DTTF/NB479: Dszquphsbqiz Day 30

- Announcements:
- Questions?
- This week:
  - Digital signatures, DSA
  - Coin flipping over the phone

RSA Signatures allow you to recover the message from the signature; ElGamal signatures don't

Sig = f(user, message)

RSA

- Alice chooses:
  - p,q, n=pq,
  - e: gcd(n, (p-1)(q-1))=1,
  - d: ed ≡ 1(mod ((p-1)(q-1))
- Publishes n, e
- Alice's signature:
  - $y \equiv m^d (\text{mod } n)$. Delivers (m, y)
- Bob's verification:
  - Does $m \equiv y^e$ (mod n)?

ElGamal

- Alice chooses:
  - p,primitive root α, secret **a**, and $\beta \equiv \alpha^a$ (mod p)
  - Publishes (p, α, β), keeps **a** secret
- Alice's signature:
  - Chooses k: random, gcd(k, p-1)=1
  - Sends m, (r,s), where:
    - $r \equiv \alpha^k$ (mod p)
    - $s \equiv k^{-1}(m - ar)$ (mod p-1)
- Bob's verification:
  - Does $\beta^r r^s \equiv \alpha^m$ (mod p)?

It's quicker to sign a short digest than to sign a long message

- Note that we need to choose $n > m$ in RSA, $p > m$ in ElGamal
  - Problem: m could be long!
  - But h(m) is short!
- So Alice sends (m, sig(h(m)))

- Eve intercepts this, wants to sign m' with Alice's signature, so needs sig(h(m')) = sig(h(m)), and thus h(m)=h(m')
  - Why can't she do this?

Birthday attacks can be successful on signatures that are too short

- Slightly different paradigm: two rooms with r people each. What's the probability that someone in this room has the same birthday as someone in the other room.
- Approximation: $$1-e^{\frac{-r^2}{N}}$$
  - Note that we divide by N, not 2N.
  - But setting the probability = 0.5 and solving for r, we get r=c*sqrt(n) again (where c=sqrt(ln 2)~.83)

  - Consider a 50-bit hash. Only need 2^25 documents
  - These are relatively easy to generate, actually.

Birthday attacks on signatures that are too short

- *Mallory* generates 2 groups of documents:

r "good docs"

r "fraudulent docs"

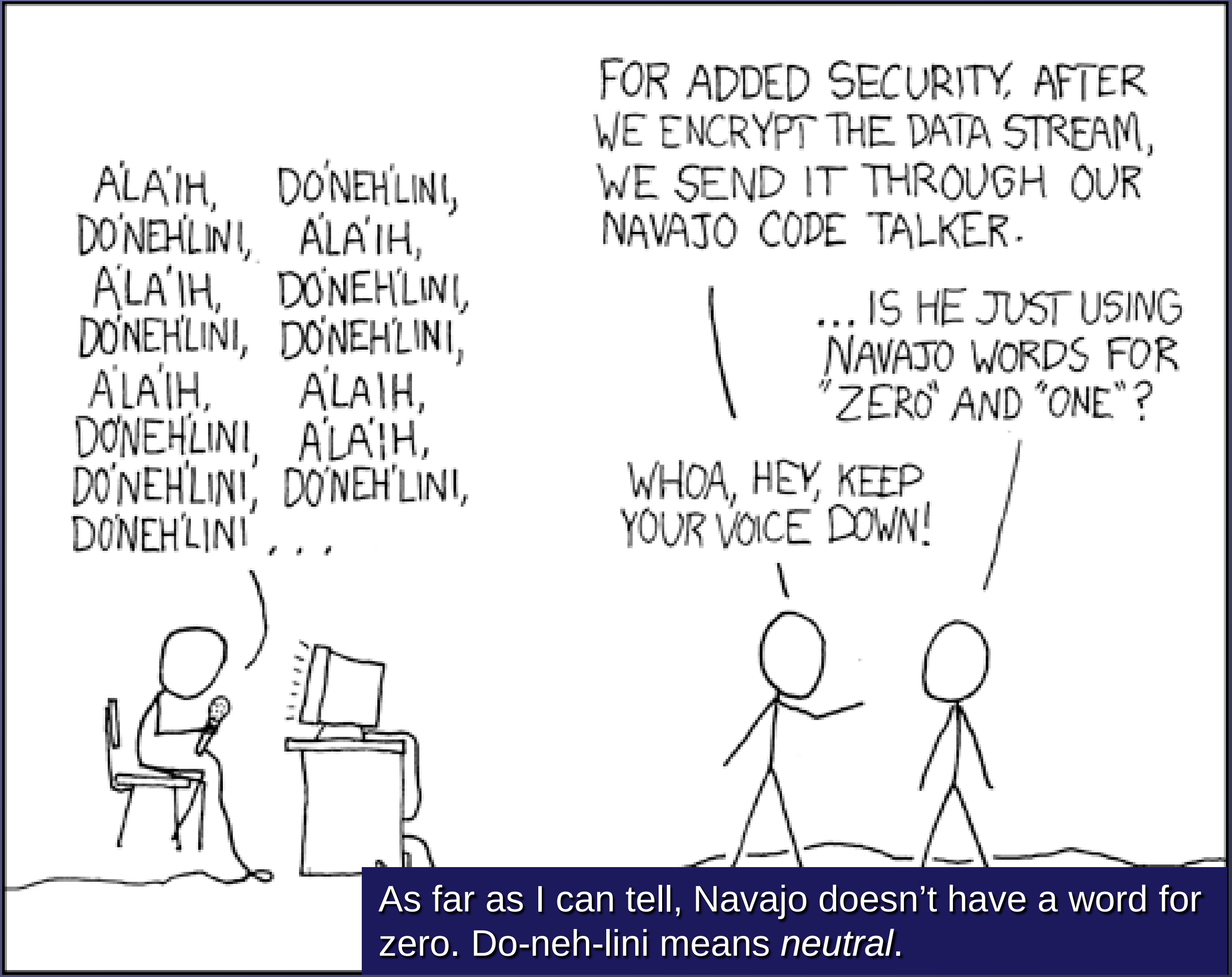
- Want a match (m_1, m_2) between them such that $h(m_1) = h(m_2)$
- *Mallory* sends $(m_1, h(m_1))$ to Alice, who returns signed copy: $(m_1, sig(h(m_1)))$.
- *Mallory* replaces m_1 with m_2 and uses $sig(h(m_1)$ as the signature.
  - The pair $(m_2, sig(h(m_1))$ looks like Alice's valid signature!
- Alice's defense? What can she do to defend herself?

