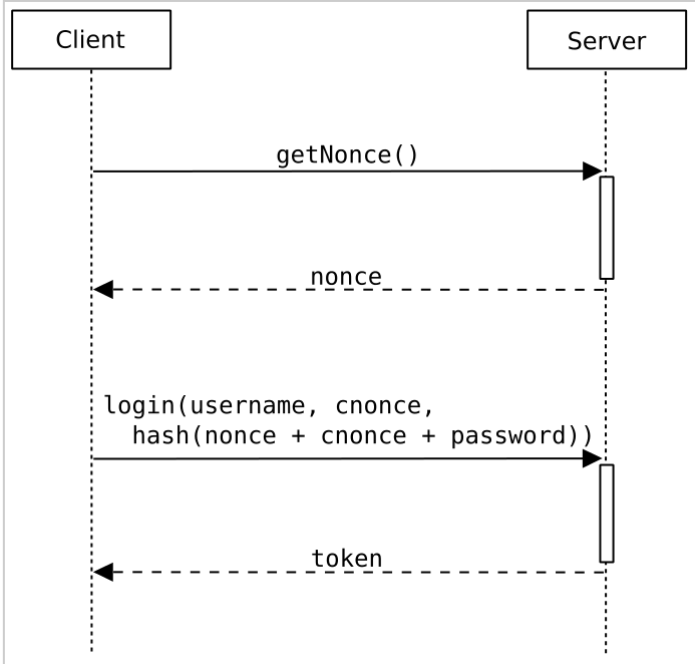
Cryptographic nonce

From Wikipedia, the free encyclopedia

In cryptography, a **nonce** is an arbitrary number that may only be used once. It is similar in spirit to a nonce word, hence the name. It is often a random or pseudo-random number issued in an authentication protocol to ensure that old communications cannot be reused in replay attacks. They can also be useful as initialization vectors and in cryptographic hash function.



Typical client-server communication during a nonce-based authentication process including both a server nonce and a client nonce.

Contents

Definition

A nonce is an arbitrary number used only once in a cryptographic communication, in the spirit of a nonce word. They are often random or pseudo-random numbers. Many nonces also include a timestamp to ensure exact timeliness, though this requires clock synchronization between organizations. The addition of a client nonce ("**cnonce**") helps to improve the security in some ways as implemented in digest access authentication. To ensure that a nonce is used only once, it should be time-variant (including a suitably fine-grained timestamp in its value), or generated with enough random bits to ensure a probabilistically insignificant chance of repeating a previously generated value. Some authors define pseudo-randomness (or unpredictability) as a requirement for a nonce.[1]

Usage

Authentication

Authentication protocols may use nonces to ensure that old communications cannot be reused in replay attacks. For instance, nonces are used in HTTP digest access authentication to calculate an MD5 digest of the password. The nonces are different each time the 401 authentication challenge response code is presented, thus making replay attacks virtually impossible. The scenario of ordering products over the Internet can provide an example of the usefulness of nonces in replay attacks. An attacker could take the encrypted information and—without needing to decrypt—could continue to send a particular order to the supplier, thereby ordering products over and over again under the same name and purchase information. The nonce is used to give 'originality' to a given message so that if the company receives any other orders from the same person with the same nonce, it will discard those as invalid orders.

A nonce may be used to ensure security for a stream cipher. Where the same key is used for more than one message and then a different nonce is used to ensure that the keystream is different for different messages encrypted with that key; often the message number is used.

Secret nonce values are used by the Lamport signature scheme as a signer-side secret which can be selectively revealed for comparison to public hashes for signature creation and verification.

Initialization vectors

Initialization vectors may be referred to as nonces, as they are typically random or pseudo-random.

Hashing

Nonces are used in proof-of-work systems to vary the input to a cryptographic hash function so as to obtain a hash for a certain input that fulfills certain arbitrary conditions. In doing so, it becomes far more difficult to create a "desirable" hash than to verify it, shifting the burden of work onto one side of a transaction or system. For example, proof of work, using hash functions, was considered as a means to combat email spam by forcing email senders to find a hash value for the email (which included a timestamp to prevent pre-computation of useful hashes for later use) that had an arbitrary number of leading zeroes, by hashing the same input with a large number of nonce values until a "desirable" hash was obtained.

Similarly, the bitcoin block-chain hashing algorithm can be tuned to an arbitrary difficulty by changing the required minimum/maximum value of the hash so that the number of bitcoins awarded for new blocks does not increase linearly with increased network computation power as new users join. This is likewise achieved by forcing bitcoin miners to add nonce values to the value being hashed to change the hash algorithm output. Because cryptographic hash algorithms cannot easily be predicted based on their inputs, this makes the act of blockchain hashing and the possibility of being awarded bitcoins something of a lottery, where the first "miner" to find a nonce that delivers a desirable hash is awarded valuable bitcoins.

See also

- Salt (cryptography)
- Key stretching

References

1. Nonce-Based Symmetric Encryption (http://www.cs.ucdavis.edu/~rogaway/papers/nonce.pdf)

External links

- Sam Ruby Blogging on Nonce with an implementation (http://www.intertwingly.net/blog/1585.html)
- RFC 2617 (https://tools.ietf.org/html/rfc2617) - HTTP Authentication: Basic and Digest Access Authentication
- RFC 3540 (https://tools.ietf.org/html/rfc3540) - Robust Explicit Congestion Notification (ECN) Signaling with Nonces
- RFC 4418 (https://tools.ietf.org/html/rfc4418) - UMAC: Message Authentication Code using Universal Hashing
- Web Services Security (http://docs.oasis-open.org/wss/2004/01/)

Categories: Cryptography