601.445/645
Practical Cryptographic Systems

Asymmetric Cryptography III

Instructor: Matthew Green

Housekeeping

- A2
  - Due 23rd February, 11:59pm
- Quiz moved to Weds
  - All reading material before RSA reading

News?

News?

BBC

Home News Sport Business Innovation Culture Arts Travel Earth Audio Video Live

US politicians furious at UK demand for encrypted Apple data

3 days ago

Share Save

Graham Fraser
Technology Reporter

News?

FOR SUBSCRIBERS

Paranoia seeps into a federal workforce worried about getting fired and being watched

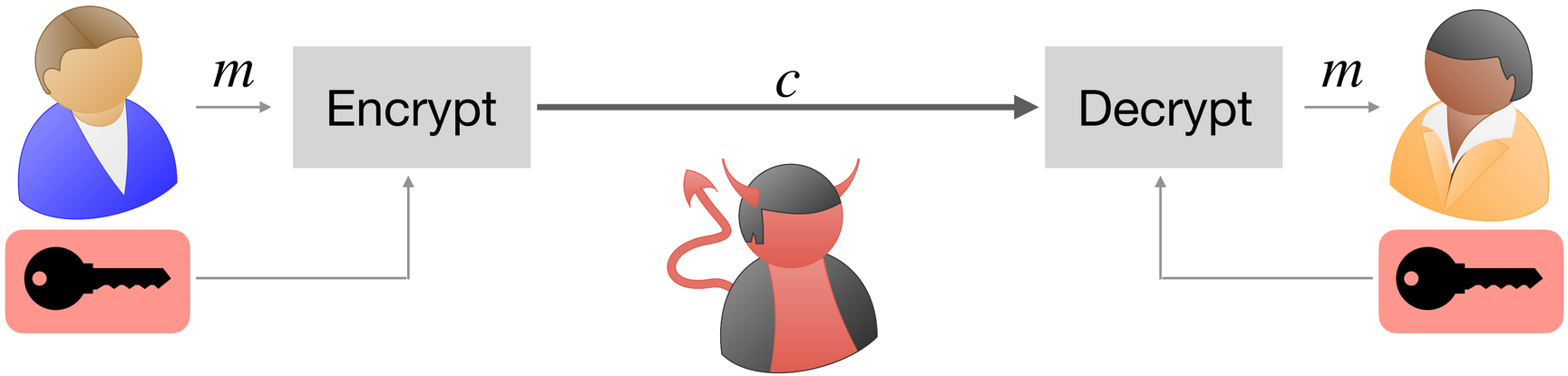
By Sean Lyngaas, Tami Luhby and Hadas Gold, CNN

