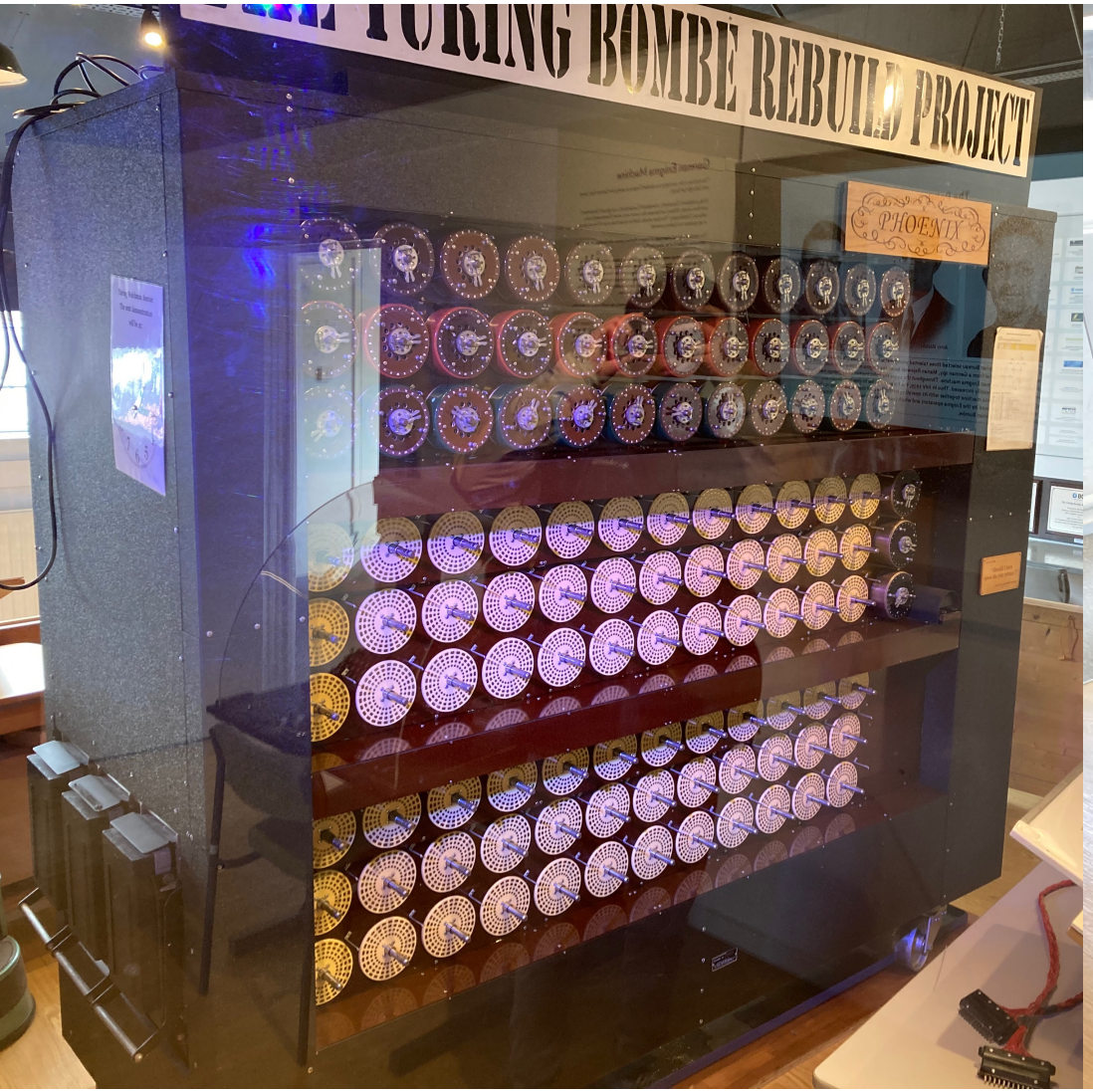
