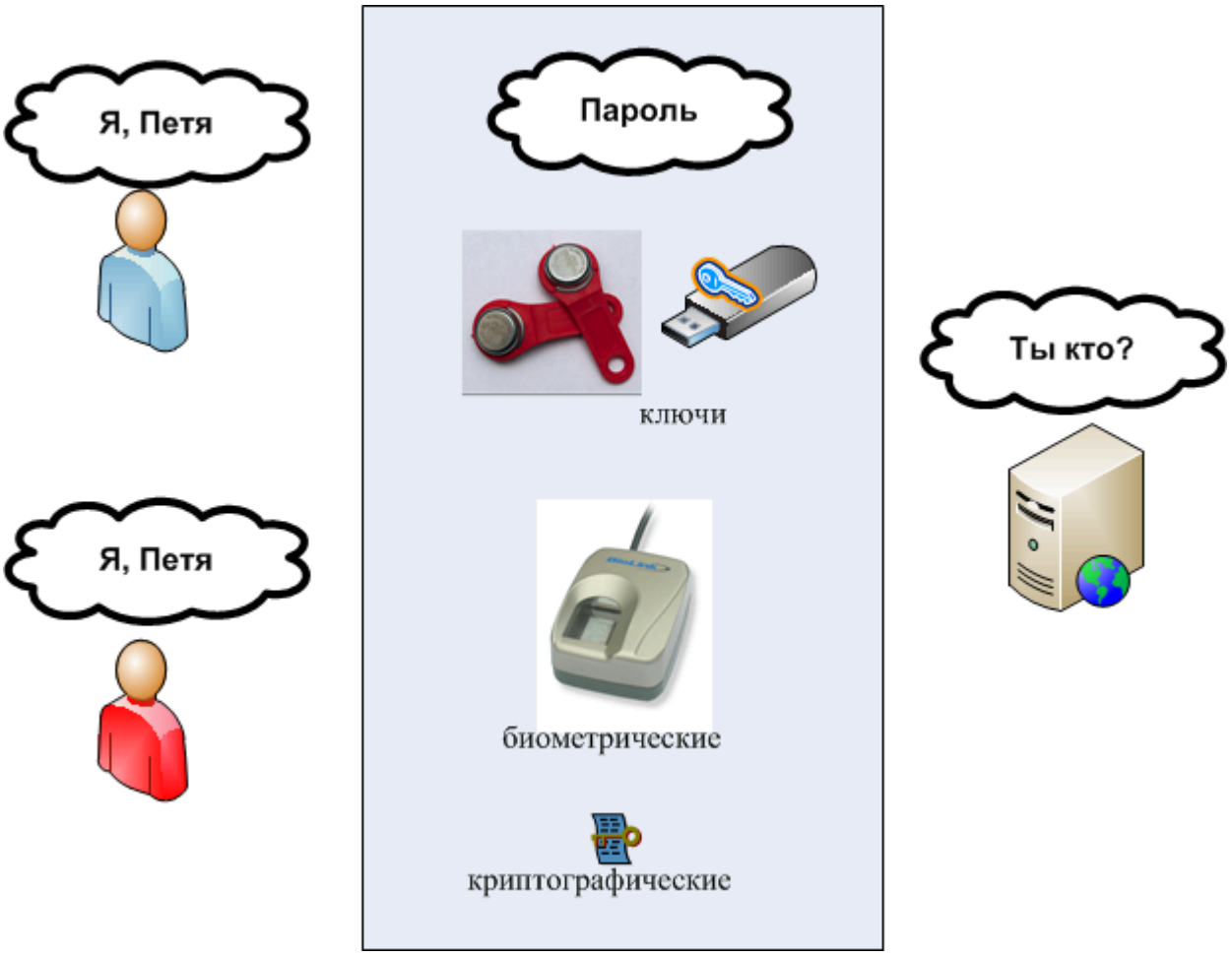
7 Аутентификация

Аутентификация (Authentication) — проверка принадлежности субъекту доступа по предъявленному им идентификатору (пароль, ключ и т.д.); подтверждение подлинности.

