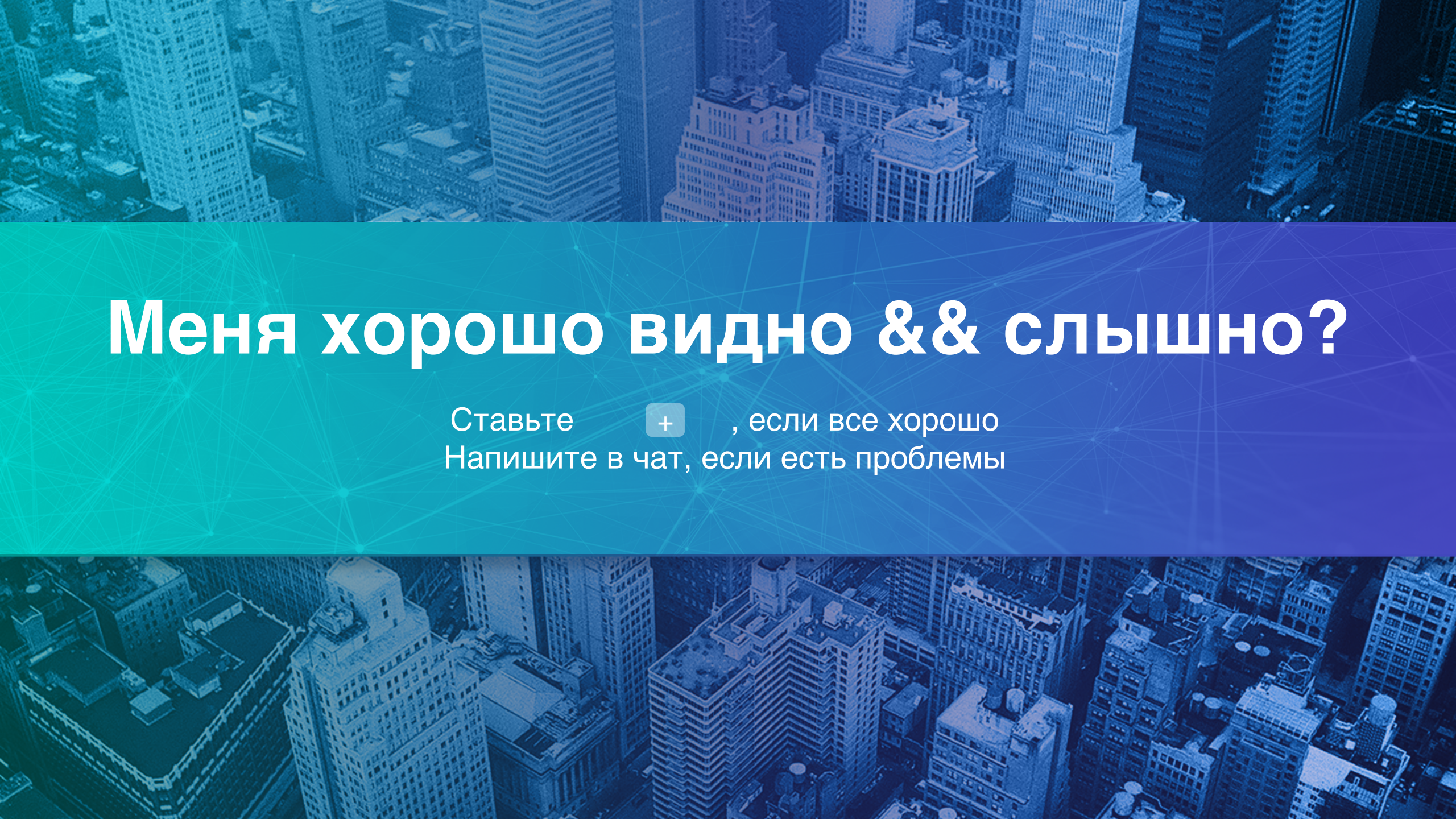
OTUS

ОНЛАЙН-ОБРАЗОВАНИЕ

Онлайн-образование

Проверить, идет ли запись!



