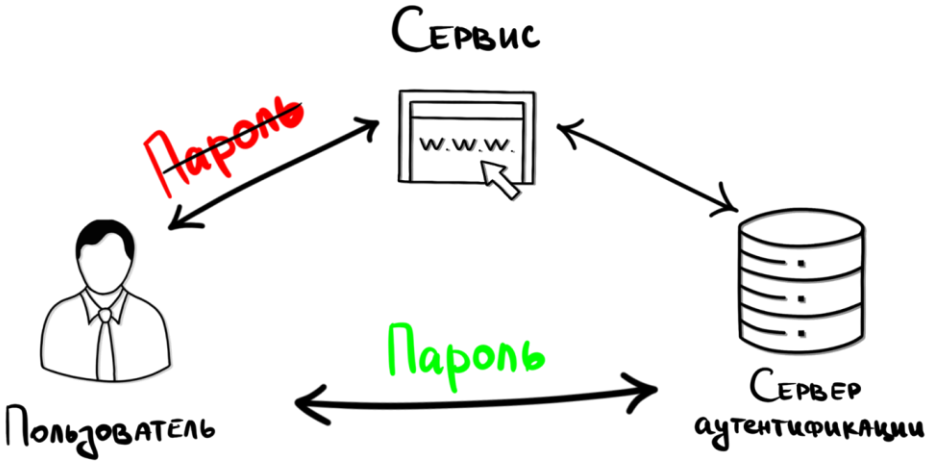
РусКрипто XXVIII НАУЧНО-ПРАКТИЧЕСКАЯ КОНФЕРЕНЦИЯ

Модель безопасности для протоколов делегированной аутентификации на основе HTTP

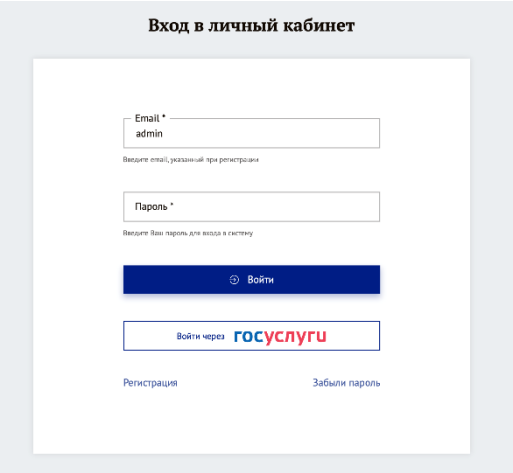
Мурадян Давид Каренович, инженер-аналитик, КриптоПро
Ахметзянова Лилия Руслановна, заместитель начальника отдела криптографических исследований, КриптоПро
Алексеев Евгений Константинович, начальник отдела криптографических исследований, КриптоПро

Протоколы делегированной аутентификации (ПДА)



- Пользователь аутентифицируется перед Сервисом.
- Сервис делегирует аутентификацию Серверу аутентификации.
- Сервис не получает от Пользователя его аутентификационные данные.

Делегированная аутентификация



[1]


ТЕХНИЧЕСКИЙ КОМИТЕТ ПО СТАНДАРТИЗАЦИИ
«КРИПТОГРАФИЧЕСКАЯ ЗАЩИТА ИНФОРМАЦИИ»

МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ
ТК 26
МР.26.2.002
–2024

Информационная технология

КРИПТОГРАФИЧЕСКАЯ ЗАЩИТА ИНФОРМАЦИИ
Использование российских криптографических алгоритмов в протоколах OpenID Connect

© Технический комитет по стандартизации
«Криптографическая защита информации»

Москва
2024

[2]

[3]

[1] МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ПО ИСПОЛЬЗОВАНИЮ ЕСИА, Версия 3.48

[2] МР.26.2.002-2024. Информационная технология. Криптографическая защита информации. Использование российских криптографических алгоритмов в протоколах OpenID Connect.

[3] https://openid.net/

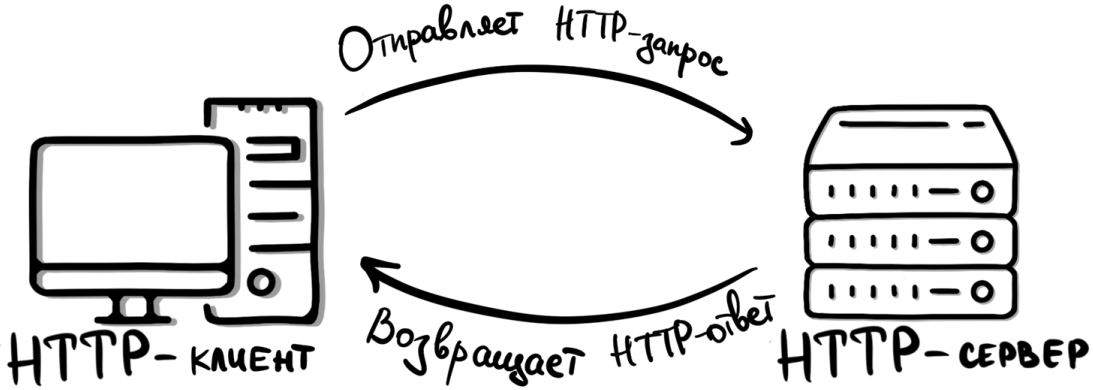
Существующие подходы к анализу

	Количественные оценки стойкости	Наличие исследований
Формальные верификаторы	−	+ [1]
Идеальные функциональности	±	+ [2]
Модели на основе экспериментатора	+	−

[1] Daniel Fett, Ralf Kuesters, Guido Schmitz (2016) A Comprehensive Formal Security Analysis of OAuth 2.0, https://arxiv.org/abs/1601.01229

