

The background is a dark blue field filled with various sizes of semi-transparent gears. On the left side, there is a vertical strip with a colorful, abstract, and pixelated pattern in shades of orange, yellow, and brown.

# Zero Knowledge Cryptography

Vivek Haldar

ICS 280 - March 8, 2002

# Breaking news: Exploit in OpenSSH

- ✦ One-off error
- ✦ Allows local user to gain root privileges
- ✦ Remote exploit not ruled out
- ✦ One line patch solves the problem

```
channel_lookup(int id)
{
    Channel *c;
-   if (id < 0 || id > channels_alloc) {
+   if (id < 0 || id >= channels_alloc) {
        log("channel_lookup: %d: bad id", id);
        return NULL;
    }
}
```



# Outline

## ★ Some theory

- What is a proof?
- What is knowledge?
- Interactive proofs
- Zero knowledge proofs
- ZK proofs for NP languages
- ZK proofs of knowledge

## ★ Applications

# A motivating example

- ✦ Alice wants to show Bob cleartext of one of her files
- ✦ Alice doesn't want to give Bob her key
- ✦ If Alice gives Bob plaintext, Bob doesn't know if it's the "real" file
- ✦ Communication among *mutually distrusting* parties



# El Grande Kahuna

To show possession of a secret to another party without *giving away the secret*

# Outline

## ★ Some theory

- ★ What is a proof?
- ★ What is knowledge?
- ★ Interactive proofs
- ★ Zero knowledge proofs
- ★ ZK proofs for NP languages
- ★ ZK proofs of knowledge

## ★ Applications



# What is a proof?

- ✦ In mathematics: a ***fixed*** sequence of statements flowing logically
- ✦ In real life proofs have a much wider meaning
- ✦ Not fixed, but rather a ***process*** by which validity is established
- ✦ E.g. cross-examination of a witness



# Properties of a proof system

- ✦ **Soundness** - I'll never believe a false statement
- ✦ **Completeness** - I'll believe all true statements
- ✦ Goedel's incompleteness theorem - any "sufficiently complex" proof system cannot be both complete and sound
- ✦ Turing's halting problem



# The Prover and the Verifier

- ✦ Proofs are defined by their *verification procedure*
- ✦ The onus is on the prover to convince the verifier
- ✦ Verification is typically simple - proving is typically hard

# The complexity class NP

- ★ NP can be characterized as the set of languages for which an efficient procedure exists to **check** (not **compute**) if a string belongs to that language
- ★ Given a string  $x$  from a language  $L$  and a **certificate**  $y$  it is easy to check if  $x$  belongs to  $L$



☀ Sidebar:

# Theory of Computation 101

- ☀ NP = set of languages which can be recognized in polynomial time by a ***non-deterministic*** Turing machine
- ☀ Remember how a non-deterministic Turing machine (NTM) is simulated by a deterministic Turing machine (TM) ?



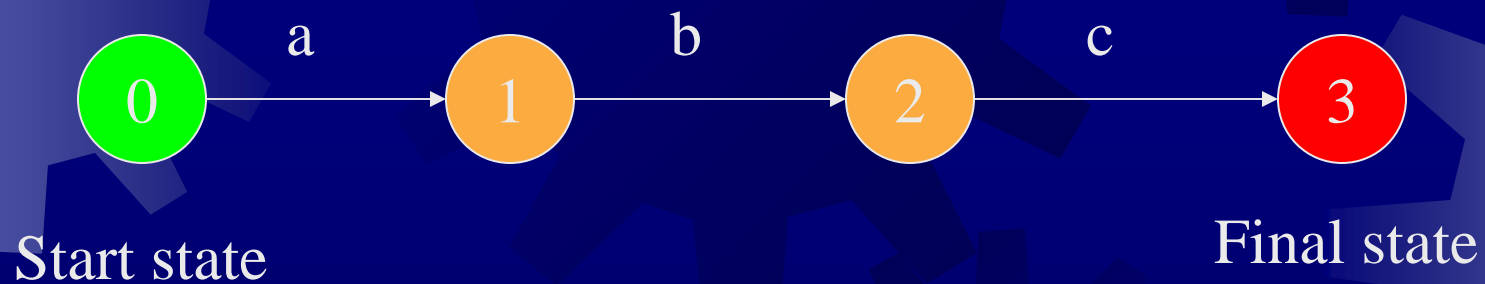
⚙️ Sidebar:

# Theory of Computation 101

Same way that a non-deterministic finite automaton (NFA) is simulated by a deterministic finite automaton (DFA)!

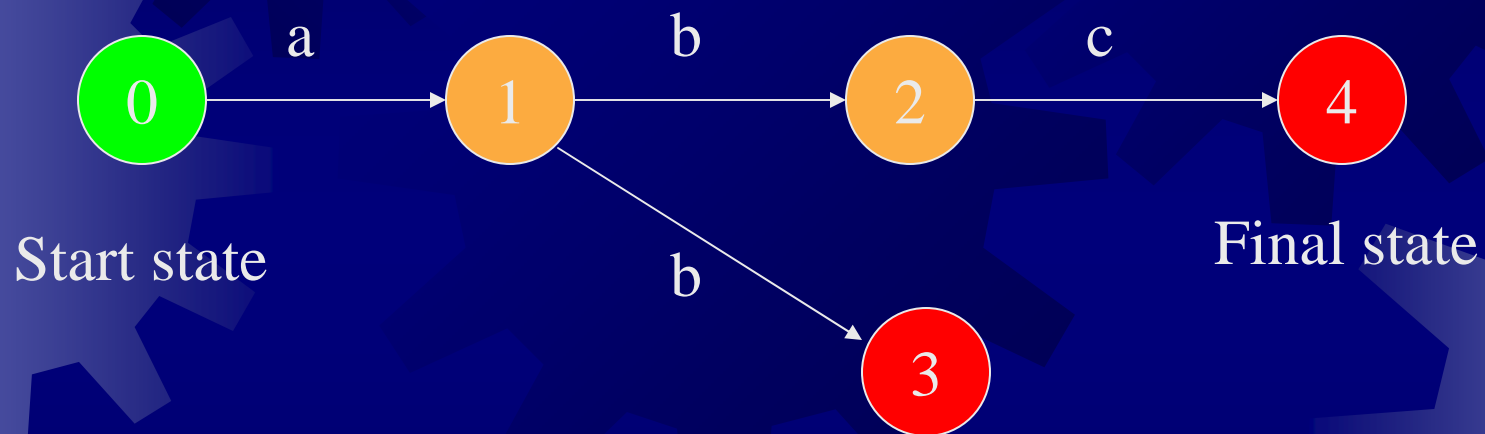
...remember?

# Sidebar: Theory of Computation 101



*Deterministic automaton*: for every state, given an input we know *unambiguously* which state to go to next

# Sidebar: Theory of Computation 101



*Non-deterministic automaton*: for every state, given an input there are *many* states to go to next - the DFA “knows” which one to take



☛ Sidebar:

# Theory of Computation 101

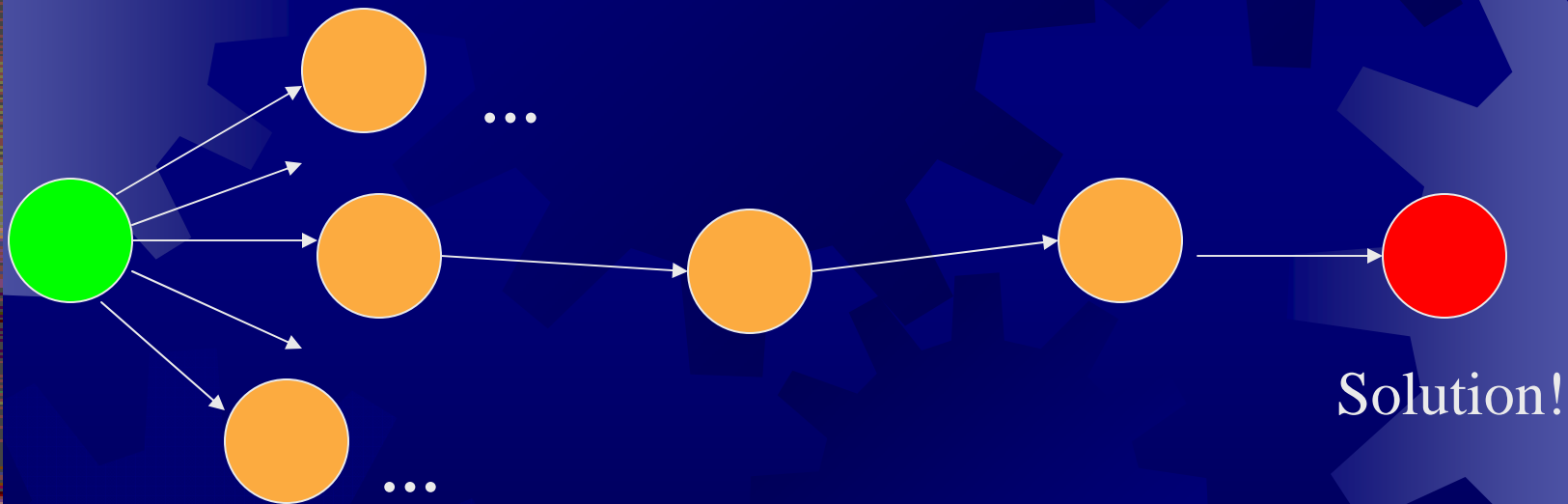
How to make a DFA do the work of an NFA

- ☀ States = Powerset of NFA states =  $2^{|\text{NFA}|}$
- ☀ Final states = any “state” which includes any of the NFA’s final states

Essentially - traverse a search tree sequentially

# Sidebar: Theory of Computation 101

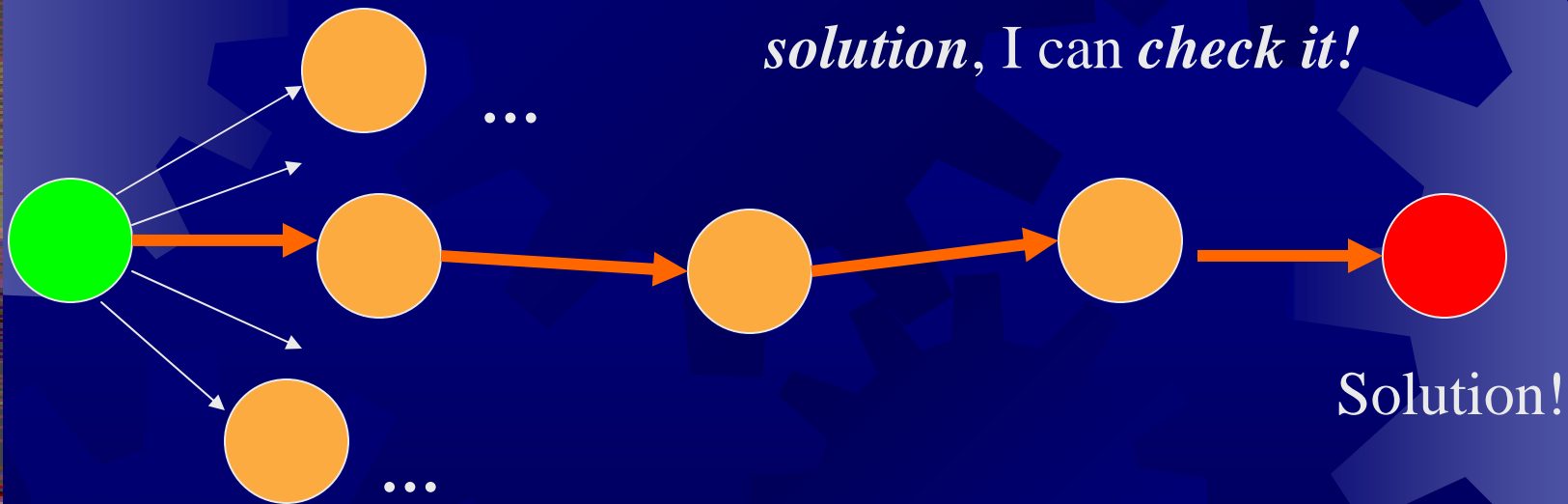
NP “search tree”



