Принципы построения отечественных криптонаборов для TLS 1.2

Алексеев Евгений Константинович,
к.ф-.м.н., начальник отдела криптографических исследований

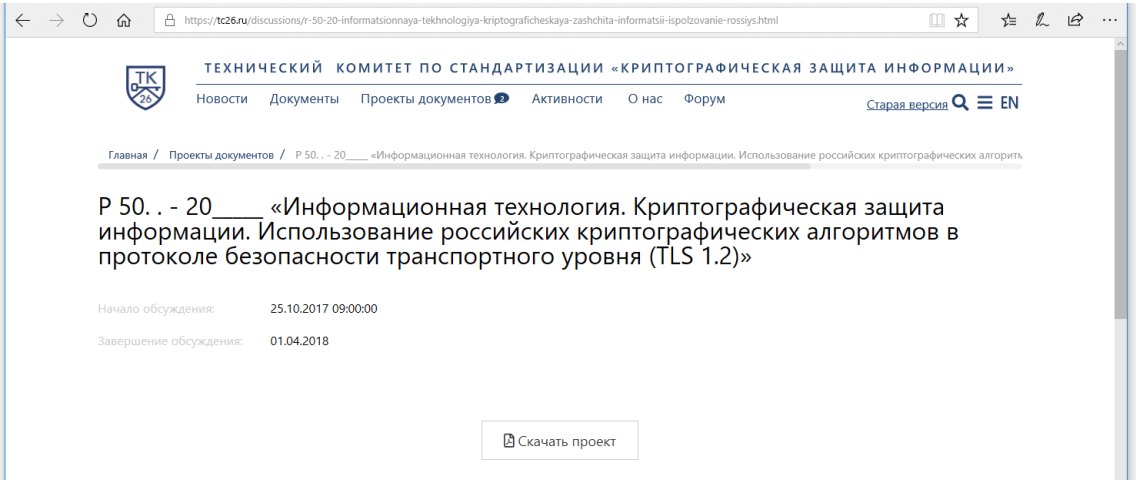
Смышляева Екатерина Сергеевна,
инженер-аналитик

КРИПТОПРО

РусКрипто’2018

Февраль 2017—март 2018

Разработка проекта рекомендаций по стандартизации, описывающего криптонаборы для TLS 1.2 на базе российских алгоритмов



КРИПТОПРО

Необходимость разработки

КРИПТОПРО

Необходимость разработки

Принципы построения

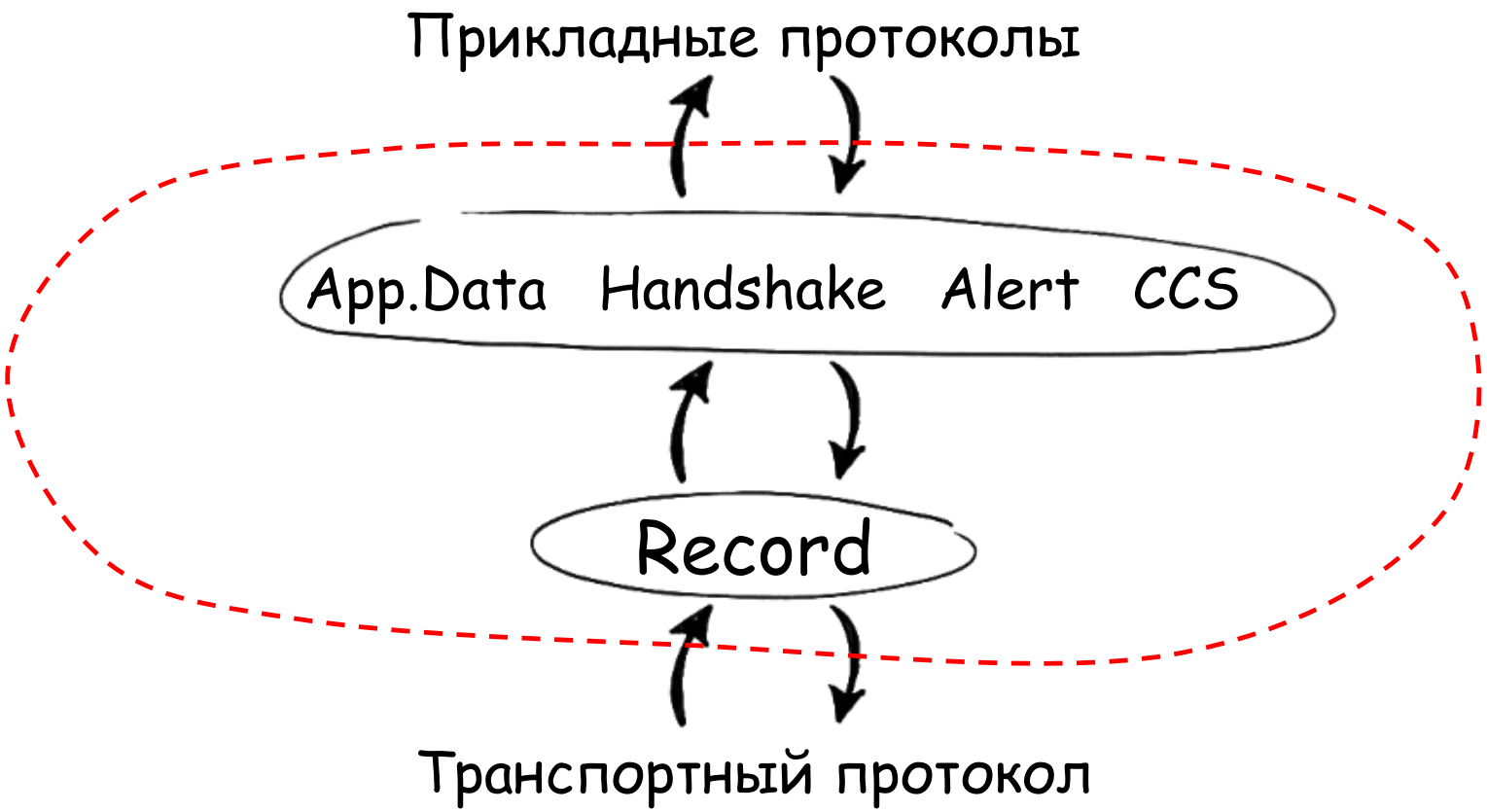
КРИПТОПРО

Что такое TLS

TLS (Transport Layer Security) предназначен для создания и обеспечения защищенного канала связи. Основан на спецификации SSL 3.0, но за время своего существования претерпел довольно много изменений.

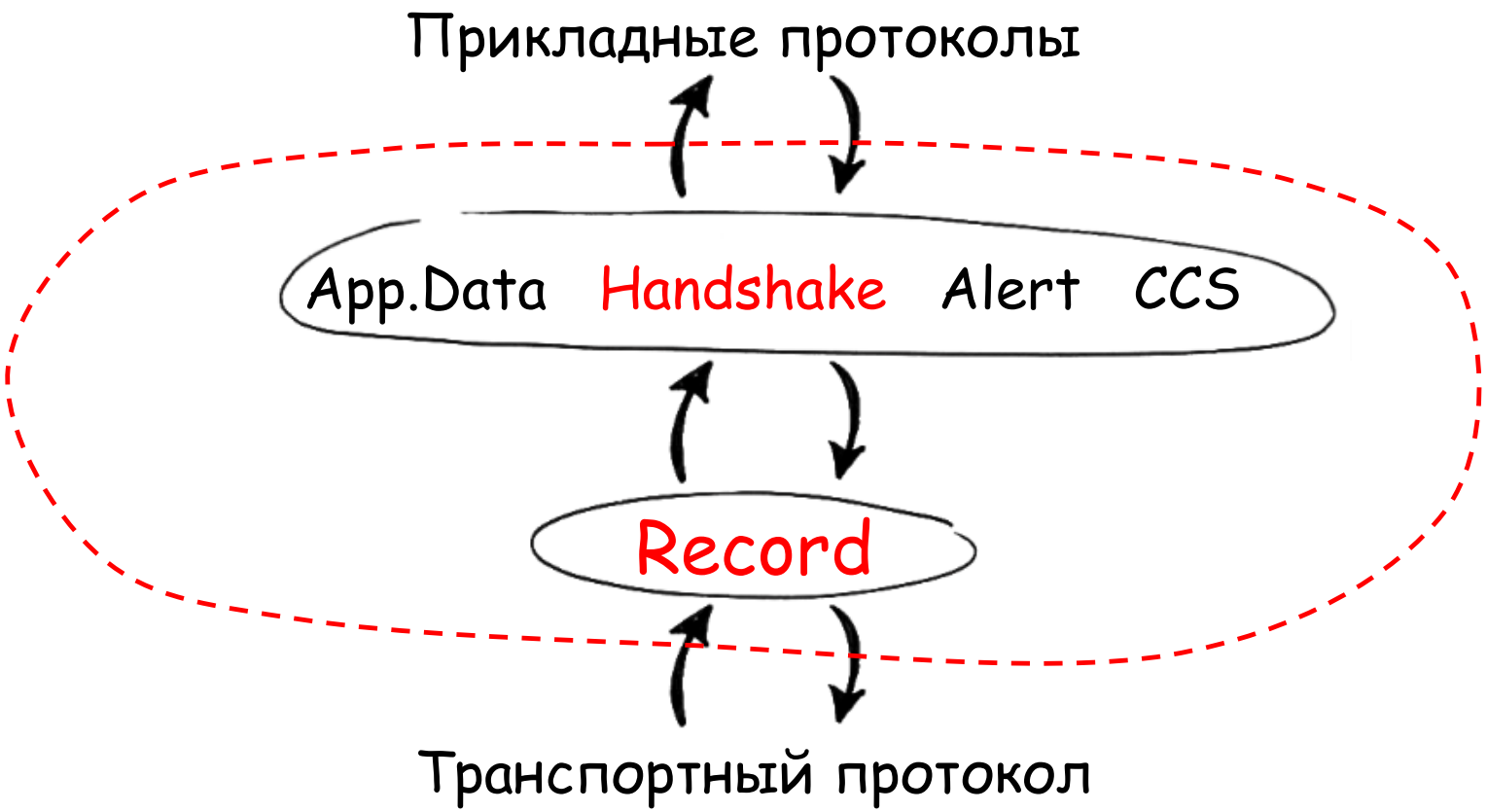
КРИПТОПРО

Что такое TLS



КРИПТОПРО

Что такое TLS



КРИПТОПРО

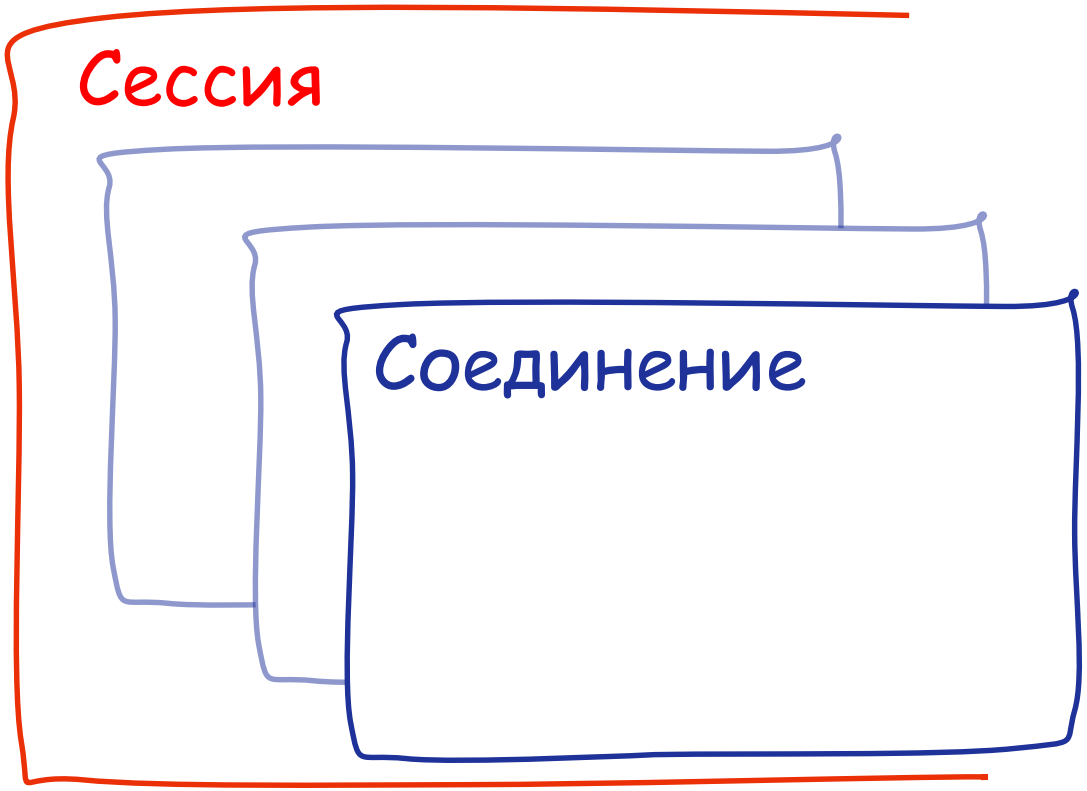
Что такое TLS

аутентификация сторон, выработка параметров безопасности

защита данных (записей)

КРИПТОПРО

Что такое TLS



КРИПТОПРО

Что такое TLS

Сессия

- *ID* сессии
- $cert_S$, $cert_C$
- CipherSuite (криптонабор)
- секретное значение *MS*

КРИПТОПРО

Что такое TLS

Соединение

- случайные значения r_S , r_C
- ключевой материал

КРИПТОПРО

Криптонаборы

При наличии фиксированного каркаса (RFC 5246) режимы работы протокола TLS имеют большую степень вариативности:

- способ выработки общего секрета PS;
- способ аутентификации;
- используемые криптопримитивы;
- большое количество опциональных расширений;
- и т.д.

Данные режимы определяются так называемыми *криптонаборами* (ciphersuite)

Криптонаборы

ТК 26 2013 год, старые криптонаборы:

- TLS_GOSTR341112_256_WITH_28147_CNT_IMIT;
- TLS_GOSTR341112_256_WITH_NULL_GOSTR3411;
- TLS_GOSTR341001_WITH_28147_CNT_IMIT;
- TLS_GOSTR341001_WITH_NULL_GOSTR3411.

КРИПТОПРО

Криптонаборы

ТК 26 2013 год, старые криптонаборы:

- TLS_GOSTR341112_256_WITH_28147_CNT_IMIT;
- TLS_GOSTR341112_256_WITH_NULL_GOSTR3411;
- TLS_GOSTR341001_WITH_28147_CNT_IMIT;
- TLS_GOSTR341001_WITH_NULL_GOSTR3411.

ТК 26 2017-2018 год, новые криптонаборы:

- TLS_GOSTR341112_256_WITH_KUZNYECHIK_CTR_OMAC;
- TLS_GOSTR341112_256_WITH_MAGMA_CTR_OMAC.

