

# Принципы построения отечественных криптонаборов для TLS 1.2

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

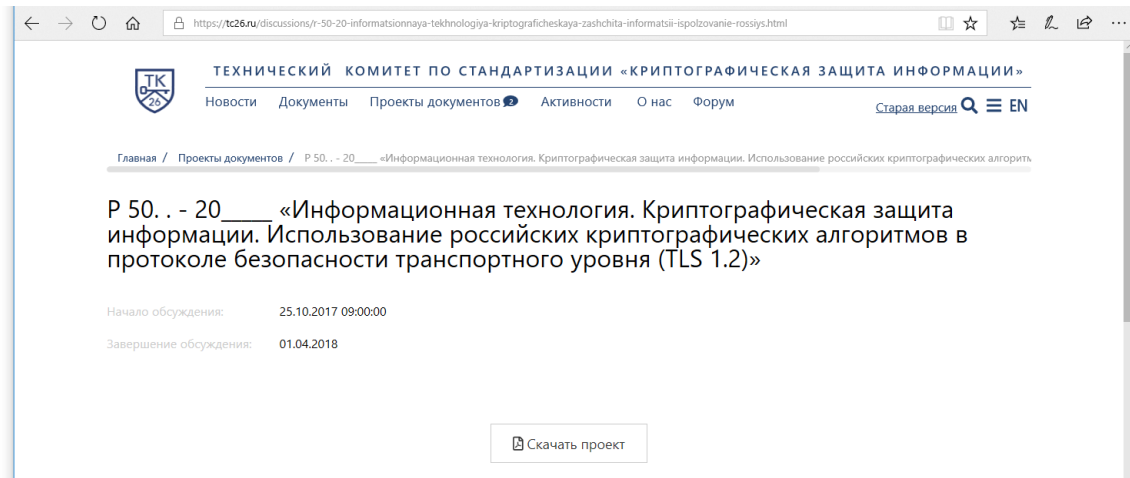
Смышляева Екатерина Сергеевна,  
инженер-аналитик



РусКрипто'2018

# Февраль 2017—март 2018

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



The screenshot shows a web browser window with the URL <https://tc26.ru/discussions/r-50-20-informatsionnaya-tehnologiya-kriptograficheskaya-zashchita-informatsii-ispolzovanie-rossijs.html>. The page header includes the logo of the Technical Committee for Standardization «Cryptographic Protection of Information» (TK 26) and navigation links: [Новости](#), [Документы](#), [Проекты документов](#) (active), [Активности](#), [О нас](#), [Форум](#). There is also a search icon and a language selector set to EN. The breadcrumb trail is: [Главная](#) / [Проекты документов](#) / [Р 50. . - 20\\_\\_\\_](#) «Информационная технология. Криптографическая защита информации. Использование российских криптографических алгоритмов». The main content area displays the title: **Р 50. . - 20\_\_\_ «Информационная технология. Криптографическая защита информации. Использование российских криптографических алгоритмов в протоколе безопасности транспортного уровня (TLS 1.2)»**. Below the title, the discussion dates are listed: **Начало обсуждения:** 25.10.2017 09:00:00 and **Завершение обсуждения:** 01.04.2018. At the bottom of the page, there is a button labeled **Скачать проект**.

# Необходимость разработки

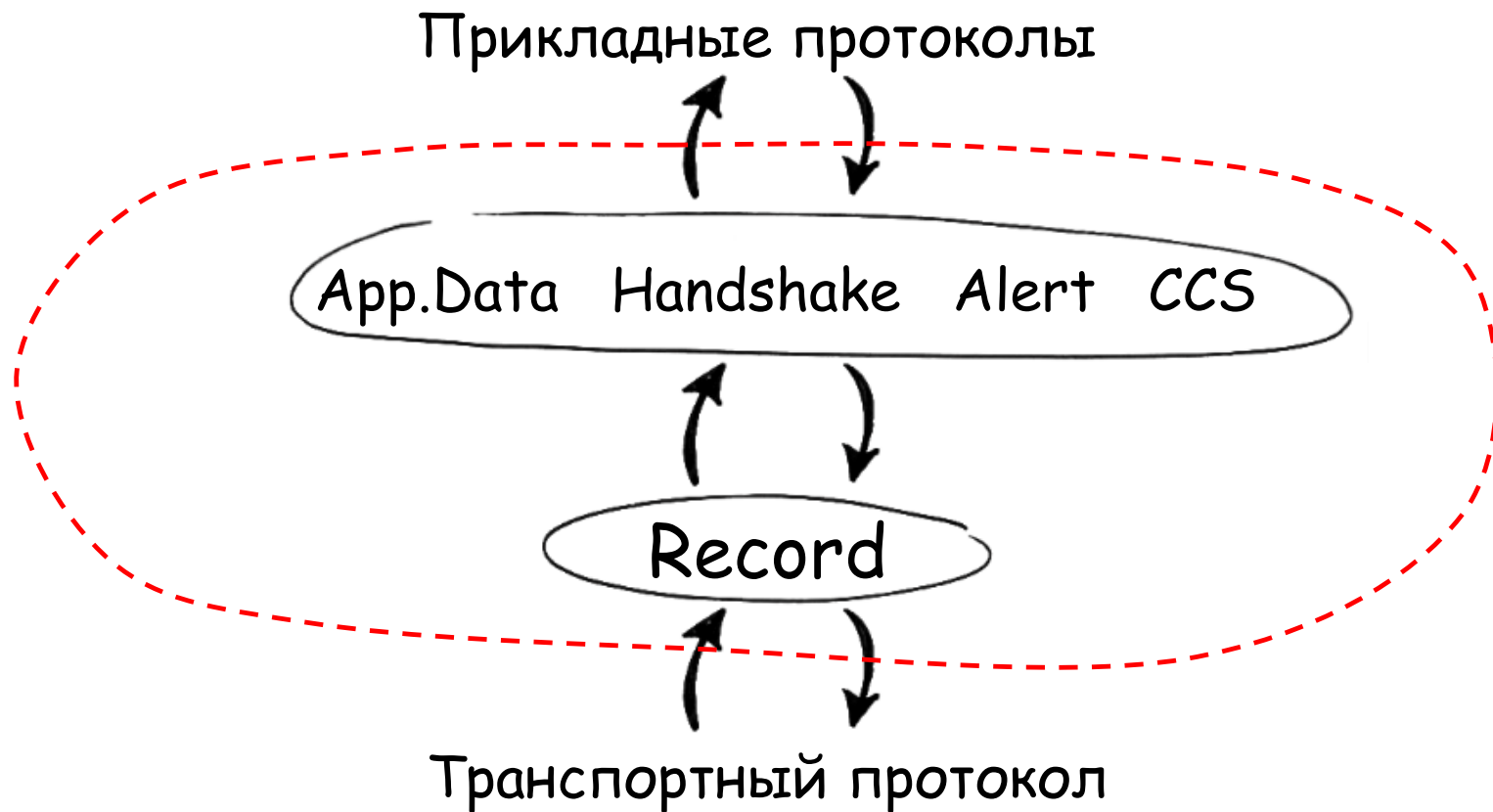
# Необходимость разработки Принципы построения

# 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.



# Что такое TLS



# Что такое TLS



# Что такое TLS

аутентификация сторон, выработка параметров безопасности

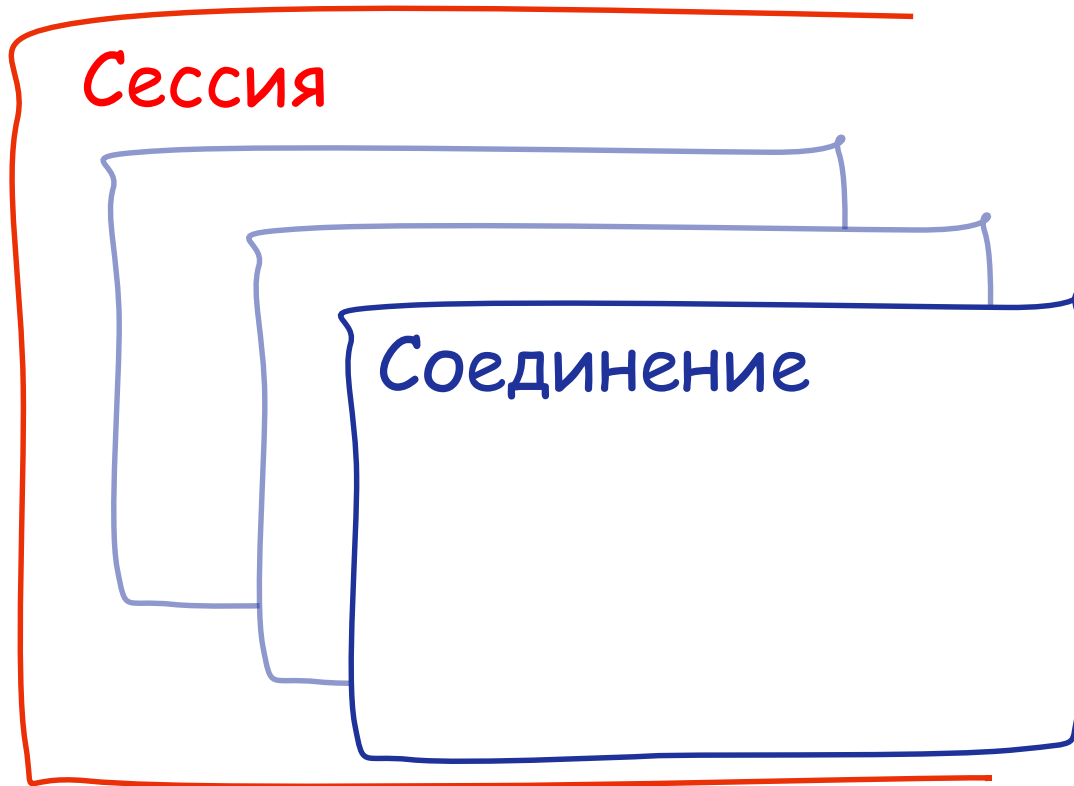
Handshake

Record

защита данных (записей)



# Что такое TLS



## Сессия

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

## Соединение

- случайные значения  $r_s$ ,  $r_c$
- ключевой материал

# 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.

# Криптонаборы

TK 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.

TK 26 2017-2018 год, новые криптонаборы:

- TLS\_GOSTR341112\_256\_WITH\_KUZNYECHIK\_CTR\_OMAC;
- TLS\_GOSTR341112\_256\_WITH\_MAGMA\_CTR\_OMAC.

# 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.

# Причины разработки новых криптонаборов

- [98] Vandouy S, Adams C., Lecture Notes on Computer Science, Heidelberg
- [99] Wagner D., on Electronic Cryptography and Privacy
- [100] White A., Encrypted and Privacy
- [101] Wikipedia, on Security
- [102] Wright C., in *Encrypted* article 35, pp
- [104] Wright C., *Encrypted A* (2006).
- [87] Sheffer Y., *Hold Layer Security*, 2015. <https://tools.ietf.org/html/rfc6989>
- [88] Sheffer Y., *Hold Security (TLS)*, <https://tools.ietf.org/html/rfc6989>
- [89] Smyth B., *Pivo Applications*. In: <https://www.usenix.org/conference/ussp12>
- [90] Stevens M., *Be Collision for File - CRYPTO 2013* Springer, Cham
- [91] Stevens M., *Lee applications*. In: [https://doi.org/10.1007/978-3-642-31131-2\\_20](https://doi.org/10.1007/978-3-642-31131-2_20)
- [92] Sun Q., *Simca: identification of and Privacy*, pp. 2007
- [93] *Thank you Bob*. <https://web.archive.org/web/19940401000000/http://www.freakab.com/data/1994/>
- [94] *The FREAK Attack*. <https://censys.io/lookup?q=freak>
- [95] Vanhoof M., *Play and TLS*. In: *Jur Association* (2011)
- [96] Vandouy S., *See WTLS*. In: *Adv Notes in Cryptography*. Springer, Berlin
- [97] <https://www.waack.com/>
- [64] Mantin L., P. Cramer R., *Computer Science*
- [65] Mantin L., *S Fast Software*, pp. 152-164.
- [66] Marlinspike, <https://www.blackhat-us-12.com/au/program/12-01-m Marlinspike-12-01-m Marlinspike.pdf>
- [67] McGrew D., *Detection of Weak Security Sympt*. Presented at <https://www.usenix.org/conference/ussp12>
- [68] Meneses A., *CRC Press* (2017)
- [69] *Microgamme attack on H*. Communicat [http://dx.doi.org/10.1007/978-3-642-31131-2\\_20](http://dx.doi.org/10.1007/978-3-642-31131-2_20)
- [70] Rescorla E., *tls13-21*, 2010. <https://tools.ietf.org/html/rfc7525>
- [71] Rescorla E., *Renegotiate*. <https://tools.ietf.org/html/rfc5246>
- [72] Nandi M., *Report* 2007
- [73] Paterson K., *Boehm*, 2010. <https://www.usenix.org/conference/ussp12>
- [74] Padding Ora <https://cve.mitre.org/cve/2015/2164/>
- [75] Paul G., *Ma In: Adams C* 2007. *Lecture Notes on Computer Science*, Heidelberg
- [64] Mantin L., P. Cramer R., *Computer Science*
- [51] Green M., *A Fast*. <https://blog.cloudflare.com/2012/08/28/heartbleed-attack/>
- [52] <http://heartbleed.com/>
- [53] Heninger N., *Du Detection of Weak Security Sympt*. Presented at <https://www.usenix.org/conference/ussp12>
- [54] <https://www.ia.ac.cn/~liang/>
- [55] Iwata T., *Kuros 2003*. LNCS, vol. 2803
- [56] Iwata T., *Kuros In proceedings of India*, December 2001. *Lecture Notes in Computer Science*, Springer, Berlin
- [57] Kobay J., *Comp V*, (eds) Fast So vol. 2365, pp. 26
- [58] Klima V., *Pika Cryptology ePrint* <https://eprint.iacr.org/2011/113>
- [48] Garman C., *Pate Recovery Attacks*, volume 2139 of *Information Theory*
- [49] Georgiev M., *Ivy dangerous code # Proceedings of the (CCS 12)*, ACM, <http://dx.doi.org/10.1145/2191231.2191231>
- [61] Lenstra A. K., *It was wrong, What* <https://eprint.iacr.org/2007/113>
- [62] Liberatore M., *ACM Conference*
- [63] Maitra S., *Paul (eds) Fast Software*, vol. 6733, pp. 19
- [40] Dai W., *An Attack*, 2002. <https://www.usenix.org/conference/ussp12>
- [41] Delignat-Lavaud *Exploits*. In: <https://bh.tl.org/>
- [42] Dierks T., *Alto RFC 4346*, 2006. <https://tools.ietf.org/html/rfc4346>
- [43] Dierks T., *Rescor RFC 5246*, August 2008. <https://tools.ietf.org/html/rfc5246>
- [45] Dolev D., *Yao A. Information Theory*
- [46] Fluhrer S., *Mant RCL*. In: *Vauden 2001*. *Lecture Notes in Computer Science*, Springer, Berlin
- [47] Fluhrer S., *R. M Generator*. In: *Ge Encryption*, FSEI Springer, Berlin
- [48] Garman C., *Pate Recovery Attacks*, volume 2139 of *Information Theory*
- [49] Georgiev M., *Ivy dangerous code # Proceedings of the (CCS 12)*, ACM, <http://dx.doi.org/10.1145/2191231.2191231>
- [61] Lenstra A. K., *It was wrong, What* <https://eprint.iacr.org/2007/113>
- [62] Liberatore M., *ACM Conference*
- [63] Maitra S., *Paul (eds) Fast Software*, vol. 6733, pp. 19
- [51] Bhargavan, *Collaboration*. ACM SIGS, pp. 456-467. <https://doi.org/10.1145/1219123.1219123>
- [32] Bhargavan, *TLS, IKE, JKE*, 2010. <https://www.usenix.org/conference/ussp12>
- [33] Bleichenbacher, *RSA Errors*. Springer-Verlag, <http://arc.pdf>
- [34] Brumley D., *Archive*. <https://eprint.iacr.org/2007/113>
- [35] Brumley D., *The 16th E* 2011.
- [25] Bellare M., *Advances in Cryptology*, volume 9 of *Information Theory*
- [37] Canetti R., *Secure channels*. LNCS, vol. 2139 (2013)
- [27] Beurdouche Piroati A., *The comprom 2015*. <http://see.inf.univr.it/~piroati/papers/2015-01-08-see.pdf>
- [28] Bhargavan, *Handshakes*. IEEE Symp <https://www.usenix.org/conference/ussp12>
- [29] Bhargavan, *Security* (7) <https://doi.org/10.1145/1219123.1219123>
- [30] Bereschini D., *Lioy A.: On the Robustness of TLS Security Protocols*. In: *Lecture Notes in Computer Science*, Springer, Berlin
- [11] <http://www.usenix.org/conference/ussp12>
- [12] <http://www.usenix.org/conference/ussp12>
- [13] *P. 50.1.1. format*. <https://tools.ietf.org/html/rfc5246>
- [14] *Informa*. <https://tools.ietf.org/html/rfc5246>
- [15] *Abdalla the Sec - ASI*, vol. 197
- [16] *Adrian, Henning Zanella Hellme Security*
- [17] *AlFard of RC2*. <https://tools.ietf.org/html/rfc5246>
- [25] Bellare M., *Advances in Cryptology*, volume 9 of *Information Theory*
- [26] Beery T., *SI* 2013 (2013)
- [27] Beurdouche Piroati A., *The comprom 2015*. <http://see.inf.univr.it/~piroati/papers/2015-01-08-see.pdf>
- [28] Bhargavan, *Handshakes*. IEEE Symp <https://www.usenix.org/conference/ussp12>
- [29] Bhargavan, *Security* (7) <https://doi.org/10.1145/1219123.1219123>
- [30] Bereschini D., *Lioy A.: On the Robustness of TLS Security Protocols*. In: *Lecture Notes in Computer Science*, Springer, Berlin
- [11] <http://www.usenix.org/conference/ussp12>
- [12] <http://www.usenix.org/conference/ussp12>
- [13] *P. 50.1.1. format*. <https://tools.ietf.org/html/rfc5246>
- [14] *Informa*. <https://tools.ietf.org/html/rfc5246>
- [15] *Abdalla the Sec - ASI*, vol. 197
- [16] *Adrian, Henning Zanella Hellme Security*
- [17] *AlFard of RC2*. <https://tools.ietf.org/html/rfc5246>
- [25] Bellare M., *Advances in Cryptology*, volume 9 of *Information Theory*
- [26] Beery T., *SI* 2013 (2013)
- [27] Beurdouche Piroati A., *The comprom 2015*. <http://see.inf.univr.it/~piroati/papers/2015-01-08-see.pdf>
- [28] Bhargavan, *Handshakes*. IEEE Symp <https://www.usenix.org/conference/ussp12>
- [29] Bhargavan, *Security* (7) <https://doi.org/10.1145/1219123.1219123>
- [30] Bereschini D., *Lioy A.: On the Robustness of TLS Security Protocols*. In: *Lecture Notes in Computer Science*, Springer, Berlin
- [2] Черемухин А. В., *Криптографические протоколы: основные свойства и уязвимости*, ПДМ, 2009, приложение № 2, сс. 115-150 (2009).
- [3] *ГОСТ Р 34.10-2012 Информационная технология. Криптографическая защита информации. Процессы формирования и проверки электронной цифровой подписи*. М.: «Стандартинформ» (2013). <http://protect.gost.ru/v.aspx?control=7&id=180151>
- [4] *ГОСТ Р 34.11-2012 Информационная технология. Криптографическая защита информации. Функция эллиптической кривой*. М.: «Стандартинформ» (2013). <http://protect.gost.ru/v.aspx?control=31&id=180209>
- [5] *ГОСТ Р 34.12-2015 Информационная технология. Криптографическая защита информации. Блочные шифры*. Федеральное агентство по техническому регулированию и метрологии. М.: «Стандартинформ» (2015).
- [6] *ГОСТ Р 34.13-2015 Информационная технология. Криптографическая защита информации. Режимы работы блочных шифров*. Федеральное агентство по техническому регулированию и метрологии. М.: «Стандартинформ» (2015).
- [7] Леонтьев С. Е., Попов В. О., Смышляев С. В., *Противодействие атакам на протокол TLS*. Системы высокой доступности, М.: «РадиоТехника», т. 8, к. 2, сс. 109-115 (2012).
- [8] *Информационная технология. Криптографическая защита информации. Рекомендации по стандартизации. Использование наборов алгоритмов шифрования на основе ГОСТ 28147-89 для протокола безопасности транспортного уровня (TLS) (2014)*.
- [9] *Проект рекомендаций по стандартизации. Информационная технология. Криптографическая защита информации. Использование российских криптографических алгоритмов в протоколе безопасности транспортного уровня (TLS 1.2) (2017)*. [https://www.tc26.ru/standards/draft/2009A2D009A26\\_TLS\\_2015.pdf](https://www.tc26.ru/standards/draft/2009A2D009A26_TLS_2015.pdf).
- [10] Смышляев С.В., Алексеев Е.К., Ахметзянова Л.Р., Смышляева Е.С., Мешков Д.А.: *О защите на ключ*. Часть 1. Блог компании «КриптоПро», 2017. <http://cryptopro.ru/blog/2017/05/17/o-nagruzke-na-kluch-chart-1>.