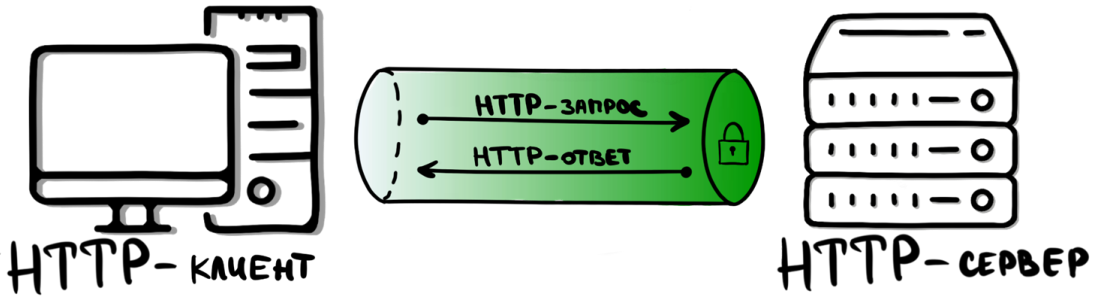
[2] Suresh Chari, Charanjit S Jutla, and Arnab Roy. Universally composable security analysis of OAuth v2.0. http://eprint.iacr.org/2011/526, September 2011. IACR Cryptology ePrint Archive: Report 2011/526.

Особенности механизма



Взаимодействие по протоколу HTTP.

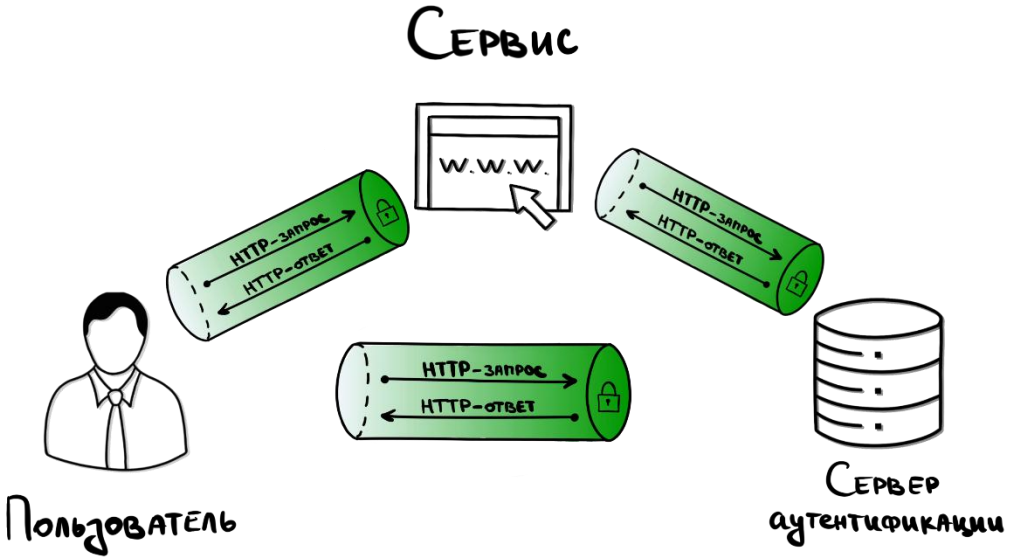
Особенности механизма



Присутствует защита канала между HTTP-клиентом и HTTP-сервером с односторонней аутентификацией HTTP-сервера.

