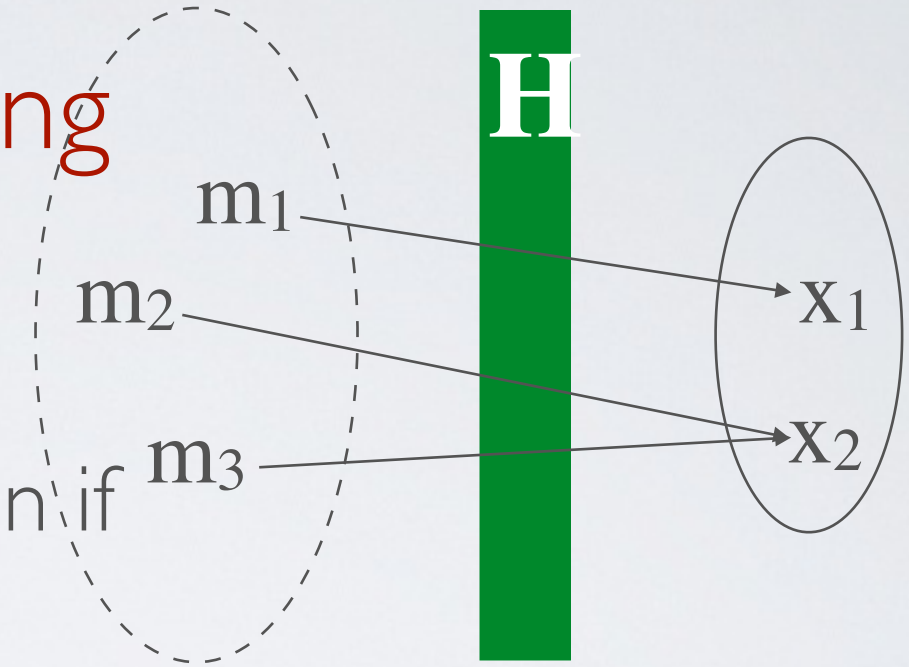
Applied Message Digests

Protocols, Attacks, Implementation Flaws

Kc Udonsi

Cryptographic hashing

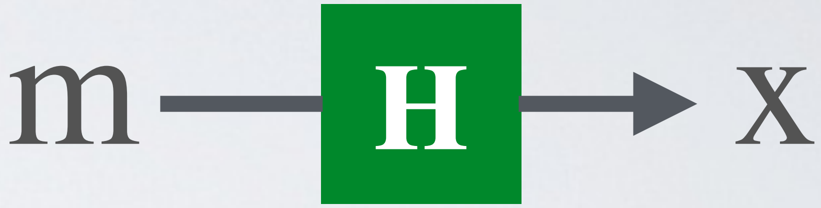


$H(m) = x$ is a hash function if

- H is one-way function
- m is a message of any length
- x is a message digest of a fixed length

➡ H is a lossy compression function necessarily there exists x, m_1 and m_2 | $H(m_1) = H(m_2) = x$

Preimage resistance and collision resistance



PR - Preimage Resistance (a.k.a One Way)

➡ given H and x, hard to find m
e.g. password storage

2PR - Second Preimage Resistance (a.k.a Weak Collision Resistance)

➡ given H, m and x, hard to find m' such that $H(m) = H(m') = x$
e.g. virus identification

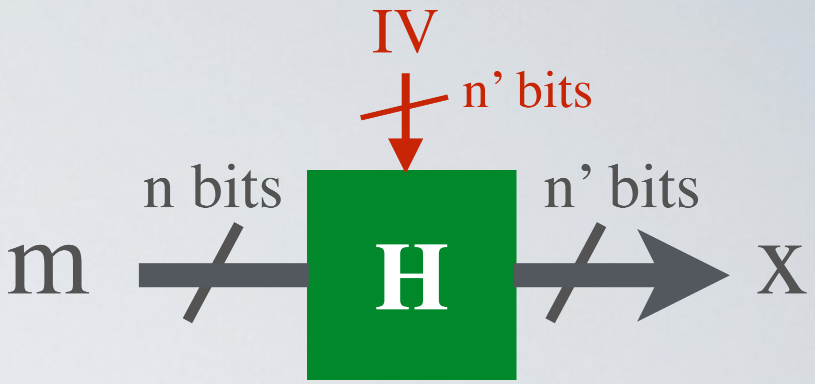
CR - Collision Resistance (a.k.a Strong Collision Resistance)

