DL systems over finite fields III

Key sizes and DSA

Tanja Lange

Eindhoven University of Technology

2MMC10 – Cryptology

Key size recommendations

			Future System Use	
	Parameter	Legacy	Near Term	Long Term
Symmetric Key Size	k	80	128	256
Hash Output Size	m	160	256	512
MAC Output Size	m	80	128	256
RSA Problem	$\log_2(n) \geq$	1024	3072	15360
Finite Field DLP	$\log_2(p^n) \geq$	1024	3072	15360
	$\log_2(p), \log_2(q) \geq$	160	256	512
ECDLP	$\log_2(q) \geq$	160	256	512
Pairing	$\log_2(p^{k \cdot n}) \geq$	1024	6144	15360
	$\log_2(p), \log_2(q) \geq$	160	256	512

- Source: ECRYPT-CSA "Algorithms, Key Size and Protocols Report" (2018). Bigger overview https://www.keylength.com/.
- These recommendations take into account attacks known today.
- Use extrapolations to larger problem sizes.
- Attacker power typically limited to 2^{128} operations (2^{80} for legacy).
- More to come on pairings and long-term security ...

Digital Signature Algorithm (DSA)

- Standardized by NIST and others. Predecessor of ECDSA.
- Designed by NSA, though this was not publicly acknowledged.
- Attempts to work around patent by Schnorr.
- Reduces signature size by working in subgroup G with $|G| = \ell \ll p$. p chosen to protect against index calculus, ℓ against Pollard rho.

KeyGen:

1. Pick random $0 < a < \ell$.
2. Compute $h_A = g^a$.
3. Output public key h_A, private key a.

Sign:

1. Pick random $0 < k < \ell$, compute $r = g^k$. Put $\bar{r} \equiv r \bmod \ell$.
2. Compute $s \equiv k^{-1}(H(m) + a\bar{r}) \bmod \ell$.
3. Send $(\bar{r}, s)$. These are 2 elements $< \ell$.

Verify:

1. Compute $w \equiv s^{-1} \bmod \ell, u_1 \equiv H(m)w \bmod \ell, u_2 \equiv \bar{r}w \bmod \ell$.
2. Compute $r' = g^{u_1} h_A^{u_2}$ and accept if $r' \equiv \bar{r} \bmod \ell$.

Summary: current state of the art

- Currently used crypto (check the lock icon in your browser) starts with elliptic-curve Diffie-Hellman (ECDH), RSA, or Diffie-Hellman (DH) in finite fields.
- Older standards are RSA or elliptic curves from NIST (or Brainpool), e.g. NIST P256 or ECDSA.
- Internet currently moving over to Curve25519 and Ed25519
- For symmetric crypto TLS (the protocol behind https) uses AES or ChaCha20 and some MAC, e.g. AES-GCM or ChaCha20-Poly1305. High-end devices have support for AES-GCM, smaller ones do better with ChaCha20-Poly1305.
- Security is getting better. Some obstacles: bugs; untrustworthy hardware;

Summary: current state of the art

- Currently used crypto (check the lock icon in your browser) starts with elliptic-curve Diffie-Hellman (ECDH), RSA, or Diffie-Hellman (DH) in finite fields.
- Older standards are RSA or elliptic curves from NIST (or Brainpool), e.g. NIST P256 or ECDSA.
- Internet currently moving over to Curve25519 and Ed25519
- For symmetric crypto TLS (the protocol behind https) uses AES or ChaCha20 and some MAC, e.g. AES-GCM or ChaCha20-Poly1305. High-end devices have support for AES-GCM, smaller ones do better with ChaCha20-Poly1305.
- Security is getting better. Some obstacles: bugs; untrustworthy hardware; let alone anti-security measures such as laws restricting encryption in China, Iran, Russia, but also western countries like Australia and UK. Even NL has atempts to weaken encryption.