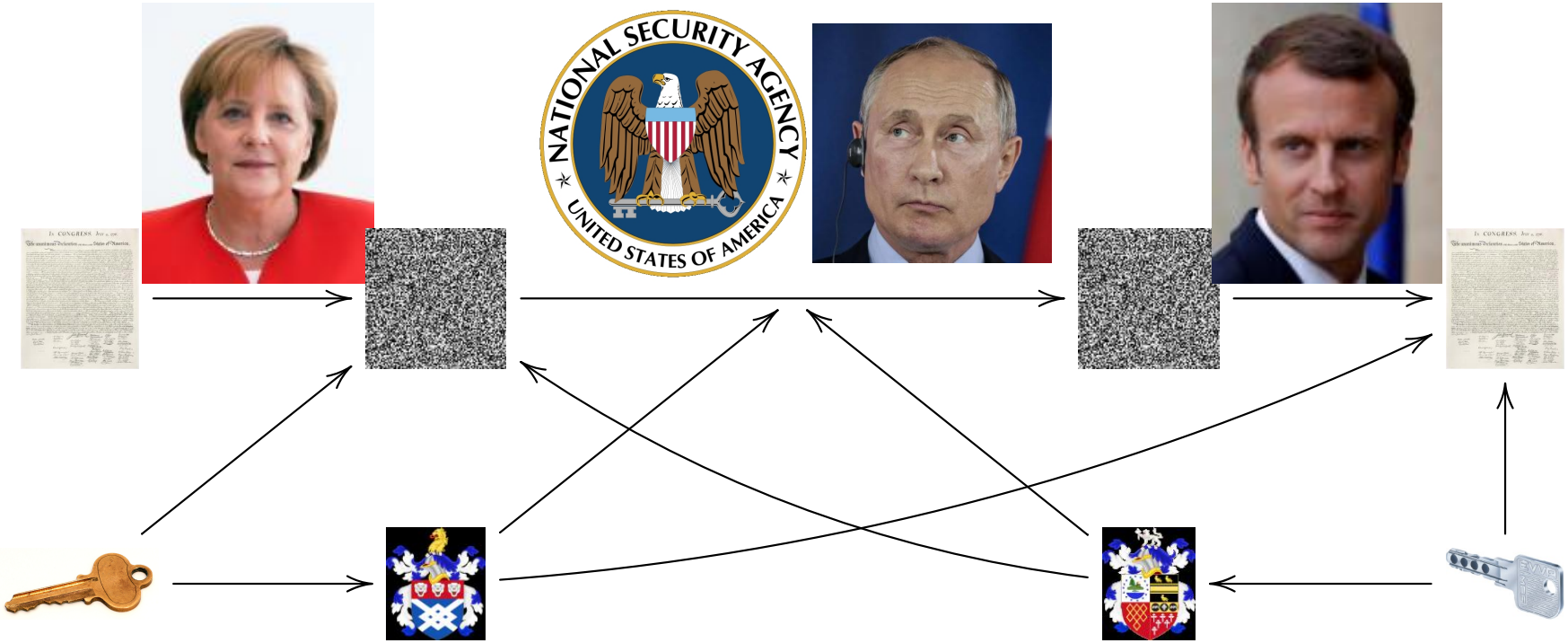
Diffie-Hellman key exchange

Tanja Lange

Eindhoven University of Technology