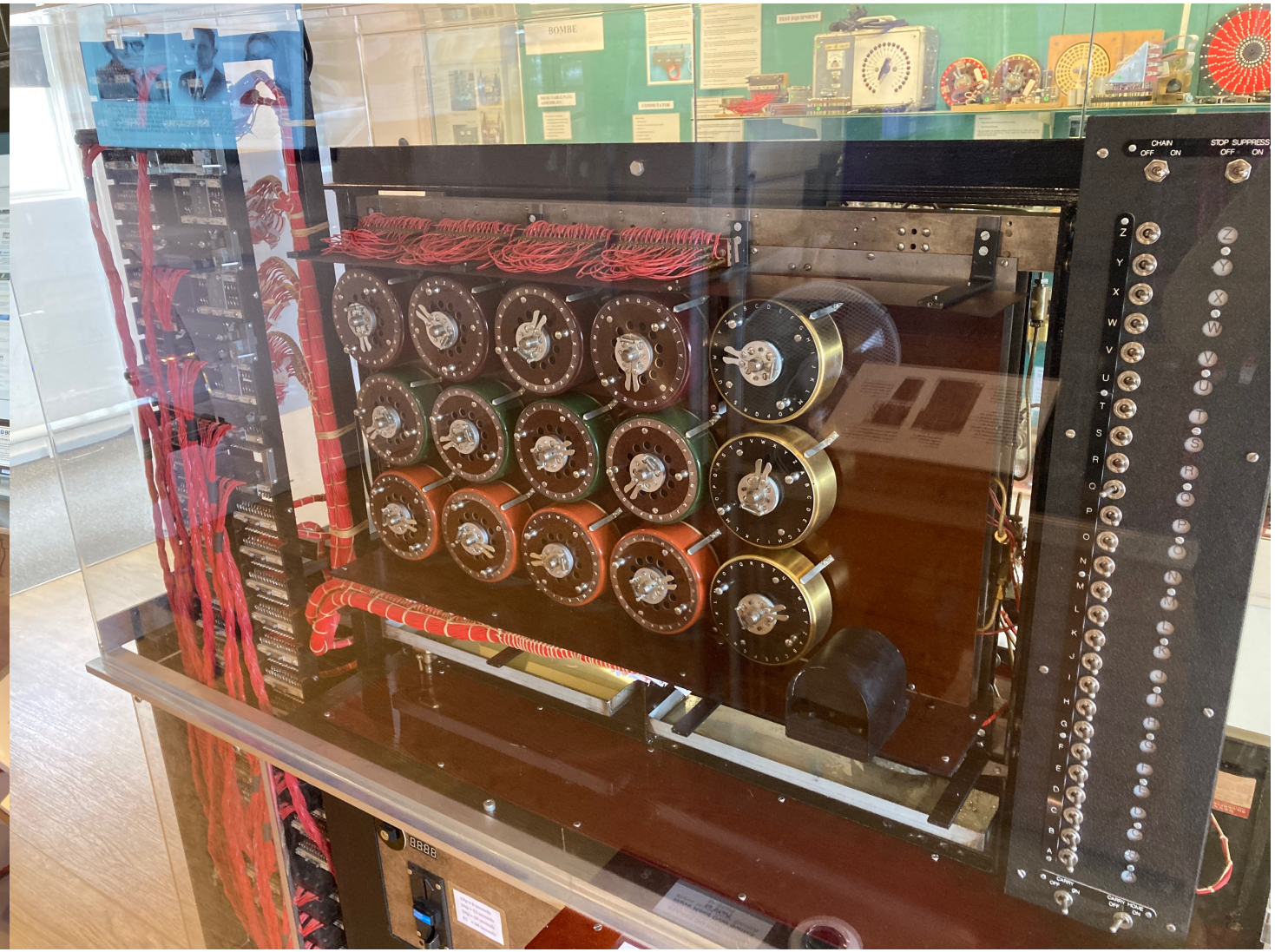
Turing's Bombe