Методы аутентификации:

- **парольные** (PIN коде и т.д.) - уникальная последовательность символов, которую пользователь должен знать.
- **"ключи"** - в случае электронных систем это электронный ключ, который хранится на носителе (смарт-карты, электронные таблетки iButton, USB-токены и т. д.)
- **биометрические** (отпечаток пальца, рисунок радужной оболочки глаза, форма лица, параметры голоса и т. д.)
- **криптографические**



Аутентификация по многоразовым паролям

Используется один пароль многократно.

Хотя аутентификация может использоваться не только к удаленным системам, методы аутентификации будем рассматривать сразу на примерах к удаленным системам

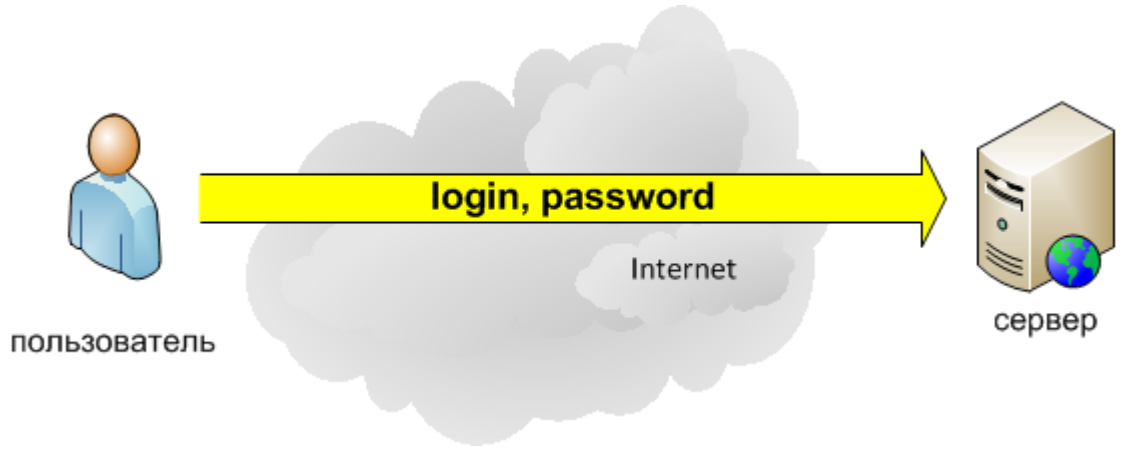
Протоколы аутентификации

PAP (Password Authentication Protocol)

PAP - аутентификация по имени и паролю пользователя. Протокол PAP ненадежен при использовании в сетях, т.к. пароли можно перехватить.

Алгоритм PAP:

1. клиент посылает имя и пароль серверу
2. сервер сверяет присланный пароль с паролем в своем хранилище



Преимущества:

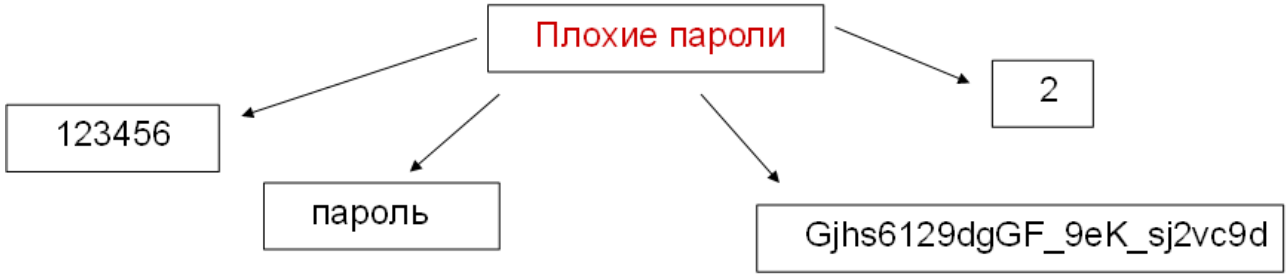
1. простота

Недостатки и пути решения:

1. подбор паролей
2. просмотр паролей в системе
3. перехват паролей при передачи
4. пароль можно «подсмотреть» при вводе
5. человеческий фактор – человек не может запомнить сложные пароли (записывает), диктует открытым способом (по телефону) и т.д.
6. каждый раз нужно набирать на клавиатуре
7. нужна предварительная регистрация пользователя в системе

Решение проблемы "подбора паролей" :

1. использовать "сильные" пароли
2. блокировка при неправильных попытках (например: 5 раз) ввода пароля



Почему эти пароли плохие:

"2" - один символ, легко перебрать.