5 minute read · Published 6:00 AM EST, Mon February 17, 2025

MORE FROM CNN

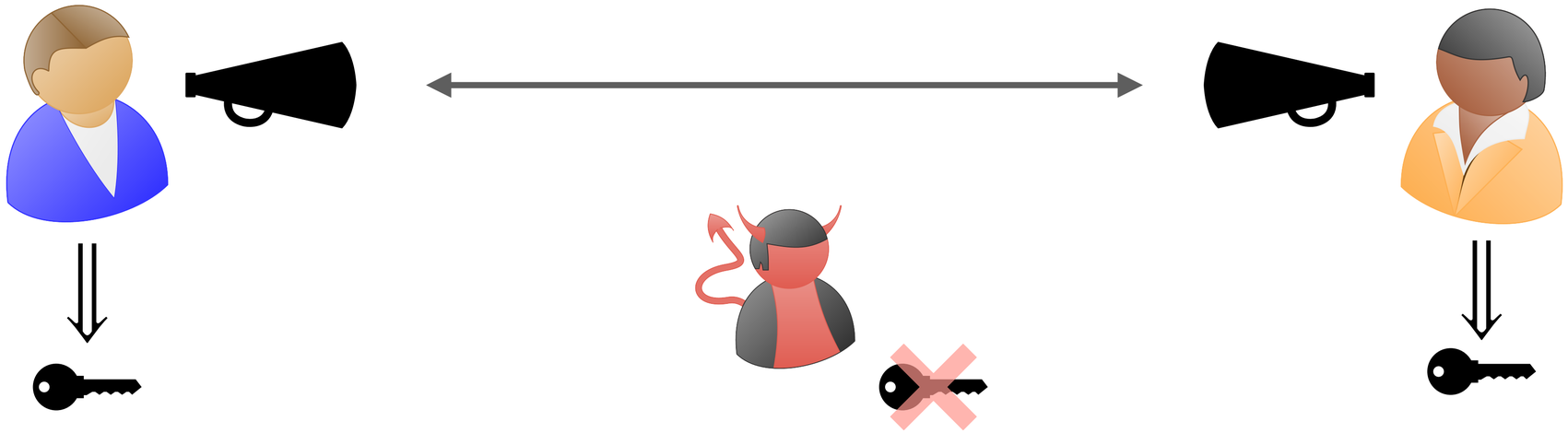
DOGE's power expands as fe

Review



How do parties agree on a common key?

Key Exchange

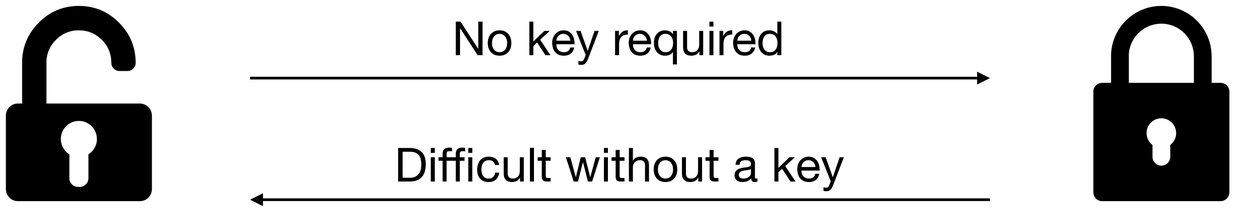


Diffie-Hellman Key Exchange

Agreeing on a common secret over an untrusted/public channel

Key Idea: Exploiting asymmetry

Often present in the real world!



Discrete Logarithm problem

- **Discrete logarithm problem**

  Given: $x \in_R 0, \ldots, p-2$

  $$\langle g \rangle = \mathbb{G} \qquad order(g) = p - 1$$

  $$h = g^x$$

  Find: x

  This problem is hard if for all p.p.t. adversaries, all attackers find x with "small" probability

See Boneh-Shoup, Appendix A

Discrete Logarithm problem

This means that “reversing” exponentiation is assumed to have super-polynomial running time.

How about the exponentiation itself?

- **Discrete logarithm problem**

  Given: $x \in_R 0, \ldots, p-2$

  $$\langle g \rangle = \mathbb{G} \qquad order(g) = p - 1$$

  $$h = g^x$$

  Find: x

  This problem is hard if for all p.p.t. adversaries, all attackers find x with “small” probability

See Boneh-Shoup, Appendix A

Discrete Logarithm problem

This means that “reversing” exponentiation is assumed to have super-polynomial running time.

How about the exponentiation itself?

- **Discrete logarithm problem**

  Given: $x \in_R 0, \ldots, p-2$

  $\langle g \rangle = \mathbb{G}$ $order(g) = p$

  $h = g^x$

  Find: x

Note that for this to hold, the size of *p* must be pretty large!