Меня хорошо видно && слышно?

Ставьте + , если все хорошо
Напишите в чат, если есть проблемы

Уязвимости класса: Open Redirect, CSRF

Пархомец Павел

Penetration tester

Awillix LLC

тг: gremlin_97

Павел Пархомец

- Yandex hall of fame
- Awillix LLC (Специалист по тестированию на проникновение)
- Победитель международных конкурсов HITB AI Challenge и Kaspersky SecurIT Cup’19
- Разрабатываю курс по информационной безопасности для лицеистов при НИЯУ МИФИ
- Тренер команд по наступательной безопасности
- Специализируюсь на эксплуатации уязвимостей веб-приложений
- CTF игрок (Sploit00n, Eun014)
- eJPT, OSCP, eWPT

Правила вебинара

Активно участвуем

Задаем вопрос в чат или голосом

Off-topic обсуждаем в Slack #канал группы или #general

Вопросы вижу в чате, могу ответить не сразу

План вебинара

1)Open redirect

- Что это
- Последствия
- Откуда появляется
- Как искать
- Противодействие
- Решение таска

2)CSRF

- Что это
- К чему приводит
- Пример
- Насколько серьезны
- Противодействие

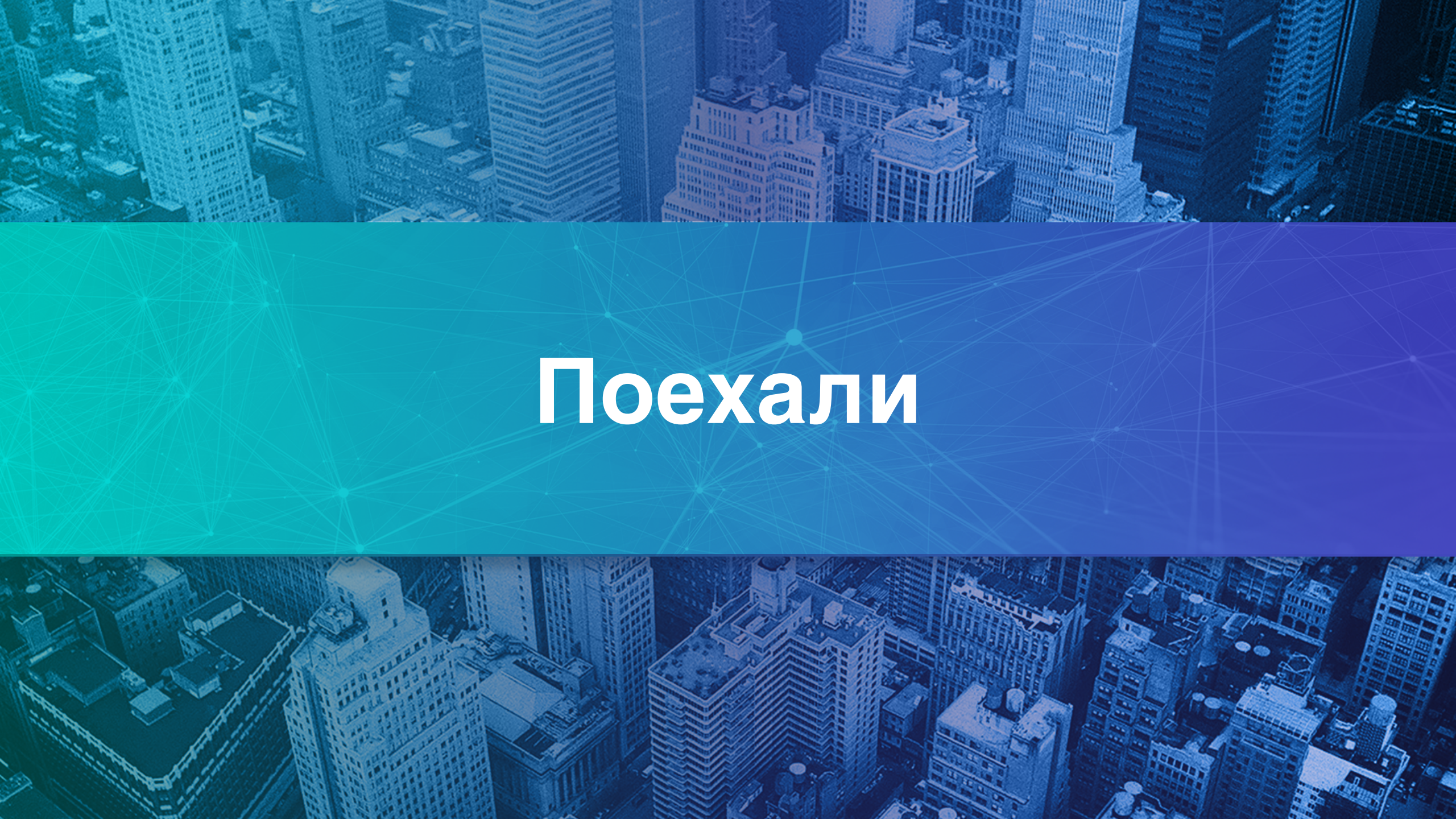
Цели вебинара

1. Познакомиться с Open Redirect
2. Познакомиться с CSRF

Смысл | Зачем вам это уметь

1. Создавать безопасные приложения

2. Помогать создавать безопасные приложения


Поехали

План вебинара

1)Open redirect

- Что это
- Откуда появляется
- Как искать
- Противодействие
- Решение таска

2)CSRF

- Что это
- К чему приводит
- Пример
- Насколько серьезны
- Противодействие

Open Redirect | Что это

Open Redirect Vulnerabilities — уязвимость web-сайта, позволяющая путём манипуляции параметров в URL перенаправить пользователя на ресурс, непредусмотренный разработчиком. В результате злоумышленник может отправить жертве письмо со ссылкой на один сайт, а в результате редиректа жертва попадёт на другой сайт.

Open Redirect | Цель

Цель - Фишинг

План вебинара

1)Open redirect

- Что это
- Откуда появляется
- Как искать
- Противодействие
- Решение таска

Open Redirect | Откуда появляется

```php
<?php
$redirect_url = $_GET['url'];