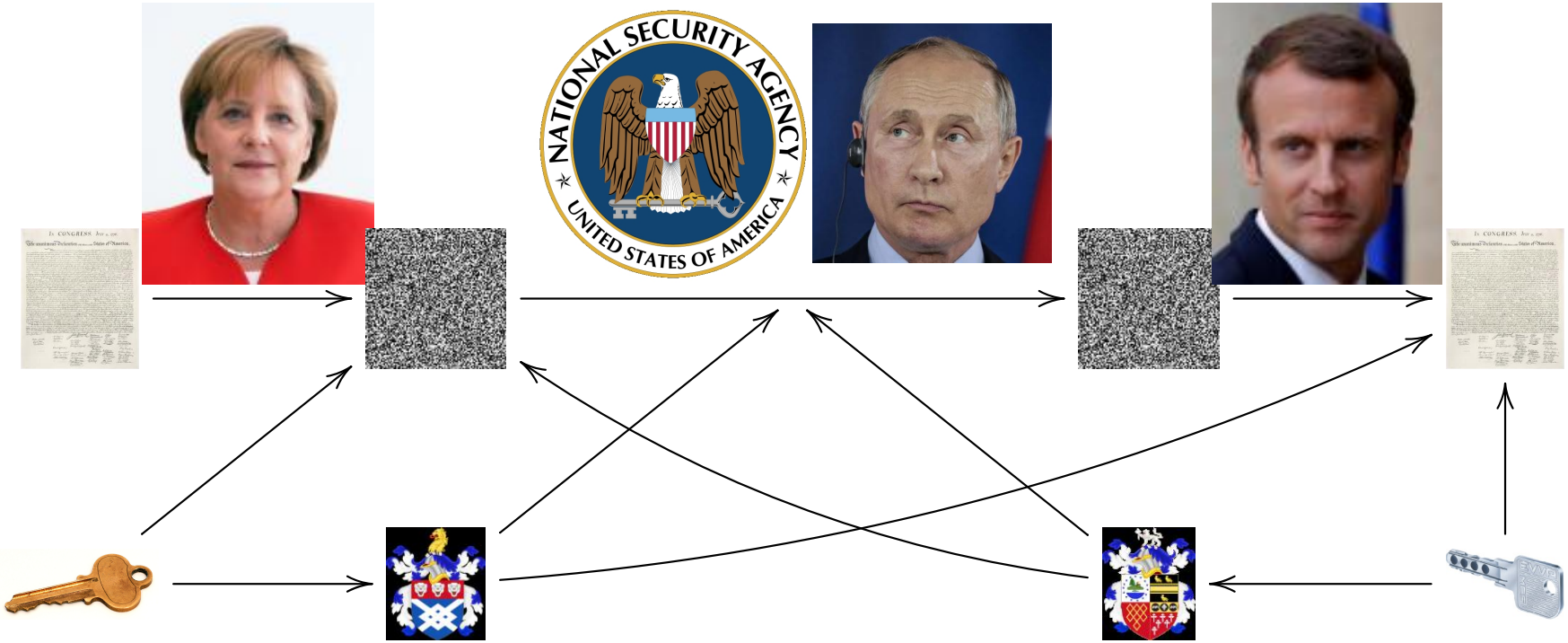
2WF80: Introduction to Cryptology

Public-key authenticated encryption ("DH" data flow)