➡ given H, hard to find m and m' such that $H(m) = H(m') = x$
e.g. digital signatures

CR → 2PR and CR → PR

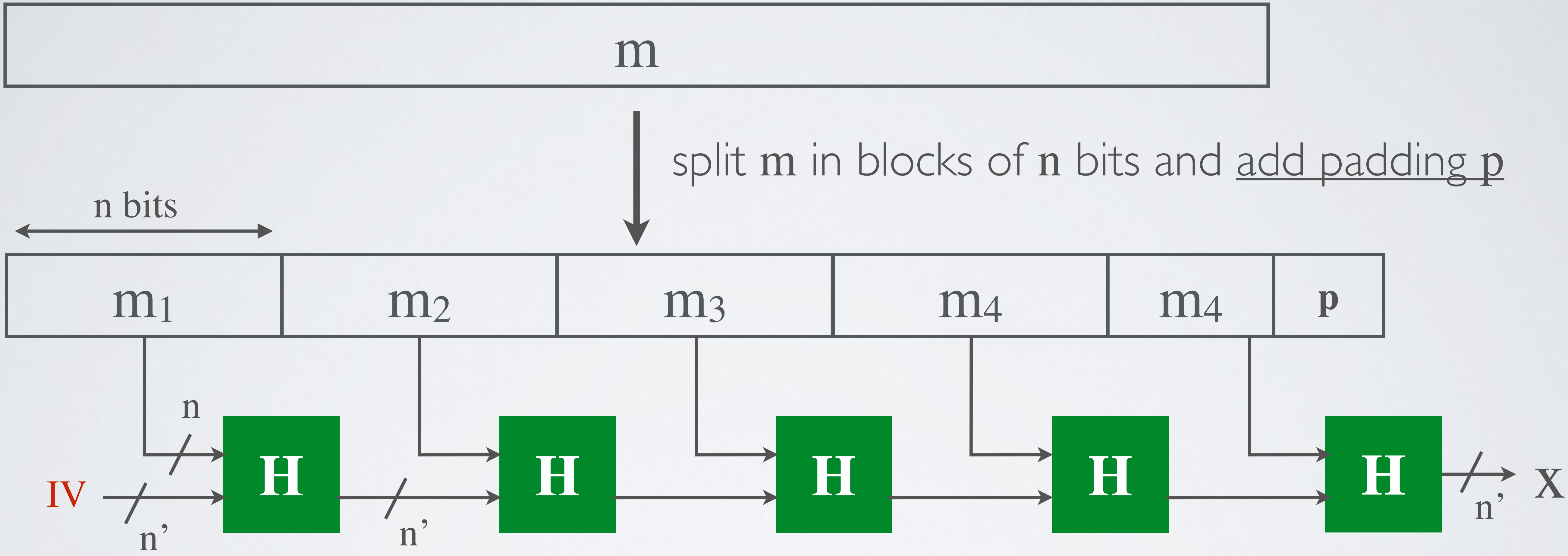
Algorithms

Common hash functions



Name	MD5	SHA-1	SHA-2				SHA-3			
Variant			SHA-224	SHA-256	SHA-384	SHA-512	SHA3-224	SHA3-256	SHA3-384	SHA3-512
Year	1992	1993	2001				2012			
Designer	Rivest	NSA	NSA				Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche			
Input n bits	512	512	512	512	1024	1024	1152	1088	832	576
Output n' bits	128	160	224	256	384	512	224	256	384	512
Speed cycle/byte	6.8	11.4	15.8		17.7		12.5			
Considered Broken	**yes**	**yes**	**no**				**no**			