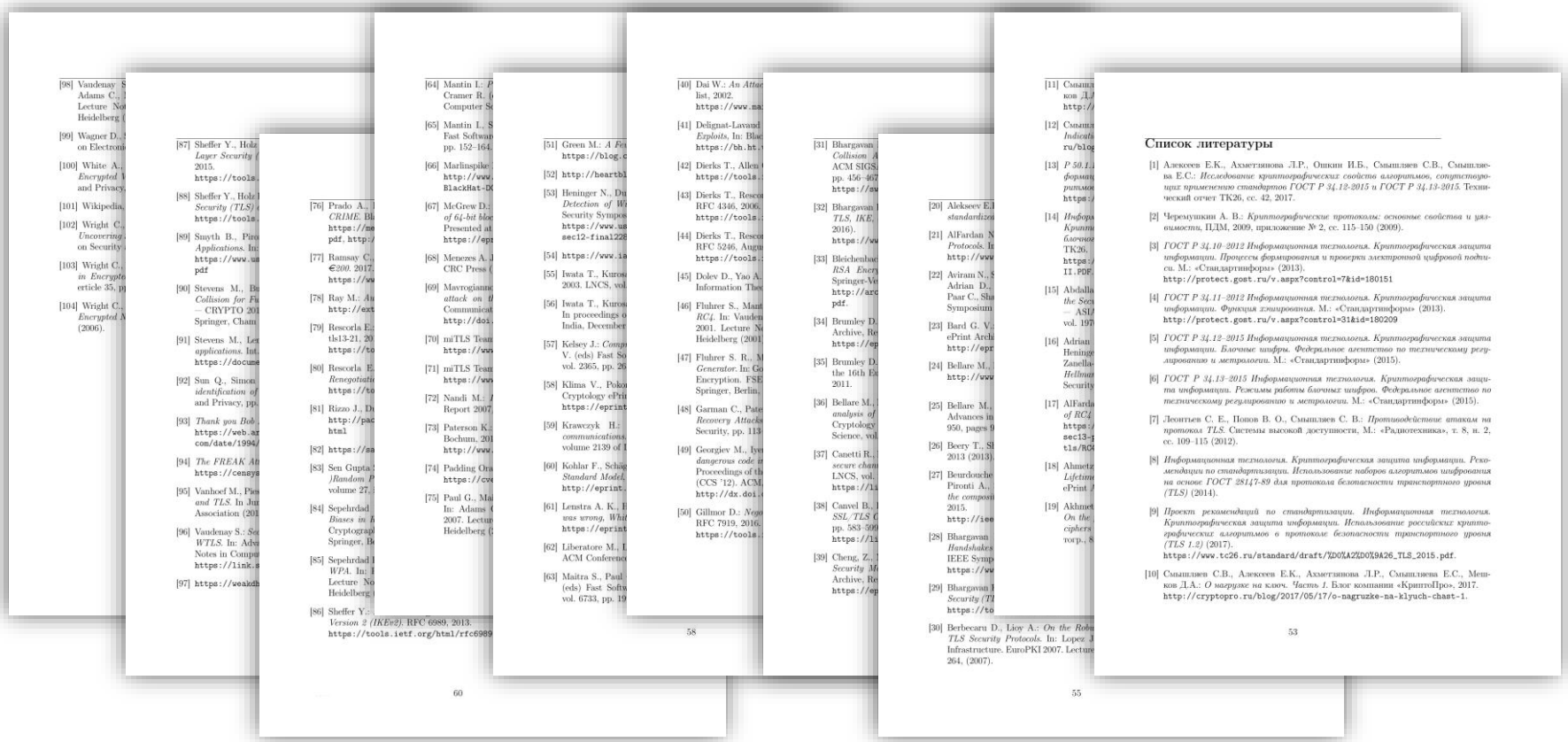
КРИПТОПРО

Причины разработки новых криптонаборов

1. В 2015 году вышли новые стандарты: блочные шифры Кузнечик, Магма и режимы их работы. Было необходимо интегрировать эти стандарты в новые криптонаборы.
2. Создание максимально самодостаточного и полного описания протокола TLS.
3. Догнать актуальную версию TLS 1.2 и учесть все уязвимости, известные на сегодняшний момент.

КРИПТОПРО

Причины разработки новых криптонаборов



уязвимости, известные на сегодняшний момент

КРИПТОПРО

Record

КРИПТОПРО

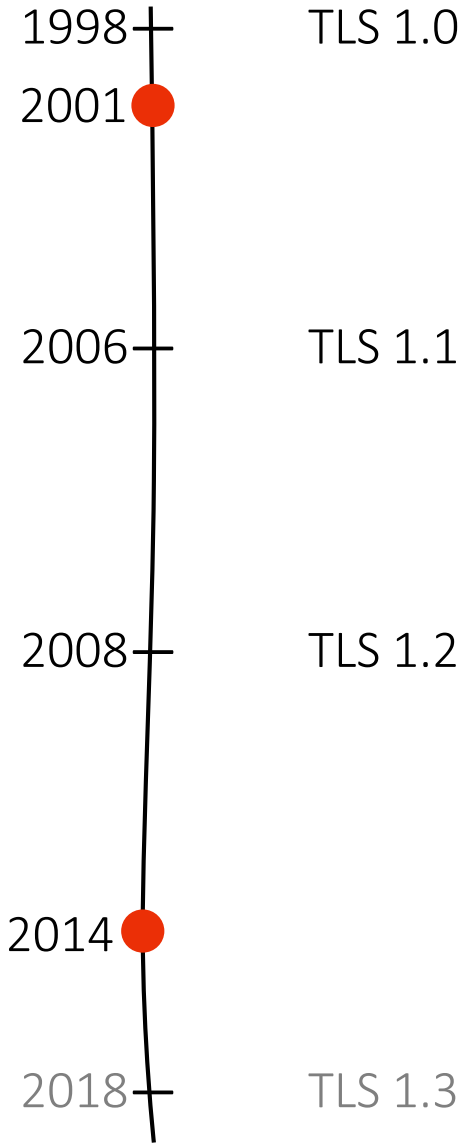
Атаки на режим CBC

КРИПТОПРО

Record/ Атака Воденея (POODLE)

Атака на режим CBC с использованием дополнения PKCS#5.

Противнику требуется после отправки модифицированного пакета данных получать информацию о результате проверки дополнения.

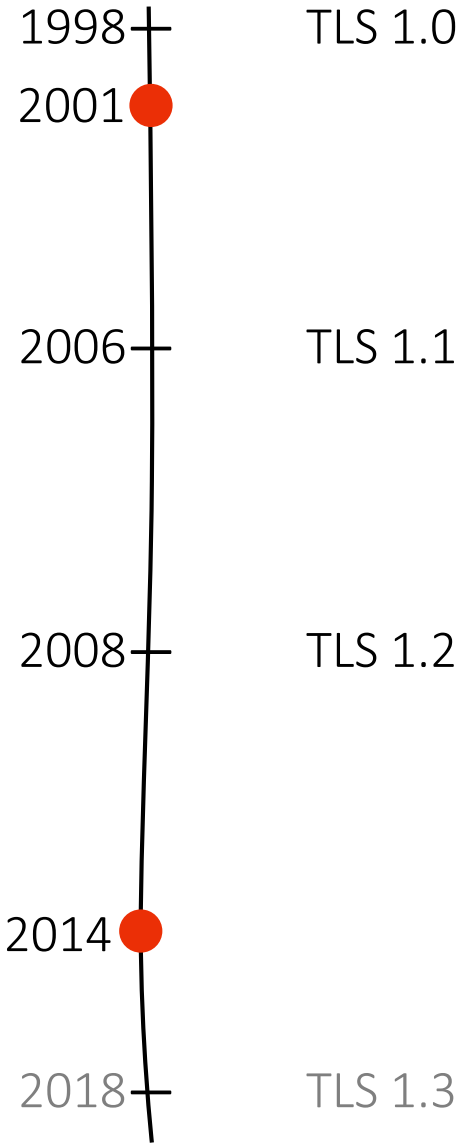


КРИПТОПРО

Record/ Атака Воденея (POODLE)

TLS 1.0

В случае некорректного дополнения посылалось специальное оповещение об ошибке decryption_failed.



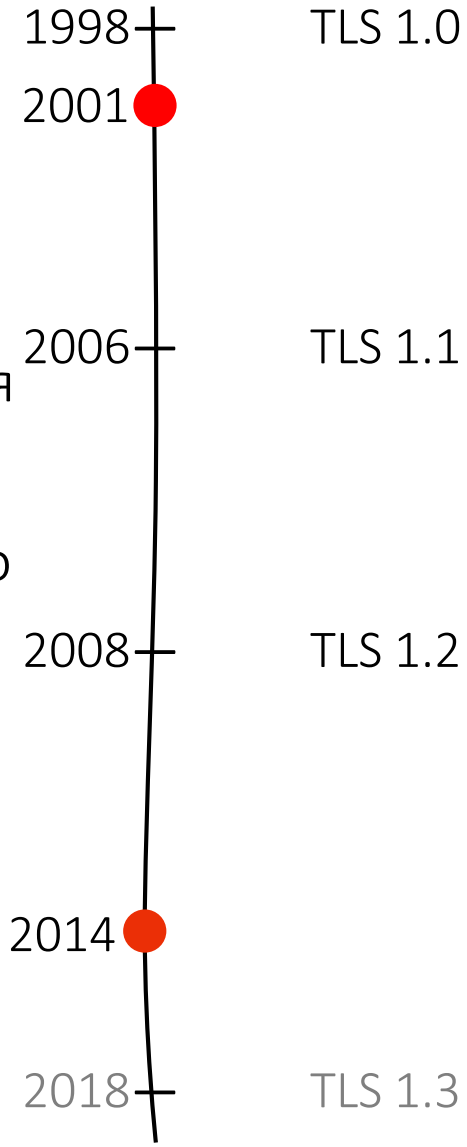
КРИПТОПРО

Record/ Атака Воденея (POODLE)

TLS 1.1

Запрещено использовать оповещение *decryption_failed*, вместо него всегда посылается оповещение *bad_record_mac*.

Опубликована модификация атаки: предложено различать типы ошибок по времени реакции сервера на модифицированный шифртекст.

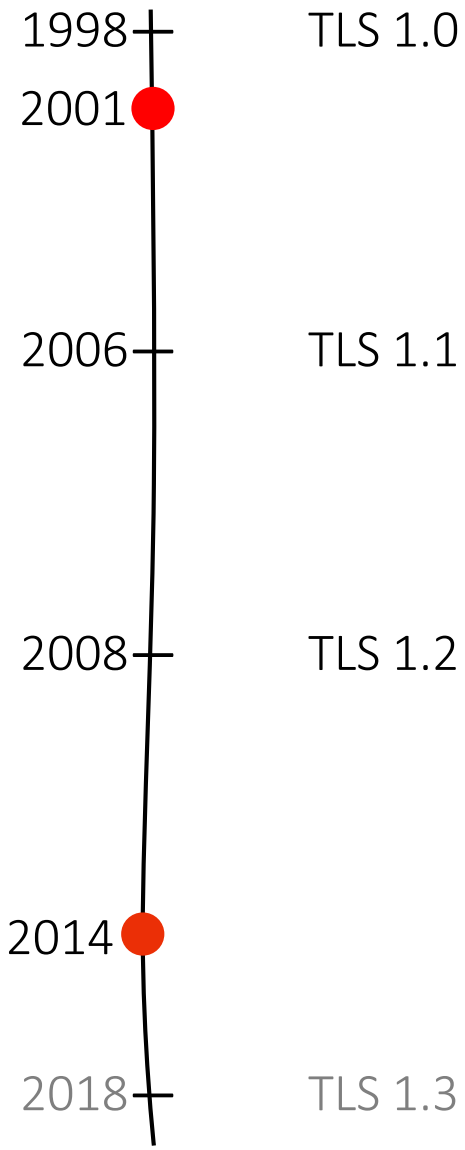


КРИПТОПРО

Record/ Атака Воденея (POODLE)

TLS 1.2

Для предотвращения временной атаки в TLS 1.2 предлагалось вычислять значение имитовставки даже в случае некорректного дополнения.



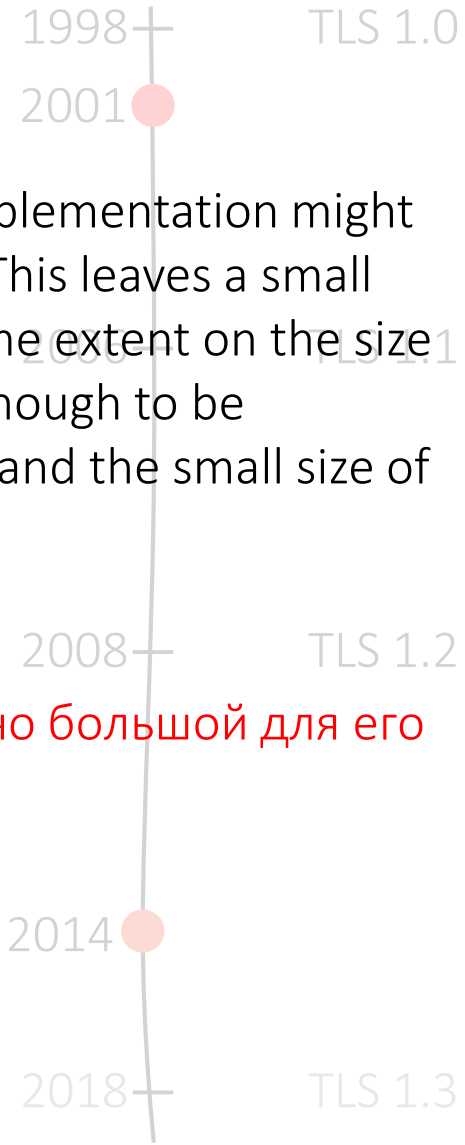
КРИПТОПРО

Record/ Атака Воденея (POODLE)

RFC 5246, пункт 6.2.3.2:

«For instance, if the pad appears to be incorrect, the implementation might assume a zero-length pad and then compute the MAC. This leaves a small timing channel, since MAC performance depends to some extent on the size of the data fragment, but it is not believed to be large enough to be exploitable, due to the large block size of existing MACs and the small size of the timing signal.»

Предполагается, что временной канал недостаточно большой для его использования.

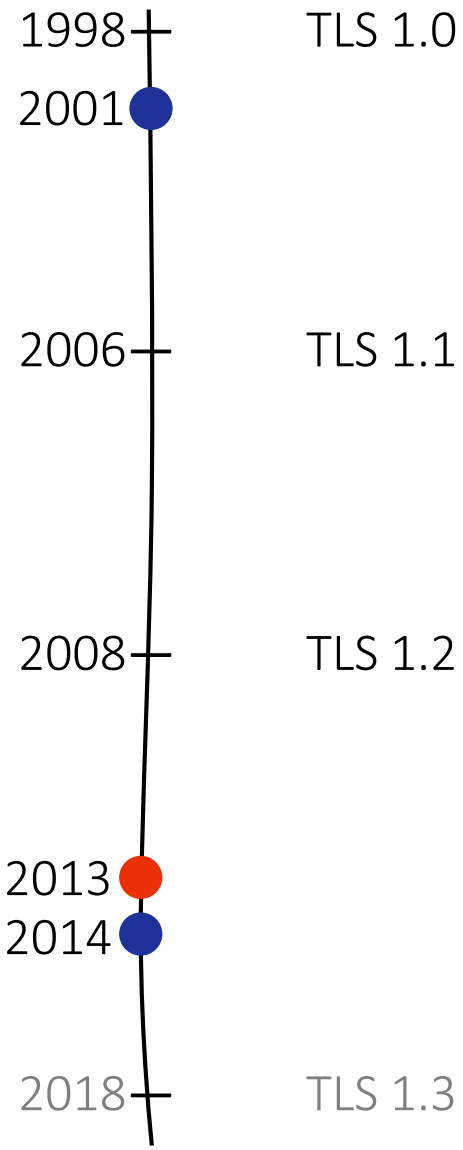


КРИПТОПРО

Record/ Атака Воденея (POODLE)/ Lucky13

Lucky13 является «атакой на контрмеры», предпринятые для противостояния атаке Воденея.

В атаке используется побочный канал по времени, возникающий за счет разницы в количестве вычислений функций сжатия SHA-1 для случая корректного и некорректного дополнения при вычислении имитовставки по алгоритму HMAC-SHA-1.



