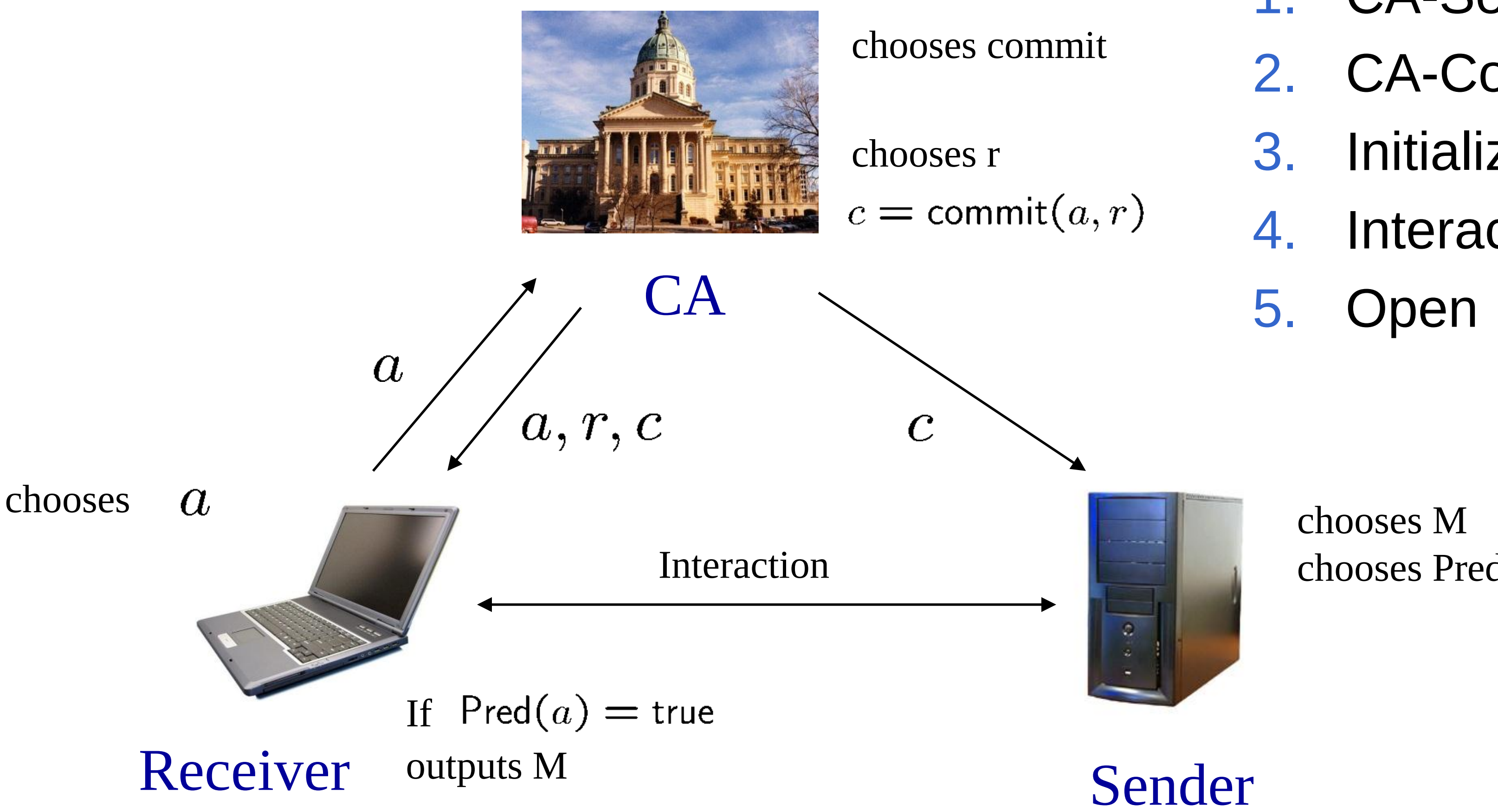
Case 2:
$\text{Pred}(a) = \text{false}$



Oblivious Commitment-Based Envelope (OCBE)

Formal Definition of OCBE

1. CA-Setup
2. CA-Commit
3. Initialization
4. Interaction
5. Open



Oblivious

- OCBE is oblivious if no adversary has a non-negligible advantage in the following game.

run steup

picks a_1, a_2

a_1, a_2 ←

chooses $b \in \{1, 2\}$
sets $a = a_b$
$c = \text{commit}(a, r)$

c →

chooses Pred, M

Pred ←

Interaction ↔

b' ←

Challenger
Receiver

Adversary
Sender

Adversary wins if b=b'

Secure Against the Receiver

- OCBE is secure against receiver if no adversary has a non-negligible advantage in the following game.

Challenger
Sender

run steup

$c = \mathsf{commit}(a, r)$

chooses $b \in \{1, 2\}$
sets $M = M_b$

picks a

$\xleftarrow{\quad a \quad}$

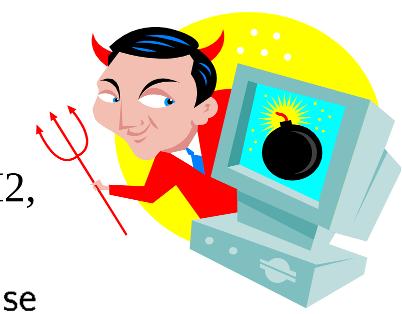
$\xrightarrow{\quad c, r \quad}$

chooses M1,M2,
Pred, s.t.
$\mathsf{Pred}(a) = \mathsf{false}$

$\xleftarrow{\text{Pred, M1, M2}}$

$\xleftrightarrow{\text{Interaction}}$

$\xleftarrow{\quad b' \quad}$



Adversary
Receiver

Adversary wins if b=b’

OCBE Protocols

- We developed the following OCBE protocols for the Pedersen commitment schemes
  - Committed value $=, >, <, \neq, \leq$, or $\geq$ a known value
  - Committed value lies in a certain range
  - Committed value satisfy conjunction of two conditions
  - Committed value satisfy disjunction of two conditions

Coming Attractions …

- Topics
  - Secure function evaluation, Oblivious transfer, secret sharing
  - Identity based encryption & quantum cryptography