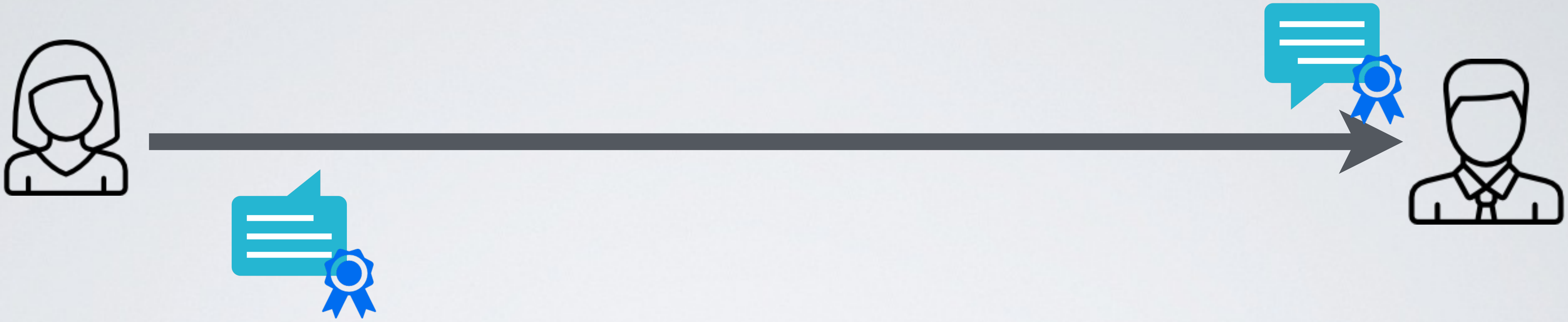
How to hash long messages ?
Merkle–Damgård construction



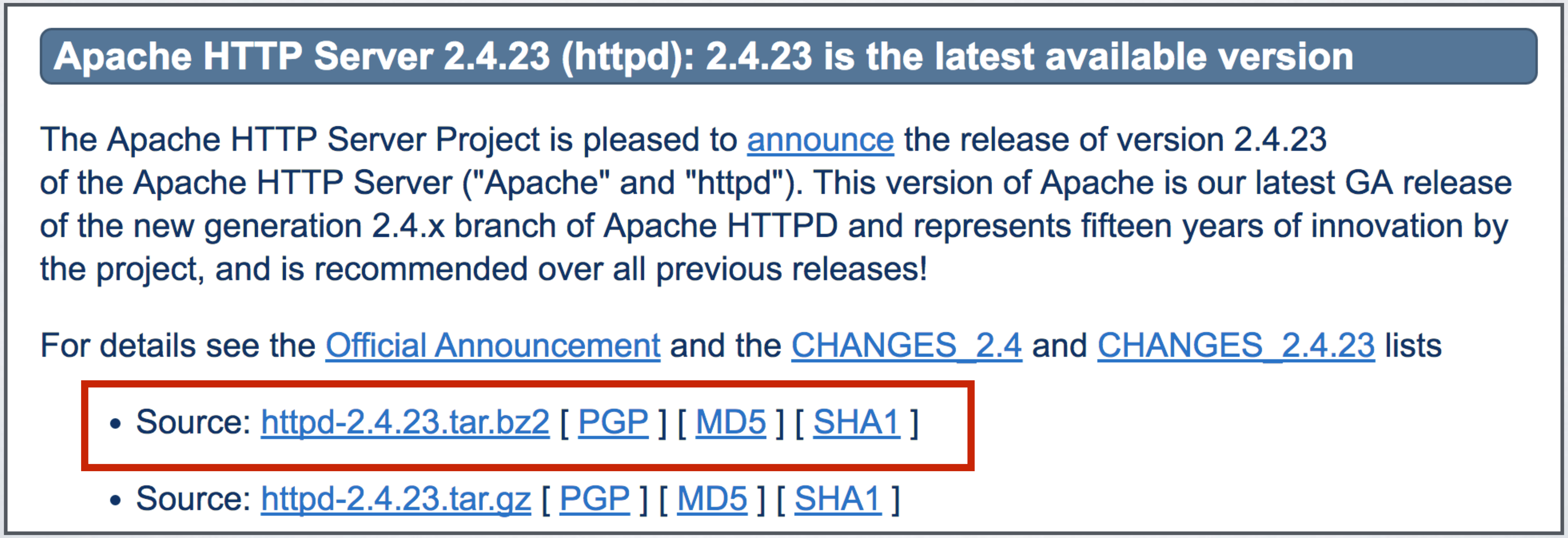
Property : if H is CR then Merkel-Damgard is CR

