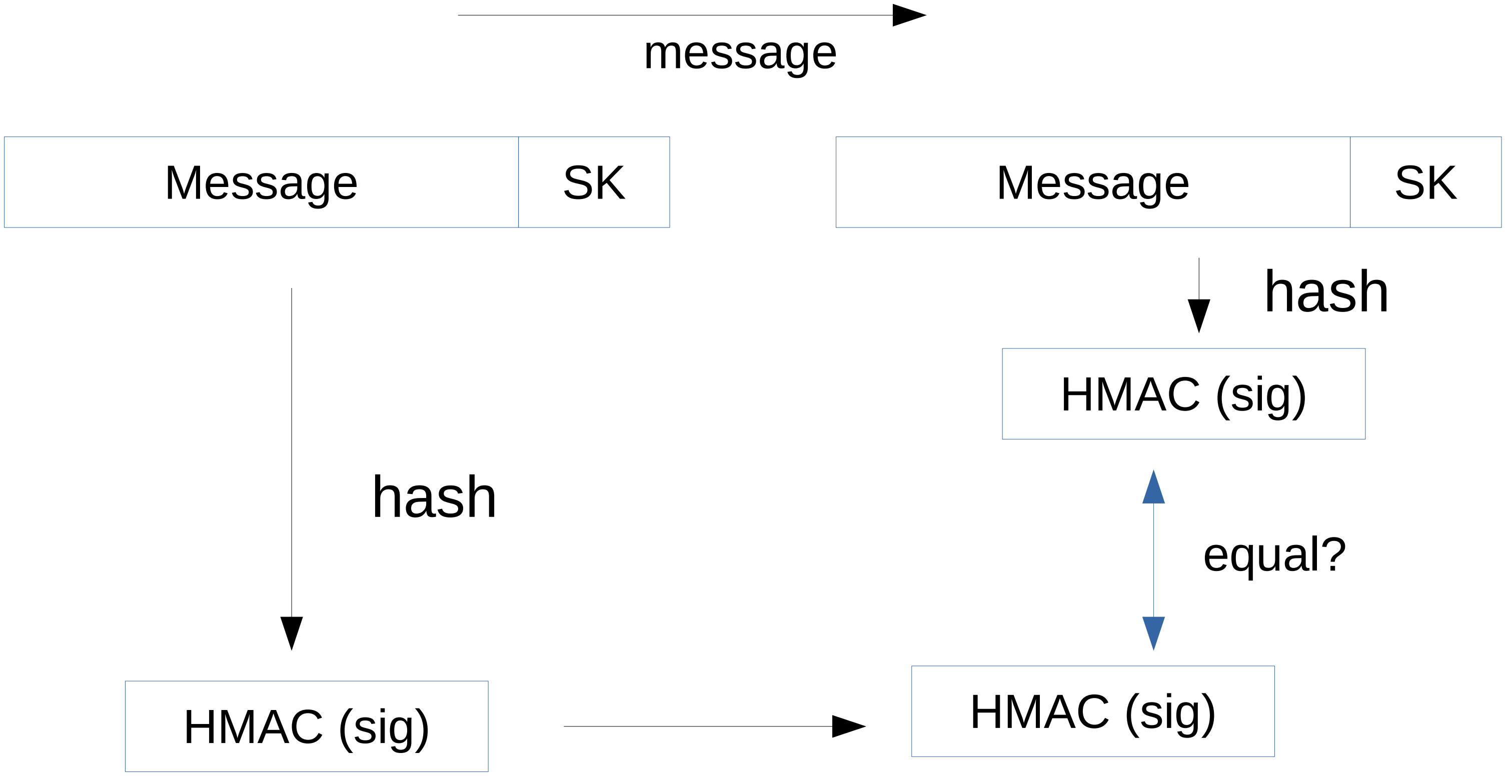
Hashed Message Authentication Codes (HMAC)

Message Authentication & Integrity (HMAC)

- Hash Message Authentication Codes (HMAC)
- A technique for verifying the integrity and authenticity of a message.
- Used with a shared secret key.
- Take a hash of the message and secret key.
- Receiver does the same, and checks that the hashes are the same.