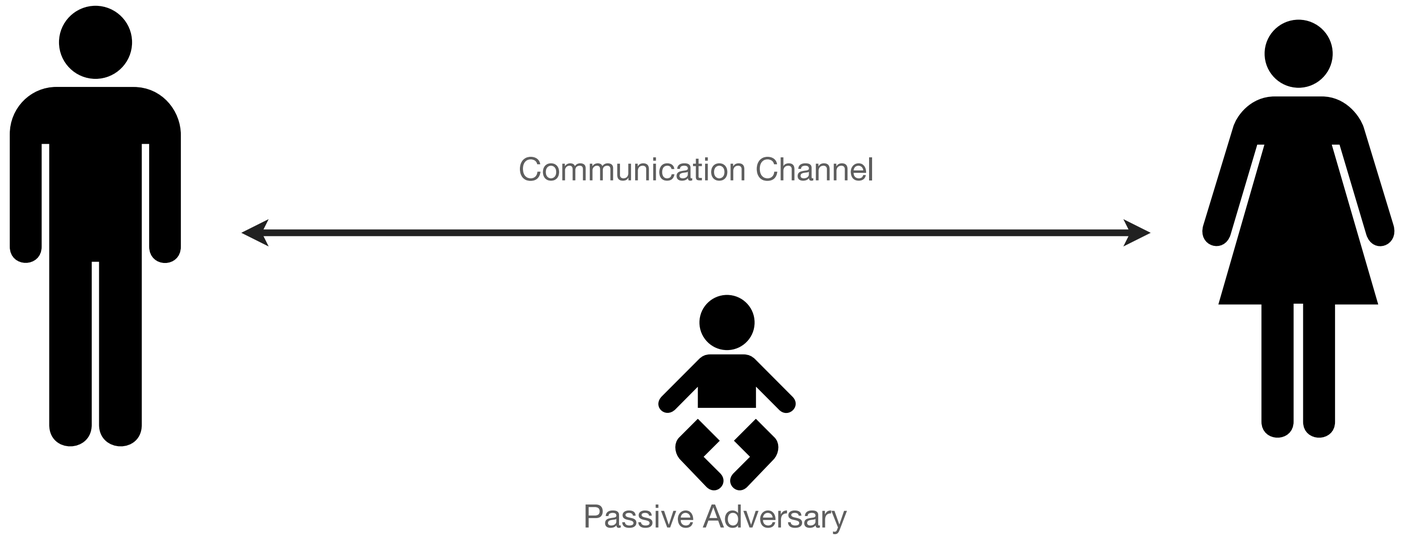
In practice, we typically assume *p* is at least 1024 bits. And 3072 bits is the minimum in modern protocols!

This problem is hard if for all p.p.t. adversaries, all attackers find x with “small” probability

See Boneh-Shoup, Appendix A

Key Agreement

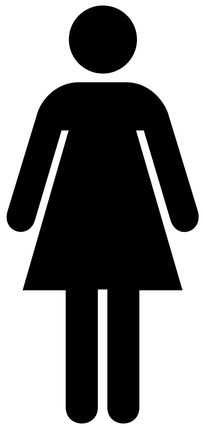
- Establish a shared key in the presence of a passive adversary