Using hash functions for Integrity

Hashing



m || H(m)



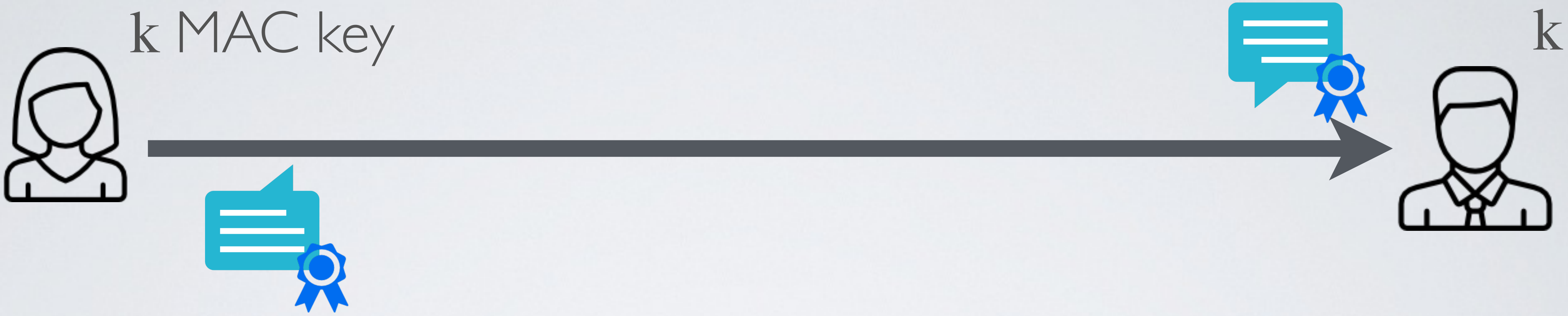
Apache HTTP Server 2.4.23 (httpd): 2.4.23 is the latest available version

The Apache HTTP Server Project is pleased to announce the release of version 2.4.23 of the Apache HTTP Server ("Apache" and "httpd"). This version of Apache is our latest GA release of the new generation 2.4.x branch of Apache HTTPD and represents fifteen years of innovation by the project, and is recommended over all previous releases!

For details see the Official Announcement and the CHANGES_2.4 and CHANGES_2.4.23 lists

- Source: httpd-2.4.23.tar.bz2 [PGP] [MD5] [SHA1]
- Source: httpd-2.4.23.tar.gz [PGP] [MD5] [SHA1]