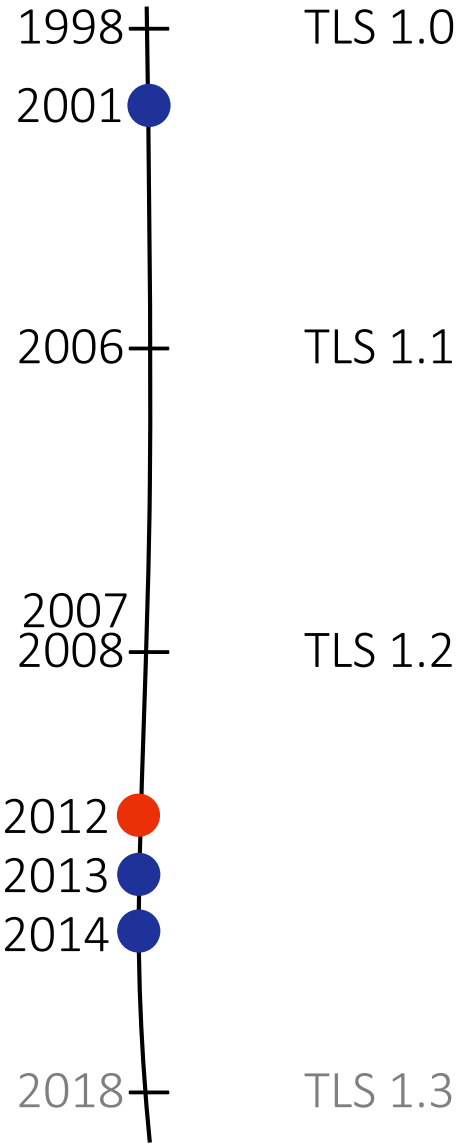
КРИПТОПРО

Record/ Атака Воденея (POODLE)/ Расширение побочного канала

Годом ранее на конференции Рускрипто’2012 специалистами КриптоПро была предложена другая атака по времени, направленная на расширение побочного канала.

RFC 5246, пункт 6.2.3.2:

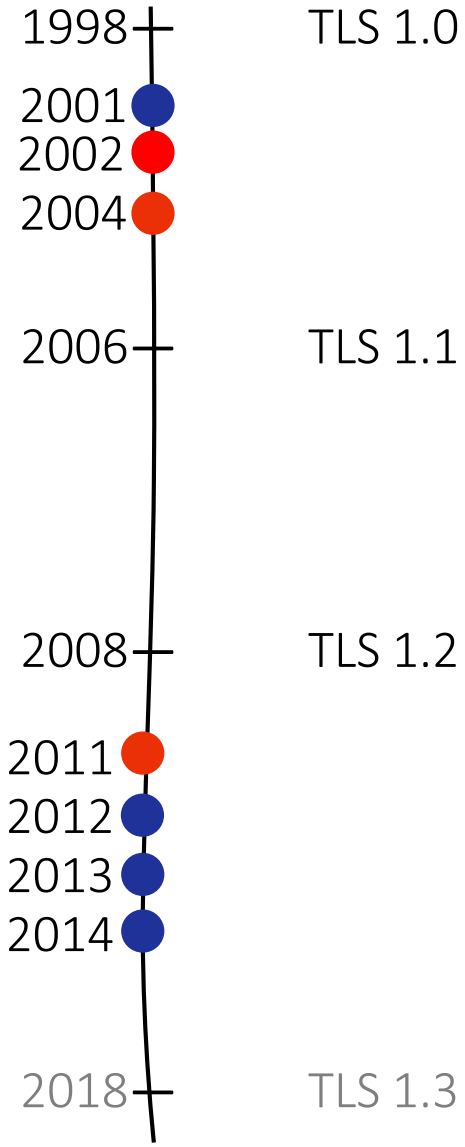
Дополнение может иметь любую длину, не превосходящую 255 байт.



КРИПТОПРО

Record/ Атака BEAST

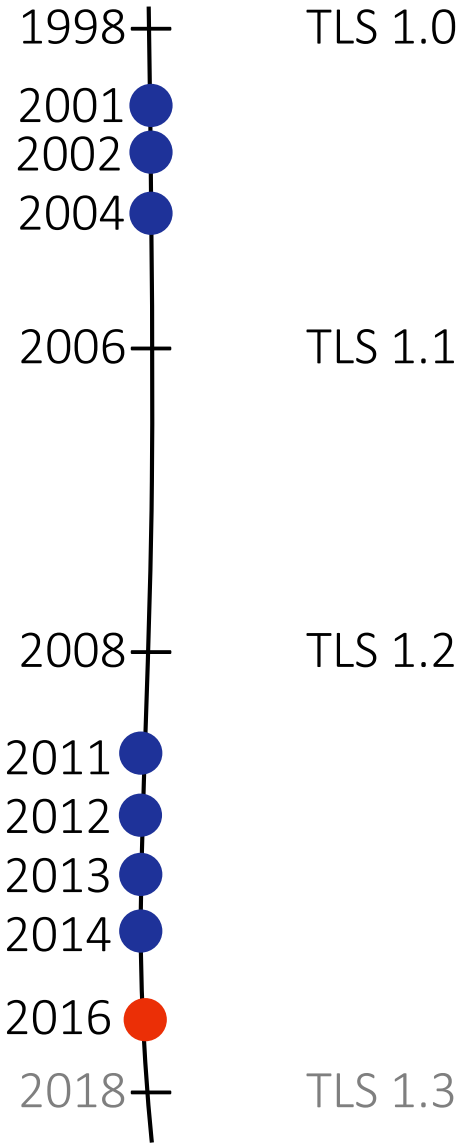
Если в режиме CBC вектор инициализации выбирается предсказуемым для противника способом (TLS 1.0, SSL 3.0), можно реализовать атаку BEAST.



КРИПТОПРО

Record/ Атака Sweet32

Атака на режим CBC с использованием произвольного 64-битного. Базируется на уязвимости режима CBC к атакам на основе парадокса задачи о днях рождения.



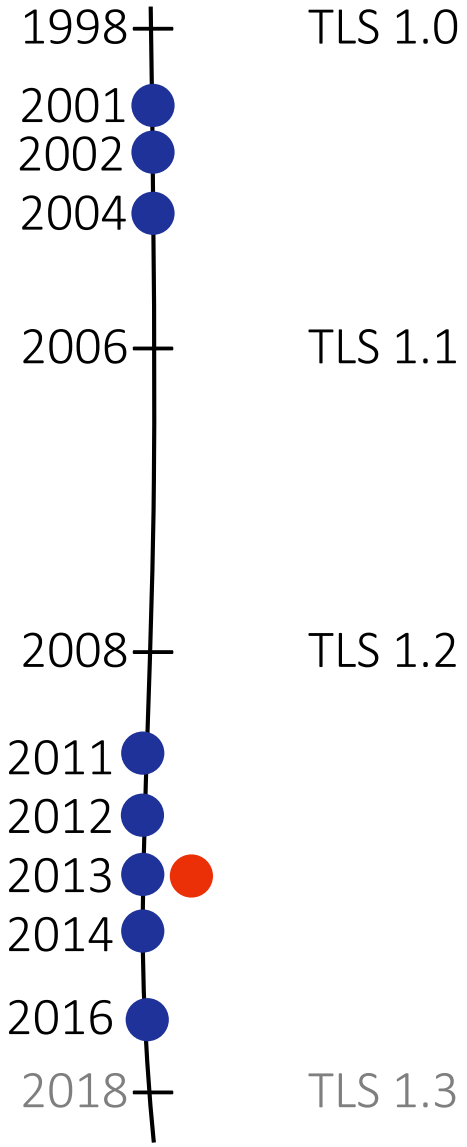
КРИПТОПРО

Одной из альтернатив для предотвращения атак, перечисленных выше, является отказ от использования режима CBC. Например, использование потокового шифра (RC4).

КРИПТОПРО

Record/ Атака Аль-Фардана и др.

Атака основывается на наличии методов, демонстрирующих существенное отклонение распределения выходной гаммы потокового шифра RC4 от равномерного распределения.



КРИПТОПРО

И новые, и старые российские криптонаборы

- не используют режим CBC;
- не используют дополнение;
- используют шифры и режимы их работы, для которых не известны какие-либо эффективно реализуемые уязвимости.

КРИПТОПРО

Record/ Новые криптонаборы

Новый протокол Record RTLS не похож ни на предыдущую отечественную, ни на зарубежные версии. В нем реализованы все передовые достижения, касающиеся задач обеспечения защищенного канала связи и минимизации нагрузки на ключ.

КРИПТОПРО

Record/ Новые криптонаборы

Ключи образуют иерархию:

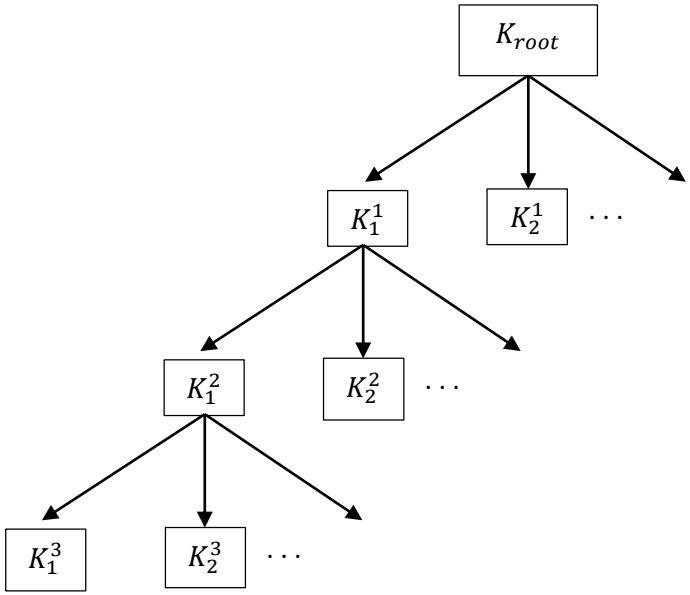
- корневой ключ;
- ключи промежуточных уровней;
- ключи обработки записей;
- секционные ключи.

Создание такой иерархии достигается за счет использования механизмов внешнего и внутреннего преобразования ключей (external и internal re-keying), которые позволяют увеличивать объем обрабатываемых данных, оставаясь в рамках безопасной нагрузки на ключ.

Re-keying Mechanisms for Symmetric Keys
draft-irtf-cfrg-re-keying-12

КРИПТОПРО

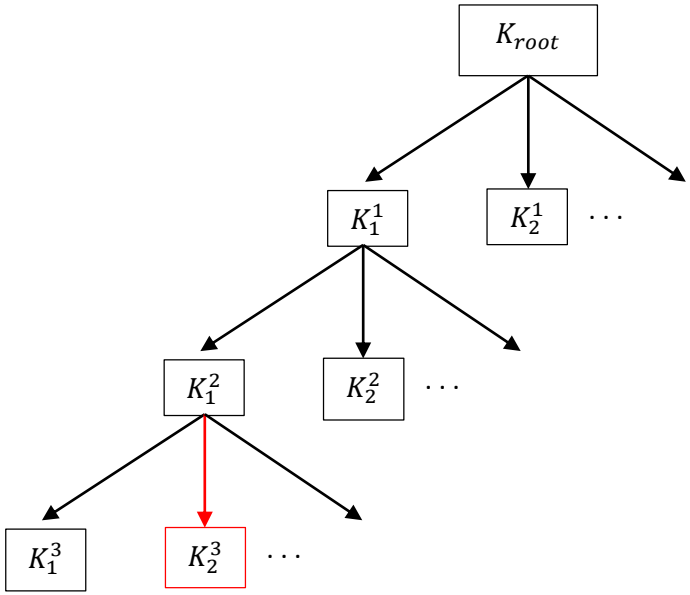
Record/ Новые криптонаборы/External re-keying (TLSTREE)



$$TLSTREE(K_{root}, i) = Divers_3(Divers_2(Divers_1(K_{root}, i \,\&\, C_1), i \,\&\, C_2), i \,\&\, C_3)$$

КРИПТОПРО

Record/ Новые криптонаборы/External re-keying (TLSTREE)



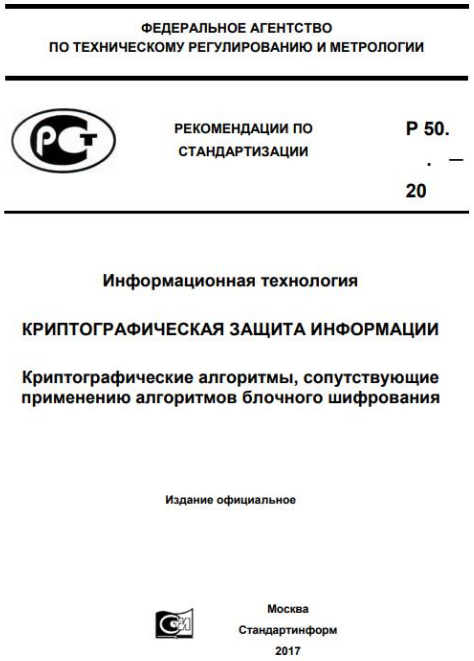
$$TLSTREE(K_{root}, i) = Divers_3(Divers_2(Divers_1(K_{root}, i \,\&\, C_1), i \,\&\, C_2), i \,\&\, C_3)$$

В целях эффективности и противодействия атакам по побочным каналам обращение к функции $Divers_j$, $j \in \{1,2,3\}$, должно производиться, **ТОЛЬКО** когда $seqnum \,\&\, C_j \neq (seqnum - 1) \,\&\, C_j$.

КРИПТОПРО

Record/ Новые криптонаборы/Internal re-keying (CTR-ACPKM)

В новых криптонаборах используется режим работы блочного шифра с преобразованием ключа CTR-ACPKM, описанный в проекте рекомендаций по стандартизации «Информационная технология. Криптографическая защита информации. Криптографические алгоритмы, сопутствующие применению алгоритмов блочного шифрования».



ФЕДЕРАЛЬНОЕ АГЕНТСТВО
ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

РЕКОМЕНДАЦИИ ПО СТАНДАРТИЗАЦИИ

Р 50. . –
20

Информационная технология

КРИПТОГРАФИЧЕСКАЯ ЗАЩИТА ИНФОРМАЦИИ

Криптографические алгоритмы, сопутствующие применению алгоритмов блочного шифрования

Издание официальное

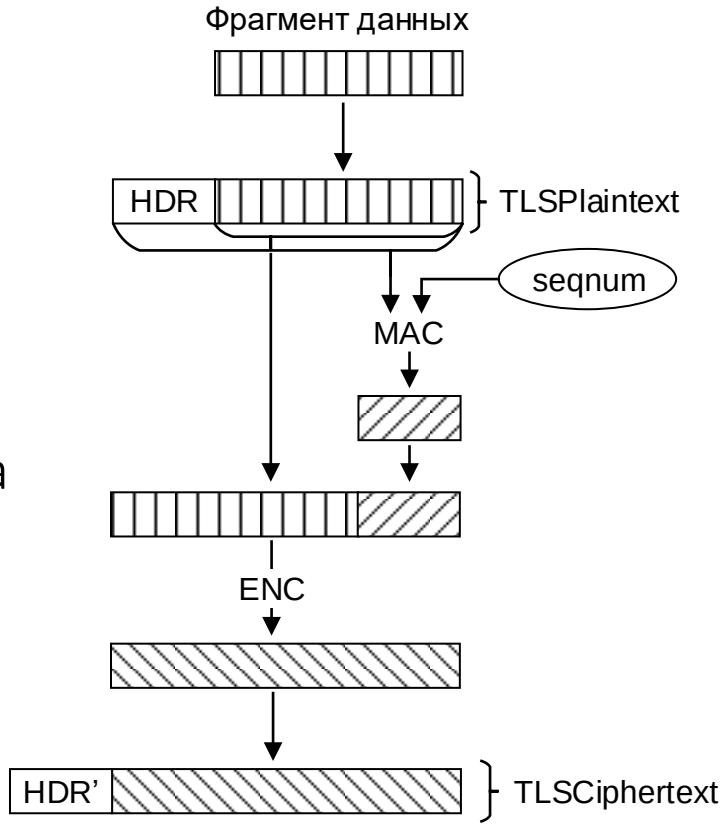
Москва
Стандартинформ
2017

КРИПТОПРО

Record/ Новые криптонаборы/ Authenticate-then-Encrypt

Используется Authenticate-then-Encrypt схема защиты данных:

- Соответствует RFC 5246.
- Схема AtE является безопасной в случае использования потокового режима.
- При анализе быстродействия и удобства реализации можно заметить, что варианты AtE и EtA с режимом CTR практически равноценны.



