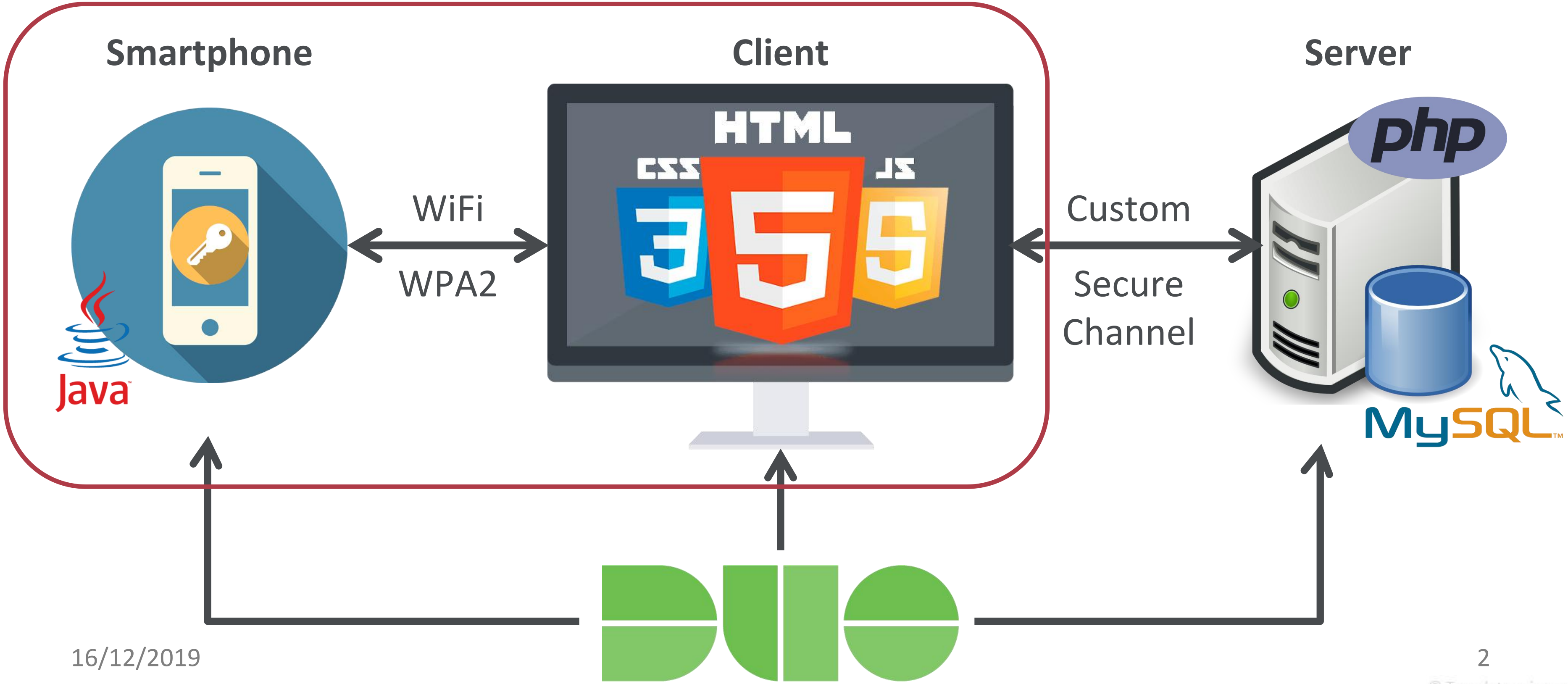
Smartphone as a security token