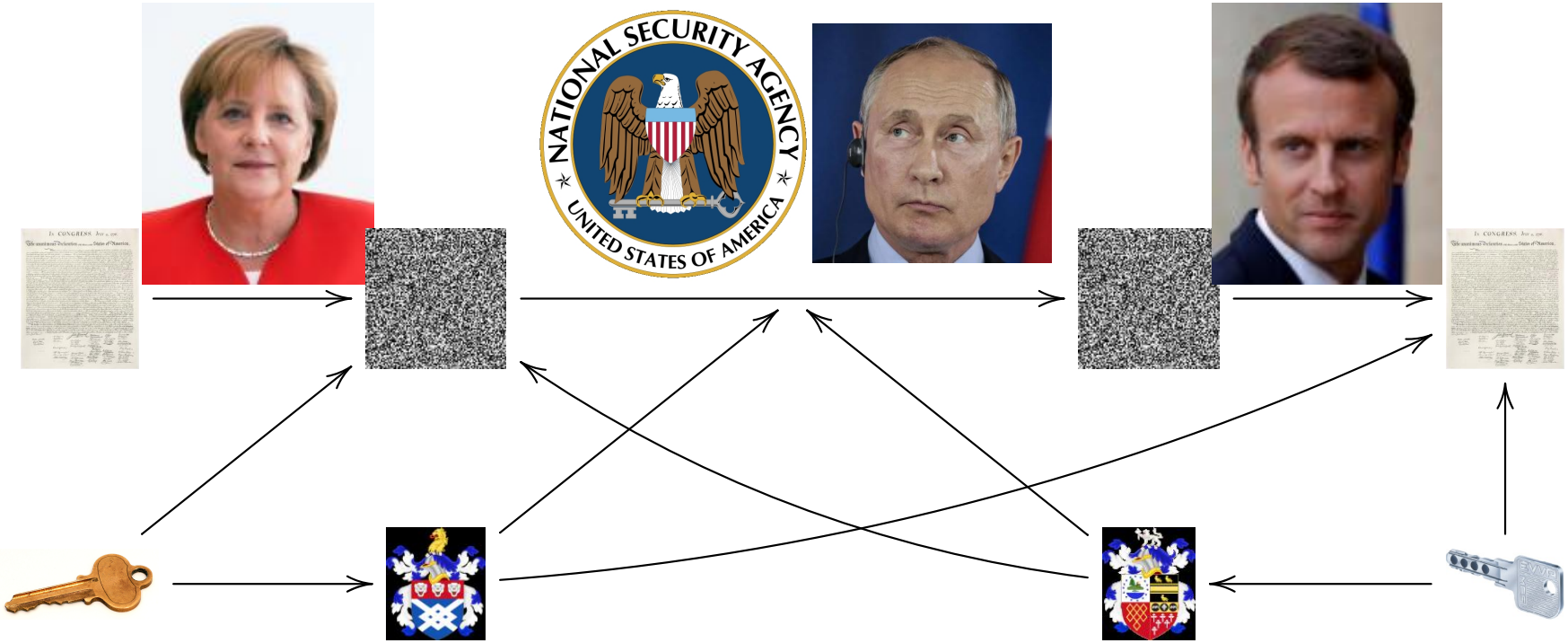
- Prerequisite: Alice has a private key  and public key  .
- Prerequisite: Bob has a private key  and public key  .
-  and  determine the same key as  and 
- Alice and Bob use this shared key to authenticate and encrypt with symmetric cryptography.

Public-key authenticated encryption ("DH" data flow)



- Everybody knows a group G generator g.
- Prerequisite: Alice has a private key $a \in \mathbb{N}$ and public key $h_A = g^a$.
- Prerequisite: Bob has a private key $b \in \mathbb{N}$ and public key $h_B = g^b$.
- $h_B^a = (g^b)^a$
- Alice and Bob use this shared key to authenticate and encrypt with symmetric cryptography.

Public-key authenticated encryption ("DH" data flow)



- Everybody knows a group G generator g.
- Prerequisite: Alice has a private key $a \in \mathbb{N}$ and public key $h_A = g^a$.
- Prerequisite: Bob has a private key $b \in \mathbb{N}$ and public key $h_B = g^b$.
- $h_B^a = (g^b)^a = g^{ab} = (g^a)^b = h_A^b$. Use hash of g^{ab} has key.
- Alice and Bob use this shared key to authenticate and encrypt with symmetric cryptography.

Diffie–Hellman key exchange

- 1976 Diffie and Hellman introduce public-key cryptography.
- To use it, standardize group G and $g \in G$.
  Everybody knows G and g as well as how to compute in G.
- Warning #1: Many G are unsafe!