header("Location: " . $redirect_url);
?>
```

Тогда уязвимая ссылка будет такой: `http://site.com/example.php?url=http://evil.com`.

Open Redirect | Откуда появляется

```
public class RedirectServlet extends HttpServlet {
  protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
    String query = request.getQueryString();
    if (query.contains("url")) {
      String url = request.getParameter("url");
      response.sendRedirect(url);
    }
  }
}
```

```
<a href="http://bank.site.com/redirect?url=http://evil.com">Click here to log in</a>
```

План вебинара

1)Open redirect

- Что это
- Откуда появляется
- Как искать
- Противодействие
- Решение таска

Open redirect I Как искать

Google:

- `allinurl` – оператор, который скажет Google искать в URL-адресе все указанные ключевые слова. Например: `allinurl:ReturnUrl` будет искать все веб-страницы, у которых в адресе будет присутствовать часть ReturnUrl.
- `site` – оператор, который говорит возвращать только те результаты, которые находятся на определенном домене или веб-сайте. Пример: `site:example.com` который ищет веб-страницы по `example.com`.
- "" – двойные кавычки – это специальные символы, который используются для указания на поиск точного сочетания слов и символов внутри кавычек.
- `*` — звездочка – знак подстановки, который олицетворяет одно или несколько слов.

```
allinurl:%3Dhttps*
allinurl:%253Dhttps*
allinurl:%3Dhttp*
allinurl:%253Dhttp*
```

Open redirect | Как искать

```
allinurl:"<keyword>=https"
allinurl:"<keyword>=http"
allinurl:<keyword>=https
allinurl:<keyword>=http
allinurl:<keyword>%3Dhttps
allinurl:<keyword>%3Dhttps*
allinurl:<keyword>%253Dhttps
allinurl:<keyword>%253Dhttps*
allinurl:<keyword>%3Dhttp
allinurl:<keyword>%3Dhttp*
allinurl:<keyword>%253Dhttp
allinurl:<keyword>%253Dhttp*
allinurl:<keyword>
```

RelayState, ReturnUrl, RedirectUri, Return, Return_url, Redirect, Redirect_uri, Redirect_url, RedirectUrl, Forward, Forward_url, SuccessUrl, Redir, Exit_url, Destination.