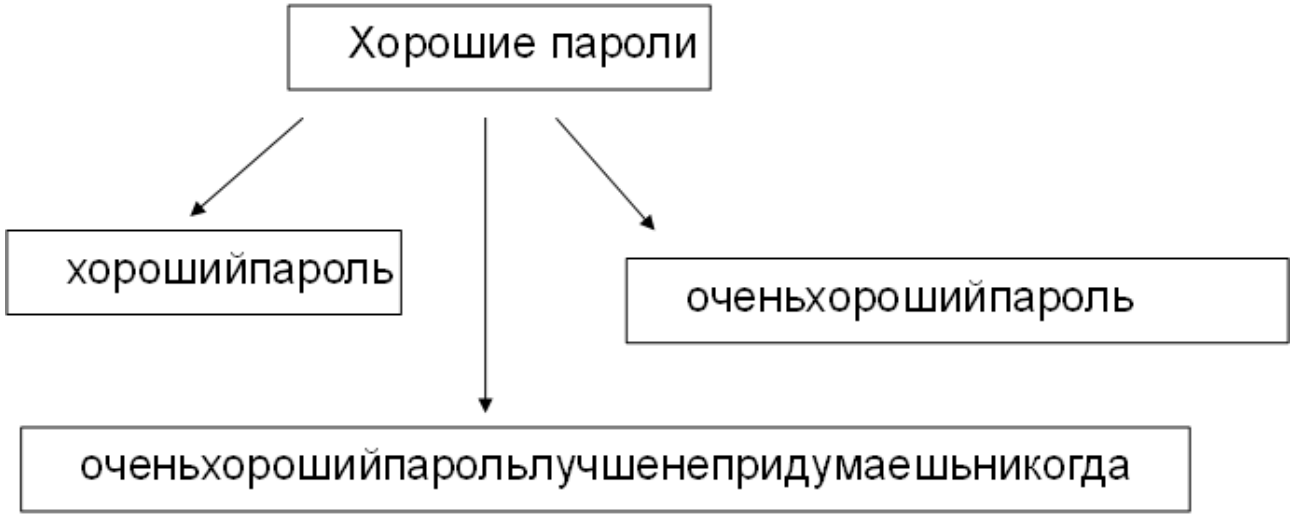
"123456" - один из популярных паролей (еще примеры - 123; 111; qwerty; qazwsx; qazwsxedc; password; "ваш логин"; "номер телефона"; "дата рождения" и т.д.).

"пароль" - словарное слово, после перебора популярных паролей, перебирают слова из словаря.

"Gjhs6129dgGF_9eK_sj2vc9d"** - пароль очень сложный, его не запомнят, а запишут и приклеят к монитору, пароль должен быть только в голове (или в сейфе).