Diffie–Hellman key exchange

- 1976 Diffie and Hellman introduce public-key cryptography.
- To use it, standardize group G and $g \in G$.
  Everybody knows G and g as well as how to compute in G.
- Warning #1: Many G are unsafe!
  - $G = (\mathbb{Q}, \cdot), g = 2,\ h_A = 65536$

Diffie–Hellman key exchange

- 1976 Diffie and Hellman introduce public-key cryptography.
- To use it, standardize group G and $g \in G$.
  Everybody knows G and g as well as how to compute in G.
- Warning #1: Many G are unsafe!
  - $G = (\mathbb{Q}, \cdot), g = 2$, $h_A = 65536$ means $a = 16$. In general, just check bitlength.
  - $G = (\mathbb{F}_p, +)$, i.e., A sends $h_A \equiv ag \bmod p$.
    See exercises for Thursday
- Diffie and Hellman suggested $G = (\mathbb{F}_p^*, \cdot)$ with g a primitive element, i.e., a generator of the whole group.

Diffie–Hellman key exchange

- 1976 Diffie and Hellman introduce public-key cryptography.
- To use it, standardize group G and $g \in G$.
  Everybody knows G and g as well as how to compute in G.
- Warning #1: Many G are unsafe!
  - $G = (\mathbb{Q}, \cdot), g = 2$, $h_A = 65536$ means $a = 16$. In general, just check bitlength.
  - $G = (\mathbb{F}_p, +)$, i.e., A sends $h_A \equiv ag \bmod p$.
    See exercises for Thursday
- Diffie and Hellman suggested $G = (\mathbb{F}_p^*, \cdot)$ with g a primitive element, i.e., a generator of the whole group.
- Used in practice $G \subset (\mathbb{F}_p^*, \cdot)$ with g an element of large prime order.
- More commonly used in practice G is group of points on an elliptic curve over $\mathbb{F}_p$. Stay on for 2MMC10 for details.

Hardness assumptions

- Computational Diffie-Hellman Problem (CDHP):
  Given g, g^a, g^b compute g^{ab}.
- Decisional Diffie-Hellman Problem (DDHP):
  Given g, g^a, g^b, and g^c decide whether $g^c = g^{ab}$.
- Discrete Logarithm Problem (DLP):
  Given g, g^a, compute a.
- If one can solve DLP, then CDHP and DDHP are easy.

Hardness assumptions

- Computational Diffie-Hellman Problem (CDHP):
  Given g, g^a, g^b compute g^{ab}.
- Decisional Diffie-Hellman Problem (DDHP):
  Given g, g^a, g^b, and g^c decide whether $g^c = g^{ab}$.
- Discrete Logarithm Problem (DLP):
  Given g, g^a, compute a.
- If one can solve DLP, then CDHP and DDHP are easy.
- If one can solve CDH, then DDHP is easy.

Practical problems

- Eve can set up a *man-in-the-middle* attack:

$$A \xleftrightarrow{g^{ae}} E \xleftrightarrow{g^{bf}} B$$

  E decrypts everything from A and reencrypts it to B and vice versa.
- This attack cannot be detected unless A and B have some long-term keys that are known to each other or compare their keys out of band.

Semi-static DH

- A cryptosystem combining public-key and symmetric-key crypto is called a *hybrid system*
- Alice publishes long-term public key $h_A = g^a$,
  keeps long-term private key a.
- Any user can encrypt to Alice using this key:
  - Pick random k and compute $r = g^k$.
  - Encrypt message using symmetric keys derived from $H(h_A^k)$.
  - Send ciphertext c along with r.
  - Alice decrypts, by obtaining symmetric key from $H(r^a) = H(g^{ak})$.

Semi-static DH

- A cryptosystem combining public-key and symmetric-key crypto is called a *hybrid system*
- Alice publishes long-term public key $h_A = g^a$,
  keeps long-term private key a.
- Any user can encrypt to Alice using this key:
  - Pick random k and compute $r = g^k$.
  - Encrypt message using symmetric keys derived from $H(h_A^k)$.
  - Send ciphertext c along with r.
  - Alice decrypts, by obtaining symmetric key from $H(r^a) = H(g^{ak})$.
- Alice's key here is static, Bob's key is ephemeral.
- Note: ephemeral does not mean one-time;
  it means that is not long term.