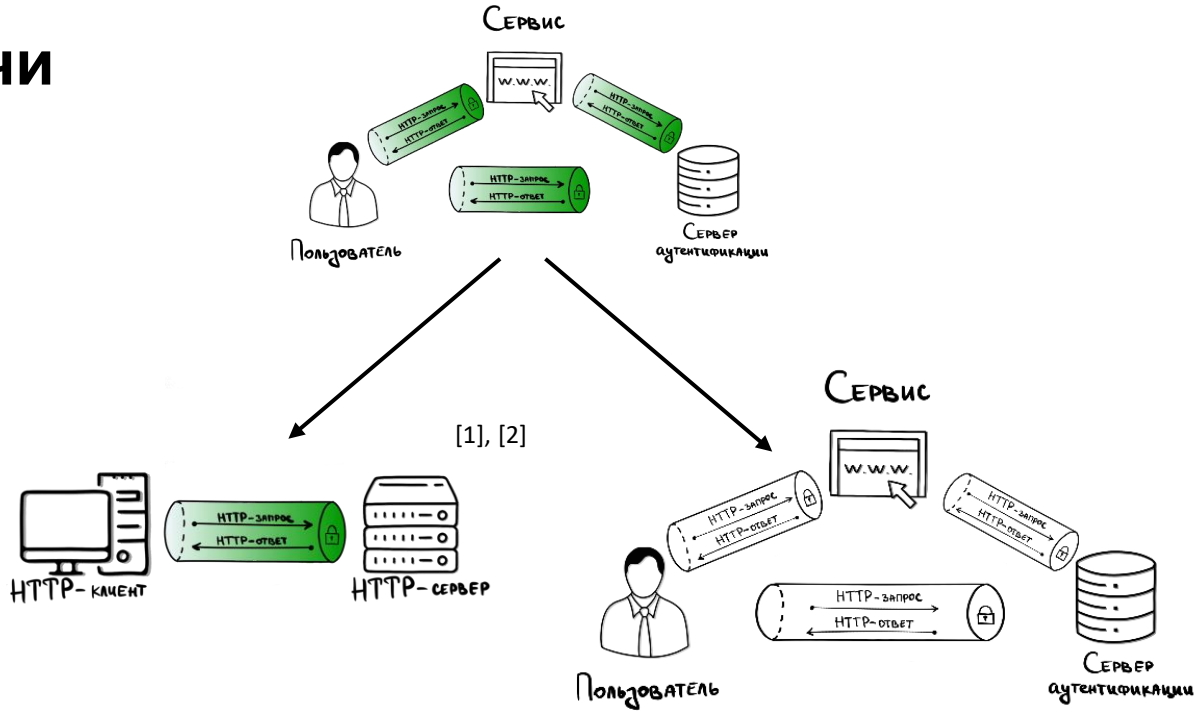
Особенности механизма

Для протоколов делегированной аутентификации один и тот же участник может выступать как в качестве HTTP-клиента, так и в качестве HTTP-сервера.



Постановка задачи

- Анализ протокола в исходном виде сложен и трудоёмок.
- Решение: декомпозиция на анализ TLS + анализ протокола с идеализированными каналами.

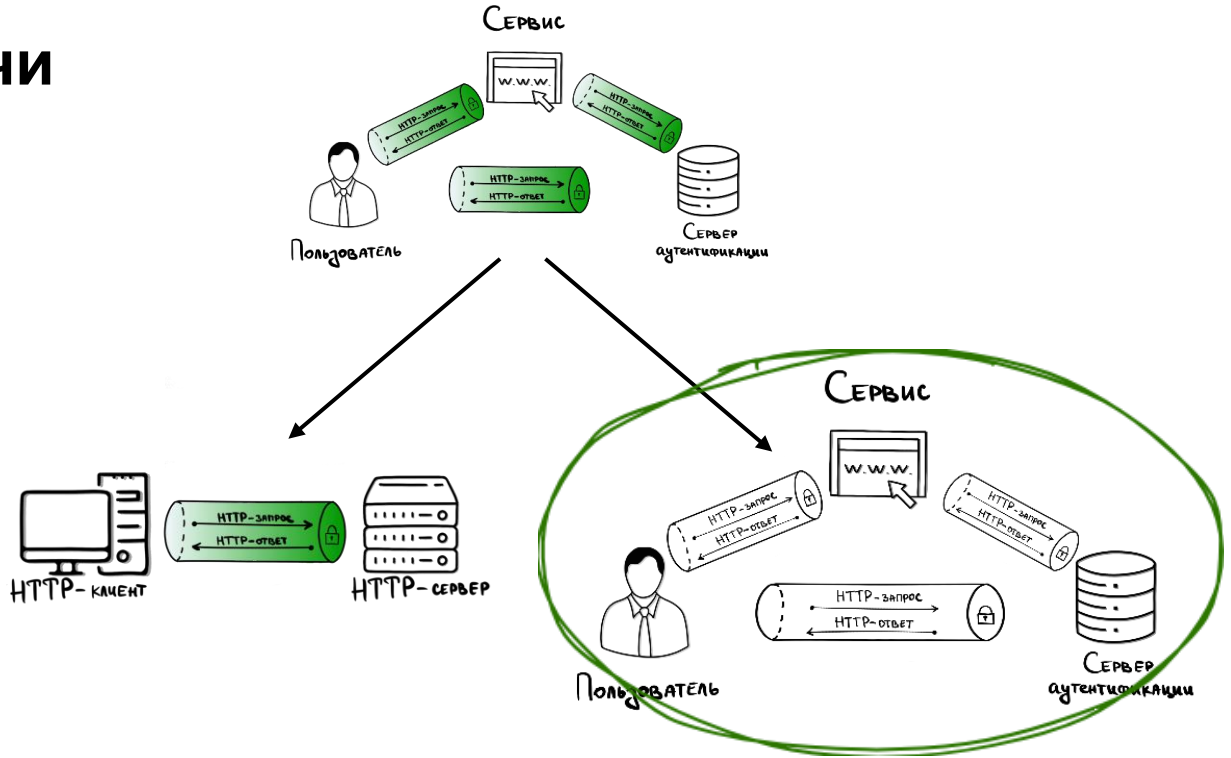


[1] Tibor Jager, Florian Kohlar, Sven Schäge, Jörg Schwenk. A Standard-Model Security Analysis of TLS-DHE.

[2] Benjamin Dowling, Marc Fischlin, Felix Günther & Douglas Stebila. A Cryptographic Analysis of the TLS 1.3 Handshake Protocol.

Постановка задачи

- Анализ протокола в исходном виде сложен и трудоёмок.
- Решение: декомпозиция на анализ TLS + анализ протокола с идеализированными каналами.