КРИПТОПРО

Handshake

КРИПТОПРО

Handshake

Основные принципы работы протокола Рукопожатия в новых криптонаборах почти не изменились по сравнению с предыдущей версией отечественных криптонаборов.

КРИПТОПРО

Client (C)	Сообщения	Server (S)
Выработка случайного значения r_C	ClientHello r_C, RI_C →	
	← ServerHello r_S, RI_S	Выработка случайного значения r_S
Этап выработки общего секрета		
	← Certificate $Cert_S$ (содержит Q_S)	
	← - - - CertificateRequest	
	← ServerHelloDone	
	- - - Certificate $Cert_C$ (содержит Q_C) →	
Выработка случайного значения PS (32 байта)		
Эфемерный ключ на кривой, соответствующей Q_S: $Q_{EPH} = k_{EPH}P$, где k_{EPH} выбирается случайно.		
$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_{EPH}, Q_S, H)$		
Экспортное представление общего секрета PS: $IV = H[25..24 + n/2];$ $PSExp = KExp15(PS, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$	ClientKeyExchange Q_{EPH},$PSExp$ →	$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_S, Q_{EPH}, H)$
		Извлечение общего секрета из экспортного представления: $IV = H[25..24 + n/2];$ $PS = KImp15(PSExp, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$
$sgn_C = SIGN_{k_C}(HM)$	CertificateVerify - - - sgn_C →	Проверка sgn_C
Этап выработки ключей из значения *PS*		
$MS = PRF(PS, label_1, HASH(HM))$		$MS = PRF(PS, label_1, HASH(HM))$;
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$
Этап подтверждения ключа		
$client\_verify\_data = PRF(MS, label_3, HASH(HM));$	ChangeCipherSpec → Finished $client\_verify\_data$ →	
	← ChangeCipherSpec ← Finished $server\_verify\_data$	$server\_verify\_data = PRF(MS, label_3, HASH(HM));$

Client (C)	Сообщения	Server (S)
Выработка случайного значения r_C	ClientHello r_C, RI_C →	
	← ServerHello r_S, RI_S	Выработка случайного значения r_S
Этап выработки общего секрета		
	← Certificate $Cert_S$ (содержит Q_S)	
	← CertificateRequest	
	← ServerHelloDone	
	Certificate → $Cert_C$ (содержит Q_C)	
Выработка случайного значения PS (32 байта)		
Эфемерный ключ на кривой, соответствующей Q_S: $Q_{EPH} = k_{EPH}P$, где k_{EPH} выбирается случайно.		
$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_{EPH}, Q_S, H)$		
Экспортное представление общего секрета PS: $IV = H[25..24 + n/2];$ $PSExp = KExp15(PS, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$	ClientKeyExchange → Q_{EPH},$PSExp$	$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_S, Q_{EPH}, H)$
		Извлечение общего секрета из экспортного представления: $IV = H[25..24 + n/2];$ $PS = KImp15(PSExp, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$
$sgn_C = SIGN_{k_C}(HM)$	CertificateVerify → sgn_C	Проверка sgn_C
Этап выработки ключей из значения *PS*		
$MS = PRF(PS, label_1, HASH(HM))$		$MS = PRF(PS, label_1, HASH(HM))$;
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$
Этап подтверждения ключа		
$client\_verify\_data = PRF(MS, label_3, HASH(HM));$	ChangeCipherSpec → Finished → $client\_verify\_data$	
	← ChangeCipherSpec ← Finished $server\_verify\_data$	$server\_verify\_data = PRF(MS, label_3, HASH(HM));$

Опциональные сообщения (для случай двусторонней аутентификации).

Client (C)	Сообщения	Server (S)
Выработка случайного значения r_C	ClientHello r_C, RI_C →	
	ServerHello r_S, RI_S ←	Выработка случайного значения r_S
Этап выработки общего секрета		
Секретный параметр	Certificate $Cert_S$ (содержит Q_S) ←	
	CertificateRequest ← - - -	
	ServerHelloDone ←	
	Certificate $Cert_C$ (содержит Q_C) - - - →	
Выработка случайного значения PS (32 байта)		Открытый ключ сервера
Эфемерный ключ на кривой, соответствующей Q_S: $Q_{EPH} = k_{EPH}P$, где k_{EPH} выбирается случайно.		
$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_{EPH}, Q_S, H)$		
Экспортное представление общего секрета PS: $IV = H[25..24 + n/2];$ $PSExp = KExp15(PS, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$	ClientKeyExchange $Q_{EPH}, PSExp$ →	$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_S, Q_{EPH}, H)$
		Извлечение общего секрета из экспортного представления: $IV = H[25..24 + n/2];$ $PS = KImp15(PSExp, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$
$sgn_C = SIGN_{k_C}(HM)$	CertificateVerify sgn_C - - - →	Проверка sgn_C
Этап выработки ключей из значения PS		
$MS = PRF(PS, label_1, HASH(HM))$		$MS = PRF(PS, label_1, HASH(HM))$
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$
Этап подтверждения ключа		
$client\_verify\_data = PRF(MS, label_3, HASH(HM))$	ChangeCipherSpec → Finished → $client\_verify\_data$	
	ChangeCipherSpec ← Finished ← $server\_verify\_data$	$server\_verify\_data = PRF(MS, label_3, HASH(HM))$

Секретный параметр — premaster secret (PS) — вырабатывается клиентом и передается серверу зашифрованным с помощью его открытого ключа Q_S.

Client (C)	Сообщения	Server (S)
Выработка случайного значения r_C	ClientHello r_C, RI_C →	
	← ServerHello r_S, RI_S	Выработка случайного значения r_S
Этап выработки общего секрета		
	← Certificate $Cert_S$ (содержит Q_S) CertificateRequest ServerHelloDone	
	 $Cert_C$ (содержит Q_C)	
Выработка случайного значения PS (32 байта)		
Эфемерный ключ на кривой, соответствующей Q_S: $Q_{EPH} = k_{EPH}P$, где k_{EPH} выбирается случайно.		
$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_{EPH}, Q_S, H)$		
Экспортное представление общего секрета PS: $IV = H[25..24 + n/2];$ $PSExp = KExp15(PS, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$	ClientKeyExchange Q_{EPH},$PSExp$ →	$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(\boldsymbol{k_S}, Q_{EPH}, H)$
		Извлечение общего секрета из экспортного представления: $IV = H[25..24 + n/2];$ $\boldsymbol{PS} = KImp15(PSExp, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$
$sgn_C = SIGN_{k_C}(HM)$	CertificateVerify sgn_C -->	Проверка sgn_C
Этап выработки ключей из значения *PS*		
$MS = PRF(PS, label_1, HASH(HM))$		$\boldsymbol{MS} = PRF(PS, label_1, HASH(HM))$;
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$		Генерация ключевого материала соединения: $\boldsymbol{KeyMaterial} = PRF(MS, label_2, r_S \| r_C)$
Этап подтверждения ключа		
$client\_verify\_data = PRF(MS, label_3, HASH(HM));$	ChangeCipherSpec Finished $client\_verify\_data$ →	
	← ChangeCipherSpec Finished $server\_verify\_data$	$server\_verify\_data = PRF(MS, label_3, HASH(HM));$

Аутентификация сервера осуществляется за счет доказательства корректного извлечения сервером секретного значения PS с помощью своего закрытого ключа, осуществляемого с помощью сообщений Finished

Client (C)	Сообщения	Server (S)
Выработка случайного значения r_C	ClientHello r_C, RI_C →	
	← ServerHello r_S, RI_S	Выработка случайного значения r_S
Этап выработки общего секрета		
	← Certificate $Cert_S$ (содержит Q_S)	
	⇠ CertificateRequest	
	← ServerHelloDone	
	⇢ Certificate $Cert_C$ (содержит Q_C)	
Выработка случайного значения PS (32 байта) Выработка эфемерной пары ключей $Q_{EPH} = k_{EPH}P$, где k_{EPH} выбирается случайно. $H = HASH(r_C \| r_S)$; $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_{EPH}, Q_S, H)$		
Экспортное представление общего секрета PS: $IV = H[25..24 + n/2];$ $PSExp = KExp15(PS, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$	ClientKeyExchange Q_{EPH},$PSExp$ →	$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_S, Q_{EPH}, H)$
		Извлечение общего секрета из экспортного представления: $IV = H[25..24 + n/2];$ $PS = KImp15(PSExp, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$
$sgn_C = SIGN_{k_C}(HM)$	CertificateVerify sgn_C ⇢	Проверка sgn_C
Этап выработки ключей из значения *PS*		
$MS = PRF(PS, label_1, HASH(HM))$		$MS = PRF(PS, label_1, HASH(HM))$;
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$
Этап подтверждения ключа		
$client\_verify\_data = PRF(MS, label_3, HASH(HM))$;	ChangeCipherSpec → Finished $client\_verify\_data$ →	
	ChangeCipherSpec ← Finished $server\_verify\_data$ ←	$server\_verify\_data = PRF(MS, label_3, HASH(HM))$;

Клиент, если это требуется, аутентифицируется с помощью подписи.

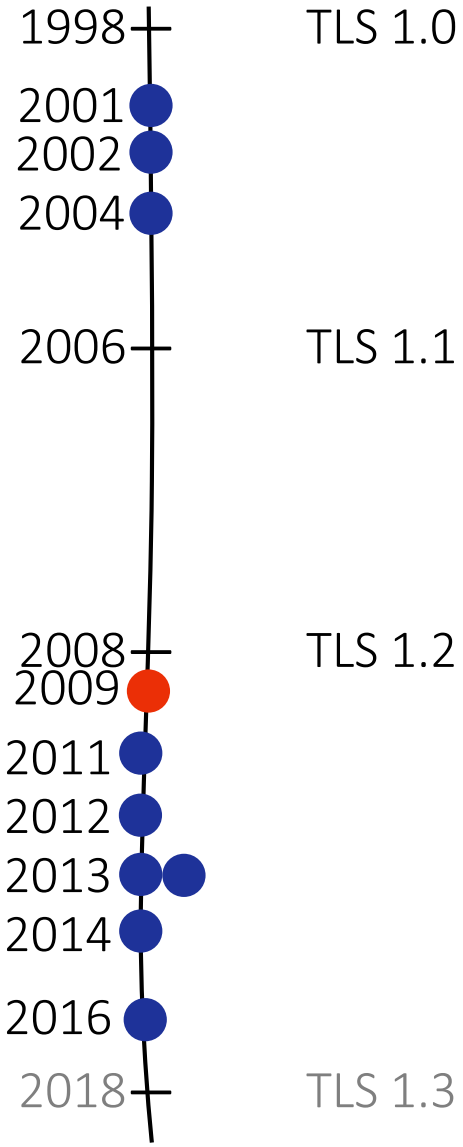
Client (C)	Сообщения	Server (S)
Выработка случайного значения r_C	ClientHello → r_C, RI_C	
	ServerHello ← r_S, RI_S	Выработка случайного значения r_S
Этап выработки общего секрета		
	Certificate ← $Cert_S$ (содержит Q_S)	
	CertificateRequest ⇠	
	ServerHelloDone ⇠	
	Certificate ⇢ $Cert_C$ (содержит Q_C)	
Выработка случайного значения PS (32 байта)		
Эфемерный ключ на кривой, соответствующей Q_S: $Q_{EPH} = k_{EPH}P$, где k_{EPH} выбирается случайно.		
$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_{EPH}, Q_S, H)$		
Экспортное представление общего секрета PS: $IV = H[25..24 + n/2];$ $PSExp = \boldsymbol{KExp15}(PS, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$	ClientKeyExchange → Q_{EPH},$PSExp$	$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_S, Q_{EPH}, H)$
		Извлечение общего секрета из экспортного представления: $IV = H[25..24 + n/2];$ $PS = \boldsymbol{KImp15}(PSExp, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$
$sgn_C = SIGN_{k_C}(HM)$	CertificateVerify ⇢ sgn_C	Проверка sgn_C
Этап выработки ключей из значения *PS*		
$MS = PRF(PS, label_1, HASH(HM))$		$MS = PRF(PS, label_1, HASH(HM))$;
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$
Этап подтверждения ключа		
$client\_verify\_data = PRF(MS, label_3, HASH(HM));$	ChangeCipherSpec → Finished → $client\_verify\_data$	
	ChangeCipherSpec ← Finished ← $server\_verify\_data$	$server\_verify\_data = PRF(MS, label_3, HASH(HM));$

Используется новый алгоритм шифрования PS, так называемый алгоритм экспорта/импорта ключа, основанный на новых шифрах и режимах их работы (определяется в рамках вышеупомянутого проекта рекомендаций);

Атаки «без доступа к ключу»

Handshake/ Атака Марша Рэя

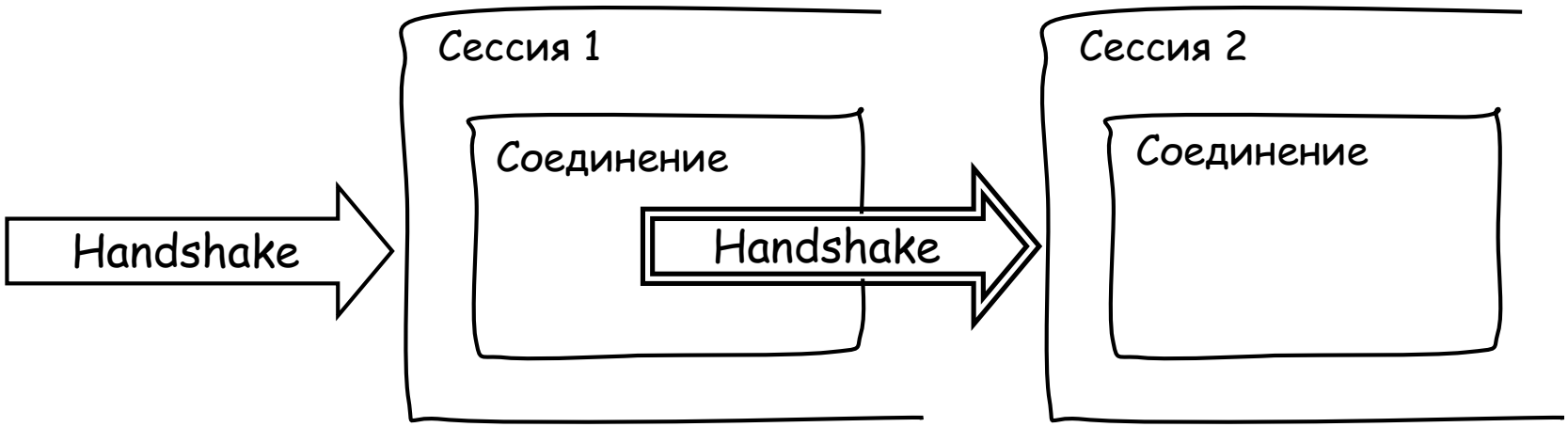
Навязывание сообщений на процедуру renegotiation (атака Марша Рэя)



КРИПТОПРО

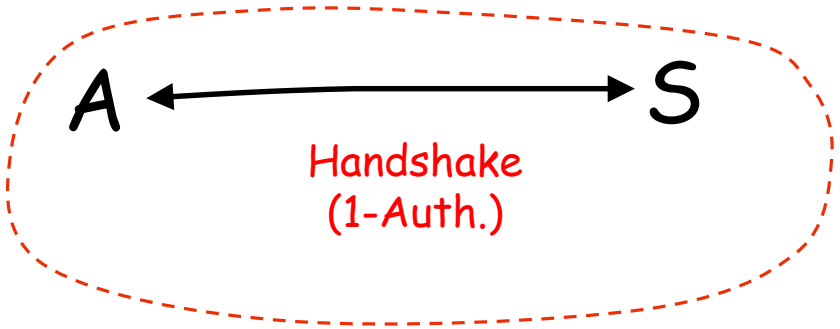
Handshake/ Атака Марша Рэя

Renegotiation — процедура выработки параметров для нового соединения, проходящая под защитой параметров текущего соединения.