уязвимости, известные на сегодняшний момент



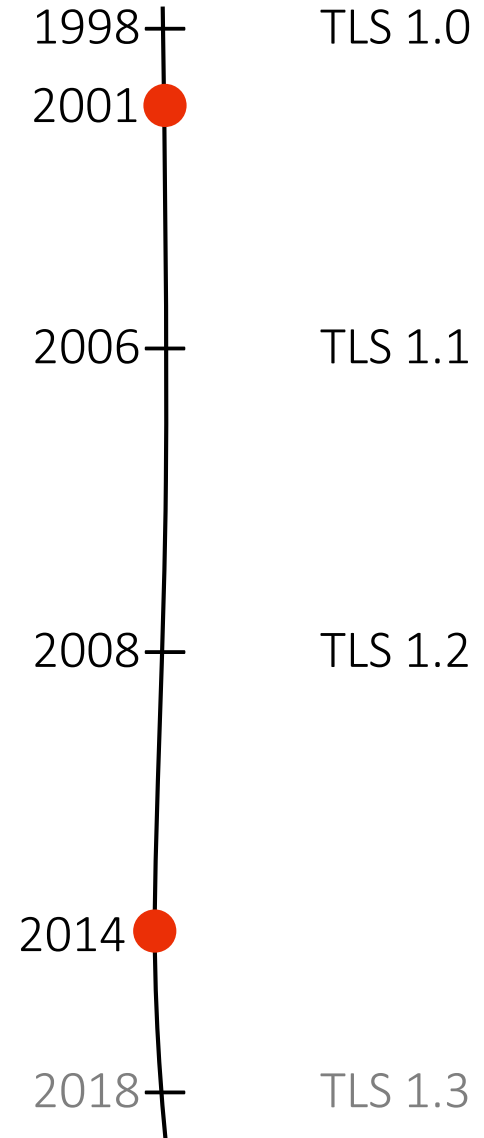
# Record

# Атаки на режим СВС

# Record/ Атака Воденя (POODLE)

Атака на режим CBC с использованием дополнения PKCS#5.

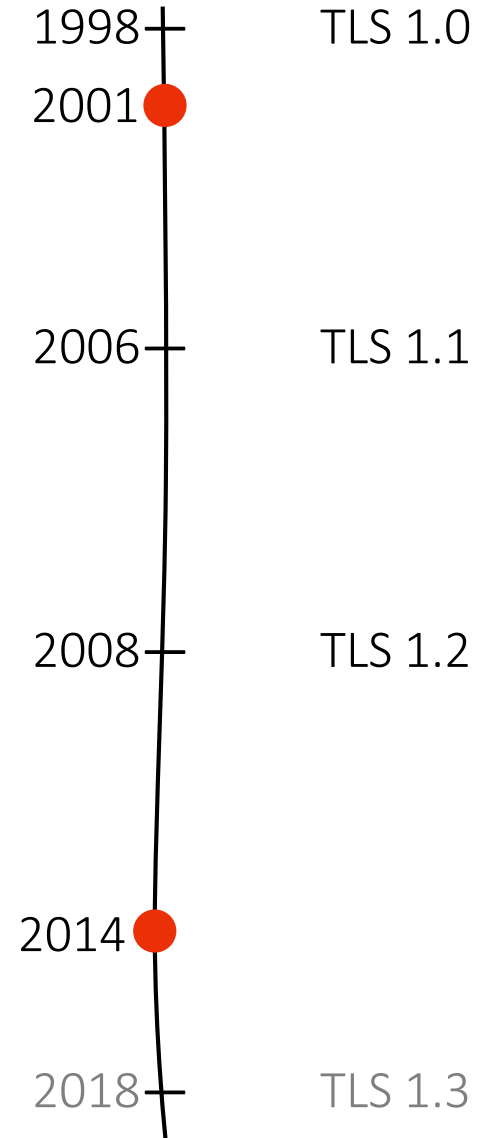
Противнику требуется после отправки модифицированного пакета данных получить информацию о результате проверки дополнения.



# Record/ Атака Воденя (POODLE)

## TLS 1.0

В случае некорректного дополнения посылалось специальное оповещение об ошибке `decryption_failed`.

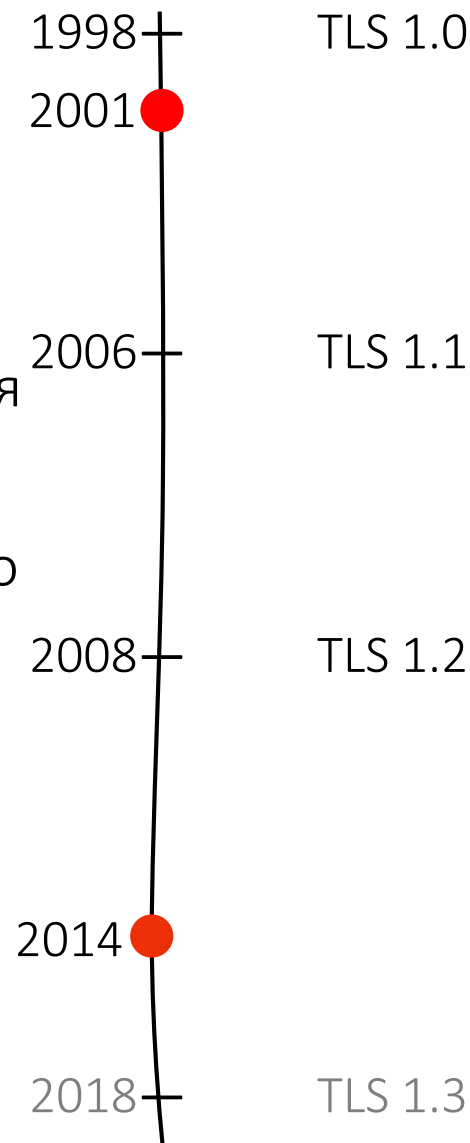


# Record/ Атака Воденя (POODLE)

## TLS 1.1

Запрещено использовать оповещение *decryption\_failed*, вместо него всегда посылается оповещение *bad\_record\_mac*.

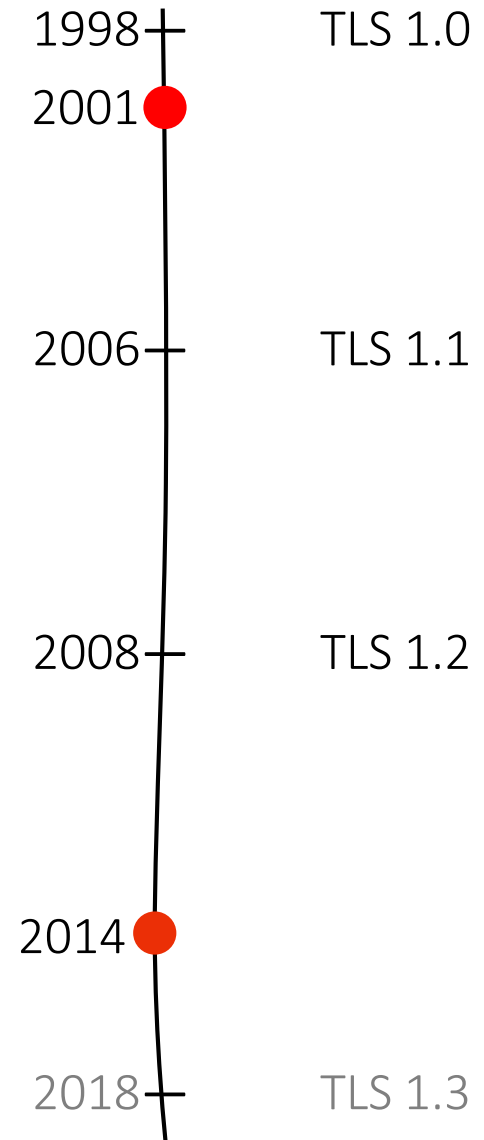
Опубликована модификация атаки: предложено различать типы ошибок по времени реакции сервера на модифицированный шифртекст.



# Record/ Атака Воденя (POODLE)

## TLS 1.2

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

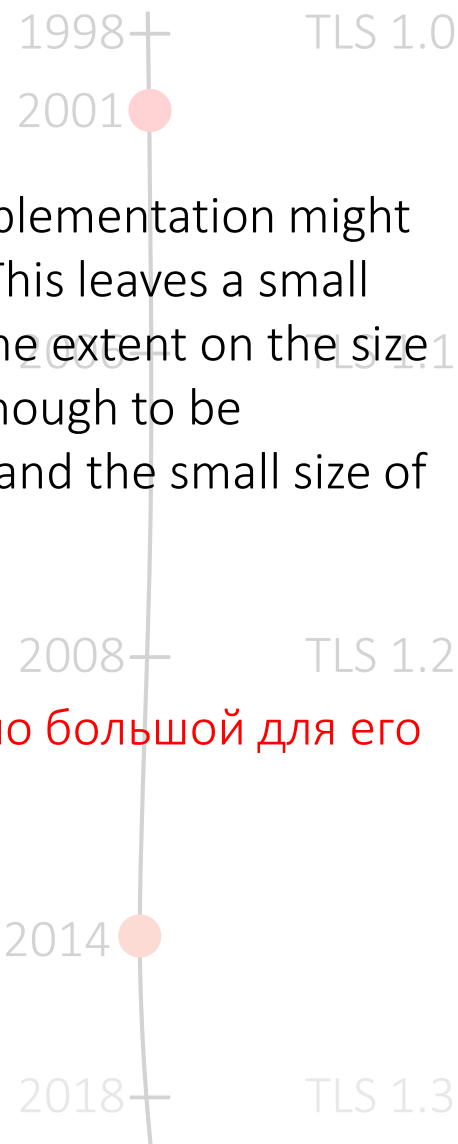


# Record/ Атака Воденя (POODLE)

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.»

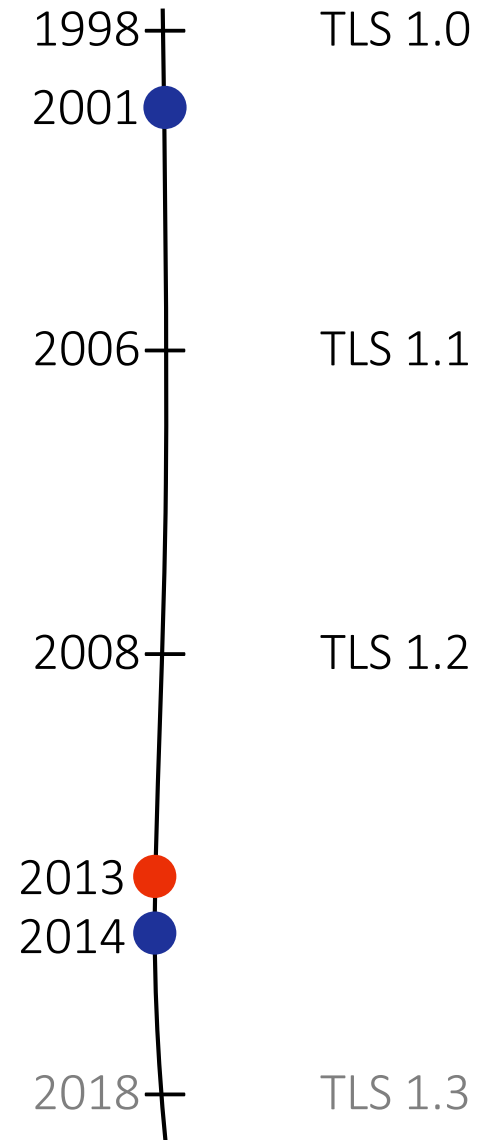
Предполагается, что временной канал недостаточно большой для его использования.