MAC - Message Authentication Code



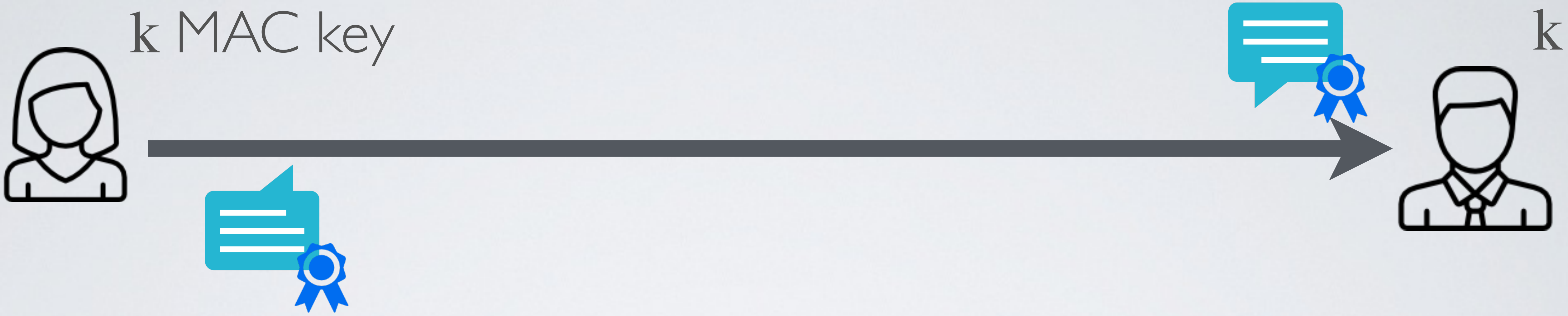
$$m \,\|\, MAC_k(m)$$

Alice an Bob share a key **k**

➡ HMAC - use a hash function on the message and the key

$$MAC_k(m) = H(\xcancel{k \,\|}\, m)$$

Good HMAC



k MAC key

k

$$m \mathbin{\|} MAC_k(m)$$

Alice an Bob share a key **k**