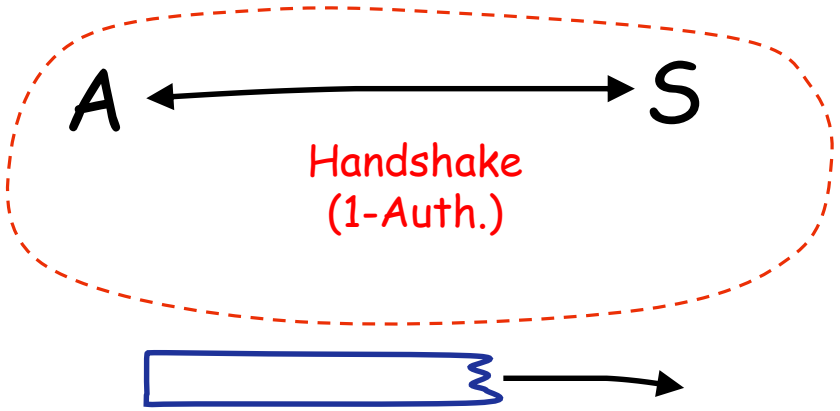
КРИПТОПРО

Handshake/ Атака Марша Рэя



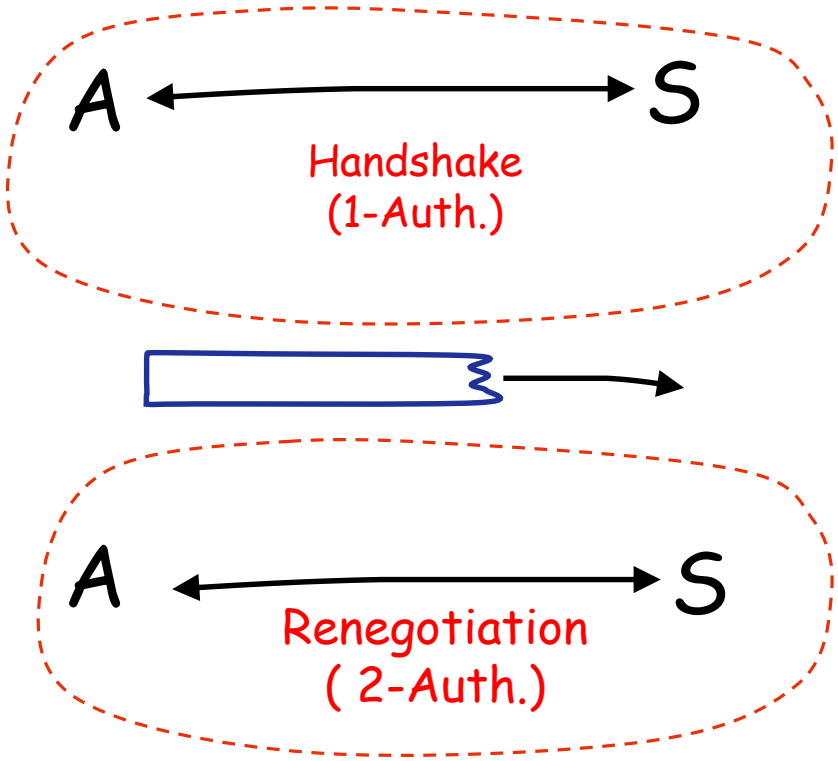
КРИПТОПРО

Handshake/ Атака Марша Рэя



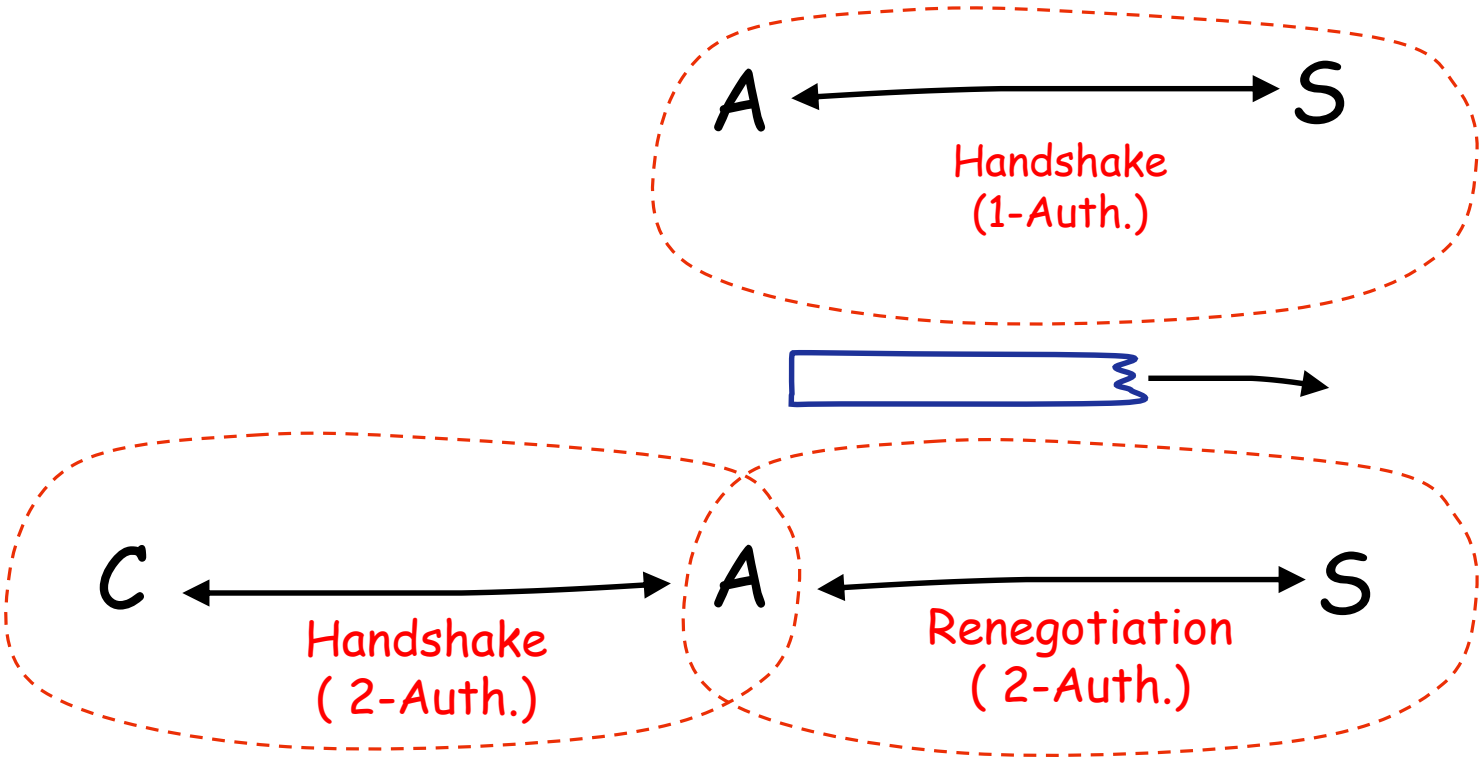
КРИПТОПРО

Handshake/ Атака Марша Рэя



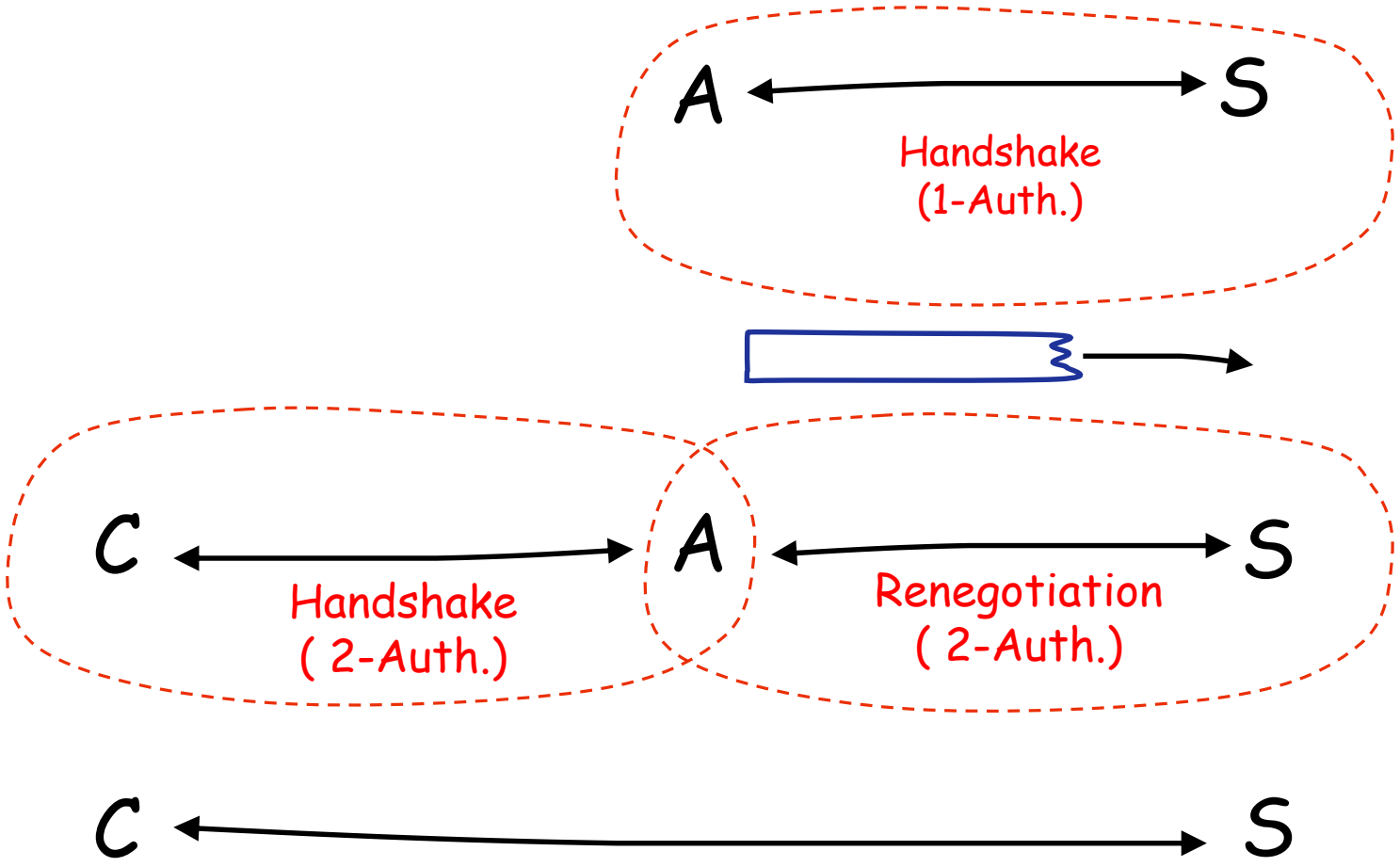
КРИПТОПРО

Handshake/ Атака Марша Рэя



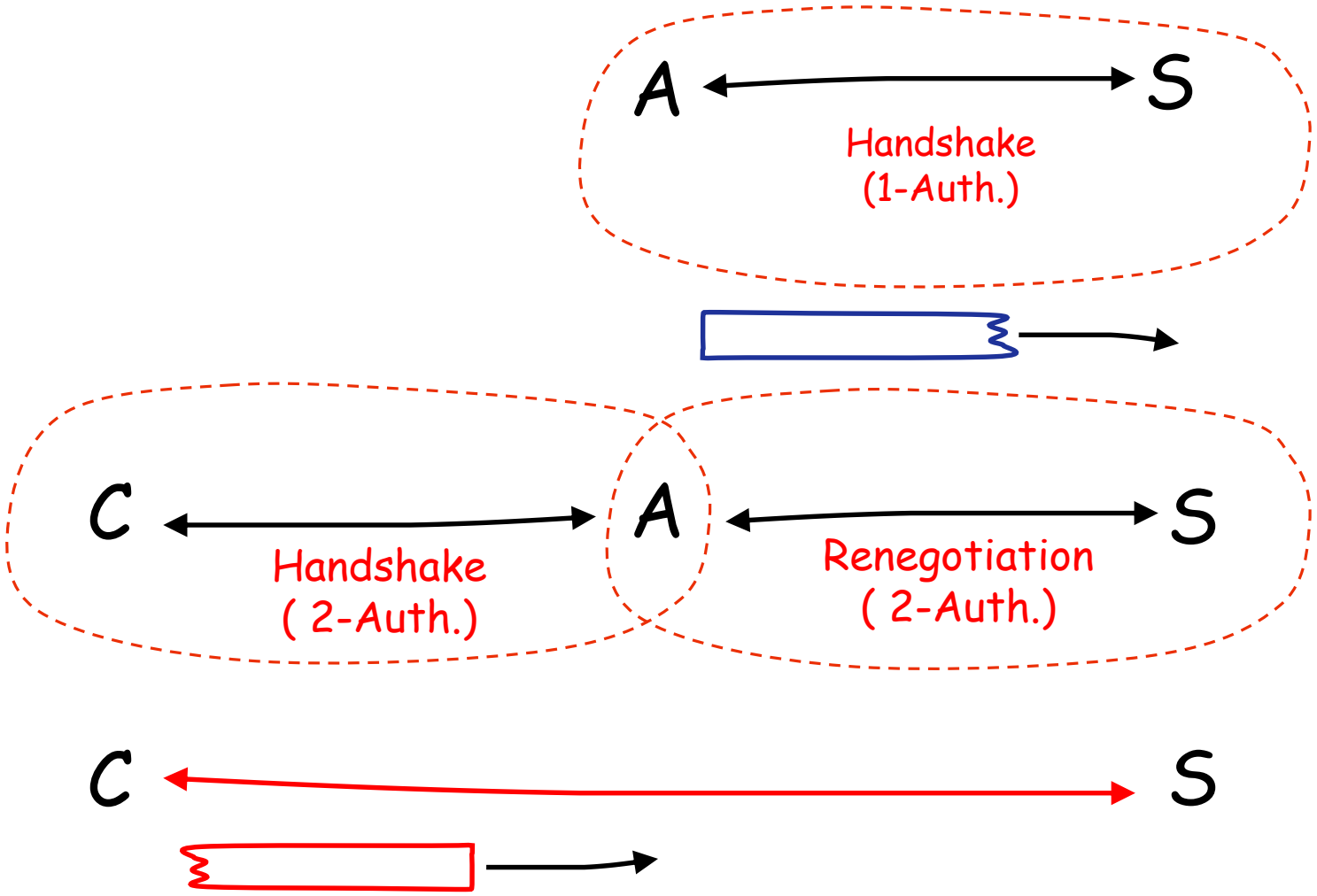
КРИПТОПРО

Handshake/ Атака Марша Рэя



КРИПТОПРО

Handshake/ Атака Марша Рэя



КРИПТОПРО

Handshake/ Атака Марша Рэя

КРИПТОПРО

Handshake/ Атака Марша Рэя

Противнику не известны новые ключи соединения, однако сообщения, пересылаемые противником до пересогласования (renegotiation) и сообщения, пересылаемые легитимным клиентом после пересогласования, будут восприниматься сервером как единый поток данных от одного клиента.