План вебинара

1)Open redirect

- Что это
- Откуда появляется
- Как искать
- Противодействие
- Решение таска

Open Redirect | Противодействие

1) Избегать динамических параметров, зависящих от пользователей
2) Белый список

План вебинара

1)Open redirect

- Что это
- Откуда появляется
- Как искать
- Противодействие
- Решение таска

Tack

http://challenge01.root-me.org/web-serveur/ch52/

План вебинара

1)Open redirect

- Что это
- Откуда появляется
- Как искать
- Противодействие
- Решение таска

2)CSRF

- Что это
- К чему приводит
- Пример
- Насколько серьезны
- Противодействие

CSRF

CSRF (англ. *Cross Site Request Forgery* — «межсайтовая подделка запроса», также известна как XSRF) — вид атак на посетителей веб-сайтов, использующий недостатки протокола HTTP. Если жертва заходит на сайт, созданный злоумышленником, от её лица тайно отправляется запрос на другой сервер (например, на сервер платёжной системы), осуществляющий некую вредоносную операцию

CSRF

Цель CSRF - выполнение атак от имени пользователя

План вебинара

1)Open redirect

- Что это
- Откуда появляется
- Как искать
- Противодействие
- Решение таска

2)CSRF

- Что это
- К чему приводит
- Пример
- Насколько серьезны
- Противодействие

CSRF I Последствия

```
Боб: Привет, Алиса! Посмотри, какой милый котик: <img
src="http://bank.example.com/withdraw?account=Alice&amount=1000000&for=Bob">
```

Если банк Алисы хранит информацию об аутентификации Алисы в куки, и если куки ещё не истекли, при попытке загрузить картинку браузер Алисы отправит куки в запросе на перевод денег на счёт Боба, чем подтвердит аутентификацию Алисы. Таким образом, транзакция будет успешно завершена, хотя её подтверждение произойдет без ведома Алисы.

CSRF

В том, что CSRF-атаки работают виноваты Cookie.