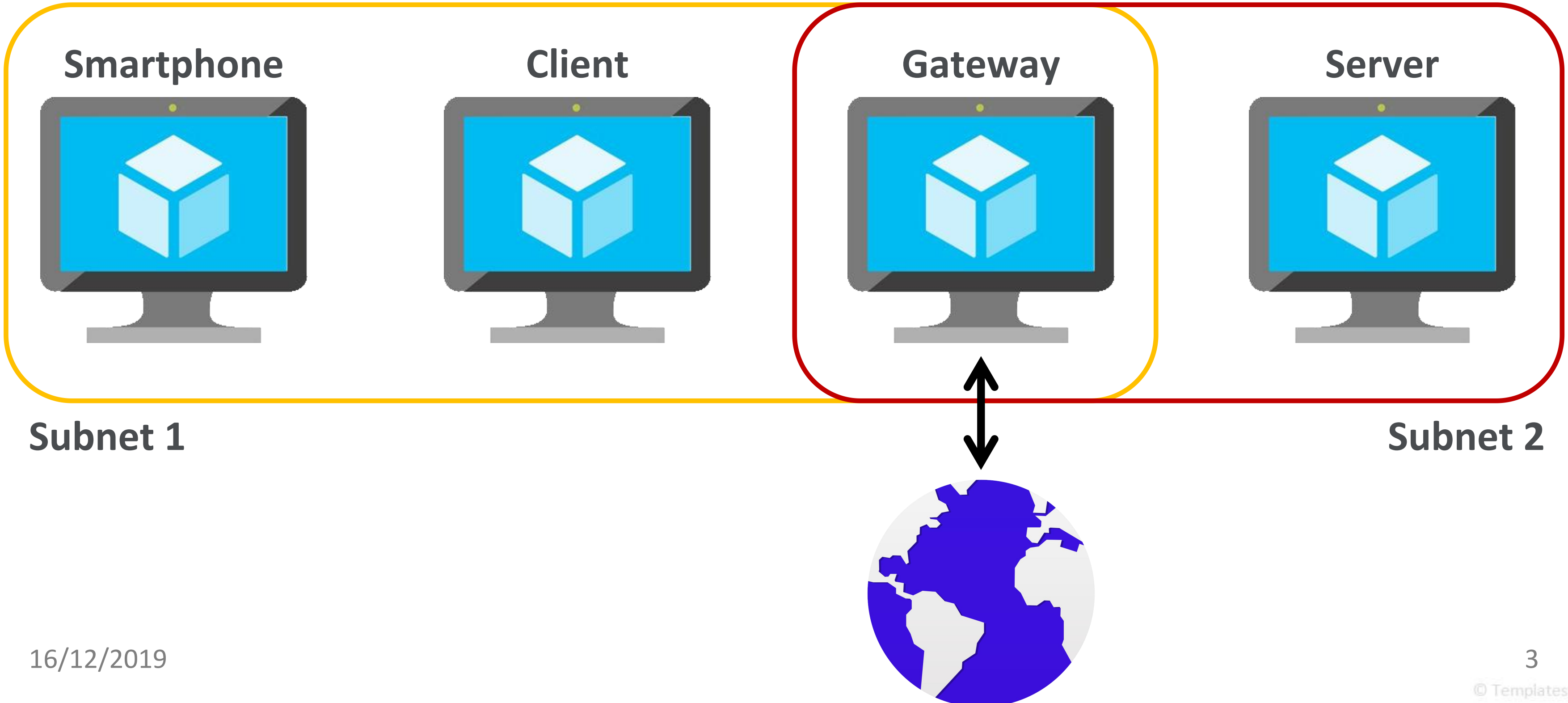
Group 18
David Gonçalves
Leonardo Troeira
Francisco do Carmo