Mostly “dead ends”

# Sidebar: Theory of Computation 101

NP “search tree”



But if you *just tell me the path in the search tree that led to a solution, I can check it!*

Mostly “dead ends”

# Outline

- ★ Some theory

- What is a proof?
- What is knowledge?
- Interactive proofs
- Zero knowledge proofs
- ZK proofs for NP languages

- ★ Applications



# What is knowledge?

- ✦ Tough question!
- ✦ Computer “scientists” are a bunch of phonies!
- ✦ We chicken out...



# What is a *gain* of knowledge?

- ✦ Defined with respect to *computational ability*
- ✦ Bob *gains knowledge* after interacting with Alice if:

After the interaction Bob can easily compute something that was tough for him earlier

# Outline

## ★ Some theory

- What is a proof?
- What is knowledge?
- **Interactive proofs**
- Zero knowledge proofs
- ZK proofs for NP languages
- ZK proofs of knowledge

## ★ Applications

# Interactive proofs

- ✱ Prover and verifier interact with each other
- ✱ For the theory heads: two Turing machines, sharing a common tape.
- ✱ **Completeness:** if  $x$  is in  $L$ :  
$$P(\text{acceptance}) \geq 2/3$$
- ✱ **Soundness:** if  $x$  is not in  $L$ :  
$$P(\text{acceptance}) \leq 1/3$$

# Interactive Proofs

- ✦ The verifier must act in polynomial time
- ✦ There are *no resource bounds* imposed on the computing power of the prover
- ✦  $IP = \text{class of languages that have interactive proofs}$
- ✦ After TOC 101, we know that:  **$NP$  is in  $IP$**
- ✦ But the interaction is uni-directional

# Outline

## ★ Some theory

- What is a proof?
- What is knowledge?
- Interactive proofs
- **Zero knowledge proofs**
- ZK proofs for NP languages

## ★ Applications

# Zero knowledge proofs

Let  $(P, V)$  be an interactive proof system for some language  $L$ . We say that  $P$  is *perfect zero knowledge* if for every probabilistic polynomial time interactive machine  $V'$  there exists a probabilistic polynomial time algorithm  $M'$  (the *simulator*) so that for every  $x$  in  $L$ :  $(P, V')(x)$  and  $M'(x)$  are identically distributed

# In plain English please!

- ✦ For every verifier interacting with a prover, there is a *simulator*
- ✦ This simulator does not have access to the interactive prover
- ✦ Yet, it can simulate the interaction between P and V
- ✦ So, V did not gain any knowledge from P - since the same output could have been generated without any access to P



# Perfect vs Computational ZK

- ✦ Earlier definition was for ***perfect zero knowledge*** - simulator and real system must have *identical* distributions
- ✦ If the distributions are not *perfectly identical*, but only *computationally indistinguishable* we have ***computational zero knowledge***

# Example: Graph Isomorphism

- ★ Alice has two isomorphic graphs :

$$G_1 = (V_1, E_1), \text{ and } G_2 = (V_2, E_2)$$

related by the isomorphism  $f$

- ★ Alice wants to prove to Bob that  $G_1$  and  $G_2$  are isomorphic without giving away the isomorphism

