Лекция 5

Организация работы сессий

Определение сессии

- Web-сессия – специальный ключ (идентификатор сессии), хранящийся клиентской программой и сопоставляемый с пользовательскими данными на стороне сервера.
- Сессии предназначены для хранения и передачи данных отдельного пользователя между динамическими страницами одного ресурса.

Причины появления сессий

- Web-сервер каждый раз при обращении к очередной странице инициализирует новую HTTP-транзакцию без возможности связывания старых данных пользователя с вызовом новой динамической страницы.

Способы хранения ключа сессии на стороне клиента

Ключ (идентификатор сессии) на стороне клиента может храниться двумя способами:

- В HTTP-Cookie (относительно безопасно)
- Как часть URL (небезопасно)

Хранение сессии на стороне сервера

- Способы хранения данных внутри сессии никак не регламентируются и могут быть представлены различными способами: в структурных файлах сервера, в БД различного типа и т.д.
- Данные сессии хранятся на сервере в виде массива.

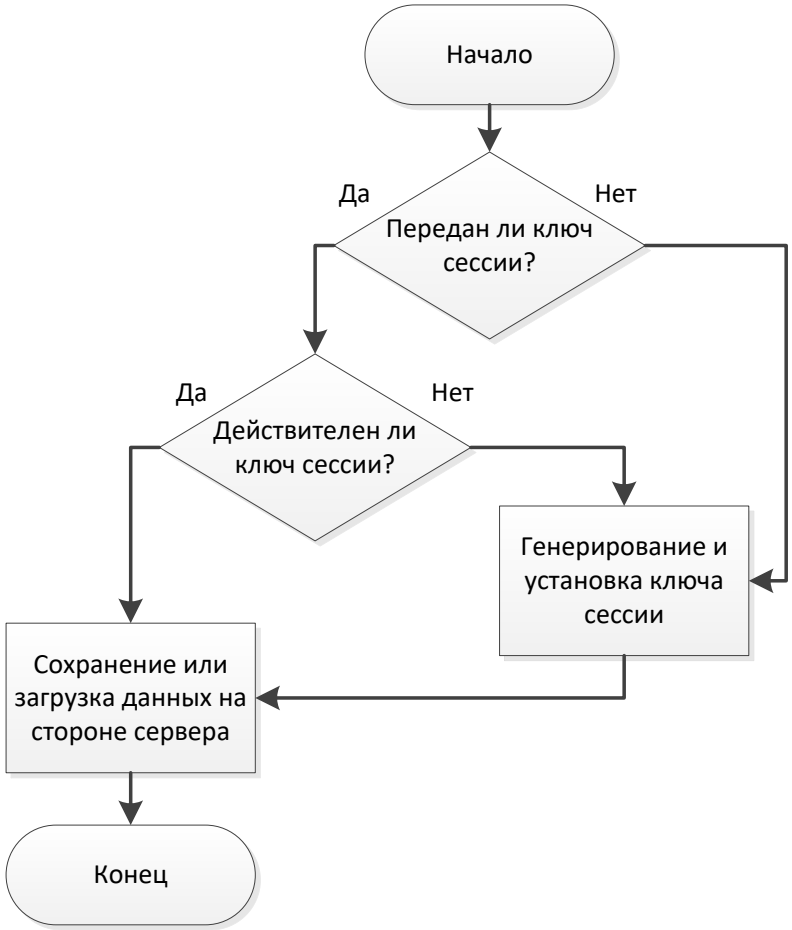
Использование сессий

Сессии могут использоваться для тех же целей, что и HTTP Cookie, но с сохранением данных на стороне сервера.

С помощью сессий также реализуются следующий функционал:

- Подтверждение авторизации пользователя на сайте
- Хранение временных данных, вводимых из нескольких форм
- Корзина в интернет-магазине
- Хранение символьного значения картинки из CAPTCHA

Механизм работы сессий



- Ключ считывается с сервера либо из строки URL, либо из Cookie
- На этапе проверки ключа проверяется, существуют ли данные на сервере, сопоставляемые с этим ключом.
- Ключ генерируются таким образом, чтобы исключить возможность его подмены.