16/12/2019

System architecture



System architecture - VMs



Solution

- Password strength, XSS, SQLI
- 2FA
- Custom Secure Channel
- Proximity

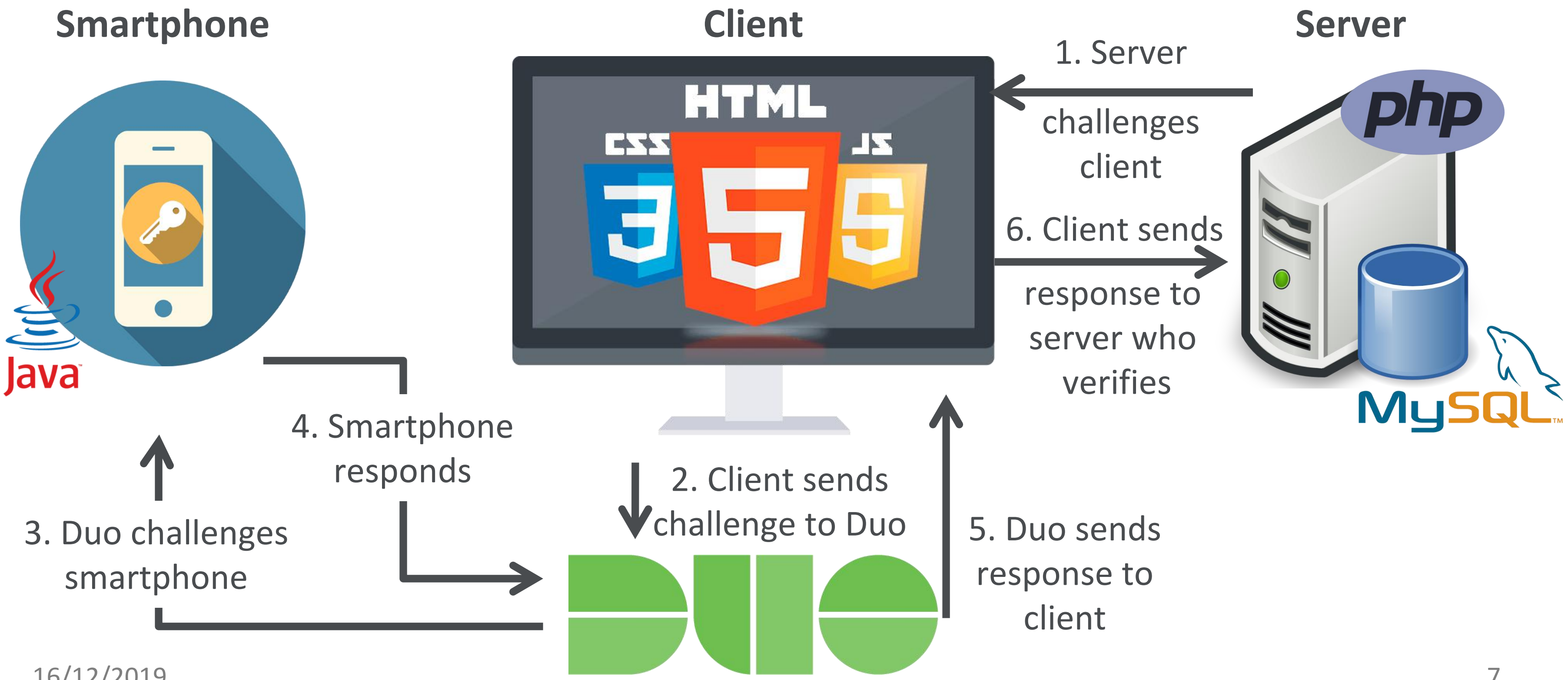
Password strength, XSS, SQLI

- Not the focus of our project, but still:
  - we use use PHP's **password_hash** which uses **bcrypt**.
  - we require a minimum of 10 characters with lowercase, uppercase, digit and symbol.
  - we escape HTML characters directly or indirectly provided by a user when output to the screen.
  - we use SQL prepared statements.