# Example: Graph Isomorphism

- ★  $P \rightarrow V$ : select and send random isomorphic copy of  $G_2$ , where  $g$  was the isomorphism
- ★  $V \rightarrow P$ : (Just got  $G'$ ) Pick  $i$  from  $\{1,2\}$  randomly, and ask  $P$  to show an isomorphism between  $G'$  and  $G_i$
- ★  $P \rightarrow V$ : if  $i=2$ , reply with  $g$ , else if  $i=1$ , reply with  $gf$

# Example: Graph Isomorphism

- ★  $P \rightarrow V$ : select and send random isomorphic copy of  $G_2$ , where  $g$  was the isomorphism
- ★  $V \rightarrow P$ : (Just get  $G$ ) Pick  $i$  from  $\{1,2\}$  randomly, send  $G_i$  if isomorphism exists
- ★  $P \rightarrow V$ : if  $i=1$ , reply with  $g$

If (as claimed) the two graphs are isomorphic, then the graph sent here is isomorphic to both input graphs

# Example: Graph Isomorphism

- ★  $P \rightarrow V$ : select and send random isomorphic copy of  $G_2$ , where  $g$  was the isomorphism
- ★  $V \rightarrow P$ : (Just  $g$  randomly, and check if isomorphism between  $G'$  and  $G_i$ )
- ★  $P \rightarrow V$ : if  $i=2$ , reply with  $g$ , else if  $i=1$ , reply with  $gf$

If the reply is a valid isomorphism between  $G_2$  and  $G_i$ , then OK



$0 + 0 = 0$ . Right?

- ✦ What if two ZK proofs are sequentially composed?
- ✦ Is the resulting interaction still zero knowledge?
- ✦ YES!
- ✦ This is important because real ZK systems iterate one round many times

# $0 + 0 = 0$ . Right? Wrong!

- ✦ What if two ZK proofs are composed *in parallel*, i.e. more than one prover talking to the same verifier?
- ✦ Is the resulting system still ZK?
- ✦ NO!!
- ✦ Why? Because the verifier can “cheat” and gain knowledge by pitting the provers (who have boundless computing power) against each other

# Outline

## ★ Some theory

- ★ What is a proof?
- ★ What is knowledge?
- ★ Interactive proofs
- ★ Zero knowledge proofs
- ★ ZK proofs for NP languages

## ★ Applications



# The Big Result

*Every* language in NP has  
a zero knowledge proof



## Sidebar: Bit Commitment

- ✦ Goal:
- ✦ Alice commits to a value
- ✦ Bob must not know this value until Alice reveals it
- ✦ Bob must be sure that Alice didn't change this value



## Sidebar: Bit Commitment

- ✦ **Secrecy:** At end of first phase: Bob does not have any knowledge of Alice's value
- ✦ **Unambiguity:** Given the transcript of the interaction during the first phase, there is at most one value which Bob may later accept as a legal "opening" of the commitment

# Goldwasser-Micali Scheme for Bit Commitment

- ★  $n = pq$ ,  $p$ ,  $q$  primes
- ★  $m$  is a quadratic residue mod  $n$
- ★ Alice gives Bob a **blob**:

$$y = f(b, x) = m^b x^2 \text{ mod } n$$

$b$  - bit being committed

$x$  - randomly chosen

Bob cannot get  $b$  from  $y$  because of intractibility of QR problem

- ★ To reveal: Alice simply reveals  $b$  and  $x$ , and Bob verifies

# ZK for NP

- ✦ If we show one NP-complete language that has a ZK proof, we're done
- ✦ E.g. Graph 3-colorability (G3C)
- ✦ Arbitrary choice among NP-complete problems

# G3C is ZK

- ✱  $P \rightarrow V$ : generate random 3-coloring of  $G$ , say  $f$ . Send  $|V|$  opaque locked boxes, where the  $i^{\text{th}}$  box has the color of the  $i^{\text{th}}$  vertex
- ✱  $V \rightarrow P$ : select a random edge and send it
- ✱  $P \rightarrow V$ : Send keys to the boxes corresponding to the vertices of this edge
- ✱  $V$ : open boxes, and if they have different elements, then OK