- ➡ Option 1 : envelope method

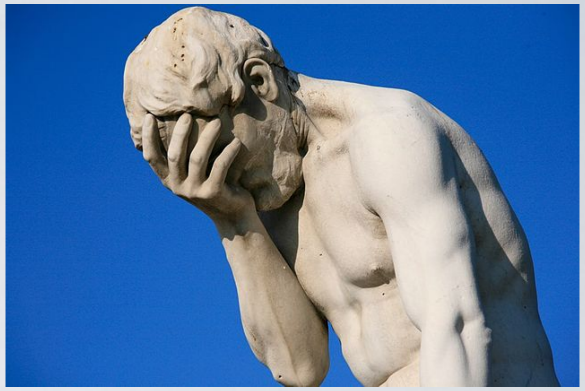
$$MAC_k(m) = H(k \mathbin{\|} m \mathbin{\|} k)$$

- ➡ Option 2 : padding method (i.e. HMAC standard)

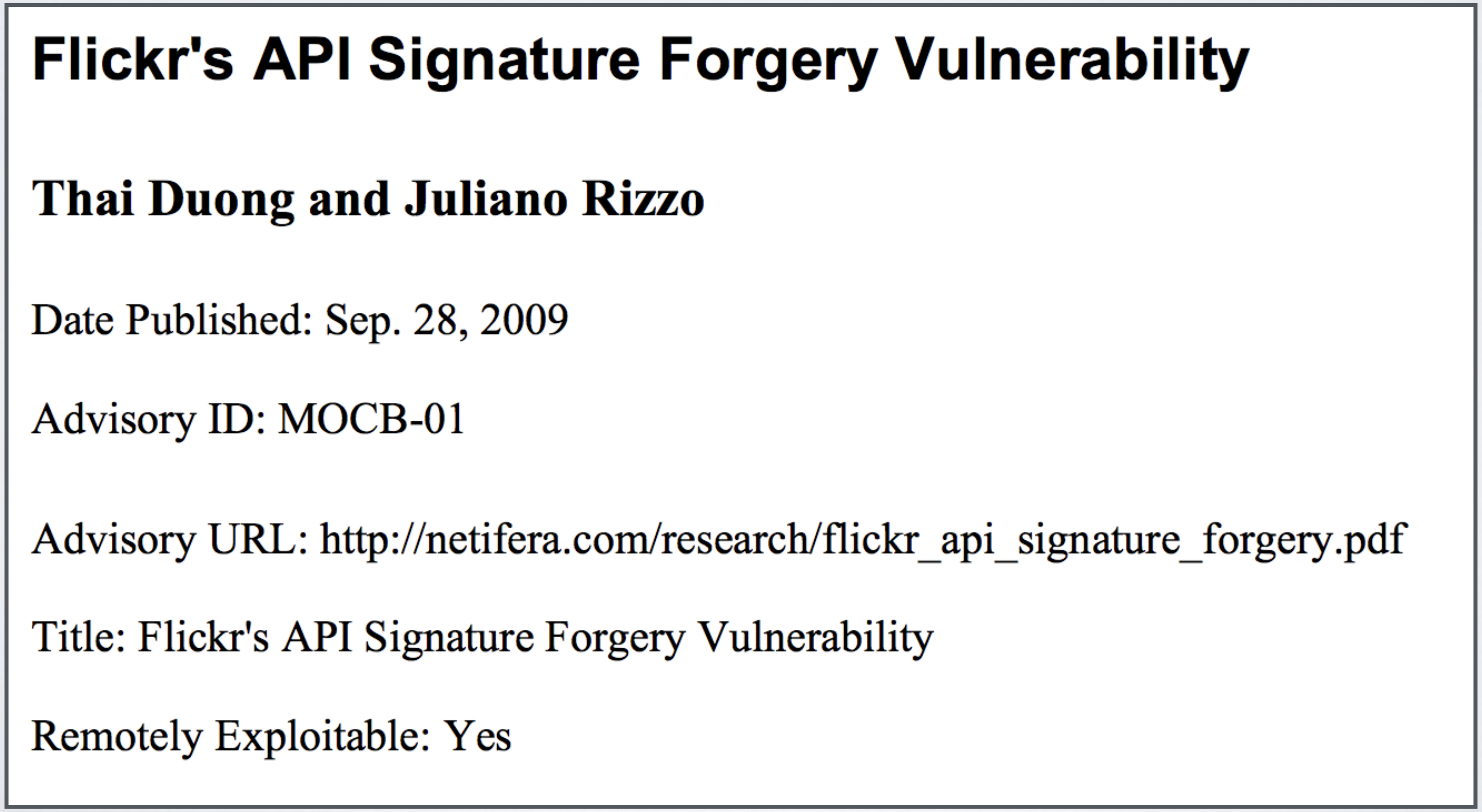
$$HMAC_k(m) \;= H((k \oplus opad) \| \; H((k \oplus ipad) \mathbin{\|} m))$$