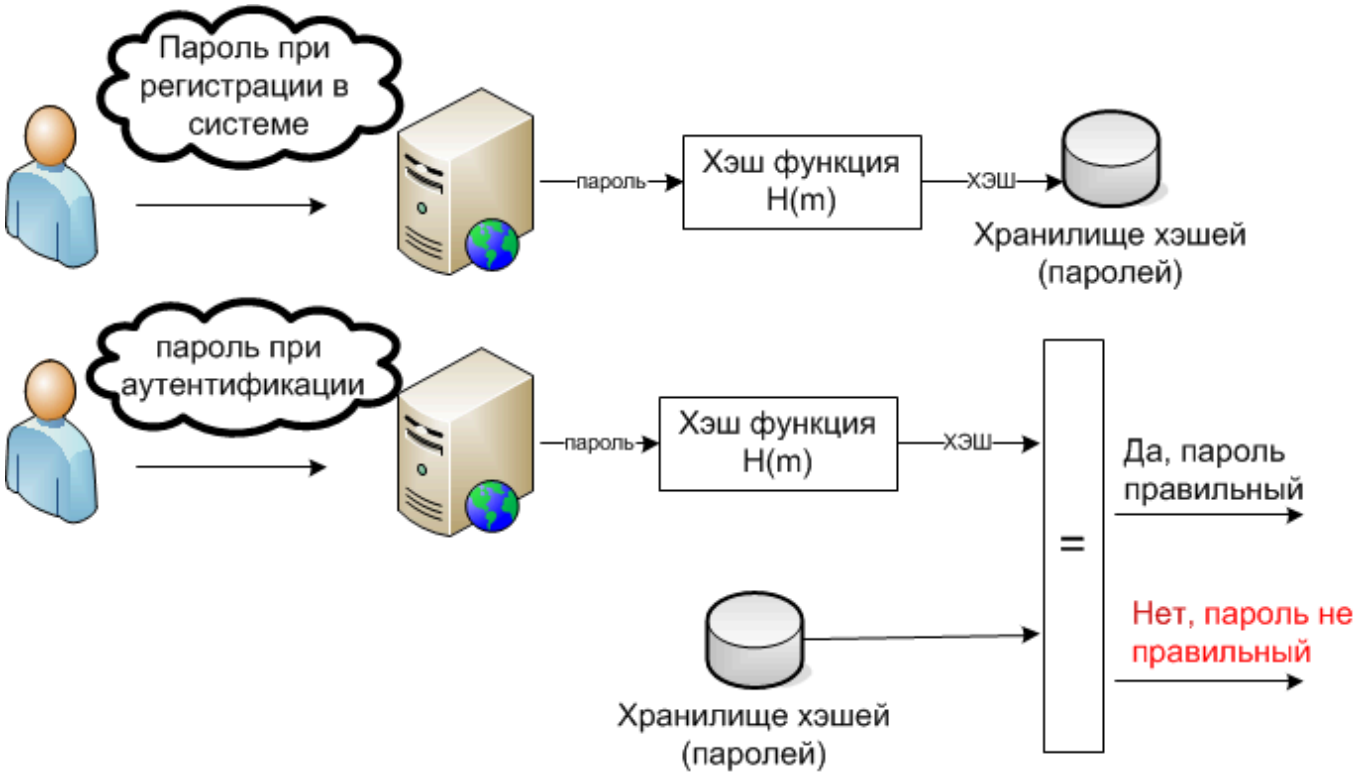
Наиболее хорошим вариантом являются пароли построенные на фразах:

1. хорошо запоминаются
2. достаточно длинные
3. словарные атаки не проходят



Решение проблемы "просмотра паролей в системе" :

4. шифрование (для расшифровывания нужно будет при себе носить ключ шифрования, при хранении на диске не защищенного ключа шифрования шифрование пароля не имеет смысла).
5. не хранить пароль в системе, а хранить его контрольную сумму или хэш.

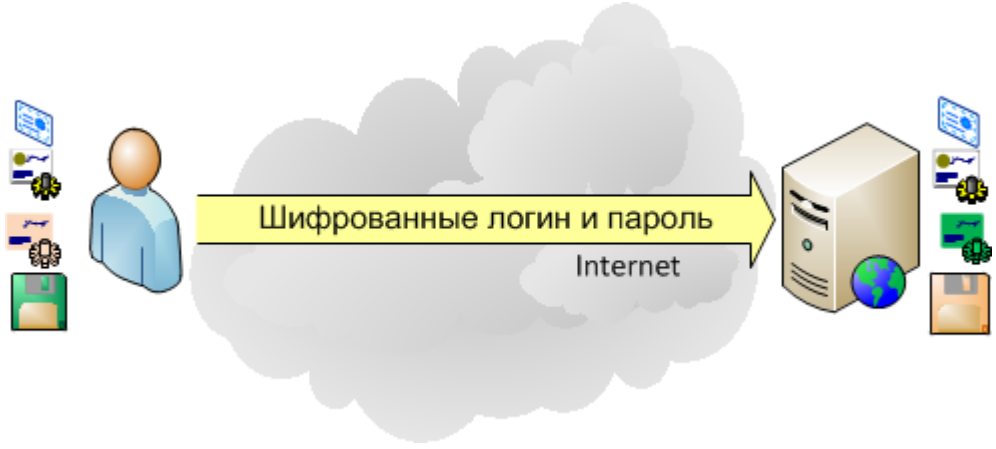


Пароли в системе не хранятся, при этом пользователь проходит аутентификацию по паролю.