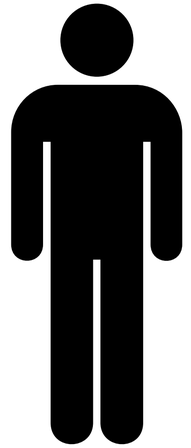


D-H Protocol

$$p, \langle g \rangle = \mathbb{Z}_p^*$$

$$a \in \mathbb{Z}_{\phi(p)}$$

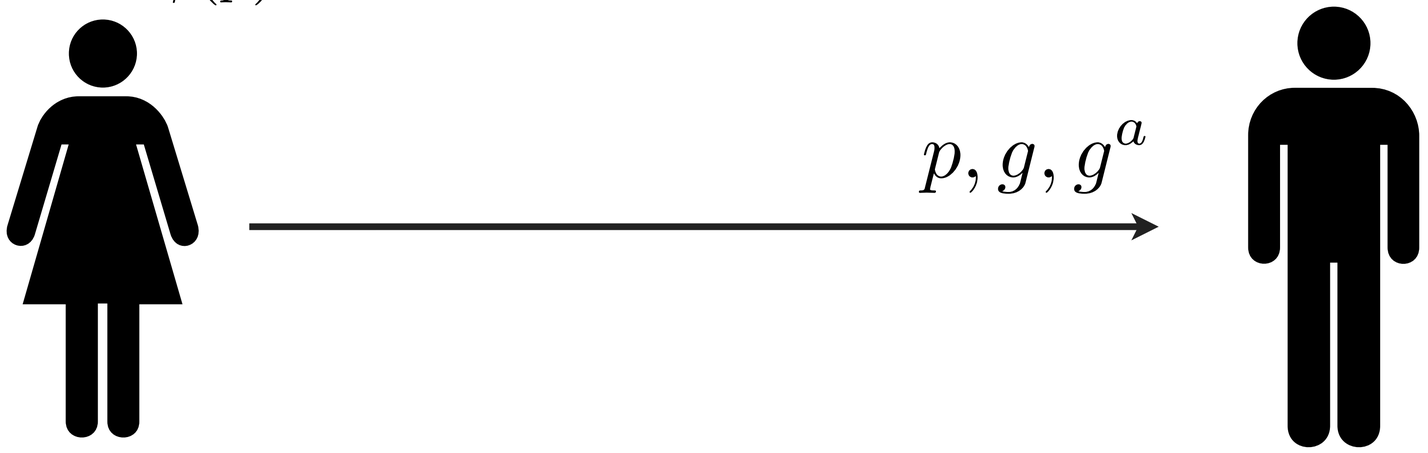




D-H Protocol

$$p, \langle g \rangle = \mathbb{Z}_p^*$$

$$a \in \mathbb{Z}_{\phi(p)}$$

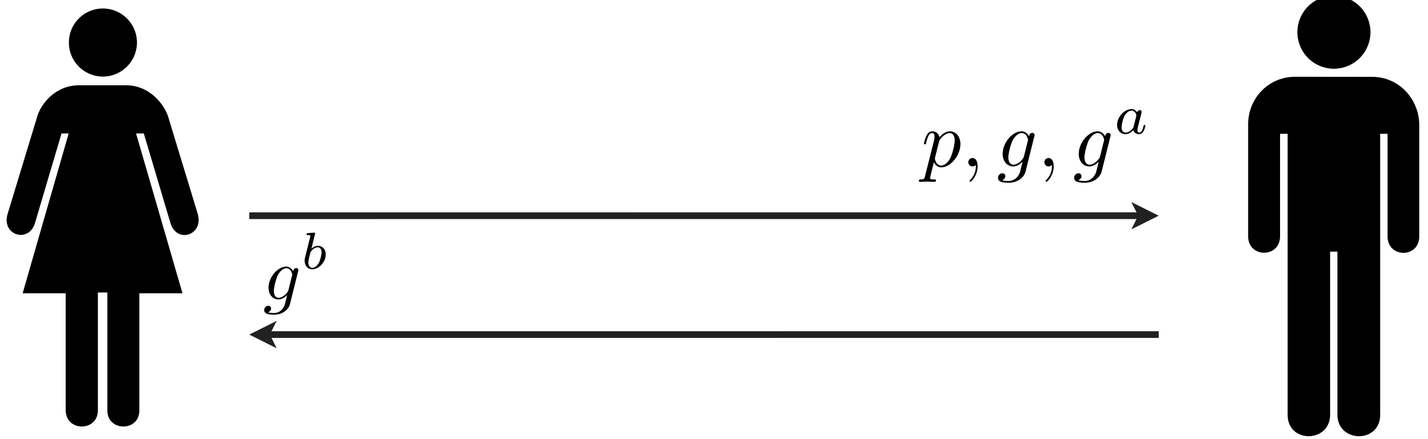


D-H Protocol

$p, \langle g \rangle = \mathbb{Z}_p^*$

$a \in \mathbb{Z}_{\phi(p)}$

$b \in \mathbb{Z}_{\phi(p)}$

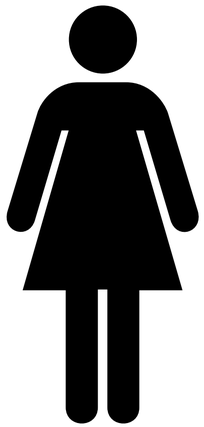


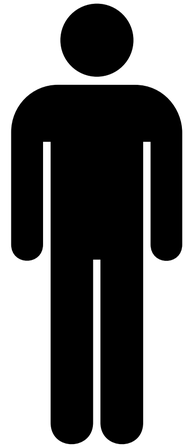
D-H Protocol

$p, \langle g \rangle = \mathbb{Z}_p^*$

$a \in \mathbb{Z}_{\phi(p)}$

$b \in \mathbb{Z}_{\phi(p)}$





p, g, g^a

g^b

g^{ba}

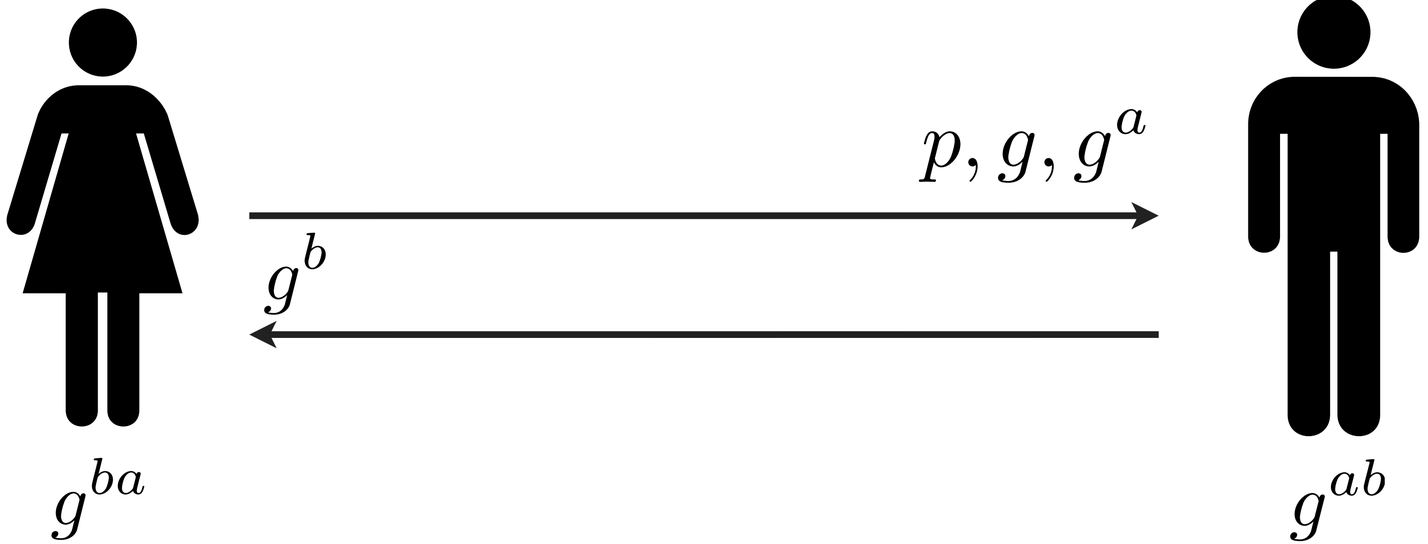
g^{ab}

D-H Protocol

$p, \langle g \rangle = \mathbb{Z}_p^*$

$a \in \mathbb{Z}_{\phi(p)}$

$b \in \mathbb{Z}_{\phi(p)}$



Usually we “hash” the shared secret value to make a secret encryption key, and then encrypt using a fast symmetric encryption scheme!

Hard problems (2)

- **Diffie-Hellman problem**

  Given: $a, b \in_R 0, \ldots, p-2$

  $$\langle g \rangle = \mathbb{G} \qquad order(g) = p - 1$$

  $$(g, g^a, g^b)$$

  Find: g^{ab}

  This problem is hard if for all p.p.t. adversaries, all attackers output a solution with “small” probability

See Boneh-Shoup, Appendix A

Hard problems (2)

- **Diffie-Hellman problem**

  Given: $a, b \in_R 0, \ldots, p-2$

  $$\langle g \rangle = \mathbb{G} \qquad order(g) = p - 1$$

  $$(g, g^a, g^b)$$

  Find: g^{ab}

  This problem is hard if for all p.p.t. adversaries, all attackers output a solution with “small” probability.

Notice this is just the Diffie-Hellman scheme re-written as a mathematical assumption!

See Boneh-Shoup, Appendix A

Hard problems (2)

- **Diffie-Hellman problem**

  Given: $a, b \in_R 0, \ldots, p-2$

  $\langle g \rangle = \mathbb{G} \qquad order(g) = p - 1$

  (g, g^a, g^b)

  Find: g^{ab}

Notice this is just the Diffie-Hellman scheme re-written as a mathematical assumption!