# Record/ Атака Воденя (POODLE)/ Lucky13

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

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.



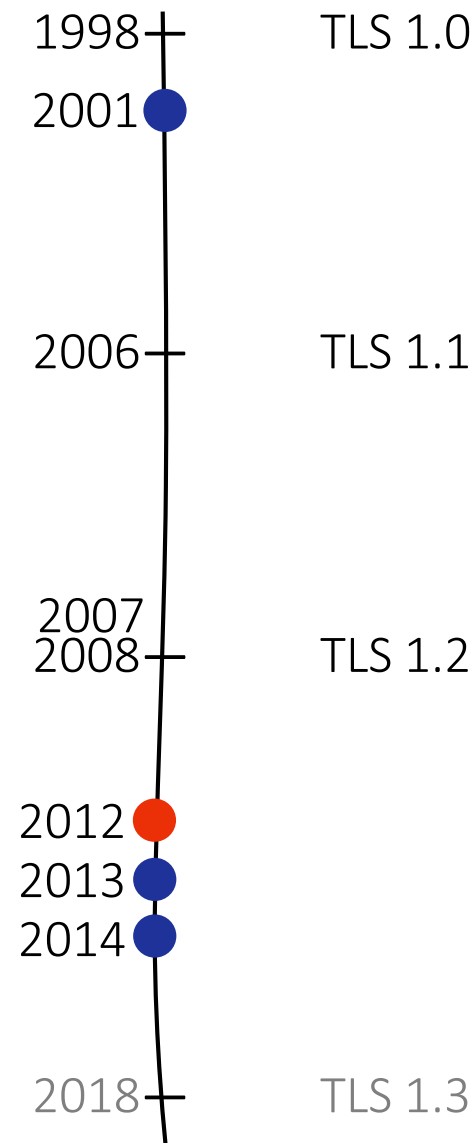


# Record/ Атака Воденя (POODLE)/ Расширение побочного канала

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

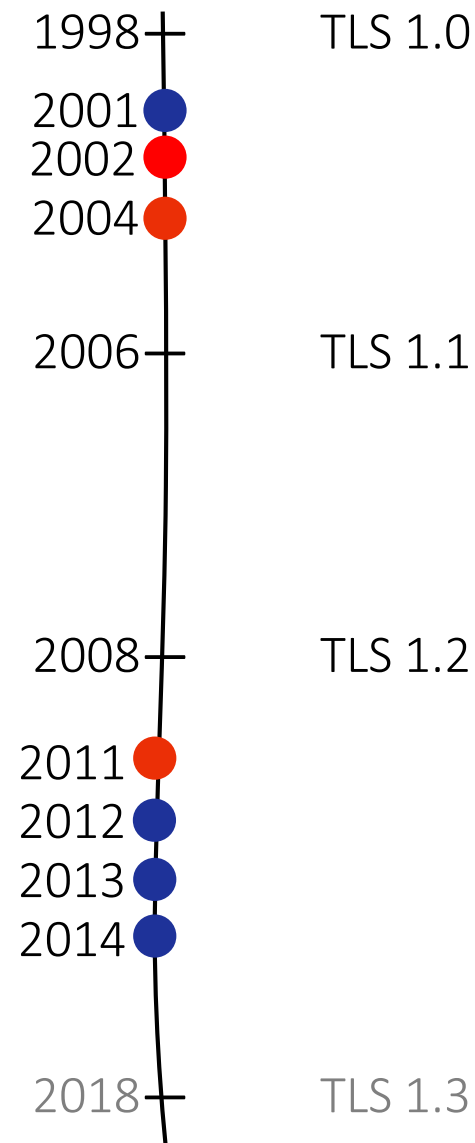
RFC 5246, пункт 6.2.3.2:

Дополнение может иметь любую длину, не превосходящую 255 байт.



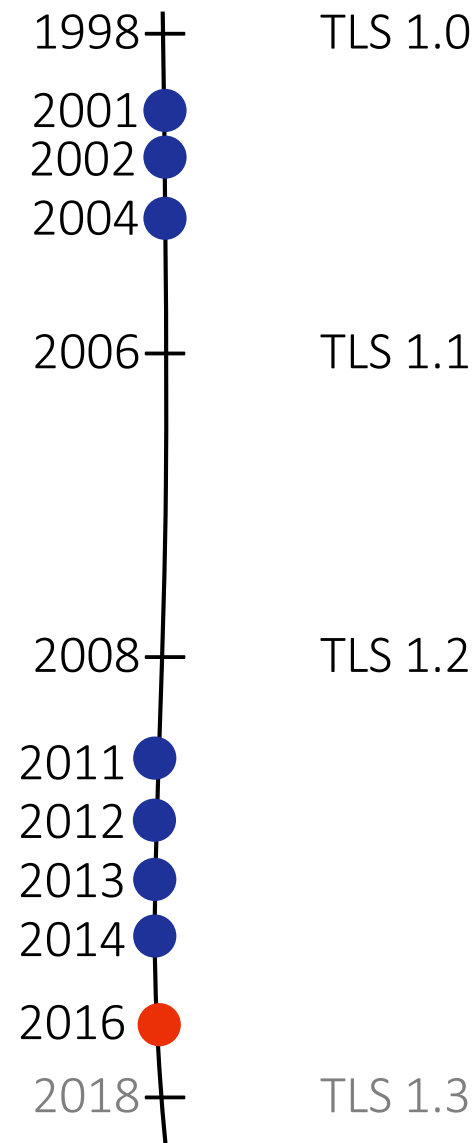
# Record/ Атака BEAST

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



# Record/ Атака Sweet32

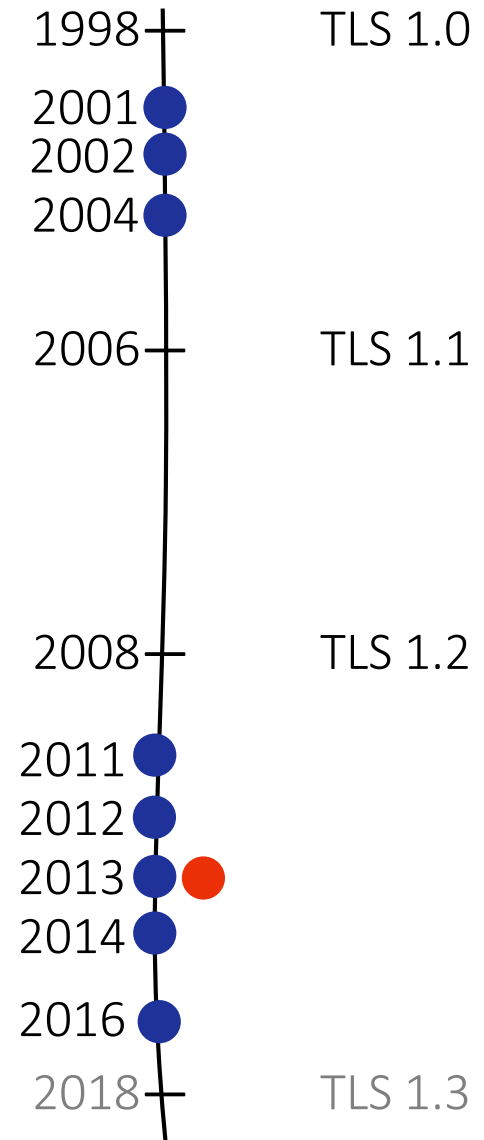
Атака на режим CBC с использованием произвольного 64-битного. Базируется на уязвимости режима CBC к атакам на основе парадокса задачи о днях рождения.



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

# Record/ Атака Аль-Фардана и др.

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



И новые, и старые российские криптонаборы

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

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.

# Record/ Новые криптонаборы

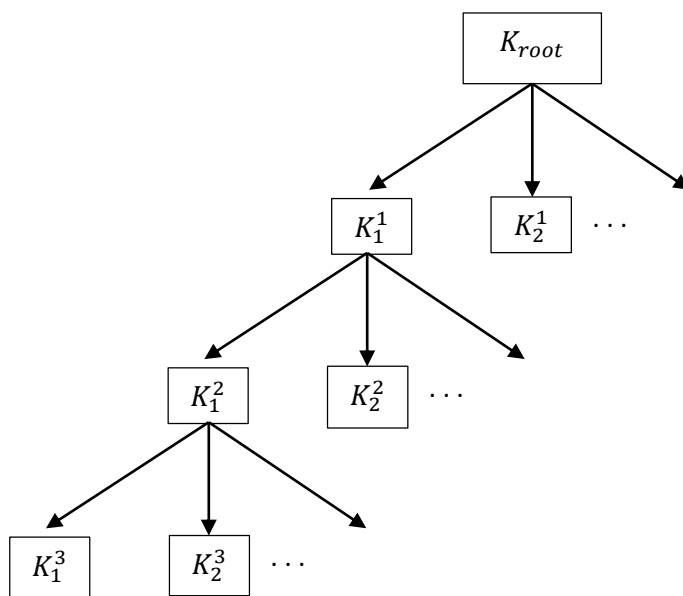
Ключи образуют иерархию:

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

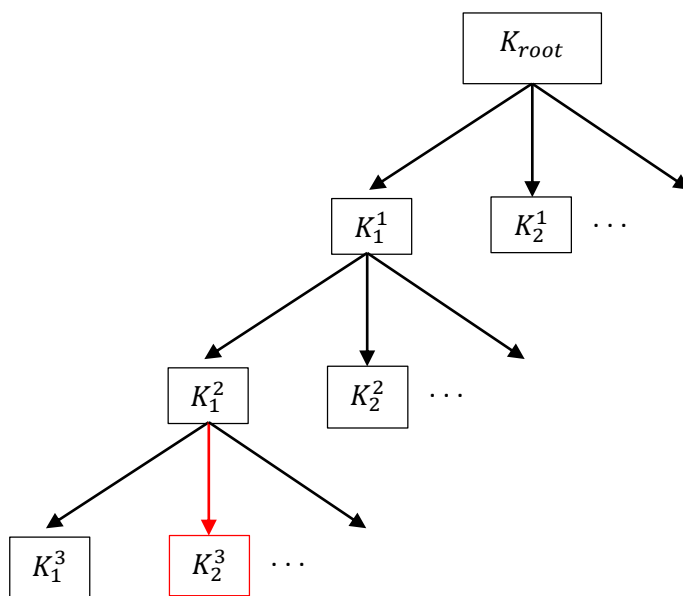
Создание такой иерархии достигается за счет использования механизмов внешнего и внутреннего преобразования ключей (external и internal re-keying), которые позволяют увеличивать объем обрабатываемых данных, оставаясь в рамках безопасной нагрузки на ключ.

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





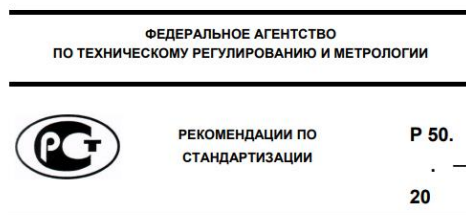
$$TLSTREE(K_{root}, i) = Divers_3(Divers_2(Divers_1(K_{root}, i \& C_1), i \& C_2), i \& C_3)$$



$$TLSTREE(K_{root}, i) = \text{Divers}_3(\text{Divers}_2(\text{Divers}_1(K_{root}, i \& C_1), i \& C_2), i \& C_3)$$

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

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



Информационная технология  
КРИПТОГРАФИЧЕСКАЯ ЗАЩИТА ИНФОРМАЦИИ  
Криптографические алгоритмы, сопутствующие  
применению алгоритмов блочного шифрования

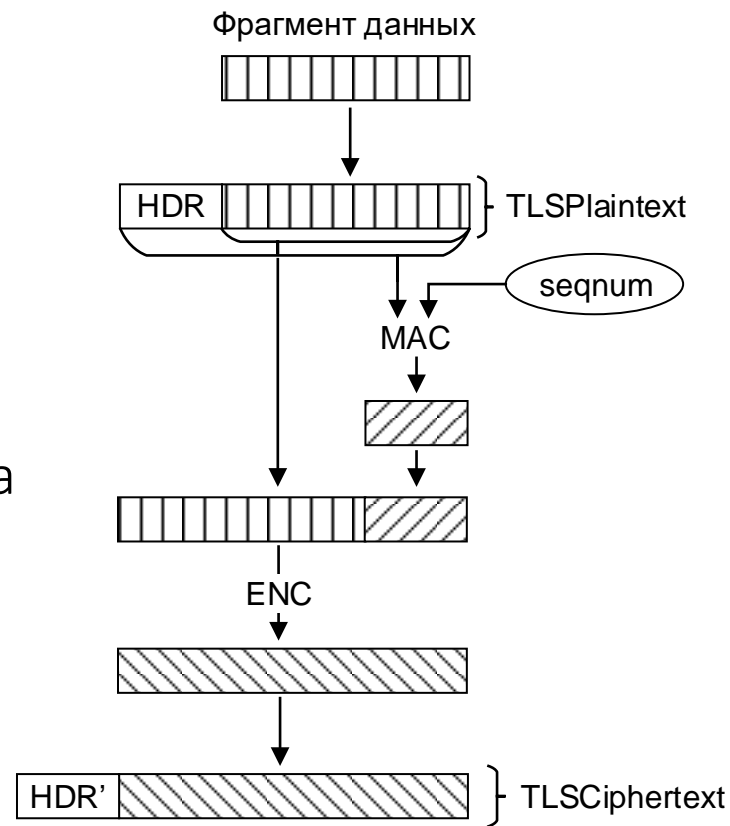
Издание официальное



Москва  
Стандартинформ  
2017

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.