КРИПТОПРО

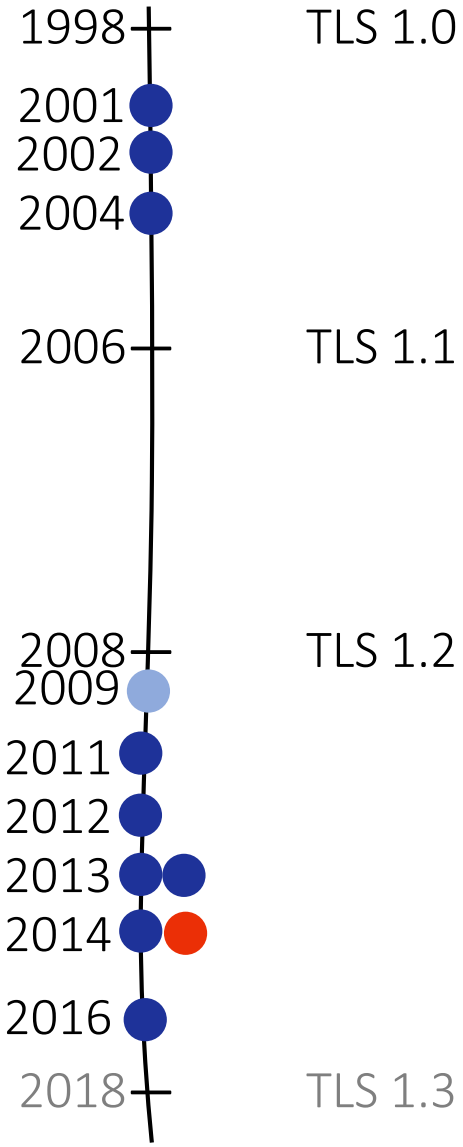
Client (C)	Сообщения	Server (S)
Выработка случайного значения r_C	ClientHello r_C, $\boldsymbol{RI_C}$ →	
	ServerHello r_S, $\boldsymbol{RI_S}$ ←	Выработка случайного значения r_S
Этап выработки общего секрета		
	Certificate $Cert_S$ (содержит Q_S) ←	
	ServerHelloDone ←	
Выработка случайного значения PS (32 байта)		
Эфемерный ключ на кривой, соответствующей Q_S: $Q_{EPH} = k_{EPH}P$, где k_{EPH} выбирается случайно.		
$H = HASH(r_C \Vert r_S);$ $K_{MAC}^{Exp} \Vert K_{ENC}^{Exp} = KEG(k_{EPH}, Q_S, H)$		
Экспортное представление общего секрета PS: $IV = H[25..24 + n/2];$ $PSExp = KExp15(PS, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$	ClientKeyExchange Q_{EPH},$PSExp$ →	$H = HASH(r_C \Vert r_S);$ $K_{MAC}^{Exp} \vert K_{ENC}^{Exp} = KEG(k_S, Q_{EPH}, H)$
		Извлечение общего секрета из экспортного представления: $IV = H[25..24 + n/2];$ $PS = KImp15(PSExp, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$
Этап выработки ключей из значения PS		
$MS = PRF(PS, label_1, HASH(HM))$		$MS = PRF(PS, label_1, HASH(HM)_{);}$
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \Vert r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \Vert r_C)$
Этап подтверждения ключа		
$client\_verify\_data = PRF(MS, label_3, HASH(\boldsymbol{HM}));$	ChangeCipherSpec → Finished → $client\_verify\_data$	
	ChangeCipherSpec ← Finished ← $server\_verify\_data$	$server\_verify\_data = PRF(MS, label_3, HASH(\boldsymbol{HM}));$

renegotiation_info (RFC 5746)

Handshake/ Атака Triple Handshake

Атака Triple Handshake состоит из навязывания серверу сообщений в течение трех этапов:

- первичного установления соединения;
- возобновления соединения (session resumption);
- пересогласования соединения (renegotiation).



КРИПТОПРО

Client (C)	Сообщения	Server (S)
Выработка случайного значения r_C	ClientHello r_C, RI_C →	
	← ServerHello r_S, RI_S	Выработка случайного значения r_S
Этап выработки общего секрета		
	← Certificate $Cert_S$ (содержит Q_S)	
	← ServerHelloDone	
Выработка случайного значения PS (32 байта)		
Эфемерный ключ на кривой, соответствующей Q_S: $Q_{EPH} = k_{EPH}P$, где k_{EPH} выбирается случайно.		
$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_{EPH}, Q_S, H)$		
Экспортное представление общего секрета PS: $IV = H[25..24 + n/2];$ $PSExp = KExp15(PS, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$	ClientKeyExchange Q_{EPH},$PSExp$ →	$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_S, Q_{EPH}, H)$
		Извлечение общего секрета из экспортного представления: $IV = H[25..24 + n/2];$ $PS = KImp15(PSExp, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$
Этап выработки ключей из значения PS		
$MS = PRF(PS, label_1, \boldsymbol{r_S} \| \boldsymbol{r_C})$		$MS = PRF(PS, label_1, \boldsymbol{r_S} \| \boldsymbol{r_C})$
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$
Этап подтверждения ключа		
$client\_verify\_data = PRF(MS, label_3, HASH(HM));$	ChangeCipherSpec → Finished $client\_verify\_data$ →	
	← ChangeCipherSpec ← Finished $server\_verify\_data$	$server\_verify\_data = PRF(MS, label_3, HASH(HM));$

Client (C)	Сообщения	Server (S)
Выработка случайного значения r_C	ClientHello r_C, RI_C →	
	← ServerHello r_S, RI_S	Выработка случайного значения r_S
Этап выработки общего секрета		
	← Certificate $Cert_S$ (содержит Q_S)	
	← ServerHelloDone	
Выработка случайного значения PS (32 байта)		
Эфемерный ключ на кривой, соответствующей Q_S: $Q_{EPH} = k_{EPH}P$, где k_{EPH} выбирается случайно.		
$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_{EPH}, Q_S, H)$		
Экспортное представление общего секрета PS: $IV = H[25..24 + n/2];$ $PSExp = KExp15(PS, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$	ClientKeyExchange Q_{EPH},$PSExp$ →	$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_S, Q_{EPH}, H)$
		Извлечение общего секрета из экспортного представления: $IV = H[25..24 + n/2];$ $PS = KImp15(PSExp, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$
Этап выработки ключей из значения *PS*		
$MS = PRF(PS, label_1, \boldsymbol{HASH(HM)})$		$MS = PRF(PS, label_1, \boldsymbol{HASH(HM)});$
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$
Этап подтверждения ключа		
$client\_verify\_data = PRF(MS, label_3, HASH(HM));$	ChangeCipherSpec → Finished $client\_verify\_data$ →	
	← ChangeCipherSpec ← Finished $server\_verify\_data$	$server\_verify\_data = PRF(MS, label_3, HASH(HM));$

extended master secret (RFC 7627)

Кросс-протокольные атаки

Handshake/ Кросс-протокольные атаки

Кросс-протокольные атаки позволяют противнику посередине заставить клиента, изначально желавшего использовать стойкие криптонаборы, установить с сервером соединение на уязвимых криптонаборах.

КРИПТОПРО

Client (C)	Сообщения	Server (S)
Выработка случайного значения r_C	ClientHello r_C, RI_C →	
	← ServerHello r_S, RI_S	Выработка случайного значения r_S
Этап выработки общего секрета		
	← Certificate $Cert_S$ (содержит Q_S)	
	← ServerHelloDone	
Выработка случайного значения PS (32 байта)		
Эфемерный ключ на кривой, соответствующей Q_S: $Q_{EPH} = k_{EPH}P$, где k_{EPH} выбирается случайно.		
$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_{EPH}, Q_S, H)$		
Экспортное представление общего секрета PS: $IV = H[25..24 + n/2];$ $PSExp = KExp15(PS, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$	ClientKeyExchange Q_{EPH},$PSExp$ →	$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_S, Q_{EPH}, H)$
		Извлечение общего секрета из экспортного представления: $IV = H[25..24 + n/2];$ $PS = KImp15(PSExp, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$
Этап выработки ключей из значения PS		
$MS = PRF(PS, label_1, HASH(HM))$		$MS = PRF(PS, label_1, HASH(HM))$;
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$
Этап подтверждения ключа		
$client\_verify\_data = PRF(MS, label_3, HASH(HM));$	ChangeCipherSpec → Finished $client\_verify\_data$ →	
	← ChangeCipherSpec Finished $server\_verify\_data$ ←	$server\_verify\_data = PRF(MS, label_3, HASH(HM));$

Client (C)	Сообщения	Server (S)
криптонабор1 (стойкий)	ClientHello → **подмена**	**криптонабор2** (уязвимый)
	ServerHello ←	
Этап выработки общего секрета		
	Certificate ← $Cert_S$ (содержит Q_S)	
	ServerHelloDone ←	
Выработка случайного значения PS (32 байта)		
Эфемерный ключ на кривой, соответствующей Q_S: $Q_{EPH} = k_{EPH}P$, где k_{EPH} выбирается случайно.		
$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_{EPH}, Q_S, H)$		
Экспортное представление общего секрета PS: $IV = H[25..24 + n/2];$ $PSExp = KExp15(PS, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$	ClientKeyExchange → Q_{EPH},$PSExp$	$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_S, Q_{EPH}, H)$
		Извлечение общего секрета из экспортного представления: $IV = H[25..24 + n/2];$ $PS = KImp15(PSExp, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$
Этап выработки ключей из значения PS		
$MS = PRF(PS, label_1, HASH(HM))$		$MS = PRF(PS, label_1, HASH(HM))$;
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$
Этап подтверждения ключа		
$client\_verify\_data = PRF(MS, label_3, HASH(HM));$	ChangeCipherSpec → Finished → $client\_verify\_data$	
	ChangeCipherSpec ← Finished ← $server\_verify\_data$	$server\_verify\_data = PRF(MS, label_3, HASH(HM));$

Client (C)	Сообщения	Server (S)
криптонабор1 (стойкий)	ClientHello → **подмена**	**криптонабор2** (уязвимый)
криптонабор1	ServerHello ← **подмена**	**криптонабор2**
Этап выработки общего секрета		
	Certificate ← $Cert_S$ (содержит Q_S)	
	ServerHelloDone	
Выработка случайного значения PS (32 байта)		
Эфемерный ключ на кривой, соответствующей Q_S: $Q_{EPH} = k_{EPH}P$, где k_{EPH} выбирается случайно.		
$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_{EPH}, Q_S, H)$		
Экспортное представление общего секрета PS: $IV = H[25..24 + n/2];$ $PSExp = KExp15(PS, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$	ClientKeyExchange → Q_{EPH},$PSExp$	$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_S, Q_{EPH}, H)$
		Извлечение общего секрета из экспортного представления: $IV = H[25..24 + n/2];$ $PS = KImp15(PSExp, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$
Этап выработки ключей из значения *PS*		
$MS = PRF(PS, label_1, HASH(HM))$		$MS = PRF(PS, label_1, HASH(HM))$;
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$
Этап подтверждения ключа		
$client\_verify\_data = PRF(MS, label_3, HASH(HM));$	ChangeCipherSpec → Finished → $client\_verify\_data$	
	ChangeCipherSpec ← Finished ← $server\_verify\_data$	$server\_verify\_data = PRF(MS, label_3, HASH(HM));$

Условие 1: сервер по определенным причинам должен согласиться работать на уязвимом криптонаборе.

Client (C)	Сообщения	Server (S)
криптонабор1 (стойкий)	ClientHello → **подмена**	**криптонабор2** (уязвимый)
криптонабор1	ServerHello ← **подмена**	**криптонабор2**
Этап выработки общего секрета		
	Certificate ← $Cert_S$ (содержит Q_S)	
	ServerHelloDone ←	
Выработка случайного значения PS (32 байта)		
Эфемерный ключ на кривой, соответствующей Q_S: $Q_{EPH} = k_{EPH}P$, где k_{EPH} выбирается случайно.		
$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_{EPH}, Q_S, H)$		
Экспортное представление общего секрета PS: $IV = H[25..24 + n/2];$ $PSExp = KExp15(PS, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$	ClientKeyExchange → Q_{EPH},$PSExp$	$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_S, Q_{EPH}, H)$
		Извлечение общего секрета из экспортного представления: $IV = H[25..24 + n/2];$ $PS = KImp15(PSExp, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$
Этап выработки ключей из значения *PS*		
$MS = PRF(PS, label_1, HASH(HM))$		$MS = PRF(PS, label_1, HASH(HM))$;
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$
Этап подтверждения ключа		
$client\_verify\_data = PRF(MS, label_3, HASH(HM));$	ChangeCipherSpec → Finished → $client\_verify\_data$	
	ChangeCipherSpec ← Finished ← $server\_verify\_data$	$server\_verify\_data = PRF(MS, label_3, HASH(HM));$

Client (C)	Сообщения	Server (S)
криптонабор1 (стойкий)	ClientHello → **подмена**	**криптонабор2** (уязвимый)
криптонабор1	ServerHello ← **подмена**	**криптонабор2**
Этап выработки общего секрета		
	Certificate ← $Cert_S$ (содержит Q_S)	
	ServerKeyExchange ←	
	ServerHelloDone ←	
Выработка случайного значения PS (32 байта)		
Эфемерный ключ на кривой, соответствующей Q_S: $Q_{EPH} = k_{EPH}P$, где k_{EPH} выработан случайно		
$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(Q_S, k_{EPH}, Q_S, H)$		
Экспортное представление общего секрета PS: $IV = H[25..24 + n/2];$ $PSExp = KExp15(PS, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$	ClientKeyExchange → Q_{EPH}, $PSExp$	$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_S, Q_{EPH}, H)$
		Извлечение общего секрета из экспортного представления: $IV = H[25..24 + n/2];$ $PS = KImp15(PSExp, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$
Этап выработки ключей из значения PS		
$MS = PRF(PS, label_1, HASH(HM))$		$MS = PRF(PS, label_1, HASH(HM))$;
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$
Этап подтверждения ключа		
$client\_verify\_data = PRF(MS, label_3, HASH(HM));$	ChangeCipherSpec → Finished → $client\_verify\_data$	
	ChangeCipherSpec ← Finished ← $server\_verify\_data$	$server\_verify\_data = PRF(MS, label_3, HASH(HM));$

Набор данных, соответствующих криптонабору2 и подпись под этими данными. Не содержит идентификатор выбранного криптонабора.

Client (C)	Сообщения	Server (S)
криптонабор1 (стойкий)	ClientHello **подмена**	**криптонабор2** (уязвимый)
криптонабор1	ServerHello **подмена**	**криптонабор2**
Этап выработки общего секрета		
	Certificate $Cert_S$ (содержит Q_S)	
	ServerKeyExchange	
	ServerHelloDone	
Эфемерный ключ $Q_{EPH} = k_{EPH}P$, где k_{EPH} выбирается случайно. $H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_{EPH}, Q_S, H)$ Экспортное представление общего секрета PS: $IV = H[25..24 + n/2];$ $PSExp = KExp15(PS, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$	ClientKeyExchange Q_{EPH},$PSExp$	$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_S, Q_{EPH}, H)$
		Извлечение общего секрета из экспортного представления: $IV = H[25..24 + n/2];$ $PS = KImp15(PSExp, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$
Этап выработки ключей из значения PS		
$MS = PRF(PS, label_1, HASH(HM))$		$MS = PRF(PS, label_1, HASH(HM))$;
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$
Этап подтверждения ключа		
$client\_verify\_data = PRF(MS, label_3, HASH(HM))$;	ChangeCipherSpec Finished $client\_verify\_data$	
	ChangeCipherSpec Finished $server\_verify\_data$	$server\_verify\_data = PRF(MS, label_3, HASH(HM))$;

Условие 2: клиент не проверяет данные, пересылаемые в сообщении ServerKeyExchange на соответствие криптонабору1.