Solution

- Password strength, XSS, SQLI
- 2FA
- Custom Secure Channel
- Proximity

Two-factor authentication

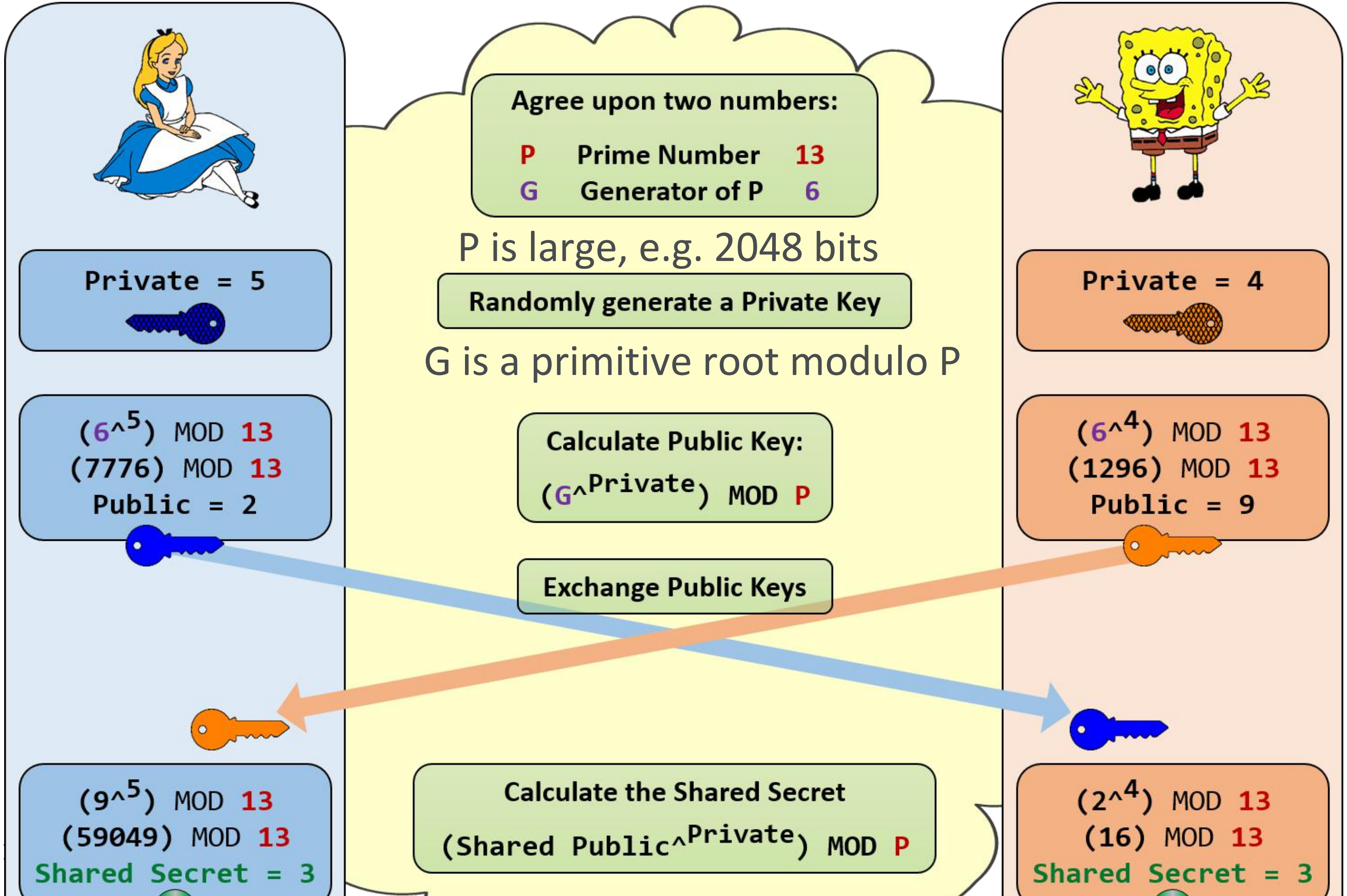


Solution