В большинстве современных систем именно так и сделано. Не только в ОС, но и в СУБД, форумах, сайтах и т.д.

Решение проблемы "перехвата паролей при передачи":

1. шифровать передаваемые пароли
2. использовать алгоритмы без передачи паролей (рассмотрены ниже (CHAP))



В настоящее время чаще всего для шифрования паролей используется протокол SSL (Secure Sockets Layer — уровень защищённых сокетов)

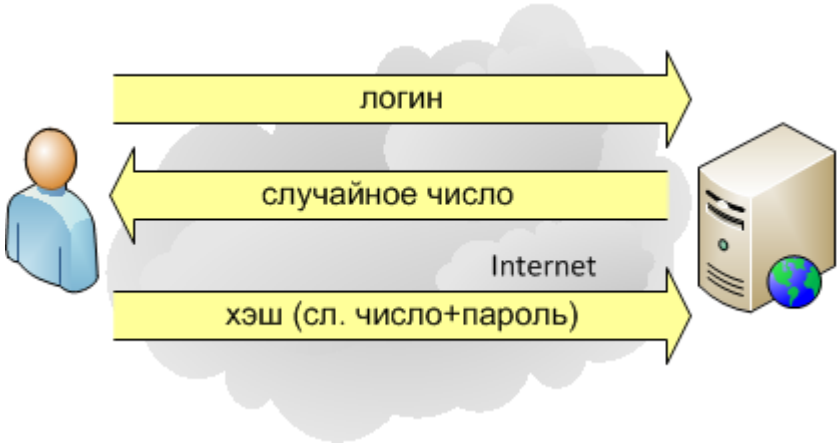
Протоколы аутентификации вызов-ответ

CHAP (Challenge Handshake Authentication Protocol)

CHAP - аутентификация без передачи пароля.

Алгоритм CHAP:

1. пользователь посылает серверу запрос на доступ (login)
2. сервер отправляет клиенту случайное число
3. на основе этого случайного числа и пароля пользователя клиент вычисляет хеш
4. клиент пересылает хеш серверу
5. сервер сверяет присланный хеш со своим вычисленным
6. в случайные промежутки времени сервер отправляет новый и повторяет шаги с 2 по 5.



Основной недостаток - необходимо хранить пароль на сервере.

CRAM - (challenge-response authentication mechanism)

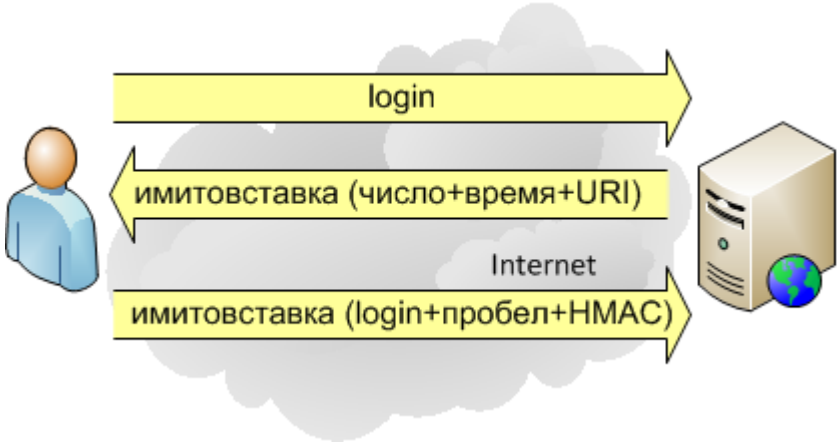
Основан на вычислении имитовставки по алгоритму **HMAC**, роль симметричного ключа выполняет **пароль**.

В зависимости от алгоритма хэширования - **CRAM-MD5, CRAM-MD4, CRAM-SHA1** и т.д.**

Алгоритм CRAM:

1. пользователь посылает серверу запрос на доступ (login)

2. сервер вычисляет имитовставку с секретным ключом-паролем пользователя для строки (случайное число + временная метка + доменное имя сервера) (например: 1896.697170952@postoffice.reston.mci.net)
3. сервер отправляет клиенту имитовставку
4. клиент вычисляет имитовставку из строки - (идентификатор клиента (login) + пробел + имитовставка сервера)
5. отправляет серверу
6. сервер сверяет полученное с ожидаемым



В CRAM вместо пароля на сервере может хранится хэш.