First attempt at a more secure cryptosystem

Scheme A: Alice and Bob share a large prime number (k =2280176348 9) as their secret key.

Alice wants to send the message $VICTORY$

1. Convert it to a number (A=01, B=02, … , Z=26) 22090320151825
2. Pad a few extra digits to make it a prime number m = 220903201518251 3
3. Ciphertext = mk = 503698255498207185946678 57
4. Assuming that factoring is hard, Eve cannot decipher this string of digits
5. Bob divides mk by k to recover the plaintext.

What can go wrong?

- Eve waits for two different ciphertexts $m_1k,\ m_2k$ to be transmitted between Alice and Bob.
- Eve computes $\gcd(m_1k, m_2k) = k$
- Eve can now quickly decipher every message between Alice and Bob!

Another attempt

Scheme B: Alice and Bob figure that working modulo primes will scramble things up and defeat the gcd. They still share a secret prime key k. The modulus p is made public.

$$\begin{aligned} \mathcal{E}(m) &\equiv mk \ (mod\ p) \\ \Delta\big(\mathcal{E}(m)\big) &\equiv \mathcal{E}(m)k^{-1} \ (mod\ p) \\ &\equiv mkk^{-1} \ (mod\ p) \\ &\equiv m \ (mod\ p) \end{aligned}$$

The gcd trick doesn't work now.

What can go wrong?

- Eve sees plaintext m transmitted in the clear.
- Eve also sees $\mathcal{E}(m) = mk \ (mod\ p)$ transmitted.
- Eve computes $m^{p-2}\ \mathcal{E}(m) \equiv m^{p-1}k \ \equiv k \ (mod\ p)$ to get the secret key!

Systems in use: DES, Triple DES, Blowfish, Twofish, Skipjack, AES, …

Private Key Cryptosystems

Alice and Bob share a secret private key k.

m → $\mathcal{E}_k(m)$ → $\Delta(\mathcal{E}_k(m)) = m$

Requires Alice and Bob to select and share the secret key in advance.

Two problems:

1. We need a distinct key for every pair of individuals/companies/governments that communicate.
2. What if two strangers wish to communicate in a secure manner?

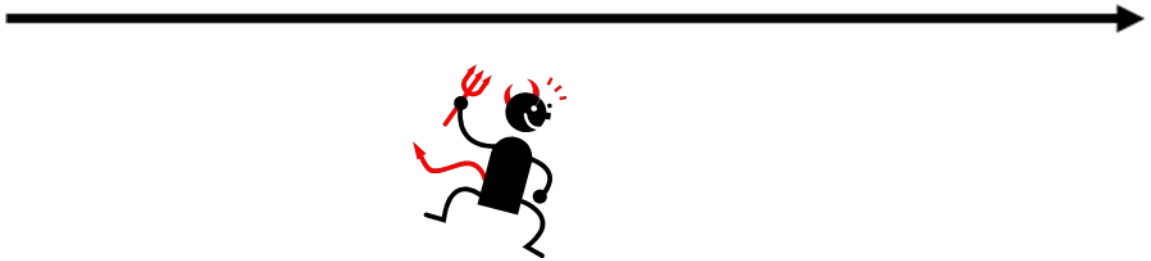
Public Key Cryptosystems

Alice has a secret private key $\mathcal{E}_A$ and a public key Δ_A
Bob has a secret private key $\mathcal{E}_B$ and a public key Δ_B

$\mathcal{E}_A(m), \Delta_A(m)$, can be computed quickly but computationally infeasible to recover m from $\Delta_A(m)$ without $\mathcal{E}_A$

$$\mathcal{E}_A\big(\Delta_A(m)\big) = m = \Delta_A\big(\mathcal{E}_A(m)\big)$$

Sending Bob a secure message



Alice has a secret private key $\mathcal{E}_A$ and a public key Δ_A
Bob has a secret private key $\mathcal{E}_B$ and a public key Δ_B

$$m \rightarrow \Delta_B(m) \rightarrow \mathcal{E}_B(\Delta_B(m)) = m$$

Bob can recover Alice's message quickly, but no one else can!

Signing a message

Alice has a secret private key $\mathcal{E}_A$ and a public key Δ_A
Bob has a secret private key $\mathcal{E}_B$ and a public key Δ_B

m → $< From\colon Alice, m, \mathcal{E}_A(From\colon Alice, m) >$ →

Bob can read m (so can Eve)
Bob can verify that the message came from Alice:

$$\Delta_A\big(\mathcal{E}_A(From\colon Alice, m)\big) \ ?=?\ From\colon Alice, m$$

Only Alice's public key will pass this verification check!

Secret and Signed

m → $< From: Alice, \Delta_B(m), \mathcal{E}_A(From: Alice, \Delta_B(m))$ →

Bob can read m but no one else can.
Bob can verify that the message came from Alice.