- Password strength, XSS, SQLI
- 2FA
- Custom Secure Channel
- Proximity

Diffie-Hellman

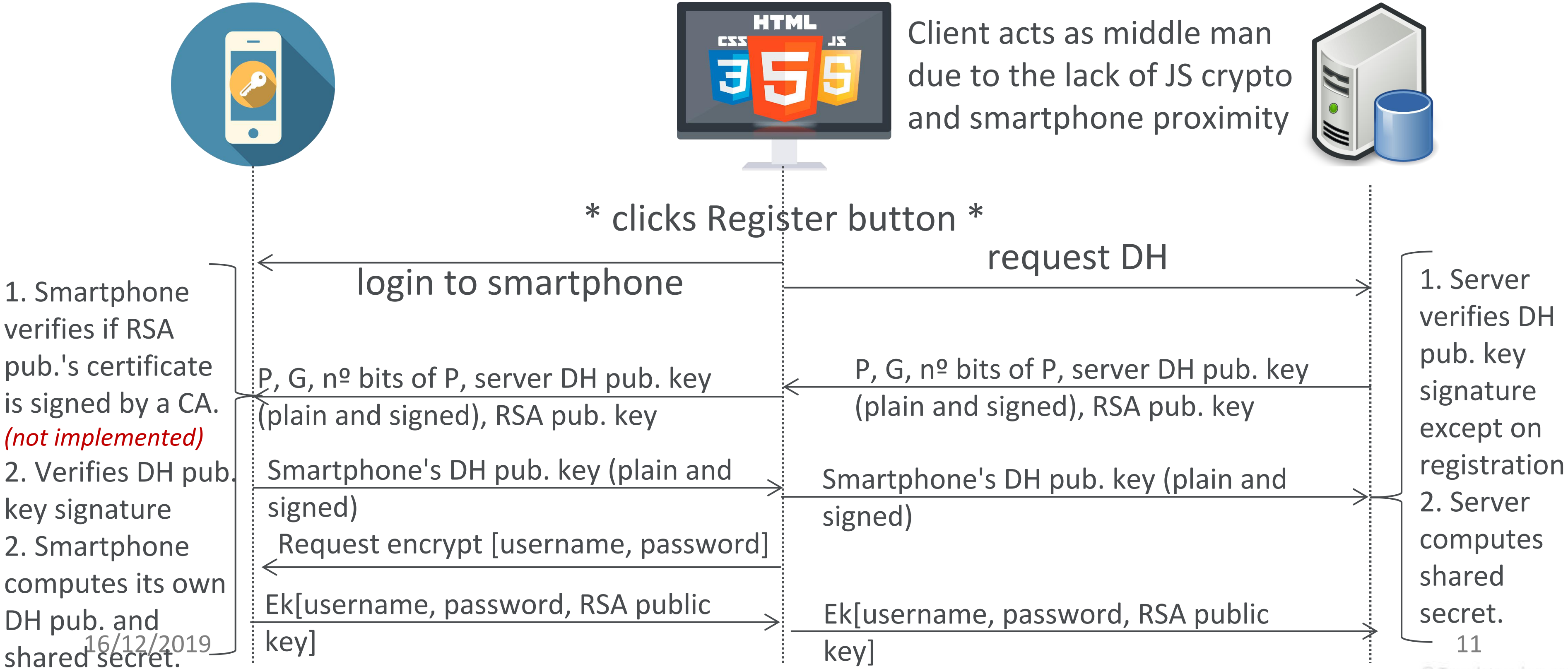
Private is large and random



Ephemeral Diffie-Hellman (DHE)