Alice's defense

- She changes a random bit herself!
- Note this changes her signature: $(m_1', \; sig(h(m_1')))$
  - Mallory is forced to generate another message with the same hash as this new document.
  - Good luck!

- Lessons:
  - Birthday attacks essentially halve the number of bits of security.
    - So SHA-1 is still secure against them
  - Make a minor change to the document you sign!

Code-talkers?



As far as I can tell, Navajo doesn't have a word for zero. Do-neh-lini means *neutral*.

http://xkcd.com/c257.html

DSA: Digital Signature Algorithm

- 1994
- Similar to ElGamal
  - signature with appendix
  - But verification is faster
  - And it’s guaranteed to be more secure

- Assume m is already hashed using SHA: so we are signing a 160-bit message, m.

DSA: Digital Signature Algorithm

- Alice's Setup:
  - m: 160-bit message
  - q: 160-bit prime
  - p: 512-bit prime, such that q is a factor of (p-1)
  - g: a primitive root of p.
  - $\alpha \equiv g^{(p-1)/q} \pmod{p}$
    - Then $\alpha^q \equiv 1 \pmod{p}$. (Why?)
  - $\beta \equiv \alpha^a$. Secret a, $0 < a < q-1$
  - Publishes: (p,q,α,β)
- Sig = (r,s)
  - random k, $0 < k < q-1$
  - $r \equiv \alpha^k \pmod{q}$
  - $s = k^{-1}(m + ar) \pmod{q}$
- Verify:
  - Compute $u1 \equiv s^{-1}m \pmod{q}$, $u2 \equiv s^{-1}r \pmod{q}$
  - Does $(\alpha^{u1}\beta^{u2} \pmod{p}) \pmod{q} = r$?

q=17

p=103

g=2

α=?

DSA: Digital Signature Algorithm

- Alice's Setup:
  - m: 160-bit message
  - q: 160-bit prime
  - p: 512-bit prime, such that q is a factor of (p-1)
  - g: a primitive root of p.
  - $\alpha \equiv g^{(p-1)/q}$ (mod p)
    - Then $\alpha^q \equiv 1$ (mod p). (Why?)
  - $\beta \equiv \alpha^a$. Secret a, 0 < a < q-1
  - Publishes: (p,q,α,β)
- Sig = (r,s)
  - random k, 0 < k < q-1
  - $r \equiv \alpha^k$ (mod q)
  - $s = k^{-1}(m + ar)$ (mod q)
- Verify:
  - Compute u1 $\equiv s^{-1}m$ (mod q), u2 $\equiv s^{-1}r$ (mod q)
  - Does $(\alpha^{u1}\beta^{u2}$ (mod p))(mod q) = r?

q=17

p=103

g=2

α=64

- Advantages over ElGamal?
  - In ElGamal, if you could solve $r = \alpha^k$ (mod p) by Pollig-Hellman, you'd have k.
  - In DSA, (p-1) has a large factor, q.
  - If you could solve the non-q factors, there would still be q possibilities for k.
  - How many ints (mod p) give a specific int (mod q)?