The RSA Public Key Cryptosystem

Invented in 1976 by Rivest, Shamir and Adleman at MIT

The NSA tried in vain to stop publication

Recently revealed to have been independently invented earlier by mathematicians at GCHQ (UK's NSA)

Depends on the assumption that factoring numbers is computationally infeasible.

If Quantum Computers become a reality, factoring will become feasible.

A Quick Review

Definition: Euler's totient function $\phi(n) =$ number of numbers in $[1, n]$ relatively prime to n.
Example:

$$\phi(1) = 1$$
$$\phi(6) = 2$$
$$\phi(17) = 16$$

Lemma:

1) If p is prime then $\phi(p) = p - 1$.
2) If p, q are primes then $\phi(pq) = (p-1)(q-1) = \phi(p)\phi(q)$

Theorem. If $\gcd(a, n) = 1$ then $a^{\phi(n)} \equiv 1 \ (mod\ n)$

So, if $n = pq\ (p, q\ primes)\ and \gcd(a, n) = 1\ \ then\ \ a^{(p-1)(q-1)} \equiv 1 (mod\ n)$

The RSA Public Key Cryptosystem

In advance:

1. Generate two primes p, q and compute $n = pq$
2. Pick e: $\gcd\big(e, (p-1)(q-1)\big) = 1$
3. Compute d: $de \equiv 1 \pmod{(p-1)(q-1)}$
4. The pair (n, e) is the public key.
5. The pair (n, d) is the secret key.

To encrypt plaintext m

1. Check that $\gcd(m, n) = 1$. If not, pad m with a few extra bits.
2. $\mathcal{E}(m) = rem(m^e, n)$

To decrypt ciphertext c

1. $\Delta(c) = rem(c^d, n)$

$$\phi(n) = \phi(pq) = \phi(p)\phi(q) = (p-1)(q-1)$$

If $\gcd(m, n) = p$ or q, then you've factored n!
If $\gcd(m, n) = n$, the ciphertext is 0.

Note: $\Delta\big(\mathcal{E}(m)\big) \equiv (m^e)^d \pmod{n}$

$$\equiv m^{de} \pmod{n}$$
$$\equiv m^{1+k(p-1)(q-1)} \pmod{n}$$
$$\equiv m \cdot m^{k\phi(n)} \pmod{n}$$
$$\equiv m \pmod{n}$$

Similarly, $\mathcal{E}\big(\Delta(m)\big) \equiv m \pmod{n}$

The RSA-129 Challenge from 1976

114381625757888867669235779976146612010218296721204236256256184293576935245733897830597123563958705058989075147599290026879543541

Cracked in 1994 after 8 months of computation by 600 volunteers from more than 20 countries on 6 continents!

“The magic words are squeamish ossifrage”

=

349052951084765094914784961990389813341776463849338784399082057 7

×

327691329932667095499619881908344614131776429679929425397982885 33

Since then RSA-768 cracked in 2010. Current recommendations are to use 2048-bit keys.

RSA-250 Broken in 2020

RSA-250 has 250 decimal digits (829 bits), and was factored in February 2020 by Fabrice Boudot, Pierrick Gaudry, Aurore Guillevic, Nadia Heninger, Emmanuel Thomé, and Paul Zimmermann.[39]

```
RSA-250 = 2140324650240744961264423072839333563008614715144755017797754920881418023447
          1401366433455190958046796109928518724709145876873962619215573630474547705208
          0511905649310668769159001975940569345745223058932597669747168173806936489469
          987157849495937497937
```

```
RSA-250 = 6413528947707158027879019017057738908482501474294344720811685963202453234463
          0238623598752668347708737661925585694639798853367
        × 3337202759497815655622601060535511422794076034476755466678452098702384172921
          0037080257448673296881877565718986258036932062711
```

RSA-260 [edit]

RSA-260 has 260 decimal digits (862 bits), and has not been factored so far.

```
RSA-260 = 2211282552952966643528108525502623092761208950247001539441374831912882294140
          2001986512729726569746599085900330031400051170742204560859276357953757185954
          2988389587092292384910067030341246205457845664136645406842143612930176940208
          4639106587591479425143514445819 9
```