Attacks

Length extension attack



Vulnerable : MD5, SHA-1 and SHA-2 (but not SHA-3)



Flickr's API Signature Forgery Vulnerability

Thai Duong and Juliano Rizzo

Date Published: Sep. 28, 2009

Advisory ID: MOCB-01

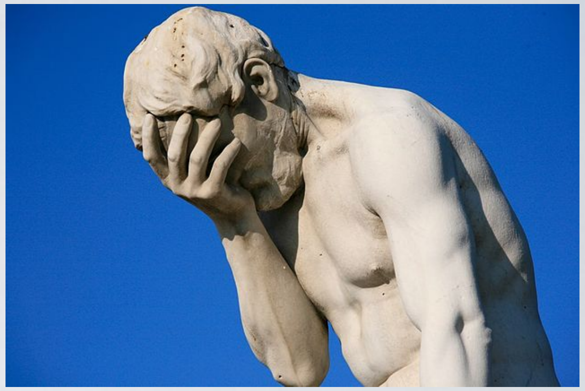
Advisory URL: http://netifera.com/research/flickr_api_signature_forgery.pdf

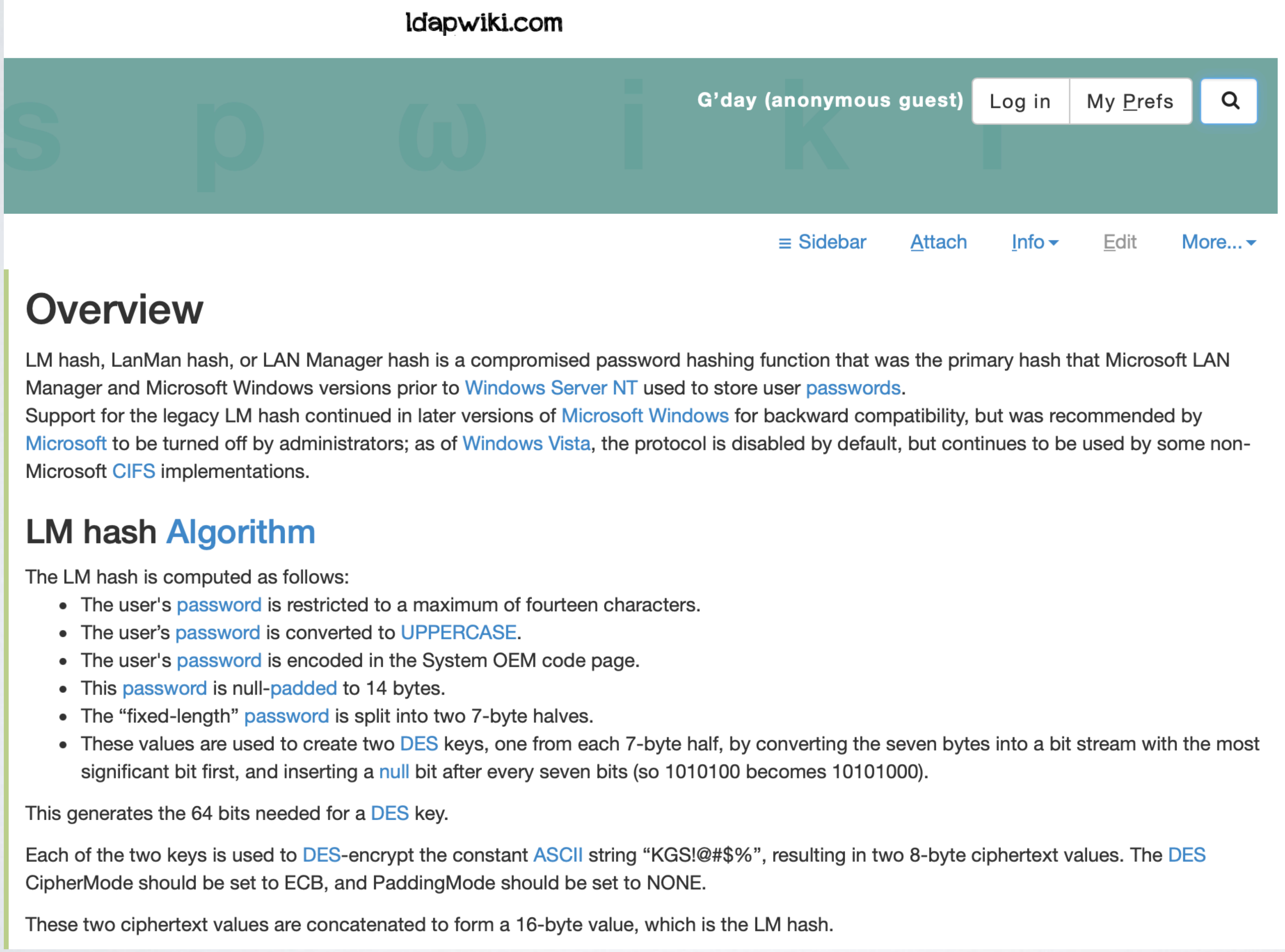
Title: Flickr's API Signature Forgery Vulnerability

Remotely Exploitable: Yes

Implementation Flaws

LM Hash





Overview

LM hash, LanMan hash, or LAN Manager hash is a compromised password hashing function that was the primary hash that Microsoft LAN Manager and Microsoft Windows versions prior to Windows Server NT used to store user passwords.

Support for the legacy LM hash continued in later versions of Microsoft Windows for backward compatibility, but was recommended by Microsoft to be turned off by administrators; as of Windows Vista, the protocol is disabled by default, but continues to be used by some non-Microsoft CIFS implementations.

LM hash Algorithm

The LM hash is computed as follows:

- The user's password is restricted to a maximum of fourteen characters.
- The user's password is converted to UPPERCASE.
- The user's password is encoded in the System OEM code page.
- This password is null-padded to 14 bytes.
- The "fixed-length" password is split into two 7-byte halves.
- These values are used to create two DES keys, one from each 7-byte half, by converting the seven bytes into a bit stream with the most significant bit first, and inserting a null bit after every seven bits (so 1010100 becomes 10101000).

This generates the 64 bits needed for a DES key.

Each of the two keys is used to DES-encrypt the constant ASCII string "KGS!@#$%", resulting in two 8-byte ciphertext values. The DES CipherMode should be set to ECB, and PaddingMode should be set to NONE.

These two ciphertext values are concatenated to form a 16-byte value, which is the LM hash.