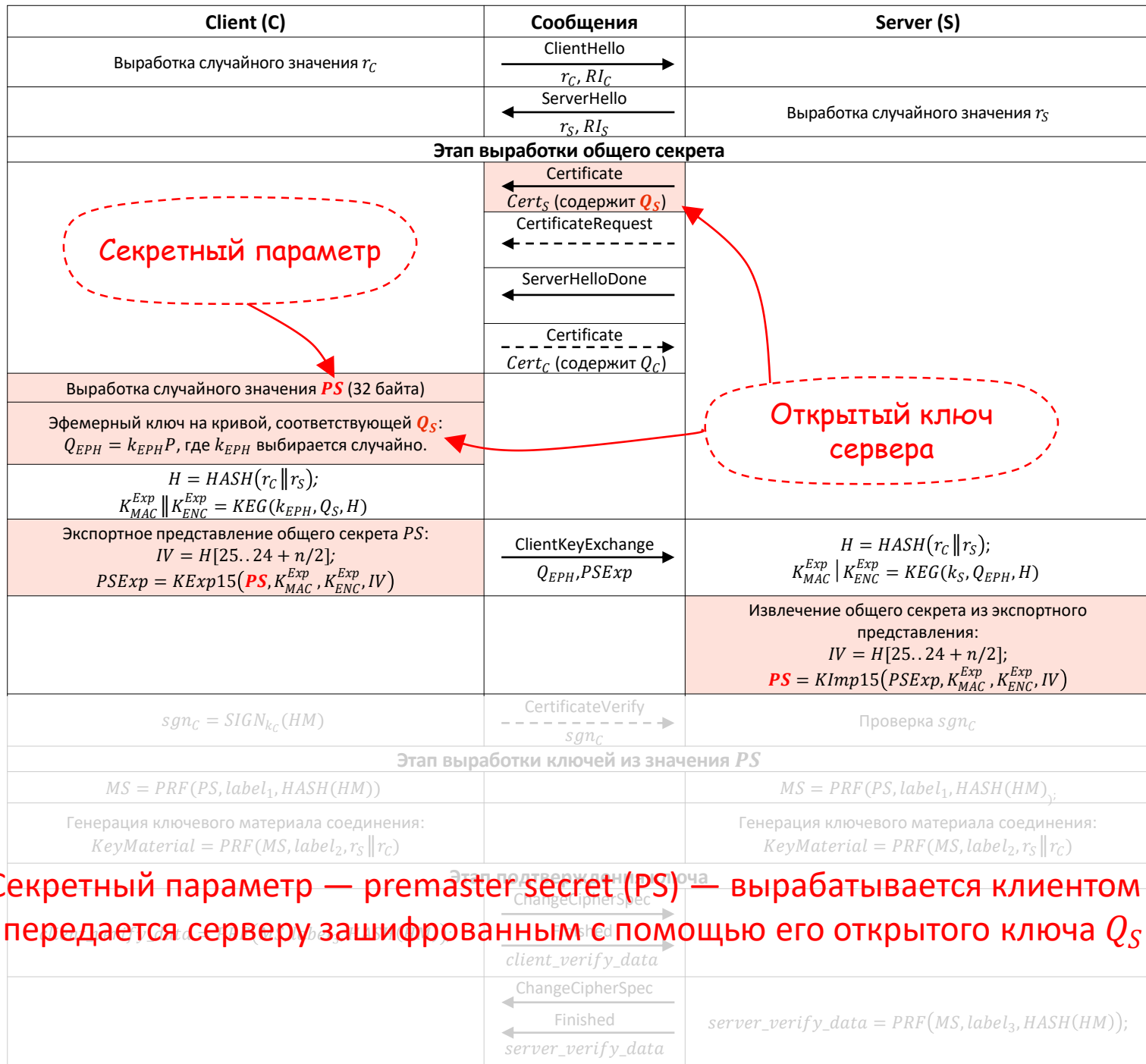
# Handshake

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

Client (C)	Сообщения	Server (S)
Выработка случайного значения $r_C$	$\xrightarrow{\text{ClientHello}}$ $r_C, RI_C$	
	$\xleftarrow{\text{ServerHello}}$ $r_S, RI_S$	Выработка случайного значения $r_S$
<b>Этап выработки общего секрета</b>		
	$\xleftarrow{\text{Certificate}}$ $Cert_S$ (содержит $Q_S$ ) $\xleftarrow{\text{CertificateRequest}}$ $\xleftarrow{\text{-----}}$	
	$\xleftarrow{\text{ServerHelloDone}}$	
	$\xrightarrow{\text{Certificate}}$ $Cert_C$ (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.		
$H = \text{HASH}(r_C \parallel r_S);$ $K_{MAC}^{Exp} \parallel K_{ENC}^{Exp} = \text{KEG}(k_{EPH}, Q_S, H)$		
Экспортное представление общего секрета $PS$ : $IV = H[25..24 + n/2];$ $PSExp = \text{KExp15}(PS, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$	$\xrightarrow{\text{ClientKeyExchange}}$ $Q_{EPH}, PSExp$	$H = \text{HASH}(r_C \parallel r_S);$ $K_{MAC}^{Exp} \parallel K_{ENC}^{Exp} = \text{KEG}(k_S, Q_{EPH}, H)$
		Извлечение общего секрета из экспортного представления: $IV = H[25..24 + n/2];$ $PS = \text{KImp15}(PSExp, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$
$sgn_C = \text{SIGN}_{k_C}(HM)$	$\xrightarrow{\text{CertificateVerify}}$ $sgn_C$	Проверка $sgn_C$
<b>Этап выработки ключей из значения <math>PS</math></b>		
$MS = \text{PRF}(PS, label_1, \text{HASH}(HM))$		$MS = \text{PRF}(PS, label_1, \text{HASH}(HM));$
Генерация ключевого материала соединения: $KeyMaterial = \text{PRF}(MS, label_2, r_S \parallel r_C)$		Генерация ключевого материала соединения: $KeyMaterial = \text{PRF}(MS, label_2, r_S \parallel r_C)$
<b>Этап подтверждения ключа</b>		
$client\_verify\_data = \text{PRF}(MS, label_3, \text{HASH}(HM));$	$\xrightarrow{\text{ChangeCipherSpec}}$ $\xrightarrow{\text{Finished}}$ $client\_verify\_data$	
	$\xleftarrow{\text{ChangeCipherSpec}}$ $\xleftarrow{\text{Finished}}$ $server\_verify\_data$	$server\_verify\_data = \text{PRF}(MS, label_3, \text{HASH}(HM));$

Client (C)	Сообщения	Server (S)
Выработка случайного значения $r_C$	ClientHello $r_C, RI_C$ →	
	ServerHello $r_S, RI_S$ ←	Выработка случайного значения $r_S$
<b>Этап выработки общего секрета</b>		
	Certificate ← $Cert_S$ (содержит $Q_S$ )	
	CertificateRequest ←-----	
	ServerHelloDone ←	
	Certificate -----> $Cert_C$ (содержит $Q_C$ )	
Выработка случайного значения $PS$ (32 байта)		<div style="border: 2px dashed red; border-radius: 50%; padding: 10px; color: red; font-weight: bold;"> Оptionальные сообщения (для случай двусторонней аутентификации). </div>
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 $sgn_C$
<b>Этап выработки ключей из значения <math>PS</math></b>		
$MS = PRF(PS, label_1, HASH(HM))$		$MS = PRF(PS, label_1, HASH(HM));$
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S    r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S    r_C)$
<b>Этап подтверждения ключа</b>		
$client\_verify\_data = PRF(MS, label_3, HASH(HM));$	ChangeCipherSpec → Finished → $client\_verify\_data$	
	ChangeCipherSpec ← Finished ← $server\_verify\_data$	$server\_verify\_data = PRF(MS, label_3, HASH(HM));$





Секретный параметр

Открытый ключ сервера

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 $r_S$
<b>Этап выработки общего секрета</b>		
	Certificate ← Cert <sub>S</sub> (содержит $Q_S$ ) CertificateRequest ← ServerHelloDone ← Cert <sub>C</sub> (содержит $Q_C$ )	
<p style="color: red; font-weight: bold;">Аутентификация сервера осуществляется за счет доказательства корректного извлечения сервером секретного значения PS с помощью своего закрытого ключа, осуществляемого с помощью сообщений Finished</p>		
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 $sgn_C$
<b>Этап выработки ключей из значения PS</b>		
$MS = PRF(PS, label_1, HASH(HM))$		$MS = PRF(PS, label_1, HASH(HM));$
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S    r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S    r_C)$
<b>Этап подтверждения ключа</b>		
$client\_verify\_data = PRF(MS, label_3, HASH(HM));$	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 $r_S$
<b>Этап выработки общего секрета</b>		
	Certificate ← $Cert_S$ (содержит $Q_S$ )	
	CertificateRequest ← - - - - -	
	ServerHelloDone ←	
	Certificate → $Cert_C$ (содержит $Q_C$ )	
Выработка случайного значения $PS$ (32 байта) $Q_{EPH} = k_{EPH}P$ , 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 $sgn_C$
<b>Этап выработки ключей из значения <math>PS</math></b>		
$MS = PRF(PS, label_1, HASH(HM))$		$MS = PRF(PS, label_1, HASH(HM))_i;$
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S    r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S    r_C)$
<b>Этап подтверждения ключа</b>		
$client\_verify\_data = PRF(MS, label_3, HASH(HM));$	ChangeCipherSpec → Finished → $client\_verify\_data$	
	ChangeCipherSpec ← Finished ← $server\_verify\_data$	$server\_verify\_data = PRF(MS, label_3, HASH(HM));$

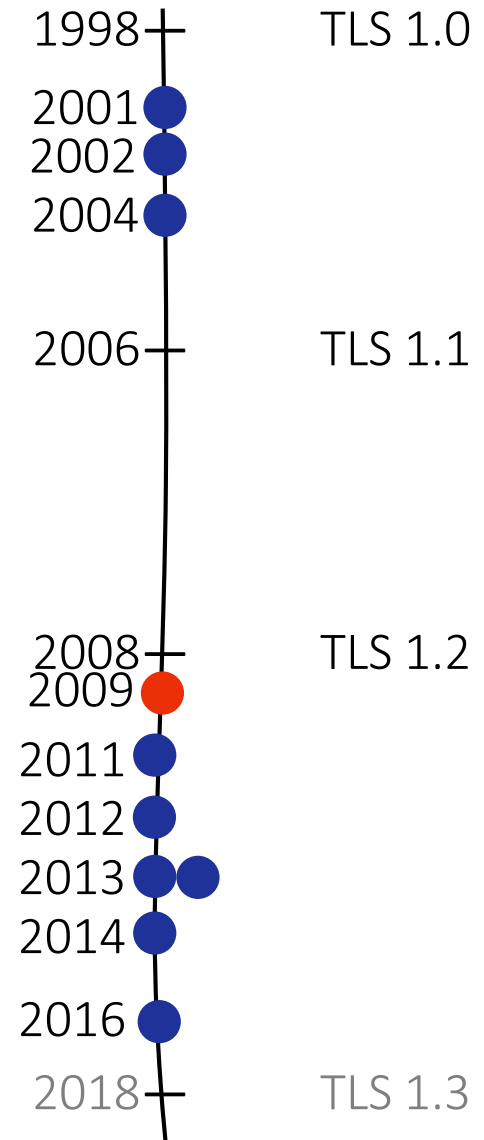
Клиент, если это требуется, аутентифицируется с помощью подписи.

Client (C)	Сообщения	Server (S)
Выработка случайного значения $r_C$	ClientHello $r_C, RI_C$	
	ServerHello $r_S, RI_S$	Выработка случайного значения $r_S$
<b>Этап выработки общего секрета</b>		
	Certificate ← Cert <sub>S</sub> (содержит $Q_S$ ) CertificateRequest ← ServerHelloDone ← Cert <sub>C</sub> (содержит $Q_C$ )	
<p style="color: red; font-weight: bold;">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);</p>		
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 $sgn_C$
<b>Этап выработки ключей из значения PS</b>		
$MS = PRF(PS, label_1, HASH(HM))$		$MS = PRF(PS, label_1, HASH(HM));$
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S    r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S    r_C)$
<b>Этап подтверждения ключа</b>		
$client\_verify\_data = PRF(MS, label_3, HASH(HM));$	ChangeCipherSpec Finished $client\_verify\_data$	
	ChangeCipherSpec Finished $server\_verify\_data$	$server\_verify\_data = PRF(MS, label_3, HASH(HM));$

Атаки «без доступа к ключу»

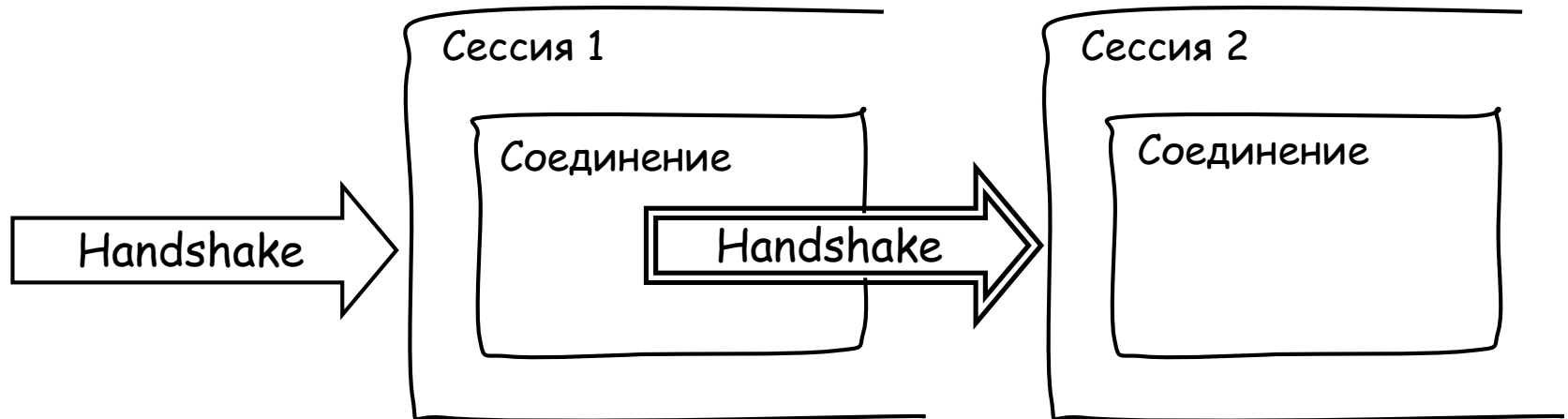
# Handshake/ Атака Марша Рэя

Навязывание сообщений на процедуру renegotiation (атака Марша Рэя)

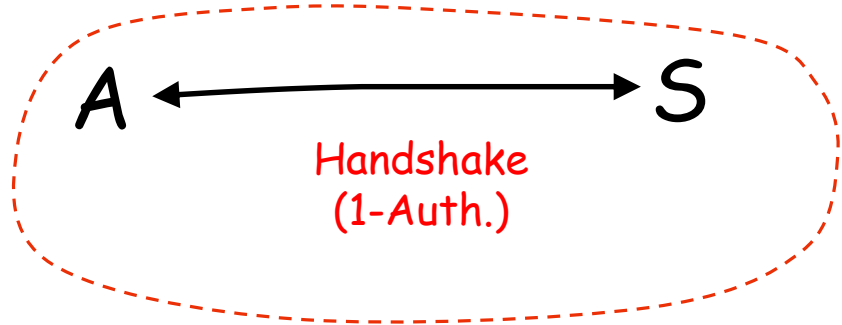


# Handshake/ Атака Марша Рэя

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

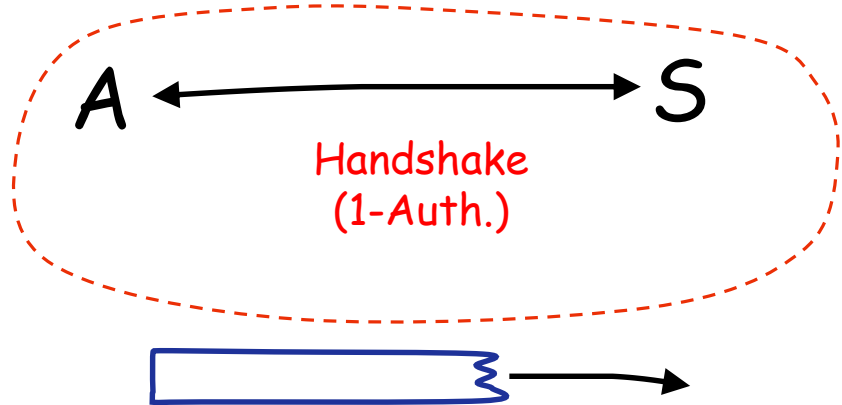


# Handshake/ Атака Марша Рэя

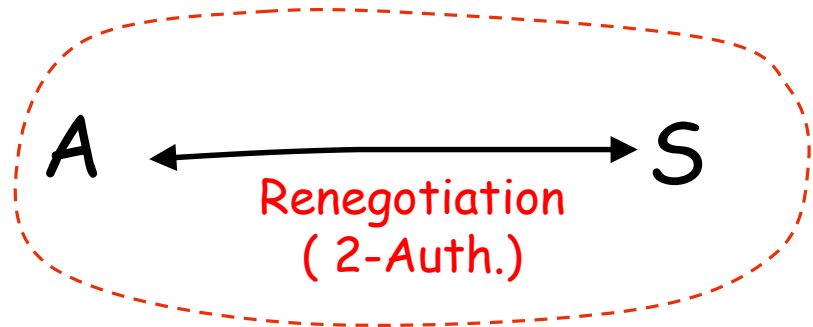
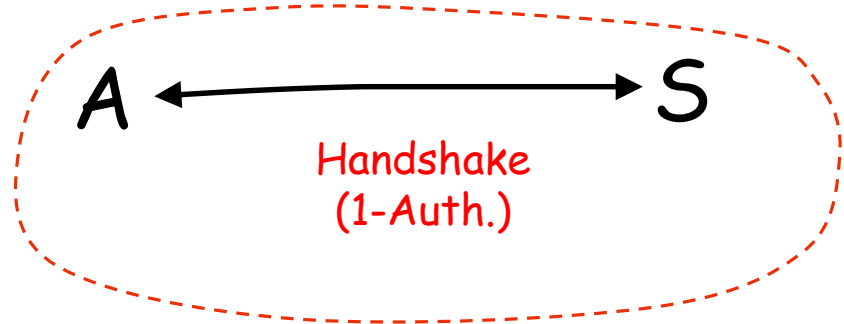