Алгоритм генерации сессии

- Ключ сессии, сохраняемый в Cookie, именуется уникальным идентификатором через uuid().
- Значения сессии хранятся внутри сервера в NoSQL базе данных. В качестве таковой предлагается Redis
- Значения сериализуются с ключом сессии

Сериализация сессии

- Сериализация — процесс перевода какой-либо структуры данных в последовательность символов ASCII
- Алгоритм сериализации сессии: UUID+переменная.
- Пример:
  - UUID=294510df…25464
  - Переменная=is_admin
  - Сериализуемый объект: 294510df…25464is_admin

Пример работы с Redis

```
cpp_redis::client client;

client.connect();

client.set("hello", "42");
client.get("hello", [](cpp_redis::reply& reply) {
  std::cout << reply << std::endl;
});
```

Генератор uuid()

```
#ifndef UUID_H
#define UUID_H

#include <string>
#include <cstdlib>

namespace MathUtils{
const std::string CHARS = "0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghij

std::string generateUUID(){
  std::string uuid = std::string(36,' ');
  int rnd = 0;
  int r = 0;

  uuid[8] = '-';
  uuid[13] = '-';
  uuid[18] = '-';
  uuid[23] = '-';

  uuid[14] = '4';

  for(int i=0;i<36;i++){
    if (i != 8 && i != 13 && i != 18 && i != 14 && i != 23) {
      if (rnd <= 0x02) {
          rnd = 0x2000000 + (std::rand() * 0x1000000) | 0;
      }
      rnd >>= 4;
      uuid[i] = CHARS[(i == 19) ? ((rnd & 0xf) & 0x3) | 0x8 : rnd & 0xf];
    }
  }
  return uuid;
}
#endif
```

Сохранение данных сессии

```
cpp_redis::client client;

client.connect();

bool is_admin;

// предположим, что переменная сессии is_admin отвечает за то, что юзер администратор
// предположим, что авторизованный юзер администратор, тогда is_admin = 1
client.set("294510dfa85bc143830d7ccc5dd25464is_admin", "1");

client.get("294510dfa85bc143830d7ccc5dd25464is_admin", [](cpp_redis::reply& reply) {
  is_admin = reply;
  if(is_admin)
    cout << "Пользователь является администратором" << endl;
  else
    cout << "Пользователь не является администратором" << endl;
});
```

Безопасность сессии

Узнав идентификатор сессии, злоумышленник может получить доступ к учётной записи другого пользователя. Способы получения идентификатора:

- Перехватка GET-запроса через HTTP-заголовок **Http-Referer** на сайте, на который осуществлён переход (в том случае, если сессия хранится как часть URL)
- Перехватка Cookies путём внедрения JavaScript-кода на одну из страниц сайта через **XSS** уязвимость ресурса или дополнительные привилегии администратора
- Прослушивание HTTP-трафика

Безопасность сессии

Существуют следующие методы обеспечения безопасности сессии:

- Запрет на хранение идентификатора сессии в части URL (только в Cookies)
- Экранирование символов, способных внедрить HTML-код на страницу
- Использование протокола **SSL** над HTTP

Другие технологии защиты сессии

- Ограничение срока действия сессии
- Привязка к IP-адресу пользователя, его браузеру и другим характеристикам

Литература

- https://github.com/cpp-redis/cpp_redis - Redis Client
- https://gist.github.com/fernandomv3/46a6d7656f50ee8d39dc - UUID

Лабораторная работа

- Разработать механизм работы с сессией.
- Конструктор сессии генерирует новый идентификатор сессии
- Методы set() и get() устанавливают и получают значение сессии, соответственно

Структура класса Session()

```
#include <cpp_redis/cpp_redis>
class Session
{
    public:
    …
    std::string set(std::string name);
    std::string get(std::string name);

    …
   private:
    cpp_redis::client client;
    …
}
```