HMAC



message

Message SK

Message SK

hash

HMAC (sig)

hash

equal?

HMAC (sig)

HMAC (sig)

authentication, integrity

Message Integrity (HMAC)

- A cryptographic hash function is applied to a combination of the message and the secret key (details later).
- If the hashes are the same we know
  - The message has not been changed
  - It originated from the peer with which we share the secret key.
- [HMACs are similar to digital signatures (later).]
- They require a shared secret key.

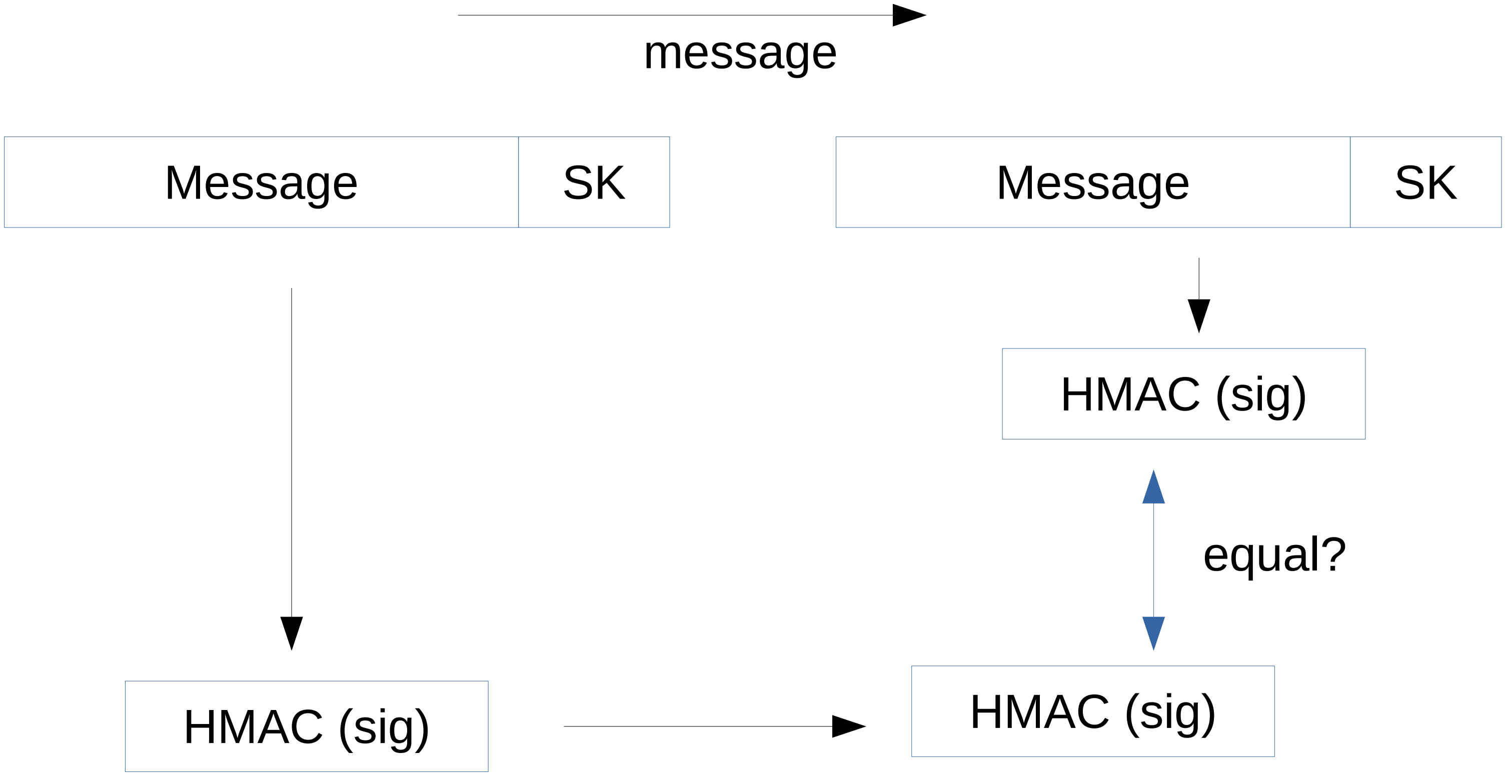
Use Case for HMAC

- Register an app with an public API such as Twitter.
- Obtain an ID and SecretKey.
- When making a call, send
  - ID (identification)
  - HMAC calculated using the SecretKey (authentication)

HMAC

- Take the hash of a combination of the message and the secret key.
- Receiver does the same, and checks that the hashes match.
- Authenticates the sender (only they have the secret key)
- Verifies the integrity of the message.

HMAC



message

Message | SK

Message | SK

HMAC (sig)

equal?

HMAC (sig)

HMAC (sig)

authentication, integrity

HMAC Algorithm

- There are a number of problems with just concatenating the message and the secret key.
- That is **hash (message || key)**
- [|| - concatenation]
- These include
  - length extension attacks, where it is possible to calculate hash **(message || key || additional data)** without knowing the key
  - where does the message end and the key begin?

Aside - Block Size for Cryptographic Hash Function

- **Each hash function has a block size.**
- For example, for SHA256, it is 512 bytes.
- (Note that this is not the same size as the hash output size which is 256 obviously.)
- A number of state variables with predefined values are combined with the first block of the message to get a new set of state variable values.
- These state variable values are combined with the second block and this is repeated for each block.
- Eventually the output is obtained by combining the last values of all the state variables.

HMAC Algorithm

- 1. Prepare the key:
  - The prepared key is a block-sized key derived from the secret key, K (block size of the hash function)
  - Either by padding to the right with 0s up to the block size,
  - Hashing down to less than or equal to the block size first and then padding to the right with zeros.