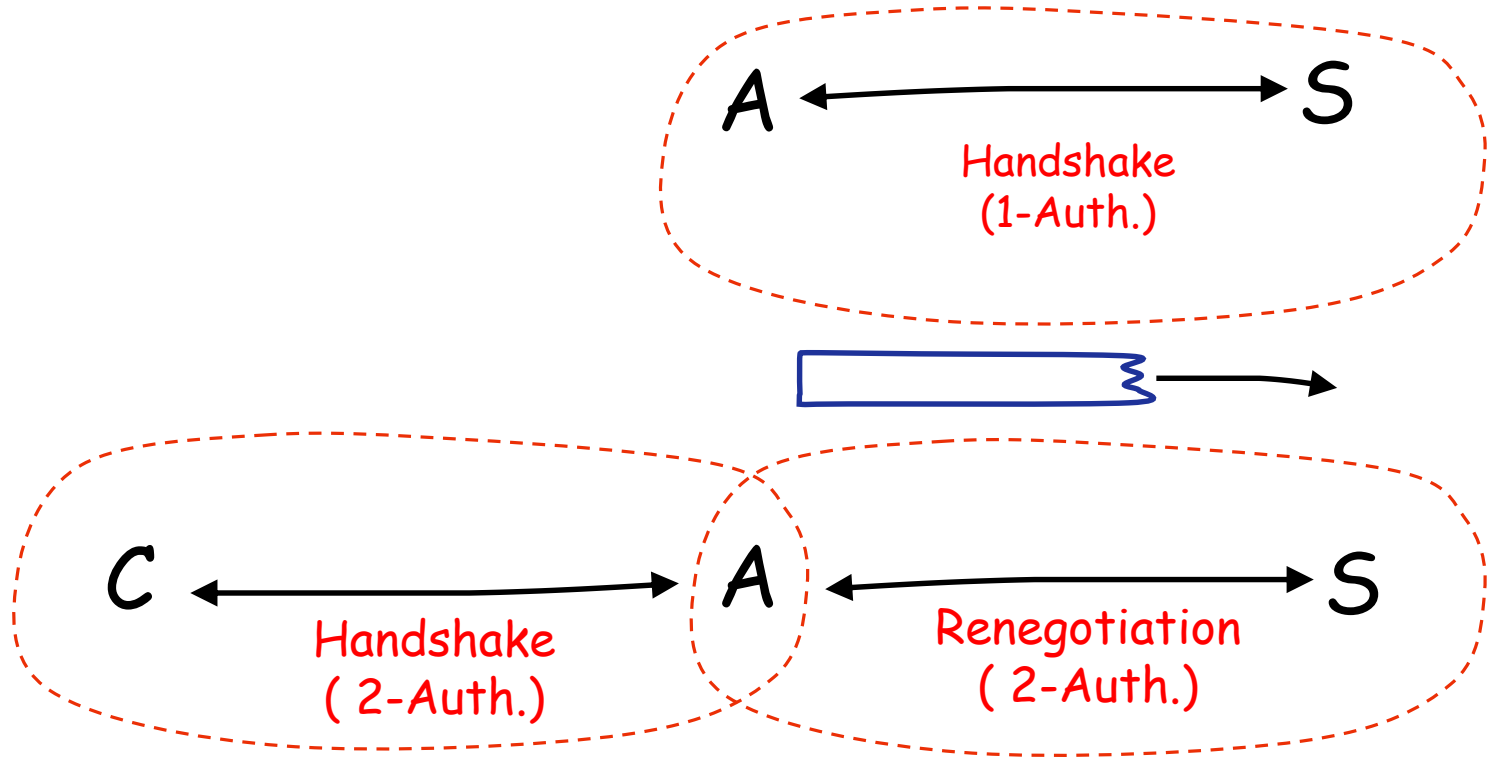
# Handshake/ Атака Марша Рэя



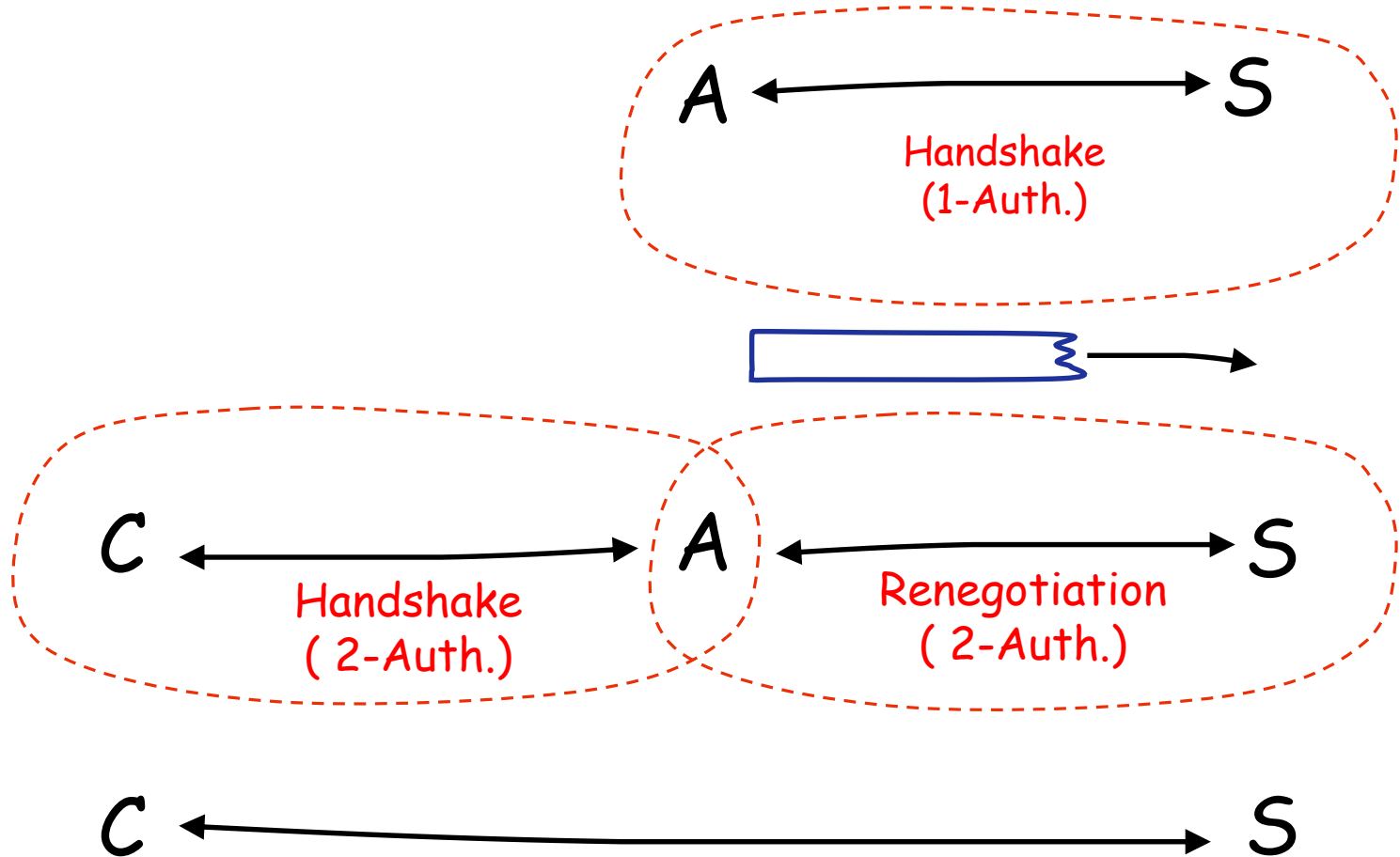
# Handshake/ Атака Марша Рэя



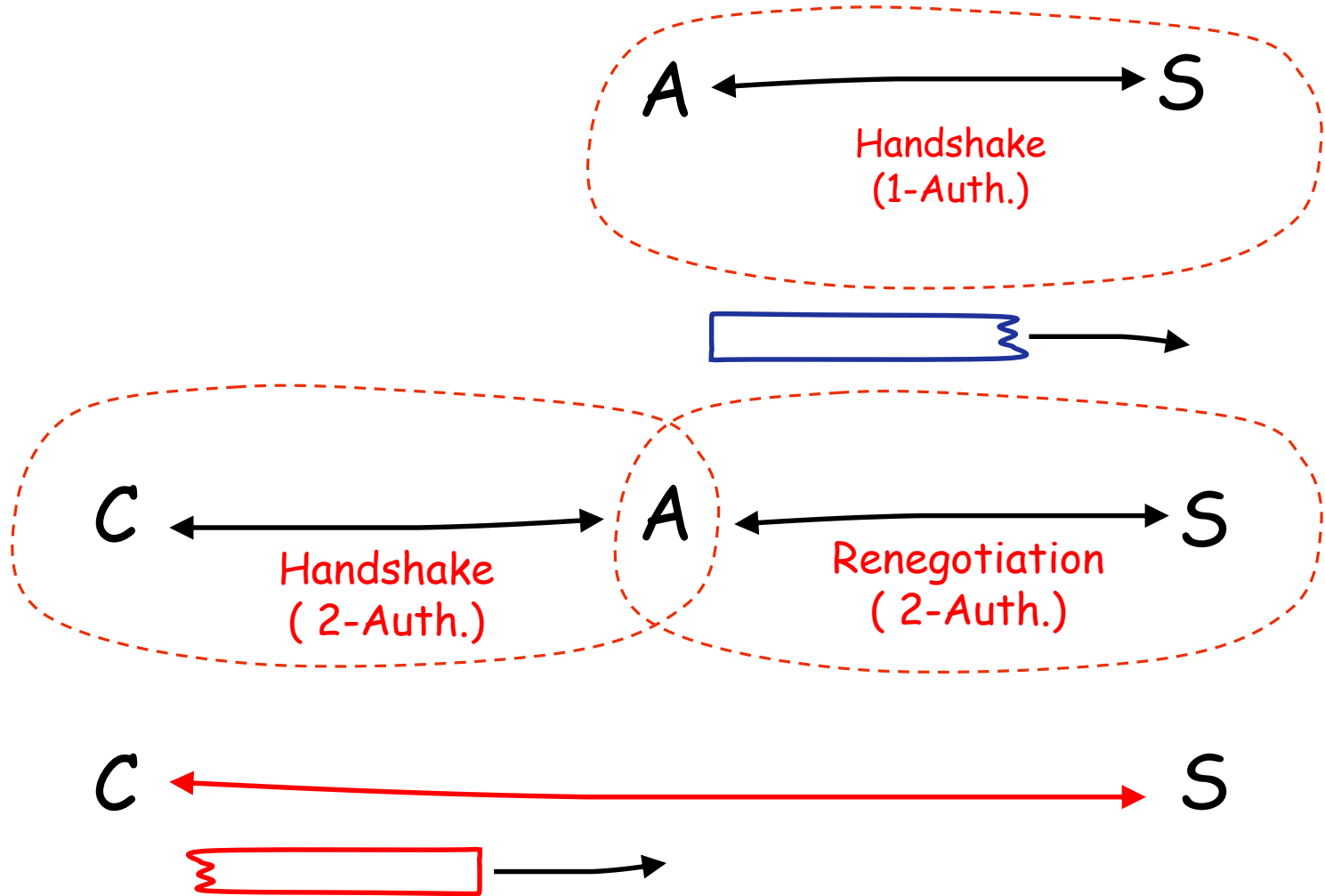
# Handshake/ Атака Марша Рэя



# Handshake/ Атака Марша Рэя



# Handshake/ Атака Марша Рэя



# Handshake/ Атака Марша Рэя



# 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.

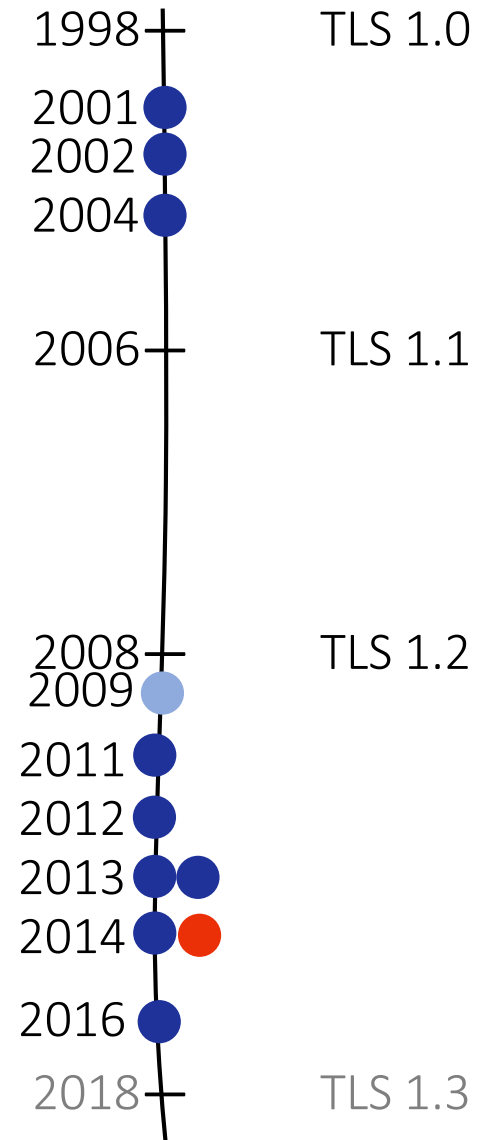
Client (C)	Сообщения	Server (S)
Выработка случайного значения $r_C$	ClientHello $r_C, RI_C$	
	ServerHello $r_S, RI_S$	Выработка случайного значения $r_S$
<b>Этап выработки общего секрета</b>		
	Certificate $Cert_S$ (содержит $Q_S$ )	
	ServerHelloDone	
Выработка случайного значения $PS$ (32 байта)		<div style="border: 2px dashed red; border-radius: 50%; padding: 10px; display: inline-block;"> <i>renegotiation_info (RFC 5746)</i> </div>
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: $IV = H[25..24 + n/2];$ $PS = KImp15(PSExp, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$
<b>Этап выработки ключей из значения <math>PS</math></b>		
$MS = PRF(PS, label_1, HASH(HM))$		$MS = PRF(PS, label_1, HASH(HM))_i$
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S    r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S    r_C)$
<b>Этап подтверждения ключа</b>		
$client\_verify\_data = PRF(MS, label_3, HASH(HM));$	ChangeCipherSpec Finished $client\_verify\_data$	
	ChangeCipherSpec Finished $server\_verify\_data$	$server\_verify\_data = PRF(MS, label_3, HASH(HM));$