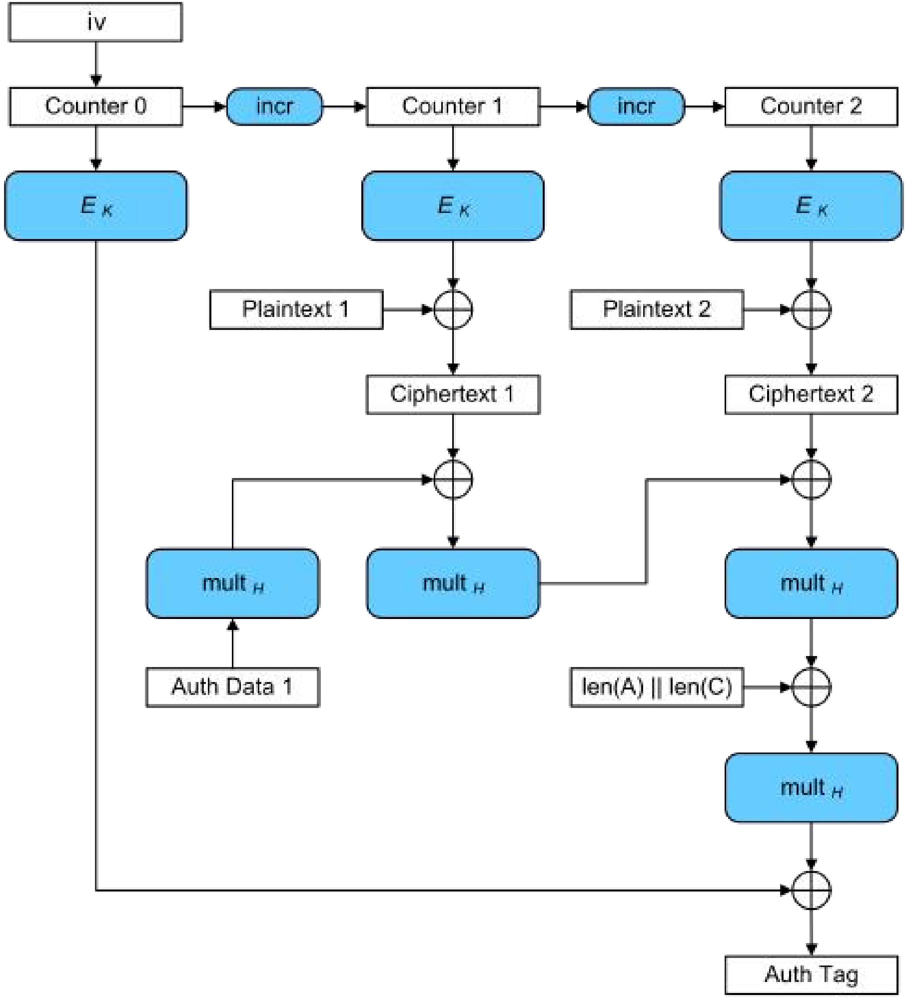
- Grants **perfect forward secrecy** by computing new private and public values for each session
  - If a session key is discovered only that session is compromised.
  - If a long-term private key is compromised, past sessions are not compromised.

Ephemeral Diffie-Hellman with RSA (DHE-RSA)



AES Galois/Counter Mode (AES-GCM)

- DH only finds a shared secret that can be used as a symmetric key, it is not an encryption algorithm.
- GCM is an authentication encryption mode of operation, it is composed by two separate functions: one for encryption (AES-CTR) and one for integrity and authentication (GMAC).


iv
Counter 0
incr
Counter 1
incr
Counter 2
E_K
E_K
E_K
Plaintext 1
Plaintext 2
Ciphertext 1
Ciphertext 2
mult $_H$
mult $_H$
mult $_H$
Auth Data 1
len(A) || len(C)
mult $_H$
Auth Tag

Solution

- Password strength, XSS, SQLI
- 2FA
- Custom Secure Channel
- Proximity

Proximity

- The client can only communicate with the server with the smartphone close by.
  - Smartphone contains RSA private key and all cryptographic methods.
- Client application logs out as soon as connection with the smartphone is lost.
- Encrypted .txt files are decrypted to volatile memory only (garbage collected as soon as smartphone connection is lost). *(not implemented)*
- Decrypted binary files are encrypted again as soon as smartphone connection is lost. *(not implemented)*

Conclusion

We implemented:

- Duo 2FA
  - But did not associate the user's smartphone with Duo at register.
- DHE-RSA-AES256-GCM
  - But did not implement freshness
  - Did not verify the server's RSA public key authenticity
  - Did not encrypt HTTP headers (including the session cookie)
- Proximity
  - But did not encrypt / decryt files