Client (C)	Сообщения	Server (S)
криптонабор1 (стойкий)	ClientHello → **подмена**	**криптонабор2** (уязвимый)
криптонабор1	ServerHello ← **подмена**	**криптонабор2**
Этап выработки общего секрета		
	Certificate ← $Cert_S$ (содержит Q_S)	
	ServerKeyExchange ←	
	ServerHelloDone ←	
	Клиент вырабатывает эфемерный ключ Q_{EPH} на основе уязвимых параметров, ничего не подозревая об этом, и отправляет секретное значение PS под защитой Q_{EPH}.	
	ClientKeyExchange → $(PS)_{Q_{EPH}}$	
Этап выработки ключей из значения PS		
$MS = PRF(PS, label_1, HASH(HM))$		$MS = PRF(PS, label_1, HASH(HM))$;
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$
Этап подтверждения ключа		
$client\_verify\_data = PRF(MS, label_3, HASH(HM))$;	ChangeCipherSpec → Finished → $client\_verify\_data$	
	ChangeCipherSpec ← Finished ← $server\_verify\_data$	$server\_verify\_data = PRF(MS, label_3, HASH(HM))$;

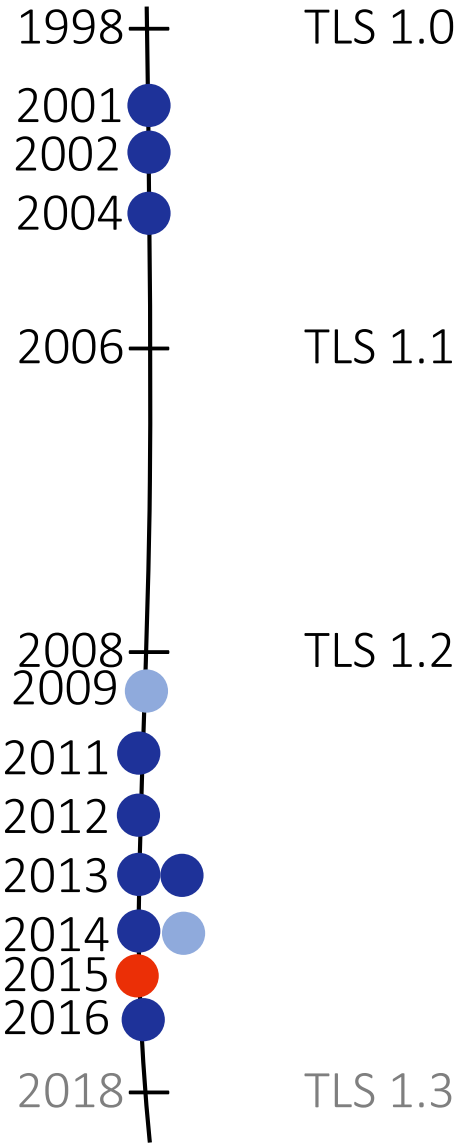
Client (C)	Сообщения	Server (S)
криптонабор1 (стойкий)	ClientHello → **подмена**	**криптонабор2** (уязвимый)
криптонабор1	ServerHello ← **подмена**	**криптонабор2**
Этап выработки общего секрета		
	Certificate ← $Cert_S$ (содержит Q_S)	
	ServerKeyExchange ←	
	ServerHelloDone ←	
	ClientKeyExchange → $(PS)_{Q_{EPH}}$	
$MS = PRF(PS, label_1, HASH(HM))$		$MS = PRF(PS, label_1, HASH(HM))$;
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$
Этап подтверждения ключа		
$client\_verify\_data = PRF(MS, label_3, HASH(HM))$;	ChangeCipherSpec → Finished → $client\_verify\_data$	
	ChangeCipherSpec ← Finished ← $server\_verify\_data$	$server\_verify\_data = PRF(MS, label_3, HASH(HM))$;

Противник получает возможность на основе некоторой уязвимости нестойкого криптонабора довольно легко восстановить эфемерный ключ и получить значение PS

Handshake/ Кросс-протокольные атаки/ FREAK

Атака FREAK: навязывание нестойких RSA_EXPORT криптонаборов (открытые RSA ключи с длиной, не превышающей 512 бит).

Противник может сделать предвычисления для наиболее распространенных простых 512-битных чисел.

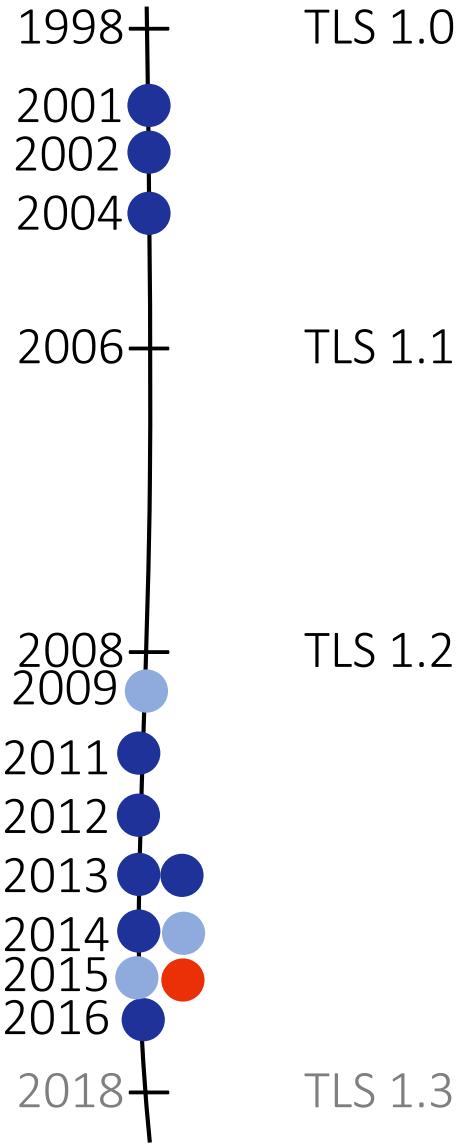


КРИПТОПРО

Handshake/ Кросс-протокольные атаки/ Logjam

Атака Logjam: навязывание нестойких DHE_EXPORT криптонаборов (ключи с длиной, не превышающей 512 бит).

Противник может сделать предвычисления для повышения эффективности алгоритма дискретного логарифмирования.

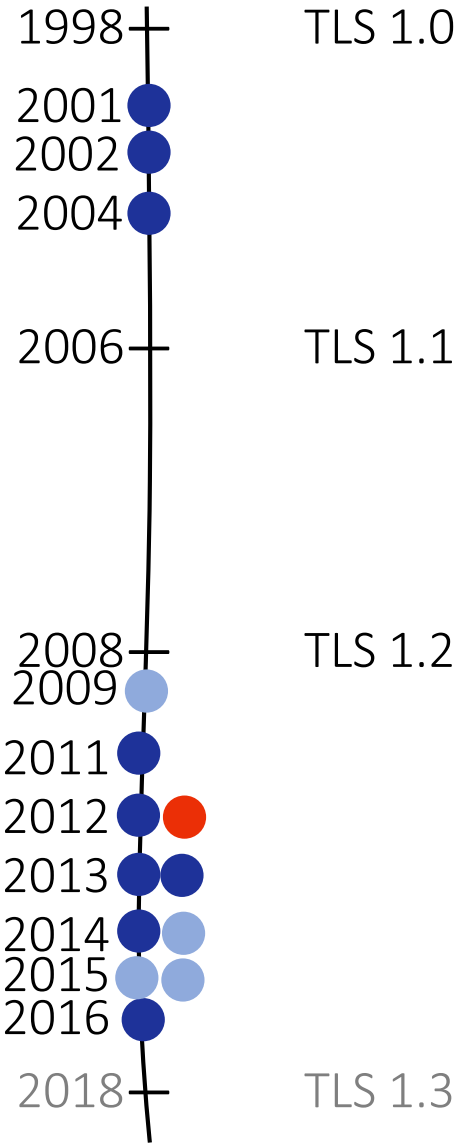


КРИПТОПРО

Handshake/ Кросс-протокольные атаки/ Подмена ECDH криптонаборов

Подмена ECDH криптонаборов на DH криптонаборы.

Атака заключалась в использовании возможности интерпретации параметров, указанных в сообщении ServerKeyExchange и соответствующих эллиптической кривой, в качестве уязвимых параметров, соответствующих мультипликативной группе целых чисел.



КРИПТОПРО

Client (C)	Сообщения	Server (S)
криптонабор1 (стойкий)	ClientHello → **подмена**	**криптонабор2** (уязвимый)
криптонабор1	ServerHello ← **подмена**	**криптонабор2**
Этап выработки общего секрета		
	Certificate ← $Cert_S$ (содержит Q_S)	
	ServerKeyExchange ←	
	ServerHelloDone ←	
	ClientKeyExchange →	
Этап выработки ключей из значения PS		
$MS = PRF(PS, label_1, HASH(HM))$		$MS = PRF(PS, label_1, HASH(HM)_{);}$
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \Vert r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \Vert r_C)$
Этап подтверждения ключа		
$client\_verify\_data = PRF\big(MS, label_3, HASH(HM)\big);$	ChangeCipherSpec → Finished → $client\_verify\_data$	
	ChangeCipherSpec ← Finished ← $server\_verify\_data$	$server\_verify\_data = PRF\big(MS, label_3, HASH(HM)\big);$

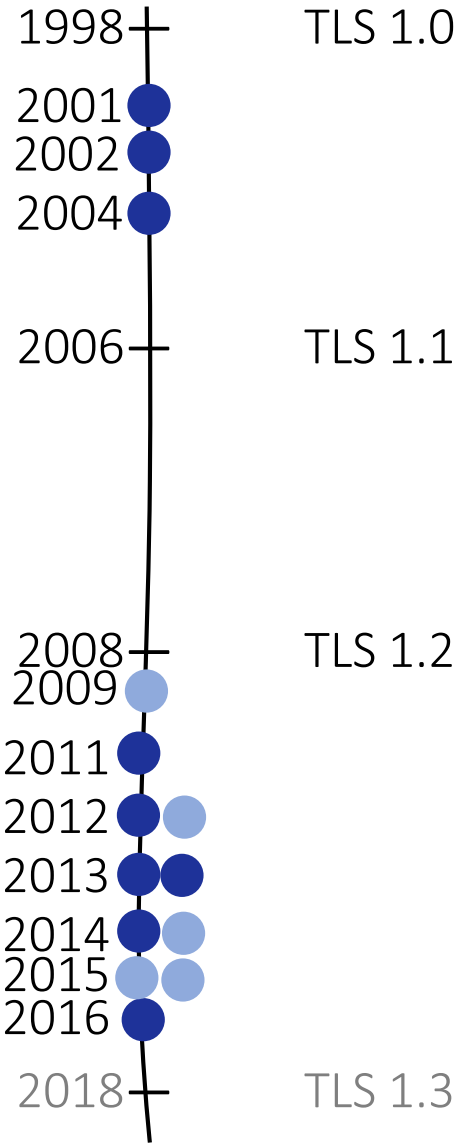
Client (C)	Сообщения	Server (S)
криптонабор1 (стойкий)	ClientHello → **подмена**	**криптонабор2** (уязвимый)
криптонабор1	ServerHello ← **подмена**	**криптонабор2**
Этап выработки общего секрета		
	Certificate ← $Cert_S$ (содержит Q_S)	
	~~ServerKeyExchange~~ ←	
	ServerHelloDone ←	
	ClientKeyExchange →	
Этап выработки ключей из значения PS		
$MS = PRF(PS, label_1, HASH(HM))$		$MS = PRF(PS, label_1, HASH(HM)_{);}$
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \Vert r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \Vert r_C)$
Этап подтверждения ключа		
$client\_verify\_data = PRF\big(MS, label_3, HASH(HM)\big);$	ChangeCipherSpec → Finished → $client\_verify\_data$	
	ChangeCipherSpec ← Finished ← $server\_verify\_data$	$server\_verify\_data = PRF\big(MS, label_3, HASH(HM)\big);$

Client (C)	Сообщения	Server (S)
криптонабор1 (стойкий)	ClientHello → **подмена**	**криптонабор2** (уязвимый)
криптонабор1	ServerHello ← **подмена**	**криптонабор2**
Этап выработки общего секрета		
	Certificate ← $Cert_S$ (содержит Q_S)	
	~~ServerKeyExchange~~ ←	
	ServerHelloDone ←	
Выработка случайного значения PS (32 байта)		
Эфемерный ключ на кривой, соответствующей Q_S: $Q_{EPH} = k_{EPH}P$, где k_{EPH} выбирается случайно.		
$H = HASH(\dots \| \dots)$		
	ClientKeyExchange →	
		$PS = Kimp15(PSExp, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$
Этап выработки ключей из значения PS		
$MS = PRF(PS, label_1, HASH(HM))$		$MS = PRF(PS, label_1, HASH(HM))$;
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$
Этап подтверждения ключа		
$client\_verify\_data = PRF(MS, label_3, HASH(HM))$;	ChangeCipherSpec → Finished → $client\_verify\_data$	
	ChangeCipherSpec ← Finished ← $server\_verify\_data$	$server\_verify\_data = PRF(MS, label_3, HASH(HM))$;

Используются параметры из сертификата сервера, которые могут соответствовать предопределенному множеству кривых. Изменение этих параметров в сертификате приводит к неудаче при попытке проверки подписи

Handshake/ Модификации эфемерного ключа

А также атаки на основе модификации эфемерного ключа (разные года)



КРИПТОПРО

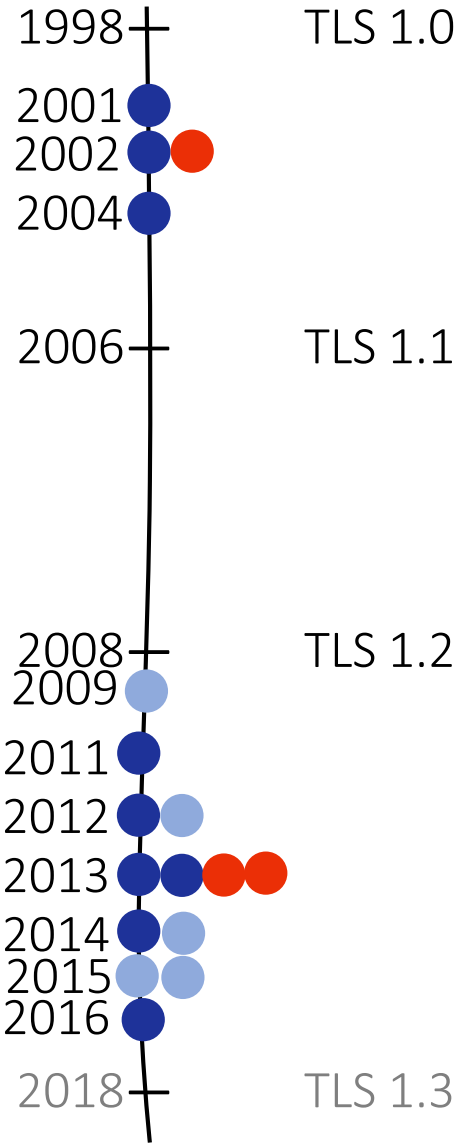
Client (C)	Сообщения	Server (S)
Выработка случайного значения r_C	ClientHello r_C, RI_C →	
	← ServerHello r_S, RI_S	Выработка случайного значения r_S
Этап выработки общего секрета		
	← Certificate $Cert_S$ (содержит Q_S)	
	← ServerHelloDone	
Выработка случайного значения PS (32 байта)		
Эфемерный ключ на кривой, соответствующей Q_S: $Q_{EPH} = k_{EPH}P$, где k_{EPH} выбирается случайно.		
$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_{EPH}, Q_S, H)$		
Экспортное представление общего секрета PS: $IV = H[25..24 + n/2];$ $PSExp = KExp15(PS, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$	ClientKeyExchange Q_{EPH},$PSExp$ →	$H = HASH(r_C \| r_S);$ $K_{MAC}^{Exp} \| K_{ENC}^{Exp} = KEG(k_S, Q_{EPH}, H)$
		Извлечение общего секрета из экспортного представления: $IV = H[25..24 + n/2];$ $PS = KImp15(PSExp, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$
Этап выработки ключей из значения *PS*		
$MS = PRF(PS, label_1, HASH(HM))$		$MS = PRF(PS, label_1, HASH(HM));$
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S \| r_C)$
Этап подтверждения ключа		
$client\_verify\_data = PRF(MS, label_3, HASH(HM));$	ChangeCipherSpec → Finished $client\_verify\_data$ →	
	← ChangeCipherSpec ← Finished $server\_verify\_data$	$server\_verify\_data = PRF(MS, label_3, HASH(HM));$

Проверка принадлежности точки целевой подгруппе точек эллиптической кривой

Атаки, основывающиеся на свойствах конкретных примитивов.