# 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).



Client (C)	Сообщения	Server (S)
Выработка случайного значения $r_C$	$\xrightarrow{\text{ClientHello}}$ $r_C, RI_C$	
	$\xleftarrow{\text{ServerHello}}$ $r_S, RI_S$	Выработка случайного значения $r_S$
<b>Этап выработки общего секрета</b>		
	$\xleftarrow{\text{Certificate}}$ $Cert_S$ (содержит $Q_S$ )	
	$\xleftarrow{\text{ServerHelloDone}}$	
Выработка случайного значения $PS$ (32 байта)		
Эфемерный ключ на кривой, соответствующей $Q_S$ : $Q_{EPH} = k_{EPH}P$ , где $k_{EPH}$ выбирается случайно.		
$H = \text{HASH}(r_C \  r_S);$ $K_{MAC}^{Exp} \  K_{ENC}^{Exp} = \text{KEG}(k_{EPH}, Q_S, H)$		
Экспортное представление общего секрета $PS$ : $IV = H[25..24 + n/2];$ $PSExp = \text{KEXP15}(PS, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$	$\xrightarrow{\text{ClientKeyExchange}}$ $Q_{EPH}, PSExp$	$H = \text{HASH}(r_C \  r_S);$ $K_{MAC}^{Exp}   K_{ENC}^{Exp} = \text{KEG}(k_S, Q_{EPH}, H)$
		Извлечение общего секрета из экспортного представления: $IV = H[25..24 + n/2];$ $PS = \text{KIMP15}(PSExp, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$
<b>Этап выработки ключей из значения <math>PS</math></b>		
$MS = \text{PRF}(PS, label_1, r_S \  r_C)$		$MS = \text{PRF}(PS, label_1, r_S \  r_C)$
Генерация ключевого материала соединения: $KeyMaterial = \text{PRF}(MS, label_2, r_S \  r_C)$		Генерация ключевого материала соединения: $KeyMaterial = \text{PRF}(MS, label_2, r_S \  r_C)$
<b>Этап подтверждения ключа</b>		
$client\_verify\_data = \text{PRF}(MS, label_3, \text{HASH}(HM));$	$\xrightarrow{\text{ChangeCipherSpec}}$ $\xrightarrow{\text{Finished}}$ $client\_verify\_data$	
	$\xleftarrow{\text{ChangeCipherSpec}}$ $\xleftarrow{\text{Finished}}$ $server\_verify\_data$	$server\_verify\_data = \text{PRF}(MS, label_3, \text{HASH}(HM));$

Client (C)	Сообщения	Server (S)
Выработка случайного значения $r_C$	$\xrightarrow{\text{ClientHello}}$ $r_C, RI_C$	
	$\xleftarrow{\text{ServerHello}}$ $r_S, RI_S$	Выработка случайного значения $r_S$
<b>Этап выработки общего секрета</b>		
	$\xleftarrow{\text{Certificate}}$ $Cert_S$ (содержит $Q_S$ )	
	$\xleftarrow{\text{ServerHelloDone}}$	
		<b>extended master secret (RFC 7627)</b>
Выработка случайного значения $PS$ (32 байта)		
Эфемерный ключ на кривой, соответствующей $Q_S$ : $Q_{EPH} = k_{EPH}P$ , где $k_{EPH}$ выбирается случайно.		
$H = \text{HASH}(r_C \parallel r_S);$ $K_{MAC}^{Exp} \parallel K_{ENC}^{Exp} = \text{KEG}(k_{EPH}, Q_S, H)$		
Экспортное представление общего секрета $PS$ : $IV = H[25..24 + n/2];$ $PSExp = \text{KEXP15}(PS, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$	$\xrightarrow{\text{ClientKeyExchange}}$ $Q_{EPH}, PSExp$	$H = \text{HASH}(r_C \parallel r_S);$ $K_{MAC}^{Exp} \parallel K_{ENC}^{Exp} = \text{KEG}(k_S, Q_{EPH}, H)$
		Извлечение общего секрета из экспортного представления: $IV = H[25..24 + n/2];$ $PS = \text{KIMP15}(PSExp, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$
<b>Этап выработки ключей из значения <math>PS</math></b>		
$MS = \text{PRF}(PS, label_1, \text{HASH}(HM))$		$MS = \text{PRF}(PS, label_1, \text{HASH}(HM));$
Генерация ключевого материала соединения: $KeyMaterial = \text{PRF}(MS, label_2, r_S \parallel r_C)$		Генерация ключевого материала соединения: $KeyMaterial = \text{PRF}(MS, label_2, r_S \parallel r_C)$
<b>Этап подтверждения ключа</b>		
$client\_verify\_data = \text{PRF}(MS, label_3, \text{HASH}(HM));$	$\xrightarrow{\text{ChangeCipherSpec}}$ $\xrightarrow{\text{Finished}}$ $client\_verify\_data$	
	$\xleftarrow{\text{ChangeCipherSpec}}$ $\xleftarrow{\text{Finished}}$ $server\_verify\_data$	$server\_verify\_data = \text{PRF}(MS, label_3, \text{HASH}(HM));$

# 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.

Client (C)	Сообщения	Server (S)
Выработка случайного значения $r_C$	$\xrightarrow{\text{ClientHello}}$ $r_C, RI_C$	
	$\xleftarrow{\text{ServerHello}}$ $r_S, RI_S$	Выработка случайного значения $r_S$
<b>Этап выработки общего секрета</b>		
	$\xleftarrow{\text{Certificate}}$ $Cert_S$ (содержит $Q_S$ )	
	$\xleftarrow{\text{ServerHelloDone}}$	
Выработка случайного значения $PS$ (32 байта)		
Эфемерный ключ на кривой, соответствующей $Q_S$ : $Q_{EPH} = k_{EPH}P$ , где $k_{EPH}$ выбирается случайно.		
$H = \text{HASH}(r_C \  r_S);$ $K_{MAC}^{Exp} \  K_{ENC}^{Exp} = \text{KEG}(k_{EPH}, Q_S, H)$		
Экспортное представление общего секрета $PS$ : $IV = H[25..24 + n/2];$ $PSExp = \text{KEXP15}(PS, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$	$\xrightarrow{\text{ClientKeyExchange}}$ $Q_{EPH}, PSExp$	$H = \text{HASH}(r_C \  r_S);$ $K_{MAC}^{Exp}   K_{ENC}^{Exp} = \text{KEG}(k_S, Q_{EPH}, H)$
		Извлечение общего секрета из экспортного представления: $IV = H[25..24 + n/2];$ $PS = \text{KIMP15}(PSExp, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$
<b>Этап выработки ключей из значения <math>PS</math></b>		
$MS = \text{PRF}(PS, label_1, \text{HASH}(HM))$		$MS = \text{PRF}(PS, label_1, \text{HASH}(HM))_i$
Генерация ключевого материала соединения: $KeyMaterial = \text{PRF}(MS, label_2, r_S \  r_C)$		Генерация ключевого материала соединения: $KeyMaterial = \text{PRF}(MS, label_2, r_S \  r_C)$
<b>Этап подтверждения ключа</b>		
$client\_verify\_data = \text{PRF}(MS, label_3, \text{HASH}(HM));$	$\xrightarrow{\text{ChangeCipherSpec}}$ $\xrightarrow{\text{Finished}}$ $client\_verify\_data$	
	$\xleftarrow{\text{ChangeCipherSpec}}$ $\xleftarrow{\text{Finished}}$ $server\_verify\_data$	$server\_verify\_data = \text{PRF}(MS, label_3, \text{HASH}(HM));$

Client (C)	Сообщения	Server (S)
криптонабор1 (стойкий)	ClientHello → подмена	криптонабор2 (уязвимый)
	← ServerHello	
<b>Этап выработки общего секрета</b>		
	← Certificate Cert <sub>S</sub> (содержит Q <sub>S</sub> )	
	← ServerHelloDone	
Выработка случайного значения PS (32 байта)		
Эфемерный ключ на кривой, соответствующей Q <sub>S</sub> : Q <sub>EPH</sub> = k <sub>EPH</sub> P, где k <sub>EPH</sub> выбирается случайно.		
$H = \text{HASH}(r_C \  r_S);$ $K_{MAC}^{Exp} \  K_{ENC}^{Exp} = \text{KEG}(k_{EPH}, Q_S, H)$		
Экспортное представление общего секрета PS: IV = H[25..24 + n/2]; PSExp = KExp15(PS, K <sub>MAC</sub> <sup>Exp</sup> , K <sub>ENC</sub> <sup>Exp</sup> , IV)	ClientKeyExchange → Q <sub>EPH</sub> , PSExp	$H = \text{HASH}(r_C \  r_S);$ $K_{MAC}^{Exp}   K_{ENC}^{Exp} = \text{KEG}(k_S, Q_{EPH}, H)$
		Извлечение общего секрета из экспортного представления: IV = H[25..24 + n/2]; PS = KImp15(PSExp, K <sub>MAC</sub> <sup>Exp</sup> , K <sub>ENC</sub> <sup>Exp</sup> , IV)
<b>Этап выработки ключей из значения PS</b>		
$MS = \text{PRF}(PS, \text{label}_1, \text{HASH}(HM))$		$MS = \text{PRF}(PS, \text{label}_1, \text{HASH}(HM))_i$
Генерация ключевого материала соединения: KeyMaterial = PRF(MS, label <sub>2</sub> , r <sub>S</sub>    r <sub>C</sub> )		Генерация ключевого материала соединения: KeyMaterial = PRF(MS, label <sub>2</sub> , r <sub>S</sub>    r <sub>C</sub> )
<b>Этап подтверждения ключа</b>		
client_verify_data = PRF(MS, label <sub>3</sub> , HASH(HM));	ChangeCipherSpec → Finished → client_verify_data	
	← ChangeCipherSpec ← Finished ← server_verify_data	server_verify_data = PRF(MS, label <sub>3</sub> , HASH(HM));

Client (C)	Сообщения	Server (S)
криптонабор1 (стойкий)	ClientHello → подмена	криптонабор2 (уязвимый)
криптонабор1	ServerHello ← подмена	криптонабор2

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

Этап выработки общего секрета		
	Certificate ← Cert <sub>S</sub> (содержит Q <sub>S</sub> )	
	ServerHelloDone	
Выработка случайного значения PS (32 байта)		
Эфемерный ключ на кривой, соответствующей Q <sub>S</sub> : $Q_{EPH} = k_{EPH}P$ , где $k_{EPH}$ выбирается случайно.		
$H = \text{HASH}(r_C \parallel r_S);$ $K_{MAC}^{Exp} \parallel K_{ENC}^{Exp} = \text{KEG}(k_{EPH}, Q_S, H)$		
Экспортное представление общего секрета PS: $IV = H[25..24 + n/2];$ $PSExp = \text{KEXP15}(PS, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$	ClientKeyExchange → $Q_{EPH}, PSExp$	$H = \text{HASH}(r_C \parallel r_S);$ $K_{MAC}^{Exp} \parallel K_{ENC}^{Exp} = \text{KEG}(k_S, Q_{EPH}, H)$
		Извлечение общего секрета из экспортного представления: $IV = H[25..24 + n/2];$ $PS = \text{KIMP15}(PSExp, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$
<b>Этап выработки ключей из значения PS</b>		
$MS = \text{PRF}(PS, label_1, \text{HASH}(HM))$		$MS = \text{PRF}(PS, label_1, \text{HASH}(HM))_i$
Генерация ключевого материала соединения: $KeyMaterial = \text{PRF}(MS, label_2, r_S \parallel r_C)$		Генерация ключевого материала соединения: $KeyMaterial = \text{PRF}(MS, label_2, r_S \parallel r_C)$
<b>Этап подтверждения ключа</b>		
$client\_verify\_data = \text{PRF}(MS, label_3, \text{HASH}(HM));$	ChangeCipherSpec → Finished → $client\_verify\_data$	
	ChangeCipherSpec ← Finished ← $server\_verify\_data$	$server\_verify\_data = \text{PRF}(MS, label_3, \text{HASH}(HM));$