Digest access authentication (DIGEST-MD5)
Схема аналогичная **CHAP.**

Протокол:

1. запрос клиента (без аутентификации)
2. ответ сервера (Unauthorized), содержащий
   "realm" - строка (например: realm=testrealm@host.com)
   "nonce" - случайное число сервера (например: nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093")
3. клиент вычисляет хэш HA1 = MD5 (username: realm: password)
4. клиент вычисляет хэш HA2 = MD5 (URI)
5. клиент вычисляет хэш для ответа Response = MD5(HA1:nonce:nc:cnonce:qop:HA2)
   "nc" - счётчика запросов
   "cnonce" - клиентское случайное значение
   "qop" - код качества защиты
6. клиент посылает ответ
7. сервер сравнивает значение полученное и вычисленное

Пример:

HA1 = MD5("Mufasa:testrealm@host.com:Circle Of Life")
= 939e7578ed9e3c518a452acee763bce9

HA2 = MD5("GET:/dir/index.html") = 39aff3a2bab6126f332b942af96d3366

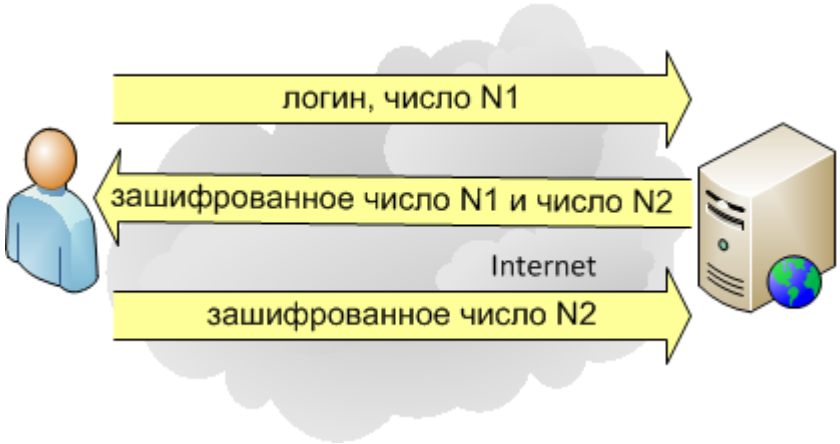
Response = MD5("939e7578ed9e3c518a452acee763bce9:\
dcd98b7102dd2f0e8b11d0f600bfb0c093:\
00000001:0a4f113b:auth:\
39aff3a2bab6126f332b942af96d3366")

= 6629fae49393a05397450978507c4ef1

Взаимная аутентификация

Т.к. сервер может быть ложным, необходимо провести взаимную аутентификацию.

1. клиент отправляет запрос серверу, содержащий его login и случайное число N1
2. сервер зашифровывает число N1, генерирует случайное число N2, и отправляет их оба клиенту
3. клиент расшифровывает числа (N1,N2) и сравнивает первое (N1) число с N1. Идентичность означает, что сервер обладает тем же уникальным ключом, что и клиент
4. клиент зашифровывает число N2 и результат отправляет серверу
5. сервер расшифровывает полученное сообщение. При совпадении результата с исходным числом N2, взаимная аутентификация прошла успешно.



Аутентификация по одноразовым паролям (One-time password)

Различные подходы к созданию одноразовых паролей:

- использующие математические алгоритмы для создания нового пароля на основе предыдущих (пароли фактически составляют цепочку, и должны быть использованы в определённом порядке).
- основанные на временной синхронизации между сервером и клиентом, обеспечивающей пароль (пароли действительны в течение короткого периода времени)
- использующие математический алгоритм, где новый пароль основан на запросе (например. случайное число, выбираемое сервером или части входящего сообщения) и/или счётчике.

Одноразовые пароли клиент может получать:

1. на бумаге
2. в токене
3. пересылкой (по СМС)