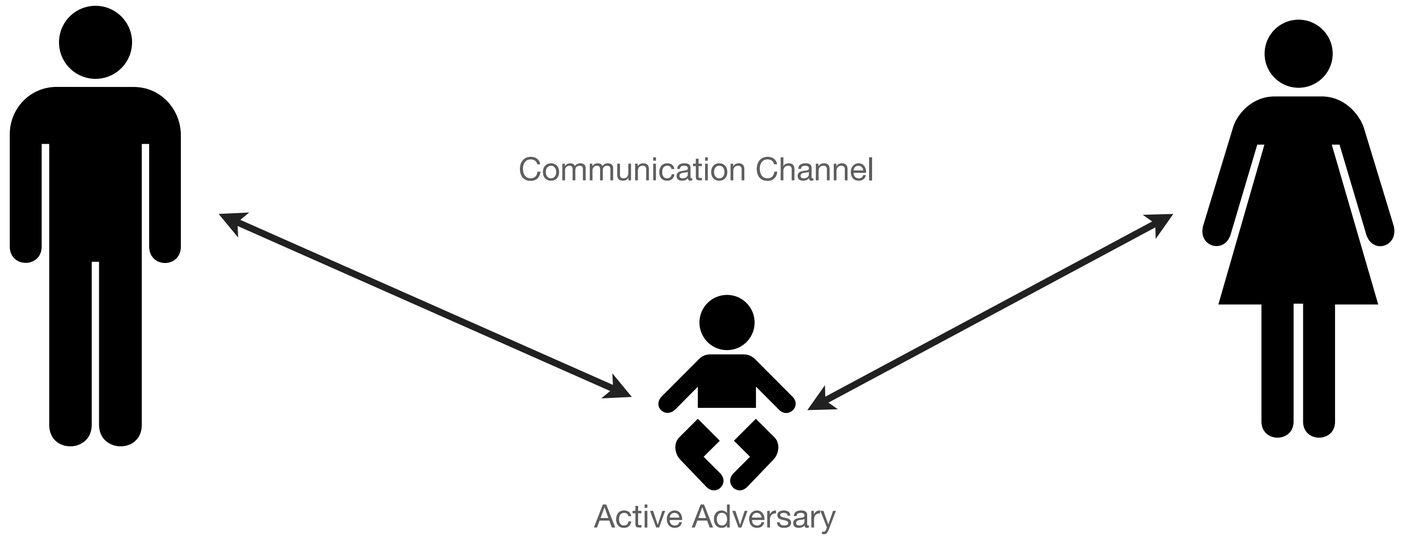
Note that for this to hold, the size of *p* must be pretty large!

In practice, we typically assume *p* is at least 1024 bits. And 3072 bits is the minimum in modern protocols!

This problem is hard if for all p.p.t. adversaries, all attackers output a solution with “small” probability.