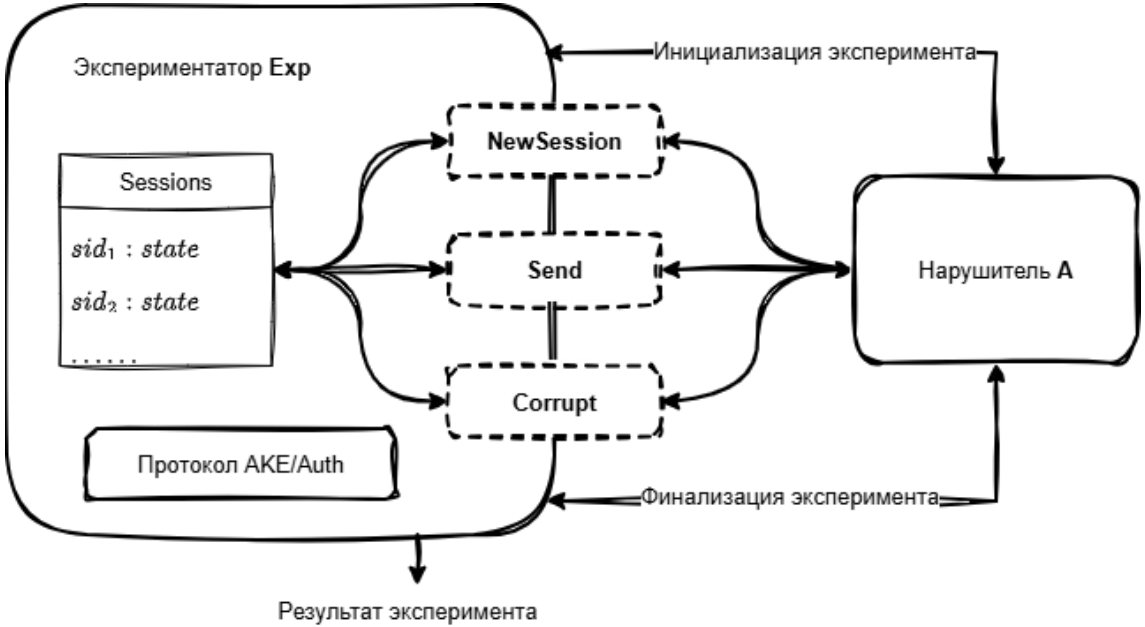
Постановка задачи

Необходимо разработать **модель безопасности** для анализа ПДА на основе HTTP.

Родственные протоколы, для которых существуют модели на основе экспериментатора:

- Протоколы AKE.
- Протоколы Auth.

Модели для родственных протоколов



Оракулы — подпрограммы экспериментатора. Задают возможности нарушителя.

Интерфейс оракула Send

$Send(msg, sid)$

1 : $state \leftarrow \pi[sid]$

2 : $\ldots$

3 : $msg', state' \leftarrow \mathcal{P}.Process(msg, state)$

4 : $\ldots$

5 : $\pi[sid] \leftarrow state'$

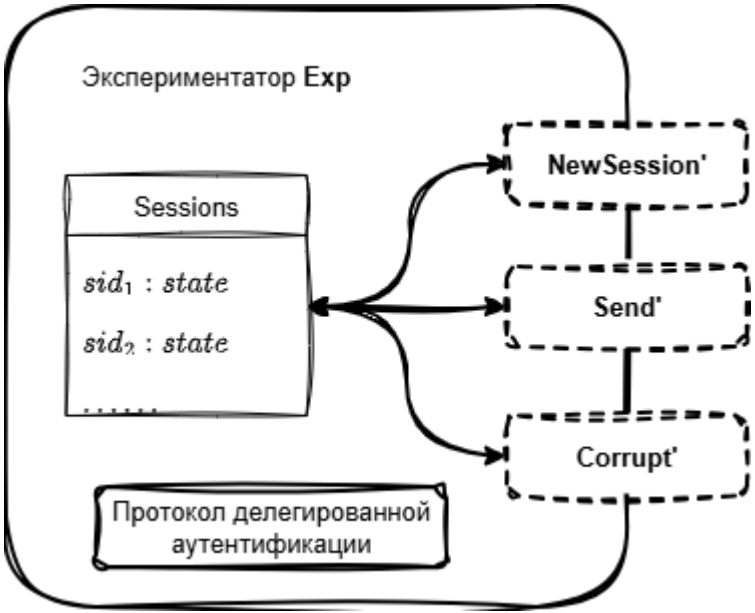
6 : $\ldots$

7 : **return** msg'

Функция обработки

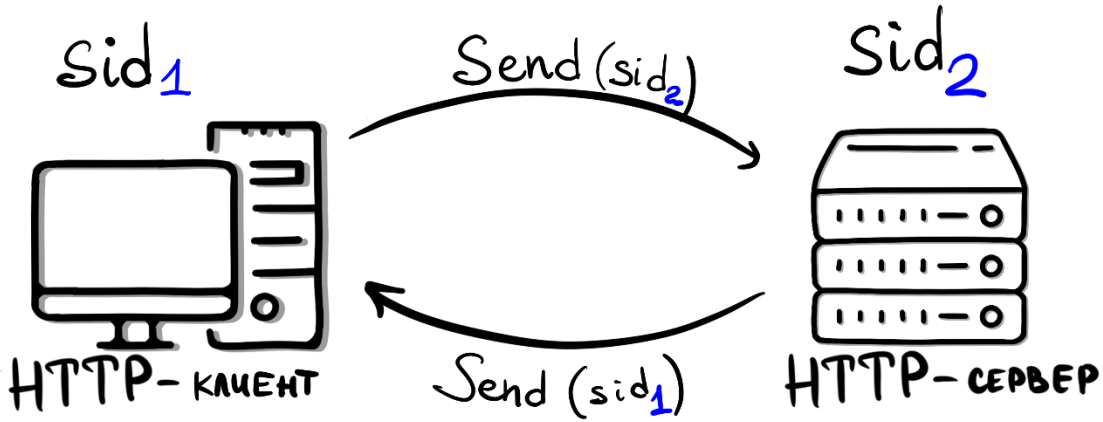
- Все сообщения возвращается нарушителю.
- Нарушитель играет роль посредника между участниками.