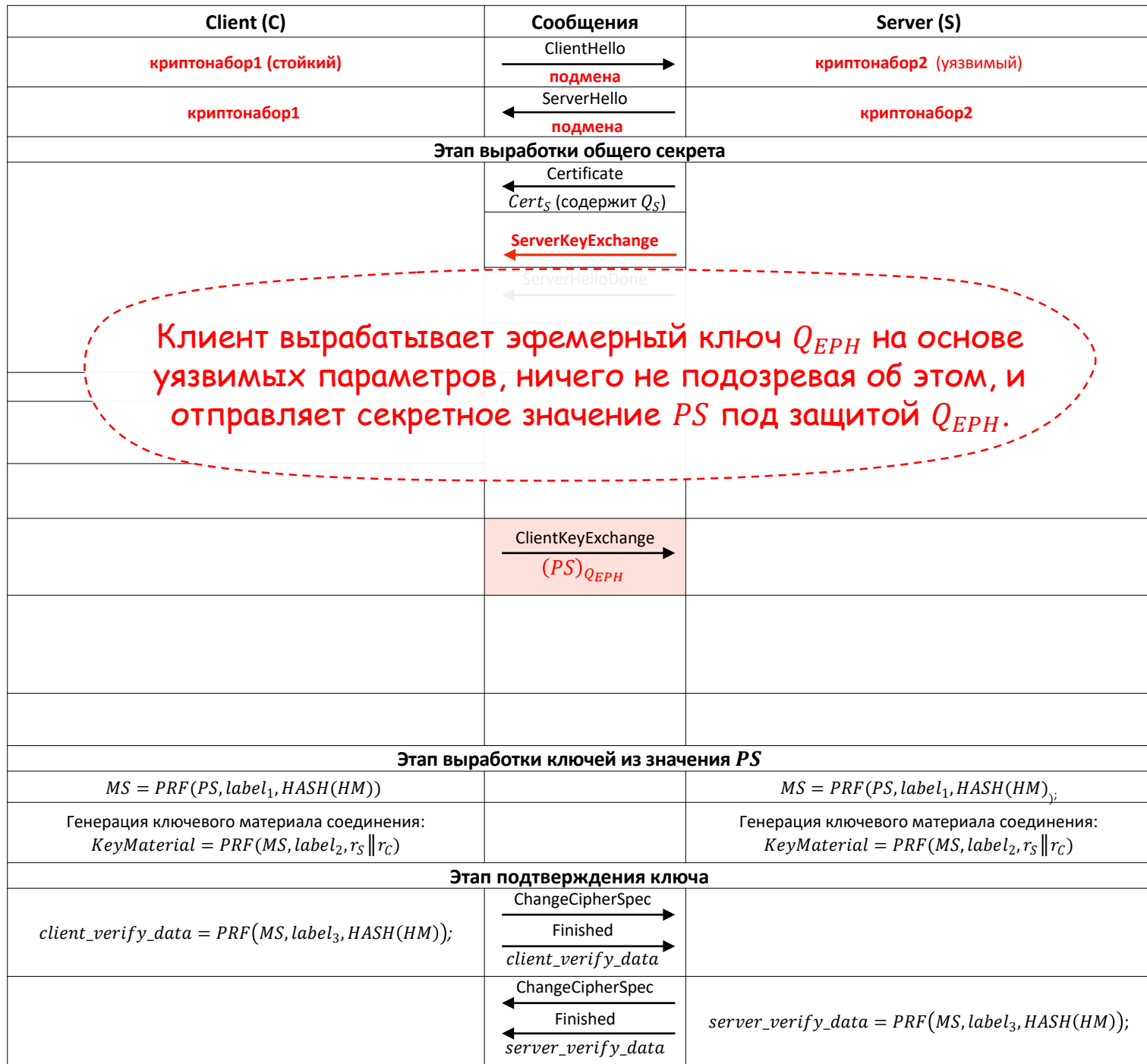


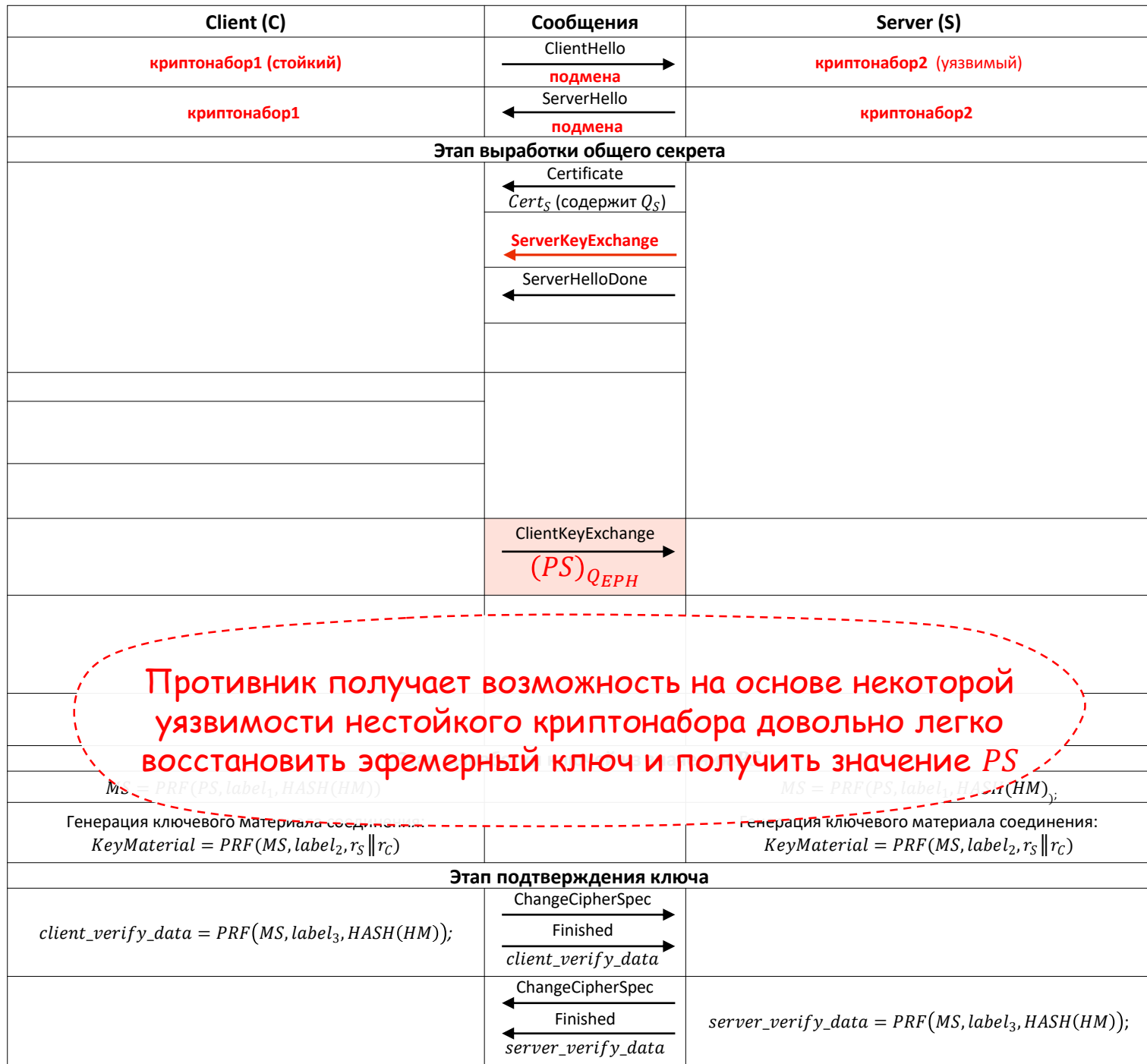
Client (C)	Сообщения	Server (S)
криптонабор1 (стойкий)	ClientHello → подмена	криптонабор2 (уязвимый)
криптонабор1	ServerHello ← подмена	криптонабор2
<b>Этап выработки общего секрета</b>		
	Certificate ← <i>Cert<sub>S</sub></i> (содержит $Q_S$ )	
	ServerHelloDone ←	
Выработка случайного значения $PS$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

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

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
<b>Этап выработки общего секрета</b>		
	Certificate ← $Cert_S$ (содержит $Q_S$ )	
	ServerKeyExchange ←	
	ServerHelloDone ←	
<div style="border: 2px solid red; padding: 10px; margin: 10px auto; width: 80%;"> <p style="color: red; font-weight: bold; text-align: center;">Условие 2: клиент не проверяет данные, пересылаемые в сообщении ServerKeyExchange на соответствие криптонабору1.</p> </div>		
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: $IV = H[25..24 + n/2];$ $PS = KImp15(PSExp, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$
<b>Этап выработки ключей из значения <math>PS</math></b>		
$MS = PRF(PS, label_1, HASH(HM))$		$MS = PRF(PS, label_1, HASH(HM))_i;$
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S    r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S    r_C)$
<b>Этап подтверждения ключа</b>		
$client\_verify\_data = PRF(MS, label_3, HASH(HM));$	ChangeCipherSpec → Finished → $client\_verify\_data$	
	ChangeCipherSpec ← Finished ← $server\_verify\_data$	$server\_verify\_data = PRF(MS, label_3, HASH(HM));$

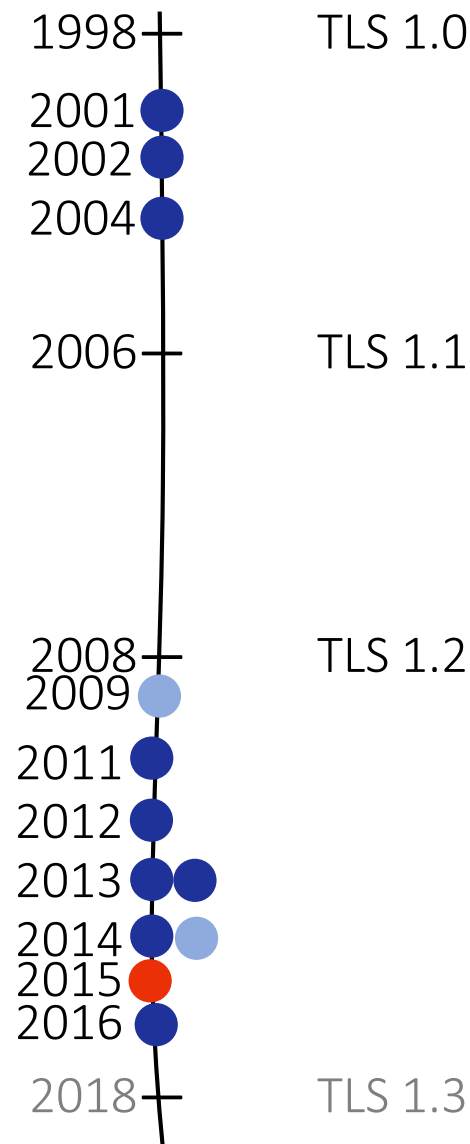




# Handshake/ Кросс-протокольные атаки/ FREAK

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

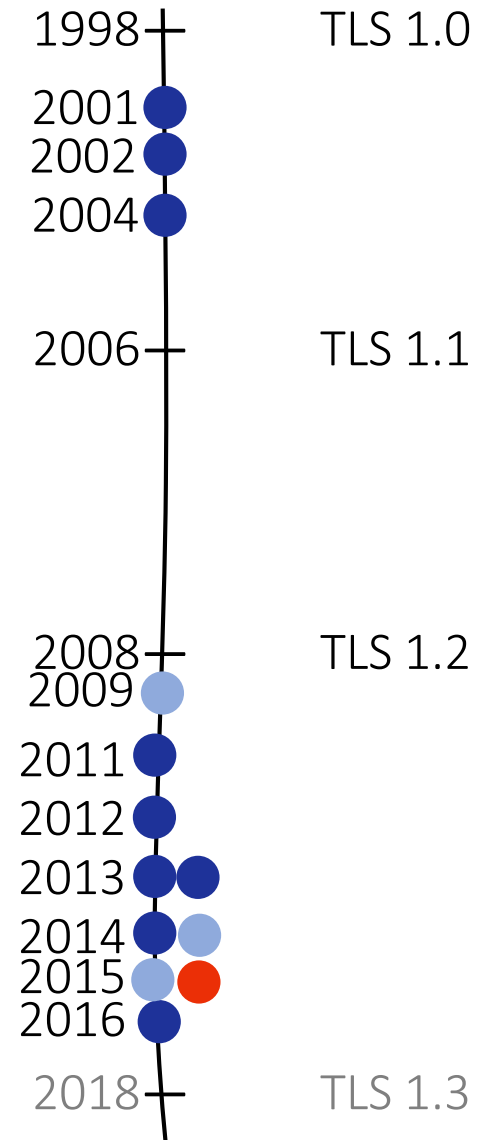
Противник может сделать предвычисления для наиболее распространенных простых 512-битных чисел.



# Handshake/ Кросс-протокольные атаки/ Logjam

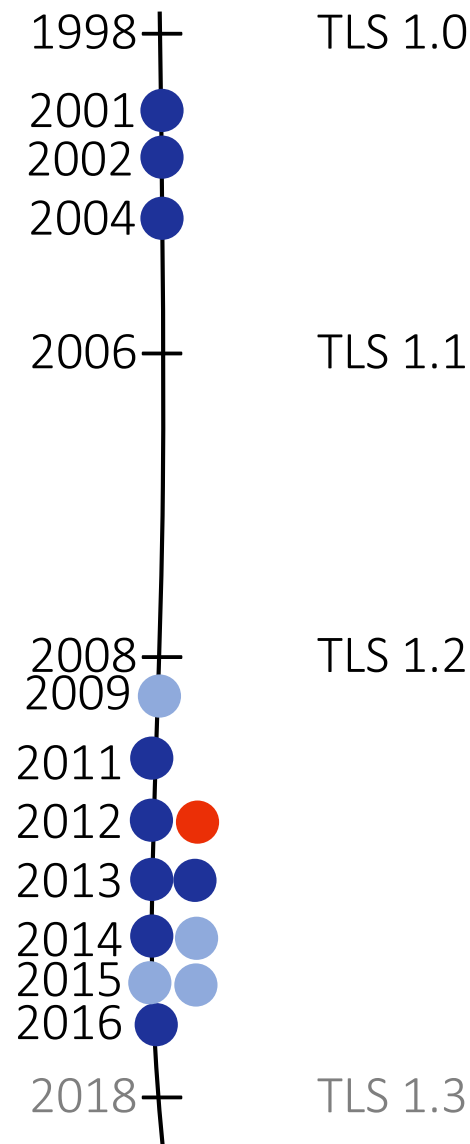
Атака Logjam: навязывание нестойких DHE\_EXPORT криптонаборов (ключи с длиной, не превышающей 512 бит).

Противник может сделать предвычисления для повышения эффективности алгоритма дискретного логарифмирования.

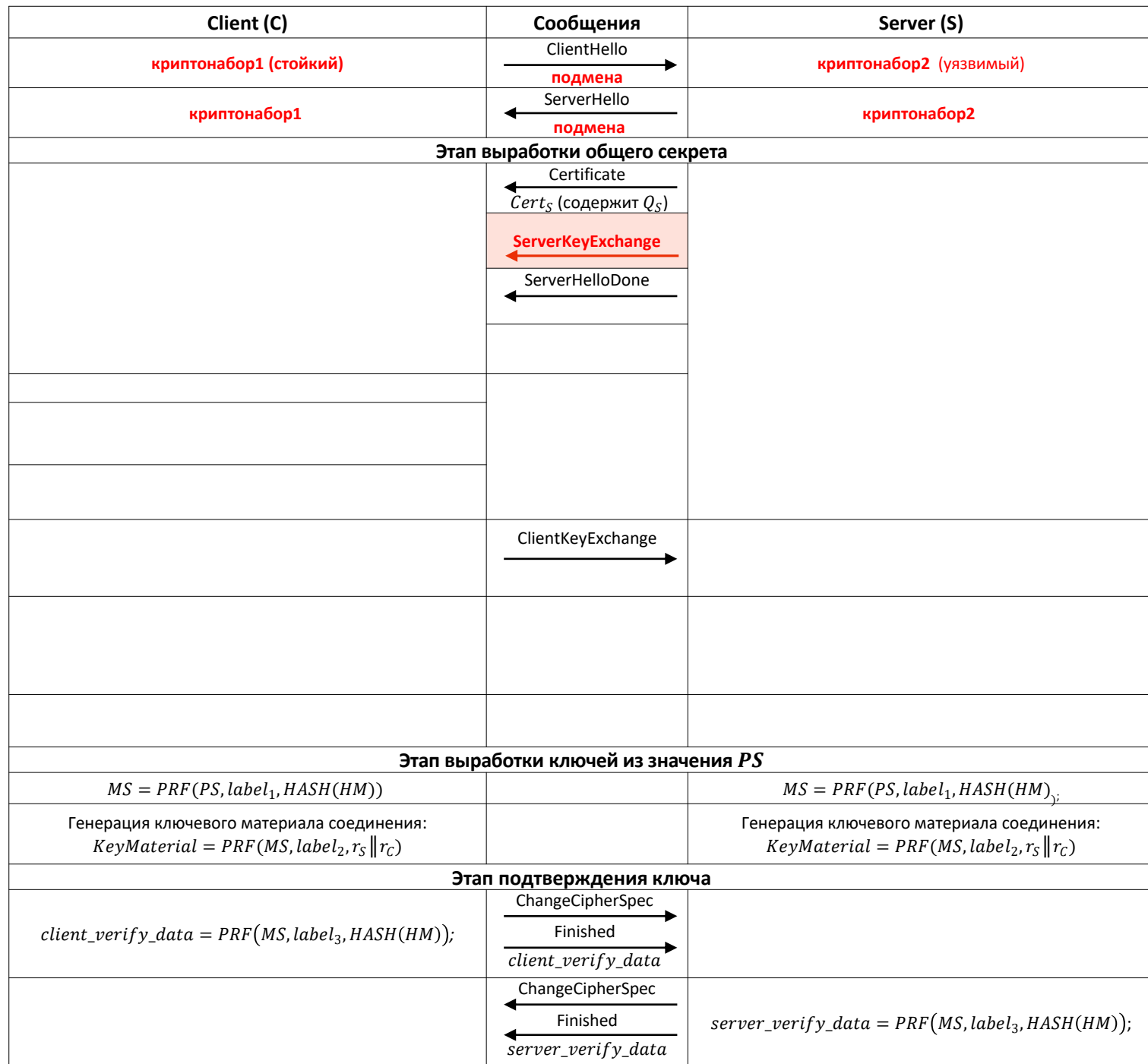


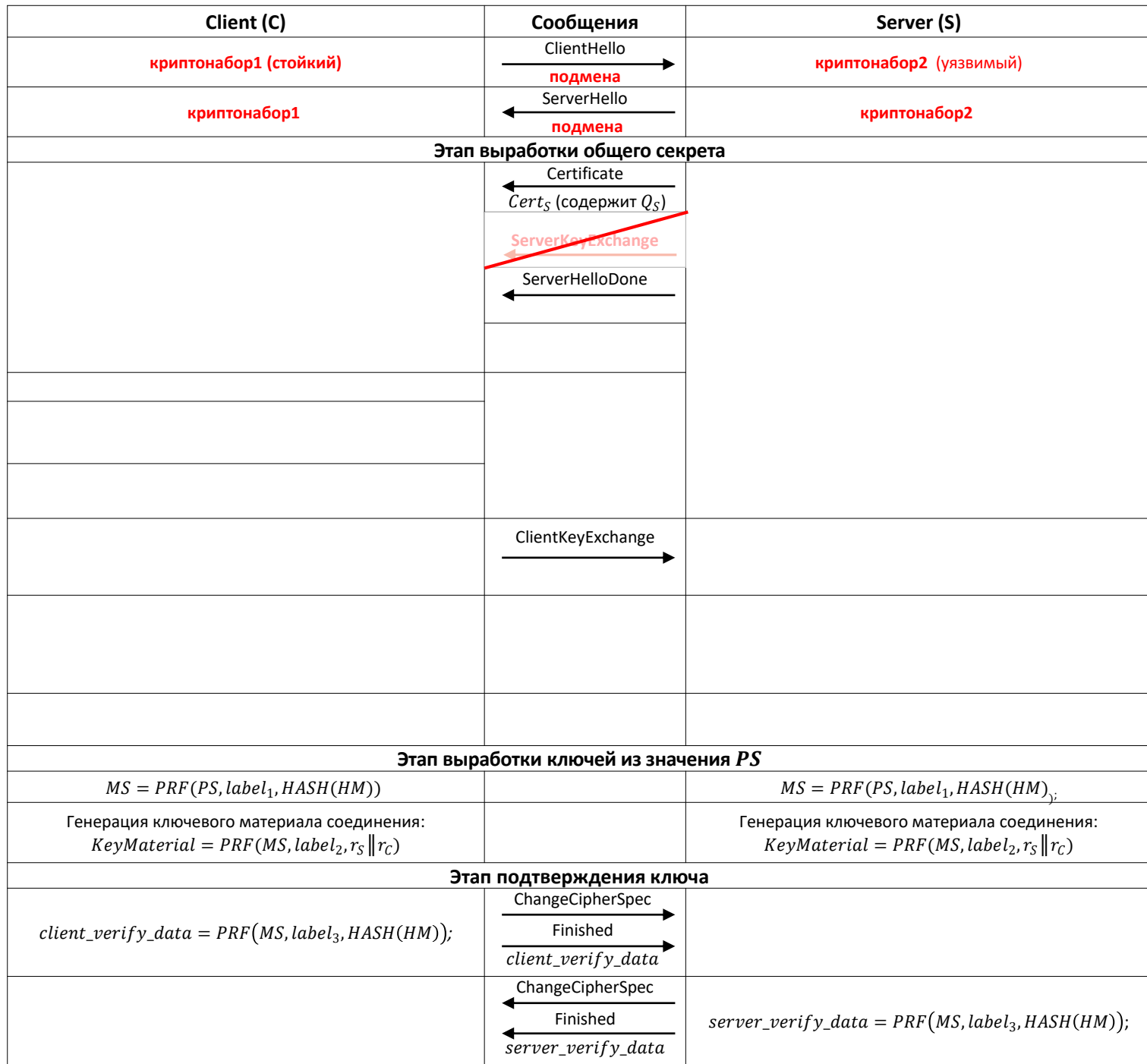
Подмена ECDH криптонаборов на DH криптонаборы.