Handshake/ Атаки на алгоритм сжатия

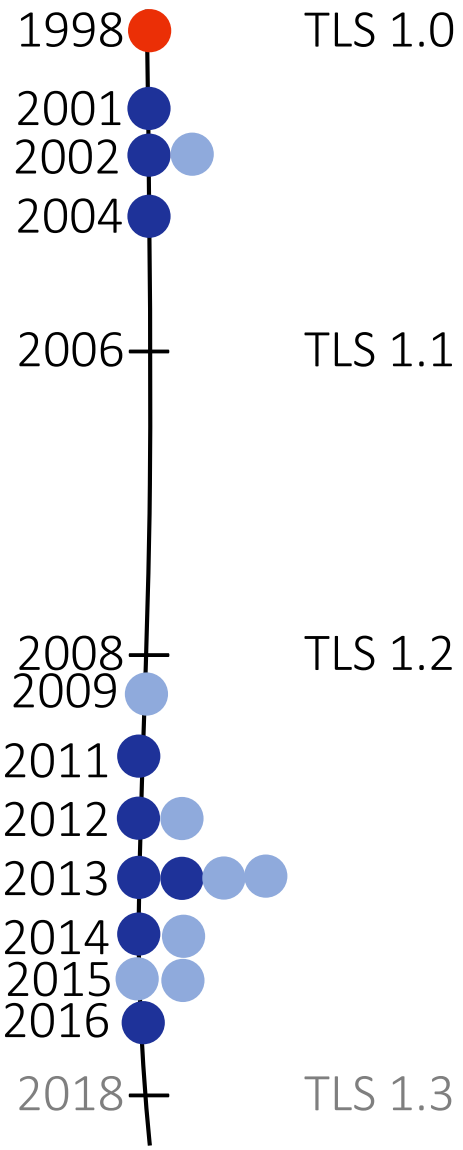
Атаки на алгоритм сжатия
(CRIME, TIME, BREACH)



КРИПТОПРО

Handshake/ Атака Блейхенбахера

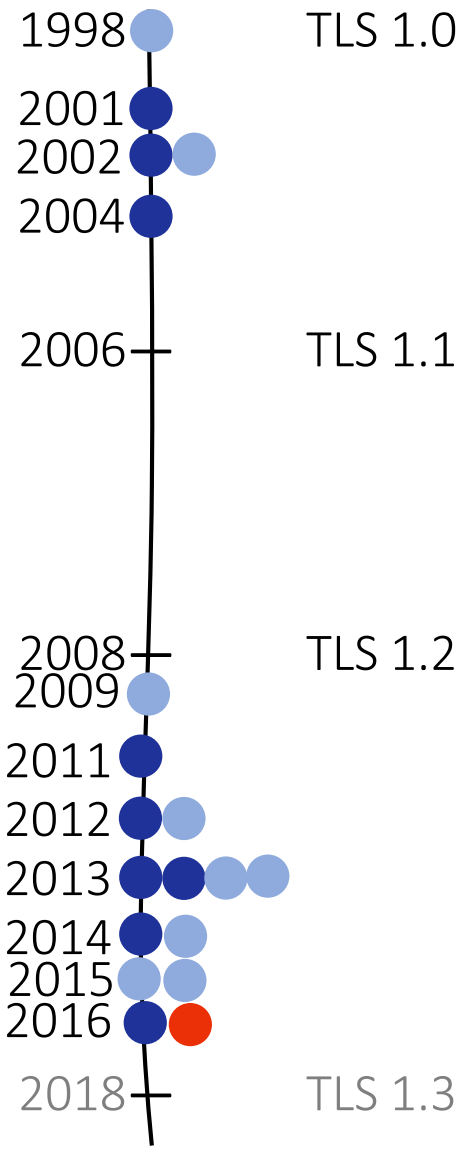
В 1998 году была опубликована атака Блейхенбахера, применимая к криптонаборам, использующим криптосистему RSA совместно с алгоритмом форматирования PKCS#1.



КРИПТОПРО

Handshake/ Атака DROWN

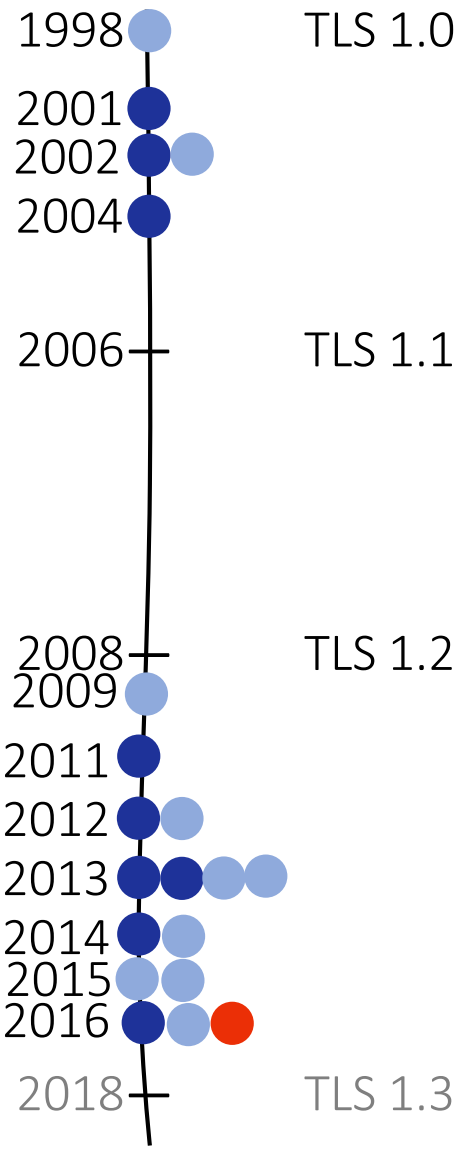
Условием проведения атаки DROWN является поддержка легитимным сервером режима работы TLS, соответствующего криптонаборам, уязвимым к атаке Блейхенбахера. В ходе данной атаки ключ сервера компрометируется во время соединения с уязвимым криптонабором.



КРИПТОПРО

Handshake/ Атака SLOTH

Атака SLOTH основывается на нестойкости распространенных хэш-функций MD5 и SHA-1 к построению коллизий различных видов.

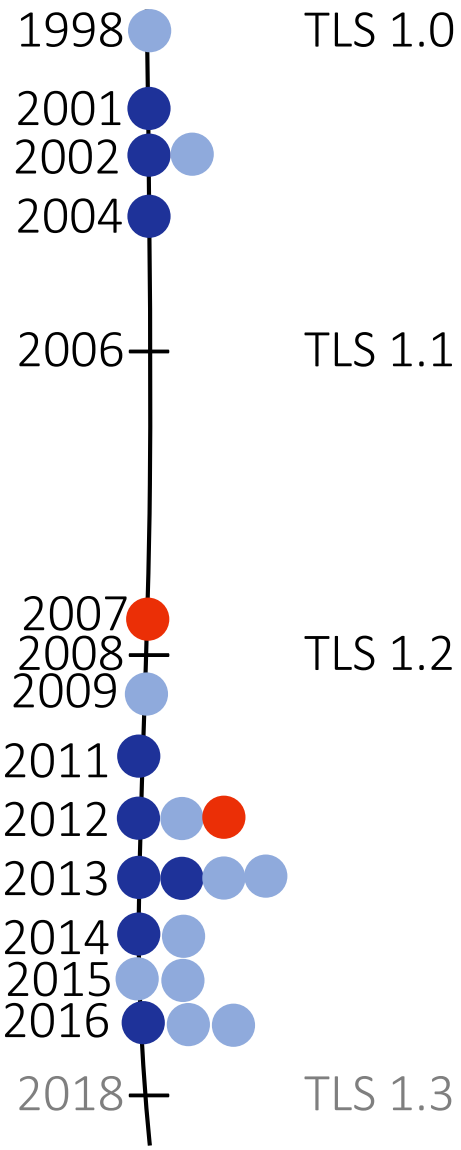


КРИПТОПРО

Alert

Прочие атаки

1. Атаки, связанные с фрагментацией оповещений
2. Атаки, связанные с обрезанием оповещений



КРИПТОПРО

Итог

Итог

- Полное и самодостаточное описание протокола.
- Структурированная информация.
- Подробный журнал работы протокола, разобранный по полям используемых структур.

КРИПТОПРО

Итог

Формирование сообщения ClientHello на стороне клиента:

```
msg_type:                 01
length:                   000040
body:
  client_version:
    major:                03
    minor:                03
  random:                 933EA21EC3802A561550EC78D6ED51AC
                          2439D7E749C31BC3A3456165889684CA
  session_id:
    length:               00
    vector:               --
  cipher_suites:
    length:               0004
    vector:
      CipherSuite:        FF88
      CipherSuite:        FF89
  compression_methods:
    length:               01
    vector:
      CompressionMethod:  00
  extensions:
    length:               0013
    vector:
      Extension: /* signature_algorithms */
        extension_type:   000D
        extension_data:
          length:         0006
          vector:
            supported_signature_algorithms:
              length:     0004
              vector:
                /* Первая пара алгоритмов */
                hash:     EE
                signature:
                          FF
```

КРИПТОПРО

Итог

На настоящий момент номерам разрабатываемых криптонаборов присвоены временные значения из приватной области номеров IANA:
TLS_GOSTR341112_256_WITH_KUZNYECHIK_CTR_OMAC, {0xFF, 0x89};
TLS_GOSTR341112_256_WITH_MAGMA_CTR_OMAC, {0xFF, 0x88}.

Как только проект рекомендаций будет принят, планируется сразу же приступить к разработке его RFC.

КРИПТОПРО

Планы на будущее

Ближайшими планами экспертов ТК26 является начало разработки следующего проекта рекомендаций, посвящённого криптонаборам, соответствующим версии TLS 1.3. Работа над данным документом начнется сразу после утверждения в ТК26 текущего проекта для TLS 1.2, а также после выхода нового RFC для TLS 1.3.

КРИПТОПРО

Спасибо за внимание!

КРИПТОПРО

Record/ Новые криптонаборы/ В цифрах

	MAGMA	KUZNYECHIK
Нагрузка на ключ	4 Мбайт	256 Кбайт
Количество записей, обрабатываемых на одном ключе	4096	64
Размер секции для ACPKM	1 Кбайт	4 Кбайт
Максимальная длина записи	2^{14} + 8 Байт	2^{14} + 16 Байт

КРИПТОПРО

TLS/ Record/ External re-keying (TLSTREE)

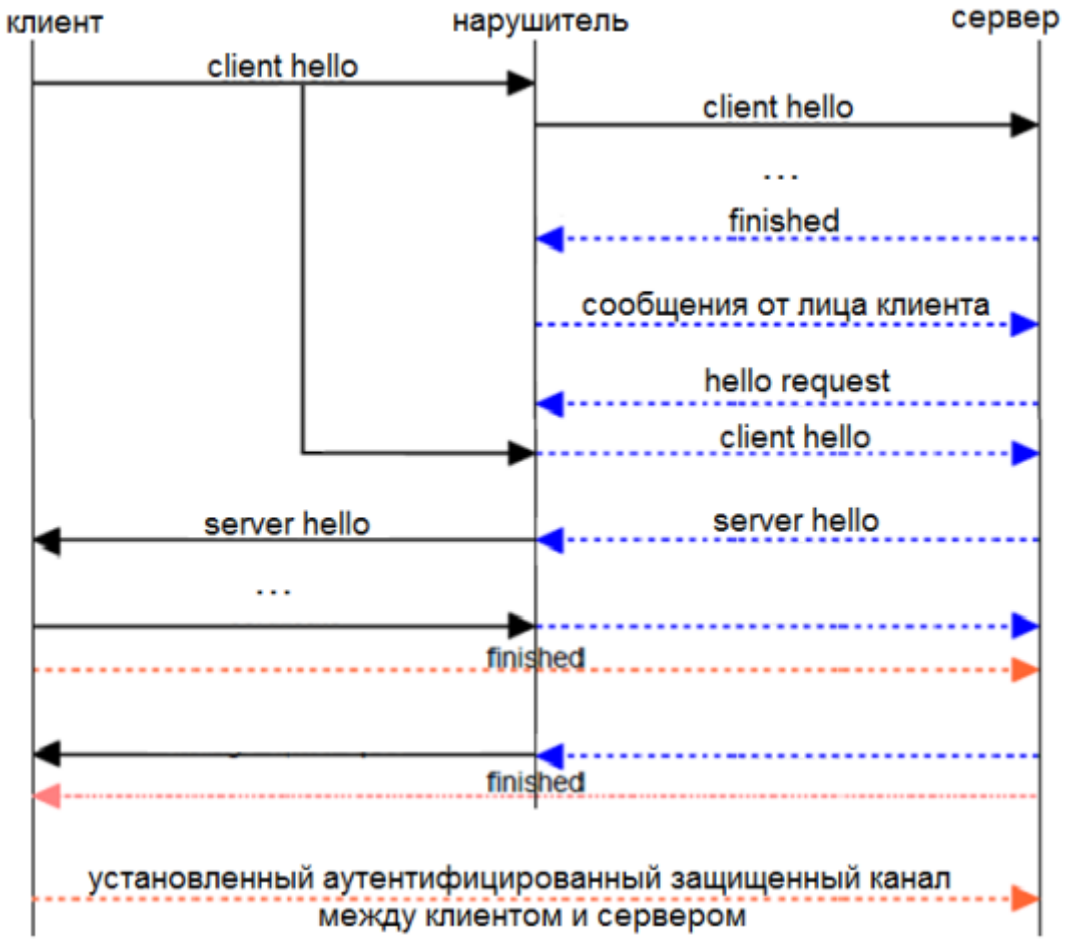
$$TLSTREE(K_{root}, i) =$$
$$= Divers_3\left(Divers_2\left(Divers_1\left(K_{root}, STR_8(i \,\&\, C_1)\right), STR_8(i \,\&\, C_2)\right), STR_8(i \,\&\, C_3)\right)$$

Как рассчитывать константы C_1, C_2, C_3:

- C_3: 2^{кол-во нулей в конце C_3} < макс. количества сообщений (64 для Магмы и 4096 для Кузнечика);
- C_2: {кол-во нулей в конце C_2} − {кол-во нулей в конце C_3} < 13
- C_1: {кол-во нулей в конце C_1} − {кол-во нулей в конце C_2} < 13

КРИПТОПРО

Handshake/ Атака Марша Рэя



КРИПТОПРО