Целевые возможности нарушителя для ПДА



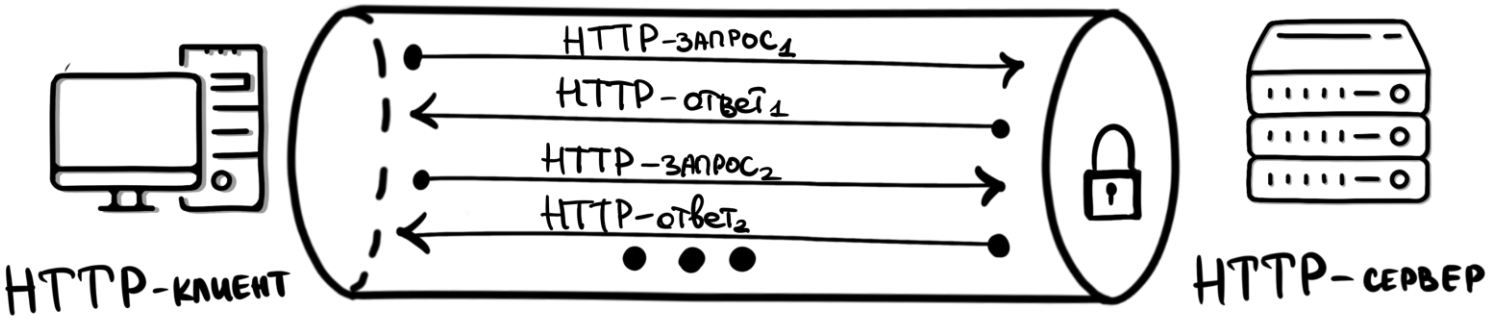
- Создание сессий между участниками.
- Запуск взаимодействия между честными участниками без получения сообщения.
- Задержка и блокировка сообщений.
- Отправление новых сообщений от лица нарушителя.
- Получение сообщений, направленных нарушителю.
- Компрометация участников.

Наивный подход



- Защищённый канал связывает два сеанса sid_1, sid_2.
- Send позволяет передать сообщение из сеанса sid_1 в сеанс sid_2 и наоборот.
- Нарушитель не узнаёт передаваемое сообщение.

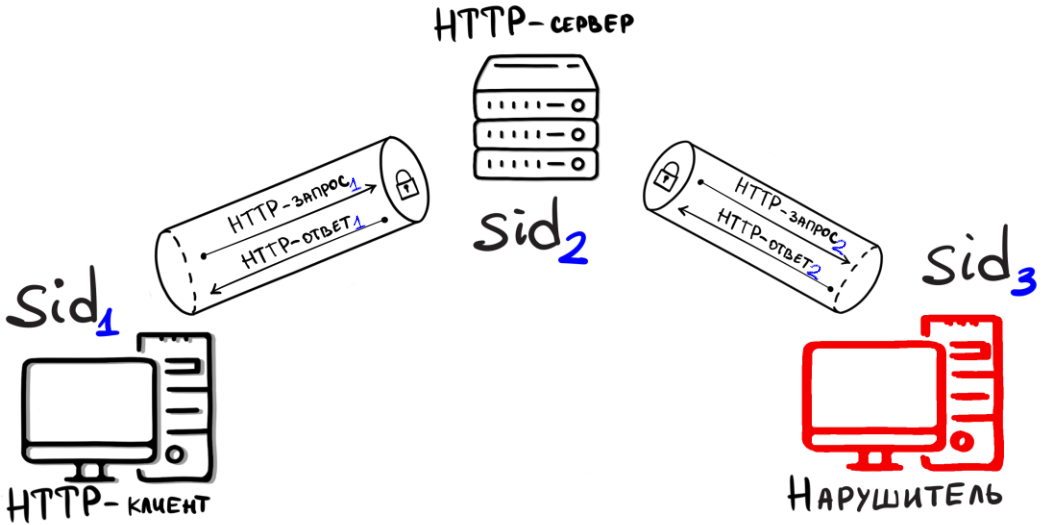
Наивный подход



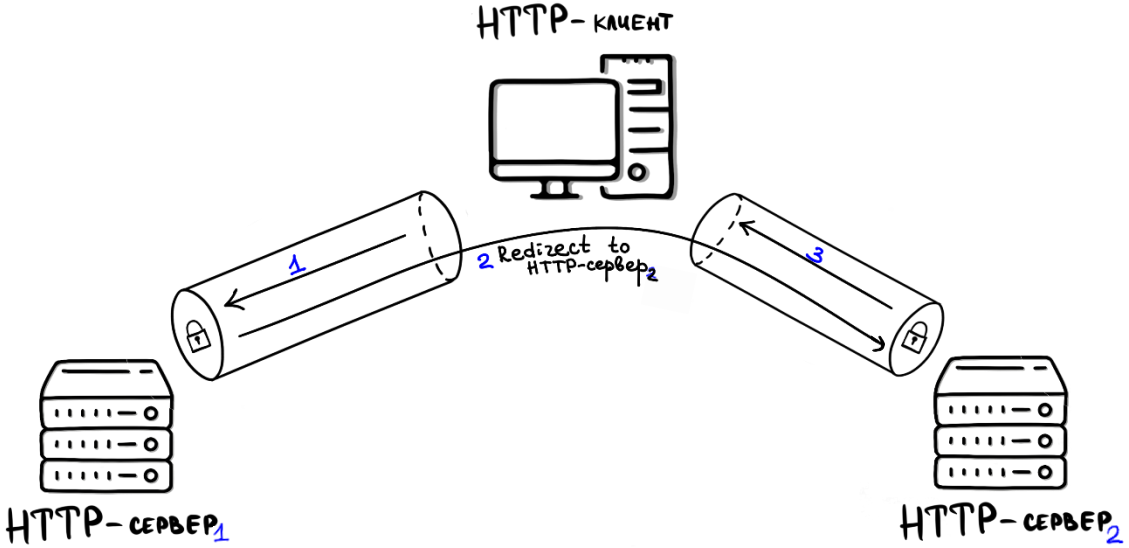
Между сеансами участников строится труба и всё взаимодействие проходит строго по трубе.