План вебинара

1)Open redirect

- Что это
- Откуда появляется
- Как искать
- Противодействие
- Решение таска

2)CSRF

- Что это
- К чему приводит
- Пример
- Насколько серьезны
- Противодействие

Пример

https://attacker.com/csrf-form.html

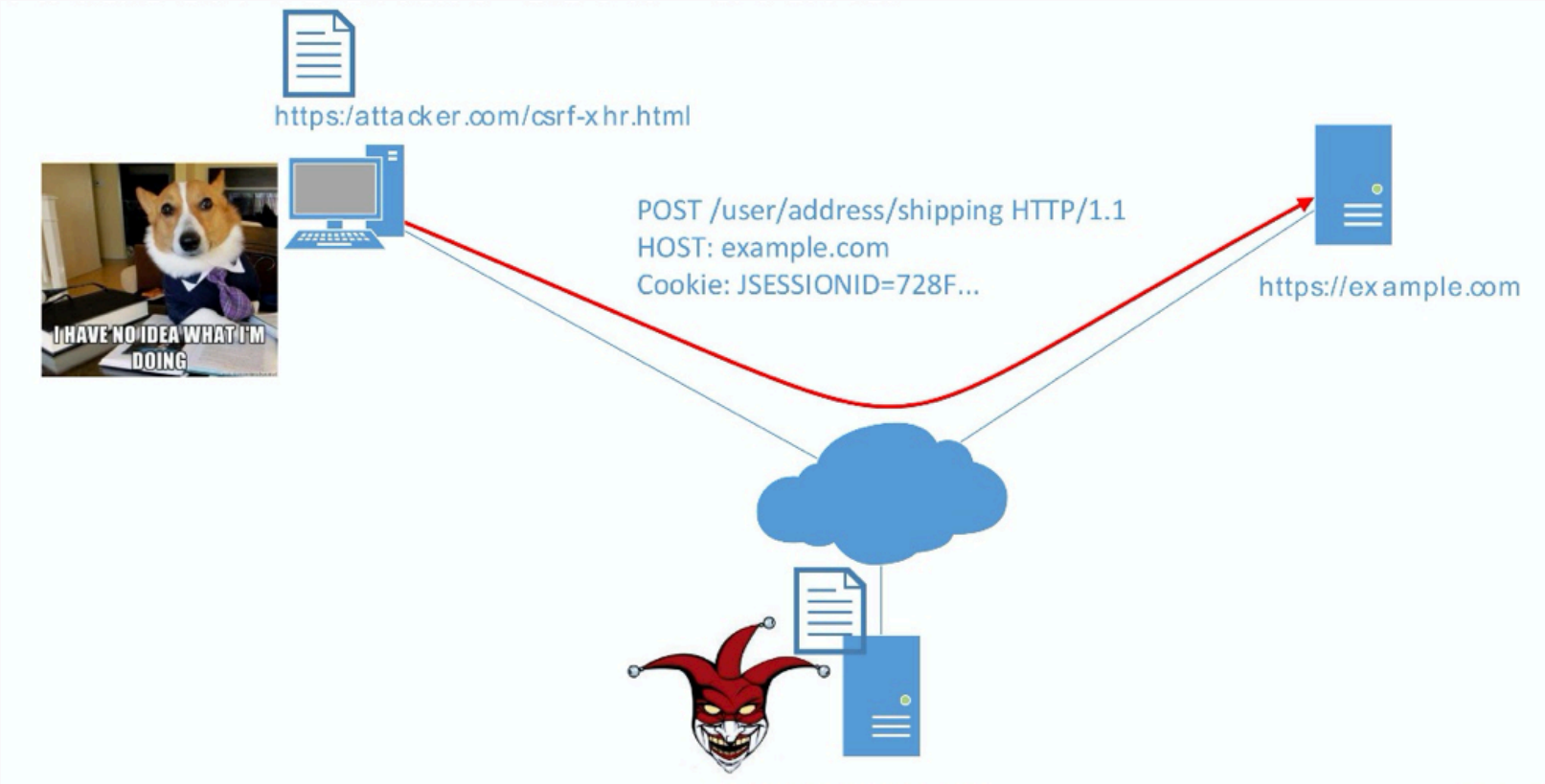
```
<html>
 <body>
  <script>history.pushState('', '', '/')</script>
  <form action="https://example.com/user/address/shipping"
method="POST">
   <input type="hidden" name="city" value="Moscow" />
   <input type="hidden" name="street" value="Prospekt&#32;Mira" />
   <input type="hidden" name="zip" value="12345" />
   <input type="submit" value="Submit request" />
  </form>
 </body>
</html>
```

Пример

https://attacker.com/csrf-xhr.html

```
<script>
  var request = new XMLHttpRequest();
  var data = 'city=Moscow&street=Prosperkt+Mira&zip=12345';
  request.open('POST', 'https://example.com/user/address/shipping', true);
  request.withCredentials = true; // INCLUDE COOKIES
  request.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
  request.send(data);
</script>
```

Пример


https:/attacker.com/csrf-xhr.html
I HAVE NO IDEA WHAT I'M DOING
POST /user/address/shipping HTTP/1.1
HOST: example.com
Cookie: JSESSIONID=728F...
https://example.com

План вебинара

1)Open redirect

- Что это
- Откуда появляется
- Как искать
- Противодействие
- Решение таска

2)CSRF

- Что это
- К чему приводит
- Пример
- Насколько серьезны
- Противодействие

Насколько серьезны?