# G3C is ZK

- ★  $P \rightarrow V$ : generate random 3-coloring of  $G$ , say  $f$ . Send  $|V|$  opaque locked boxes, where the  $i^{\text{th}}$  box has the color of the  $i^{\text{th}}$  vertex
- ★  $V \rightarrow P$ : select a random edge and send it
- ★  $P \rightarrow V$ : send  $|E|$  boxes  
corresponding to each edge
- ★  $V$ : open all boxes  
elements, then check

Each of these is a random pair from  $\{1,2,3\}$  and so yields no knowledge

# G3C is ZK

- ★  $P \rightarrow V$ : generate random 3-coloring of  $G$ , say  $f$ . Send  $|V|$  opaque locked boxes, where the  $i^{\text{th}}$  box has the color of the  $i^{\text{th}}$  vertex
- ★  $V \rightarrow P$ : select a random edge and send it
- ★  $P \rightarrow V$ : Send keys to the boxes corresponding to the vertices of this edge
- ★  $V$ : open boxes, and check that the colors of the elements, then  $O$

Verifier wants to see the colors of the vertices of this edge

# An example application

- ★ Alice sends ciphertext to many parties  $R_i$ , and wishes to show Bob that the corresponding plaintexts are the same
- ★ Ciphertext sent to is  $E_i(r_i, M_i)$ , where
  - ★  $r_i$  is random
  - ★  $M_i$  is message
- ★ Want to show that  $E_i(r_i, M_i) = E_j(r_j, M_j)$

# An example application

- ★ Consider the language

$$L = \{(C_1, C_2): \text{there exists } r_1, r_2, M \\ \text{s.t. } C_1 = E_1(r_1, M) \text{ and } C_2 = E_2(r_2, M)\}$$

- ★  $L$  is in NP

- ★ Just do a ZK proof for it!



# Outline

## ★ Some theory

- ★ What is a proof?
- ★ What is knowledge?
- ★ Interactive proofs
- ★ Zero knowledge proofs
- ★ ZK proofs for NP languages

## ★ Applications

# Identification

- ★ Alice is identified by some secret she alone is known to possess - e.g. a password
- ★ Problems
  - ★ The authenticator must be trusted
  - ★ If secret sniffed or given to untrusted party, can impersonate
- ★ Use zero knowledge!

# Fiat-Shamir Identification

One time setup:

- ☀ Trusted center published modulus  $n=pq$ , but keeps  $p$  and  $q$  secret
- ☀ Alice selects a secret prime  $s$  coprime to  $n$ , computes  $v=s^2 \bmod n$ , and registers  $v$  with the trusted center as its public key

# Fiat-Shamir Identification

Protocol messages:

$$A \rightarrow B: x = r^2 \bmod n$$

$$B \rightarrow A: e \text{ from } \{0, 1\}$$

$$A \rightarrow B: y = rs^e \bmod n$$

# Fiat-Shamir Identification

Protocol messages:

$A \rightarrow B: x =$

$B \rightarrow A: e \text{ from } \{0, 1\}$

$A \rightarrow B: y = rs^e \text{ mod } n$

If  $e=0$ , then the response  $y=r$  is independent of secret  $s$

# Fiat-Shamir Identification

Protocol messages:

$$A \rightarrow B: x = r^2 \text{ mod } n$$

$$B \rightarrow A: e \text{ from } \{0, 1\}$$

$$A \rightarrow B: y = rs^e \text{ mod } n$$

If  $e=1$ , then information pairs  $(x, y)$  can be simulated by choosing  $y$  randomly, and setting  $x=y^2 \text{ mod } n$



# References

- ★ Oded Goldreich and his fragments of a crypto book



Thank You