

Topic 17: Security

RSA and Diffie–Hellman with real numbers you can compute by hand, and why quantum computers matter.

Optional stretch material. This goes beyond the 9618 syllabus, so you will not be asked about it directly in the exam. Its purpose is to deepen ideas you already know. **Builds on:** 17.1 Encryption, protocols and digital signatures. Try the challenge questions on the last pages before you read the answers.

RSA with tiny numbers

RSA's security rests on one fact: multiplying two large primes is easy, but factoring the product is infeasible. Here it is with numbers small enough for a calculator (real keys use primes of 1000+ bits).

- | | |
|--|--|
| 1. Choose primes | $p = 3, q = 11$ |
| 2. $n = p \times q$ | $n = 33$ |
| 3. $\phi = (p-1)(q-1)$ | $\phi = 20$ |
| 4. Choose e coprime to ϕ | $e = 3$ (public key: $n = 33, e = 3$) |
| 5. Find d : $e \times d = 1 \pmod{\phi}$ | $d = 7$ ($3 \times 7 = 21 = 1 \pmod{20}$; private key) |

Encrypt message $m = 4$: $c = m^e \pmod{n} = 4^3 \pmod{33} = 64 \pmod{33} = 31$. **Decrypt**: $m = c^d \pmod{n} = 31^7 \pmod{33} = 4$, so it works. Knowing only $(33, 3)$ an attacker must factor 33 into 3×11 to find ϕ and d , trivial here, but hopeless for a 2048-bit n .

Digital signature: run it backwards. Sign by computing $\text{hash}^d \pmod{n}$ with the *private* key; anyone can check with the *public* key that the result raised to e gives the hash, proving the sender and that the data was unchanged.

Diffie–Hellman: agreeing a secret in public

Two people who have never met need a shared key, but anyone may be listening. Both publicly agree a prime $p = 23$ and a base $g = 5$, then:

Alice picks secret $a = 6$	sends $A = g^a \pmod{p} = 5^6 \pmod{23} = 8$
Bob picks secret $b = 15$	sends $B = g^b \pmod{p} = 5^{15} \pmod{23} = 19$
Alice computes $B^a \pmod{p} = 19^6 \pmod{23} = 2$	
Bob computes $A^b \pmod{p} = 8^{15} \pmod{23} = 2$	<- same shared secret

It works because $(g^a)^b = (g^b)^a = g^{ab}$. An eavesdropper sees p, g, A and B but recovering a or b means solving the **discrete logarithm problem**, infeasible for large p .

How TLS combines both, and the quantum threat

Asymmetric cryptography is slow (roughly a thousand times slower than symmetric). So HTTPS uses a **hybrid** approach: use asymmetric methods (a certificate checked with signatures, plus a Diffie–Hellman exchange) to agree a session key, then encrypt the actual traffic with fast symmetric encryption such as AES. Using a fresh DH exchange every session gives **forward secrecy**: stealing the server's long-term key later does not unlock old recordings.

A large **quantum computer** running Shor's algorithm could factor large numbers and solve discrete logarithms efficiently, breaking RSA and Diffie–Hellman. Symmetric ciphers are much less affected (Grover's algorithm only halves the effective key length, so AES-256 stays strong). **Post-quantum cryptography** based on different hard problems, such as lattices, is already being standardised and deployed.

Challenge questions

1. RSA with $p = 5$ and $q = 11$, $e = 3$. Find n , ϕ and d ($3d \equiv 1 \pmod{\phi}$), then encrypt $m = 8$.
2. Diffie–Hellman with $p = 11$ and $g = 2$. Alice's secret is 3, Bob's is 4. Find the public values and the shared secret.
3. Why does HTTPS use both asymmetric and symmetric encryption rather than just one?

Worked answers

Only look once you have had a proper go at each question.

1. $n = 55$, $\phi = 4 \times 10 = 40$. $d = 27$ because $3 \times 27 = 81 = 2 \times 40 + 1$. Ciphertext $c = 8^3 \bmod 55 = 512 \bmod 55 = \mathbf{17}$ ($55 \times 9 = 495$; $512 - 495 = 17$).
2. Alice sends $A = 2^3 \bmod 11 = 8$. Bob sends $B = 2^4 \bmod 11 = 5$. Alice: $5^3 \bmod 11 = 125 \bmod 11 = 4$. Bob: $8^4 \bmod 11 = 4096 \bmod 11 = 4$. Shared secret = **4**.
3. Asymmetric encryption solves the key distribution problem (parties can agree or protect a key over an open channel and authenticate with certificates) but is slow. Symmetric encryption is fast enough for large volumes of data but needs a shared key. So asymmetric methods set up the key and symmetric encryption protects the data.