DSA: Digital Signature Algorithm

- Alice's Setup:
  - m: 160-bit message
  - q: 160-bit prime
  - p: 512-bit prime, such that q is a factor of (p-1)
  - g: a primitive root of p.
  - $\alpha \equiv g^{(p-1)/q} \pmod{p}$
    - Then $\alpha^q \equiv 1 \pmod{p}$. (Why?)
  - $\beta \equiv \alpha^a$. Secret a, $0 < a < q-1$
  - Publishes: (p,q,α,β)
- Sig = (r,s)
  - random k, $0 < k < q-1$
  - $r \equiv \alpha^k \pmod{q}$
  - $s = k^{-1}(m + ar) \pmod{q}$
- Verify:
  - Compute $u1 \equiv s^{-1}m \pmod{q}$, $u2 \equiv s^{-1}r \pmod{q}$
  - Does $(\alpha^{u1}\beta^{u2} \pmod{p}) \pmod{q} = r$?

q=17

p=103

g=2

α=64

- How hard is it to search for a 512-bit prime p = kq + 1 for some even number k?
  - How do we search for primes?
  - 1/115 of odd 100-digit numbers are prime.
  - What fraction of odd 512-bit integers are prime?
  - Recall our discussion of the density of primes

(Day 21) Using within a primality testing scheme

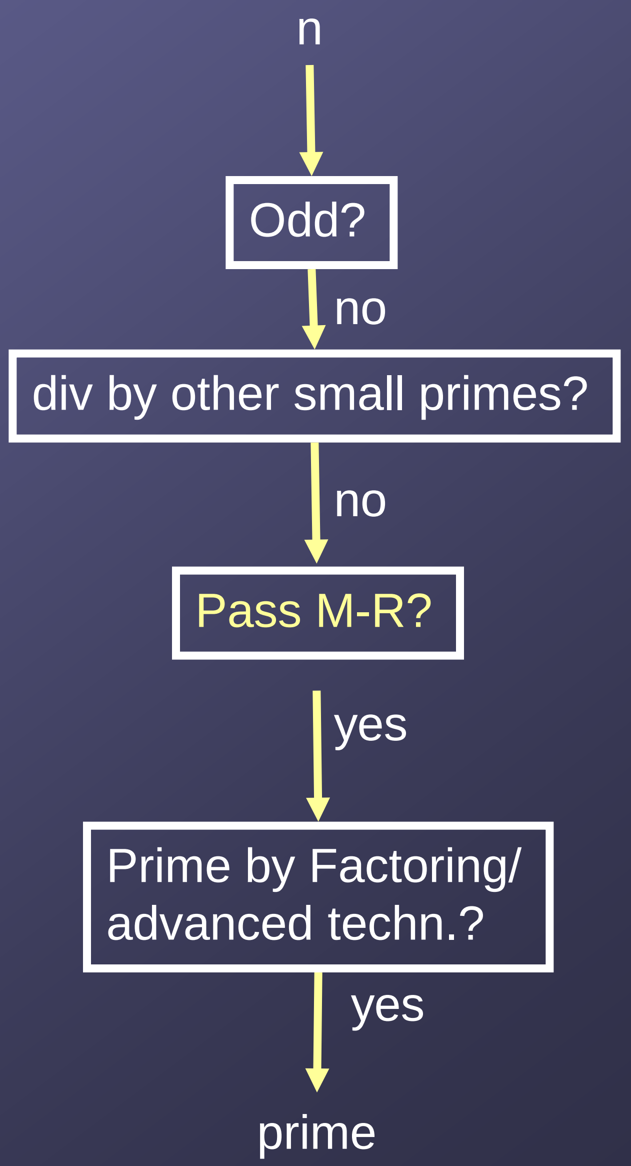
- Finding large probable primes
  - #primes < x = $\pi(x) \to \dfrac{x}{\ln(x)}$

  Density of primes: ~1/ln(x)

  For 100-digit numbers, ~1/230.

  So ~1/115 of odd 100-digit numbers are prime

  Can start with a random large odd number and iterate, applying M-R to remove composites. We'll soon find one that is a likely prime.



DSA: Digital Signature Algorithm

- Alice's Setup:
  - m: 160-bit message
  - q: 160-bit prime
  - p: 512-bit prime, such that q is a factor of (p-1)
  - g: a primitive root of p.
  - $\alpha = g^{(p-1)/q}$ (mod p)
    - Then $\alpha^q = 1$ (mod p). (Why?)
  - $\beta = \alpha^a$. Secret a, 0 < a < q-1
  - Publishes: (p,q,α,β)
- Sig = (r,s)
  - random k, 0 < k < q-1
  - $r = \alpha^k$ (mod p)
  - $s = k^{-1}(m + ar)$ (mod q)
- Verify:
  - Compute $u1 = s^{-1}m$, $u2 = s^{-1}r$
  - Does $(a^{u1}b^{u2}$ (mod p))(mod q) = r?

Show that order of ops matters:

$(\alpha^k$ (mod p))(mod q) ≠ $(\alpha^k$ (mod q))(mod p)

Easier: find

(a(mod p))(mod q) ≠ (a(mod q))(mod p)

Latest versions

- Recommended:
  - SHA-224/256/384/512 as the hash function
  - q of size 224 and 256 bits
  - p of size 2048 and 3072.