Все зависит от критичности уязвимого действия:

- Account takeover
- Privilege Escalation
- Remote code execution

План вебинара

1)Open redirect

- Что это
- Откуда появляется
- Как искать
- Противодействие
- Решение таска

2)CSRF

- Что это
- К чему приводит
- Пример
- Насколько серьезны
- Противодействие

Защита | CSRF-token

CSRF token

- Для каждой пользовательской сессии генерируется уникальный и **высокоэнтропийный** токен.
- Токен вставляется в DOM HTML страницы или отдается пользователю через API.
- Пользователь с каждым запросом, связанным с какими-либо изменениями, должен отправить токен в параметре или в HTTP-заголовке запроса.
- Так как атакующий не знает токен, то классическая CSRF-атака не работает.

Защита I CSRF-token

Формула вычисления токена:

```
token = salt + ":" + MD5(salt + ":" + secret)
```

Например:

1. В сессии хранится `secret="abcdef"`, это значение создаётся один раз.
2. Для нового токена сгенерируем `salt`, например пусть `salt="1234"`.
3. `token = "1234" + ":" + MD5("1234" + ":" + "abcdef") = "1234:5ad02792a3285252e524ccadeeda3401"`.

Защита | CSRF-token

```
<form action="http://mail.com/send" method="POST">
  <input type="hidden" name="csrf" value="1234:5ad02792a3285252e524ccadeeda3401">
  <textarea name="message">
    ...
  </textarea>
</form>
```

Защита | CSRF-token (AJAX)

```
var request = new XMLHttpRequest();

var csrfCookie = document.cookie.match(/CSRF-TOKEN=([\w-]+)/);
if (csrfCookie) {
  request.setRequestHeader("X-CSRF-TOKEN", csrfCookie[1]);
}
```

Защита | Double submit cookie

Double submit cookie

- Опять генерируется уникальный и **высокоэнтропийный** токен для каждой пользовательской сессии, но он помещается в куки.
- Пользователь должен в запросе передать одинаковые значения в куках и в параметре запроса.
- Если эти два значения совпадают в куках и в параметре, то считается, что это легитимный запрос.
- Так как атакующий просто так не может изменить куки в браузере пользователя, то классическая CSRF-атака не работает.

Защита I Samesite cookie

- У куки устанавливается дополнительный атрибут — samesite, который может иметь два значения: lax или strict.
- Суть технологии в том, что браузер не отправляет куки, если запрос осуществляется с другого домена, например, с сайта атакующего. Таким образом это опять защищает от классической CSRF-атаки.

Тест

https://docs.google.com/forms/d/e/1FAIpQLScwJ_o_LgUKi7ijPccPbzrch5RkdnYlJ6HkVUk9FNkhkwvYUA/viewform?usp=sf_link

Цели вебинара I Проверка достижения целей

1. Узнали про Open redirect и CSRF

2. Узнали методы противодействия им

Рефлексия

С какими основными мыслями и инсайтами уходите с вебинара?

Достигли ли вы цели вебинара?

Следующий вебинар

Тема: **Уязвимости класса: HTML Injection, Content Spoofing, Cross-Site Scripting**

14.08

Ссылка на вебинар будет в ЛК за 15 минут

Материалы к занятию в ЛК — можно изучать

Обязательный материал обозначен красной лентой

Список материалов для изучения

0)https://ru.wikipedia.org/wiki/Межсайтовая_подделка_запроса

1)https://habr.com/ru/company/oleg-bunin/blog/412855/

2)https://learn.javascript.ru/csrf

3)https://habr.com/ru/post/318748/

4) https://portswigger.net/web-security/websockets/cross-site-websocket-hijacking

5) https://xakep.ru/2018/10/09/open-redirect/#toc01.2

6) https://internet-lab.ru/ctf_http_open_redirect

7) https://habr.com/ru/company/otus/blog/511428/


Заполните, пожалуйста,
опрос о занятии по ссылке в чате

Спасибо за внимание!
Приходите на следующие вебинары

Пархомец Павел

Специалист по тестированию на проникновение

Awillix LLC

tg: @gremlin_97