Недостатки наивного подхода

- Сеанс, в котором будет обработано сообщение клиента, определяется содержимым HTTP-запроса.
- В общем случае нарушитель имеет возможность отправить такой HTTP-запрос, который будет обработан сервером в сеансе с тем же sid.

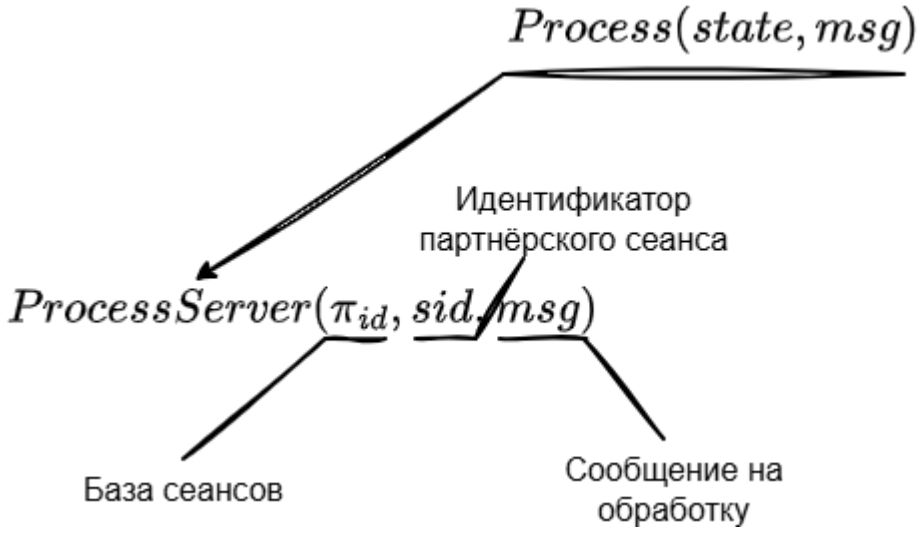


Недостатки наивного подхода



- Для передачи сообщения от HTTP-сервера1 к HTTP-серверу2 используется HTTP механизм Redirect.
- С точки зрения модели, обработка запроса к оракулу Send влияет на две сессии (с HTTP-сервером1 и HTTP-сервером2).

Решение



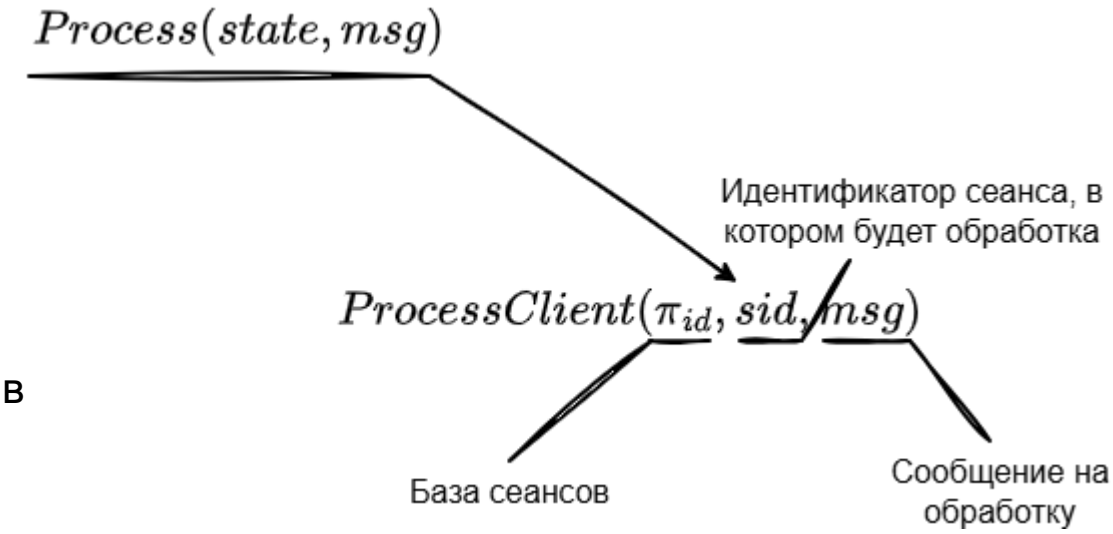
Сервер определяет сеанс, в котором нужно обработать сообщение msg на основе содержания msg ⇒ База сеансов.

Идентификатор партнёрского сеанса позволяет доставить ответ от сервера нужному клиенту ⇒ Идентификатор партнёрского сеанса.

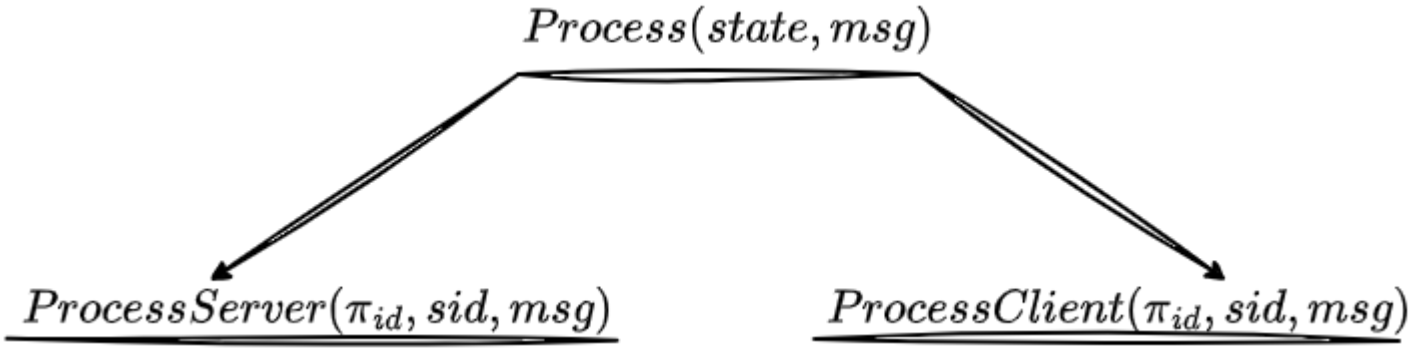
Решение

Обработка одного сообщения может влиять на несколько сеансов ⇒ База сеансов.