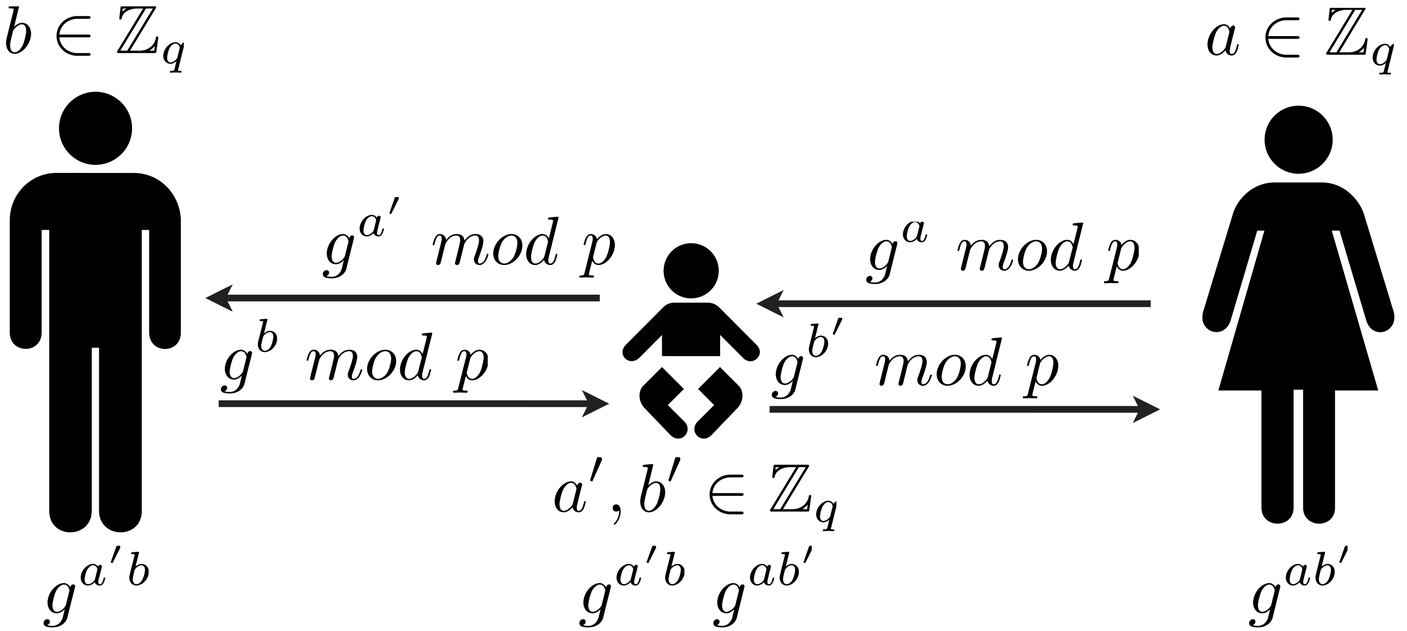
See Boneh-Shoup, Appendix A

What if we have an active adversary?



Man in the Middle

- Assume an active adversary:



Man in the Middle

- Caused by lack of authentication
  - D-H lets us establish a shared key with anyone...
    but that's the problem…
  - We don't know if the person we're talking to is the right person
- Solution?

Preventing MITM

- Verify key via separate channel
- Password-based authentication
- Authentication via PKI

Digital Signatures

- Similar to MACs, with public keys
  - Secret key used to sign data
  - Public key can verify signature
  - Advantages over MACs?

Digital Signatures

- Three algorithms:

  Keygen() -> (vk, sk)

  Sign(sk, message) -> sig

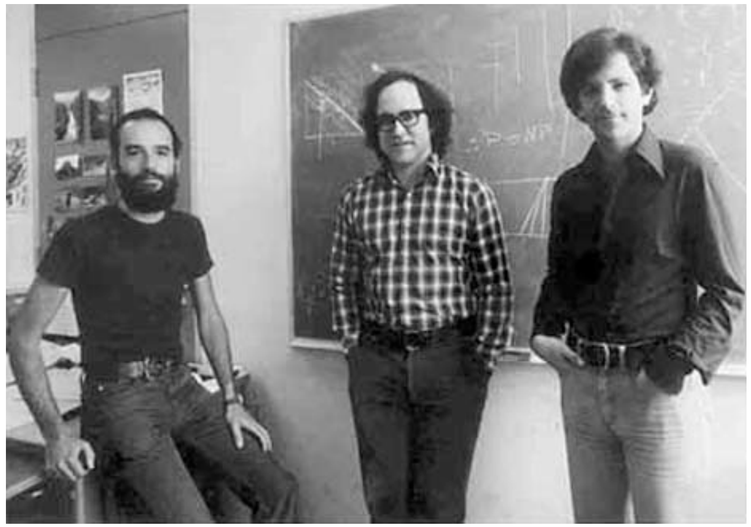
  Verify(pk, message, sig) -> True/False

Signature: definitions

- Existential Unforgeability under Chosen Message Attack
  - No efficient adversary can “forge” a signature on any new message (that it hasn’t received a signature for)
  - Even if it has access to an “oracle” that signs any message it wants
  - “Strong unforgeability” <- can’t even make a new signature for an existing message

Public Key Encryption

- What if our recipient is offline?
  - Key agreement protocols are interactive
  - e.g., want to send an email



Public Key Encryption