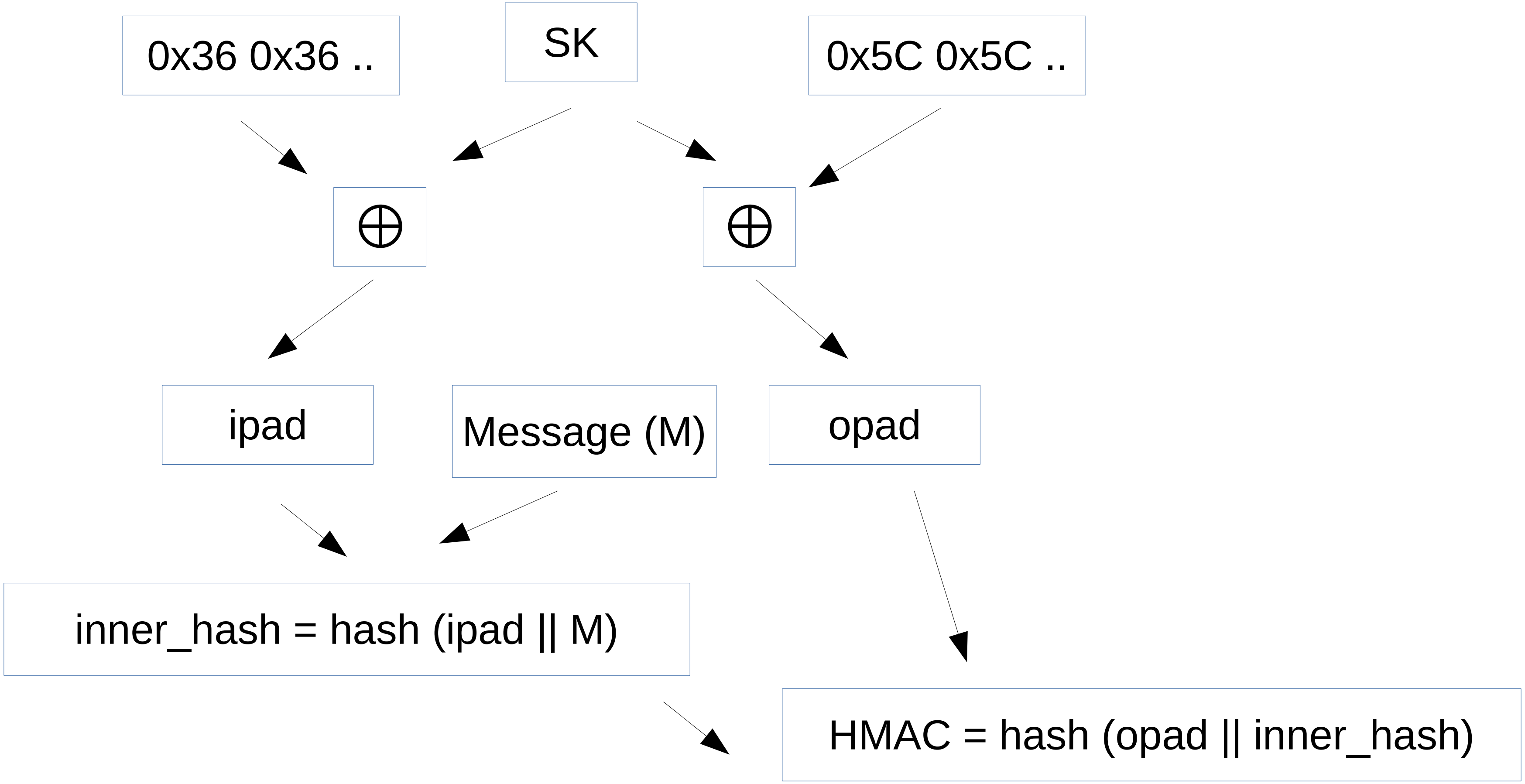
HMAC Algorithm

- Compute the inner and outer padded keys:
-
- Compute the inner padded key (ipad) by XORing the key with a byte value of 0x36 repeated to match the block size.
  - $\text{ipad} = \text{rep}(0x36) \oplus \text{key}$
- Compute the outer padded key (opad) by XORing the key with a byte value of 0x5C repeated to match the block size.
  - $\text{opad} = \text{rep}(0x5c) \oplus \text{key}$

HMAC Algorithm (cont)

- Compute the HMAC value as follows:
-
    - inner_hash = hash(ipad || message)
    - HMAC = hash (opad || inner_hash)

HMAC Algorithm


0x36 0x36 ..
SK
0x5C 0x5C ..
⊕
⊕
ipad
Message (M)
opad
inner_hash = hash (ipad || M)
HMAC = hash (opad || inner_hash)

HMAC / Java

HMAC Example

```
KeyGenerator kg = KeyGenerator.getInstance("HmacSHA256");
SecretKey sk = kg.generateKey();
String message = "Hi There" ;

Mac mac = Mac.getInstance("HmacSHA256");
mac.init(sk);
byte[] result = mac.doFinal(message.getBytes());
System.out.println(result.length);

/// Receiver
Mac mac2 = Mac.getInstance("HmacSHA256");
mac2.init(sk);
byte[] result2 = mac.doFinal(message.getBytes());

System.out.println("Check: " +
          Arrays.equals(result, result2));
```

Base64 Encoded HMAC

```
byte[] hmac = mac.doFinal(textArray);
String encodedHmac =
      Base64.getEncoder().encodeToString(hmac);
System.out.println("Encoded HMAC :" + encodedHmac);


// Base64 decode a HMAC
byte[] decodedHmac =
      Base64.getDecoder().decode(encodedHmac);
```

Summary

- What is a Hashed Message Authentication Code?
  - Provides authentication and integrity of a message without sending the shared secret (password)

Summary (Java)

- Calculate MD5 and SHA hashes (binary values).
- Get the Base64 encoded version of the hash value (text value)
- Calculate the HMAC value for a message.
- Print out the Base64 encoded version of the HMAC.