Клиент принимает сообщение в конкретном сеансе ⇒ Идентификатор сеанса, в котором будет обработка

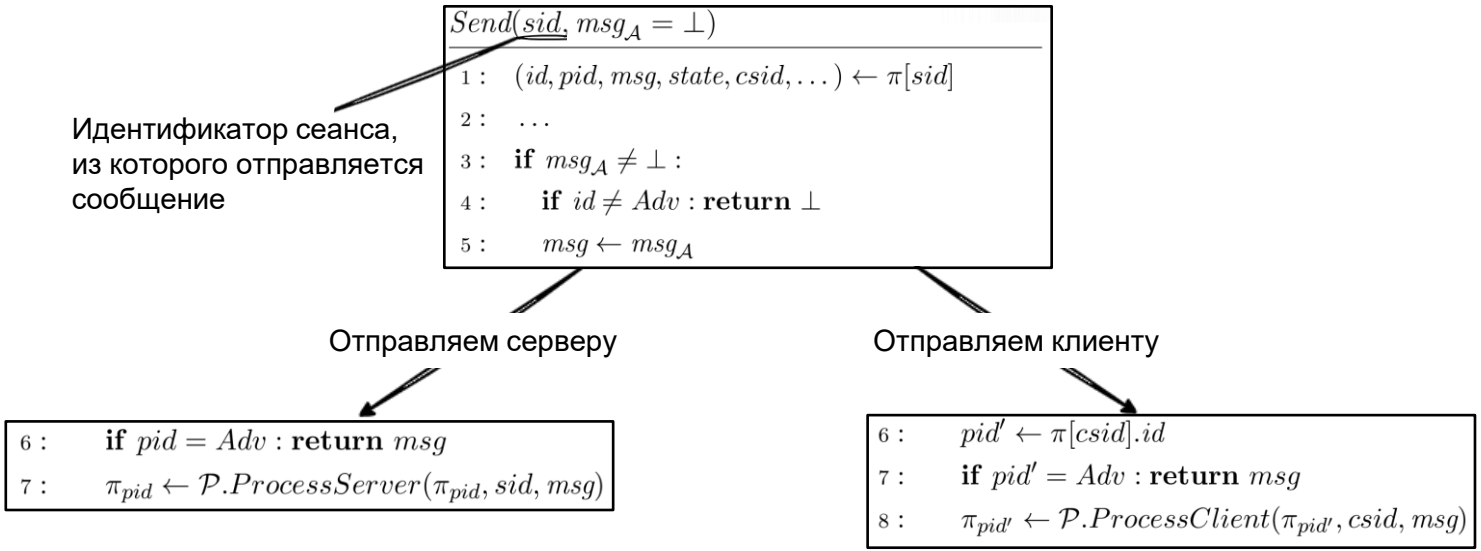


Интерфейс функций обработки



Интерфейсы функций внешне схожи, но смысл и назначение параметров разные

Интерфейс оракула Send



Угрозы безопасности

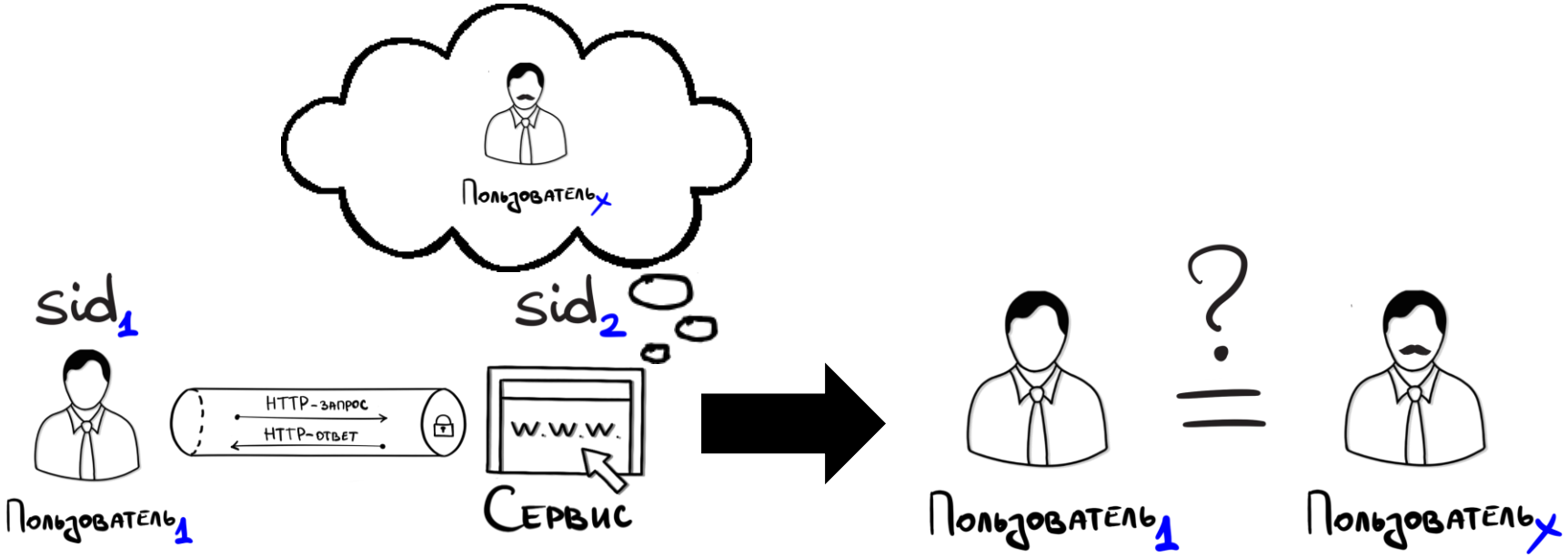
Ложная аутентификация

Успешный вход нарушителя под именем честного пользователя без знания аутентификационных данных этого пользователя на AS.

Некорректная идентификация

Жертва аутентифицируется, но сервис считает, что аутентифицировался другой пользователь.

Удобство формализации угроз безопасности



Результаты и направления будущих исследований

Сделано:

- Формализован класс протоколов делегированной аутентификации на основе HTTP.
- Разработана модель на основе экспериментатора, учитывающая особенности рассматриваемого класса механизмов.
- Проверена релевантность модели с помощью известных атака на уязвимый протокол делегированной аутентификации.

Планируется:

- Анализ OpenID connect в нашей модели.
- Интеграция двусторонней аутентификации в модель.
- Поддержка работы множества серверов аутентификации.
- Анализ механизма при компрометации серверов аутентификации
- …

РусКрипто
XXVIII НАУЧНО-ПРАКТИЧЕСКАЯ КОНФЕРЕНЦИЯ

Спасибо за внимание!

Давид Мурадян
инженер-аналитик, КриптоПро
muradyan@cryptopro.ru

Формализация механизма $\mathcal{P}$

$$\mathcal{P} = \{Initialize(n_u, n_c);$$
$$ProcessServer(\pi_{id}, sid, msg);$$
$$ProcessClient(\pi_{id}, sid, msg)\}$$

Оракулы

NewSession$(id_1, id_2 = \perp, \rho)$

```
1 : if ρ = "HTTP-client" ∧ id_2 = ⊥ :
2 :     return ⊥
3 : ctr ← ctr + 1
4 : id, pid, msg, state, status, csid ←
5 :  ← id_1, ⊥, ⊥, InitState[id_1], "start", ⊥
6 : if ρ = "HTTP-client" :
7 :     pid ← id_2
8 : π[ctr] ← (id, ρ, pid, msg, state,
9 :               status, csid)
10 : return ctr
```

Corrupt(id)

```
1 : if id ∈ AuthSrv :
2 :     return ⊥
3 : Corrupted[id] ← 1
4 : return InitState[id], π_id
```

Send$(sid, msg_{\mathcal{A}} = \perp)$

```
1 : (id, ρ, pid, msg, state,
2 :           status, csid) ← π[sid]
3 : if msg = ⊥ : return ⊥
4 : if status ∈ {"reject", "accept"} : return ⊥
5 : if status ∈ {"start"} :
6 :     π_id ← P.ProcessClient(π_id, ⊥, pid)
7 : if msg_A ≠ ⊥ :
8 :     if id ≠ Adv : return ⊥
9 :     msg ← msg_A
10 : if ρ = "HTTP-client" :
11 :     if pid = Adv : return msg
12 :     π_pid ← P.ProcessServer(π_pid, sid, msg)
13 : if ρ = "HTTP-server" :
14 :     pid' ← π[csid].id
15 :     if pid' = Adv : return msg
16 :     π_pid' ← P.ProcessClient(π_pid', csid, msg)
17 : msg ← ⊥
18 : π[ctr] ← (id, ρ, pid, msg, state,
19 :               status, csid)
20 : return 1
```

Эксперимент

$\mathbf{Exp}^{Model}_{\mathcal{P},auth}$

1 : $ctr \leftarrow -1$
2 : $\pi \leftarrow \perp$
3 : $Corrupted \leftarrow \perp$
4 : User, Client, AuthSrv,
5 : $\quad InitState, InitState_{pub} \leftarrow Initialize(n_U, n_C)$
6 : **for** $U_i \in$ User :
7 : $\quad Corrupted[U_i] = 0$
8 : **for** $C_i \in$ Client :
9 : $\quad Corrupted[C_i] = 0$
10 : $Corrupted[AS_i] = 0$
11 : $Corrupted[Adv] = 1$
12 : $\mathcal{A}^{Corrupt,NewSession,Send}$(User, Client, AuthSrv, $InitState_{pub}$)
13 : **if** $auth(\pi)$
14 : $\quad$ **return** 1
15 : **else**
16 : $\quad$ **return** 0

Предикат угрозы

Определение 4. Таблица всех сеансов π в рамках некоторого эксперимента удовлетворяет предикату $auth$, если $\exists$ sid, для которого одновременно выполнены следующие условия:

1. $id \in \mathsf{Client}$, $pid \in \mathsf{User}$, $\rho =$ "HTTP-server", $\pi[csid].id \in \mathsf{User}$. Далее примем обозначения $client = id$, $result = pid$, $user = \pi[csid].id$.
2. $(Corrupted[user] = 0) \vee (Corrupted[result] = 0)$.
3. $Corrupted[client] = 0$
4. $result \neq user$.