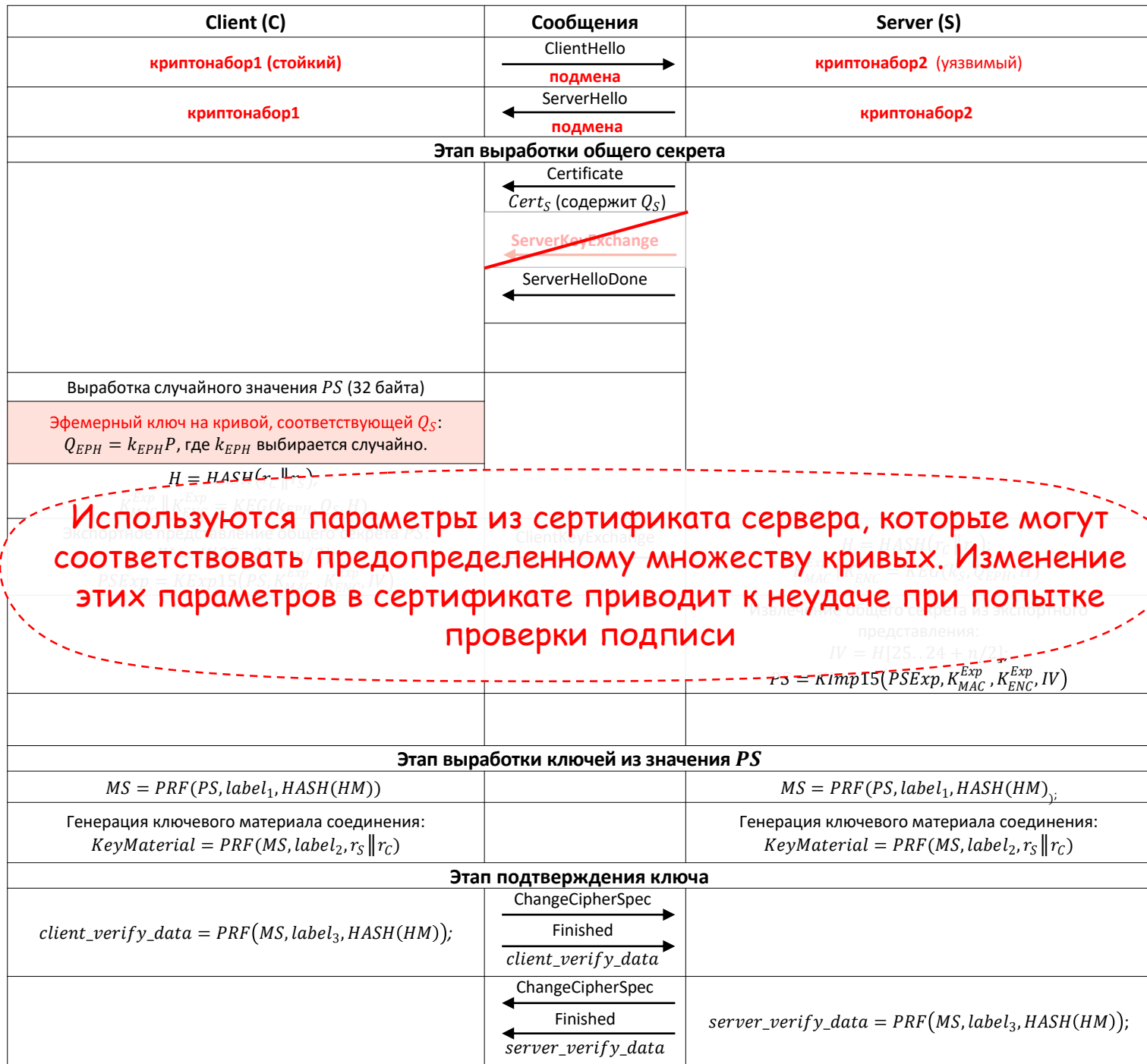
Атака заключалась в использовании возможности интерпретации параметров, указанных в сообщении ServerKeyExchange и соответствующих эллиптической кривой, в качестве уязвимых параметров, соответствующих мультипликативной группе целых чисел.





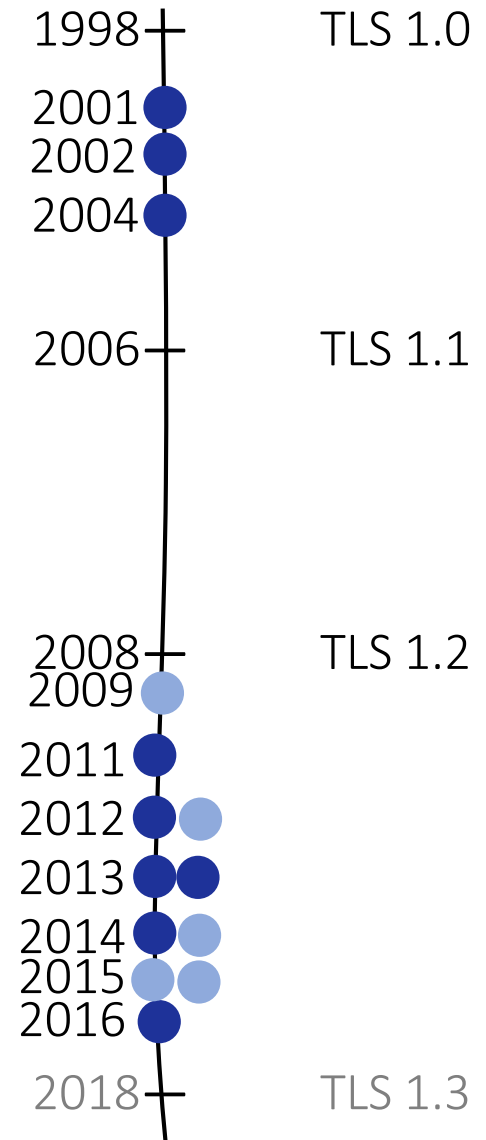






# Handshake/ Модификации эфемерного ключа

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

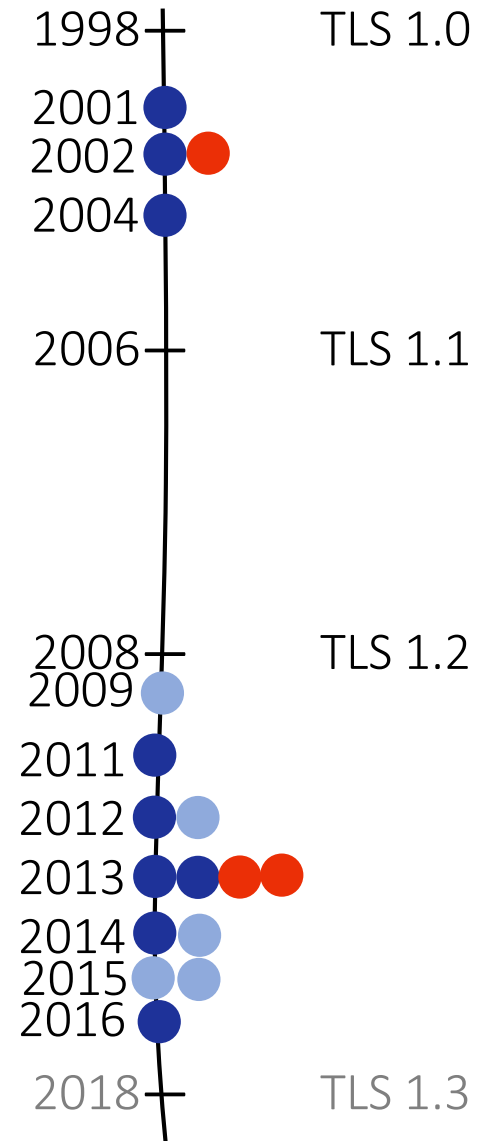


Client (C)	Сообщения	Server (S)
Выработка случайного значения $r_C$	ClientHello $r_C, RI_C$ →	
	ServerHello $r_S, RI_S$ ←	Выработка случайного значения $r_S$
<b>Этап выработки общего секрета</b>		
	Certificate ← $Cert_S$ (содержит $Q_S$ )	
	ServerHelloDone ←	
Выработка случайного значения $PS$ (32 байта)		<div style="border: 2px dashed red; border-radius: 50%; padding: 10px; color: red; font-weight: bold;">           Проверка принадлежности точки целевой подгруппе точек эллиптической кривой         </div>
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: $IV = H[25..24 + n/2];$ $PS = KImp15(PSExp, K_{MAC}^{Exp}, K_{ENC}^{Exp}, IV)$
<b>Этап выработки ключей из значения <math>PS</math></b>		
$MS = PRF(PS, label_1, HASH(HM))$		$MS = PRF(PS, label_1, HASH(HM));$
Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S    r_C)$		Генерация ключевого материала соединения: $KeyMaterial = PRF(MS, label_2, r_S    r_C)$
<b>Этап подтверждения ключа</b>		
$client\_verify\_data = PRF(MS, label_3, HASH(HM));$	ChangeCipherSpec → Finished → $client\_verify\_data$	
	ChangeCipherSpec ← Finished ← $server\_verify\_data$	$server\_verify\_data = PRF(MS, label_3, HASH(HM));$

**Атаки, основывающиеся на свойствах  
конкретных примитивов.**

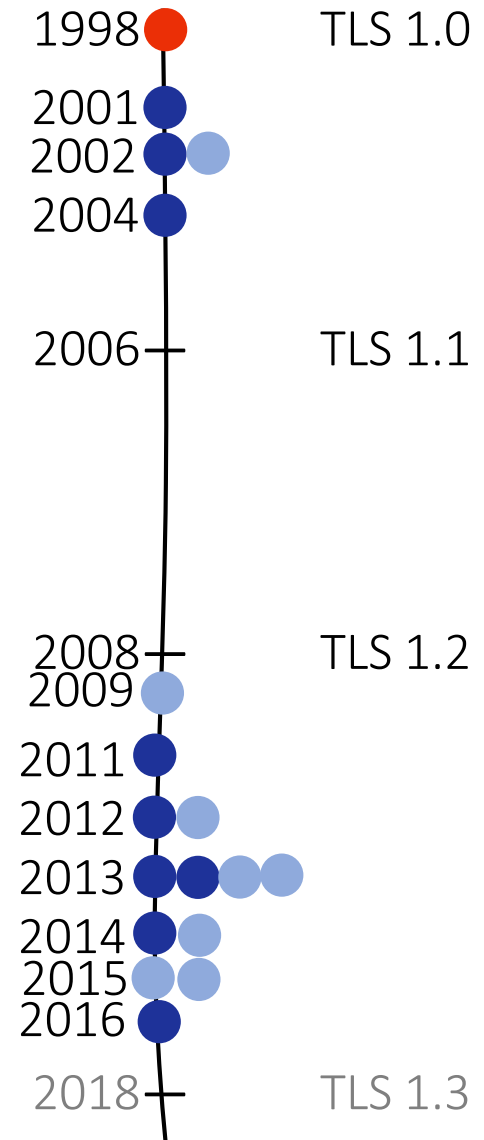
# Handshake/ Атаки на алгоритм сжатия

Атаки на алгоритм сжатия  
(CRIME, TIME, BREACH)



# Handshake/ Атака Блейхенбахера

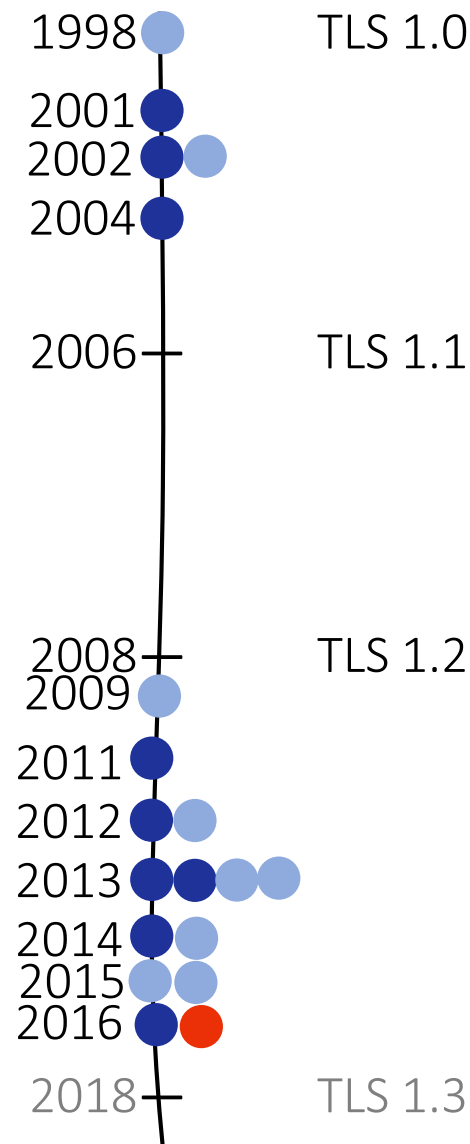
В 1998 году была опубликована атака Блейхенбахера, применимая к криптонаборам, использующим криптосистему RSA совместно с алгоритмом форматирования PKCS#1.





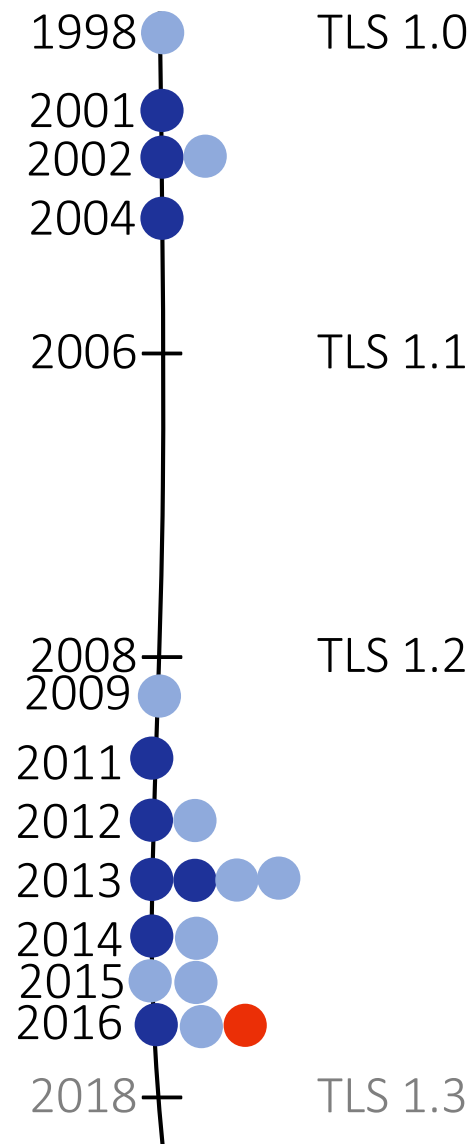
# Handshake/ Атака DROWN

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.



# Handshake/ Атака SLOTH

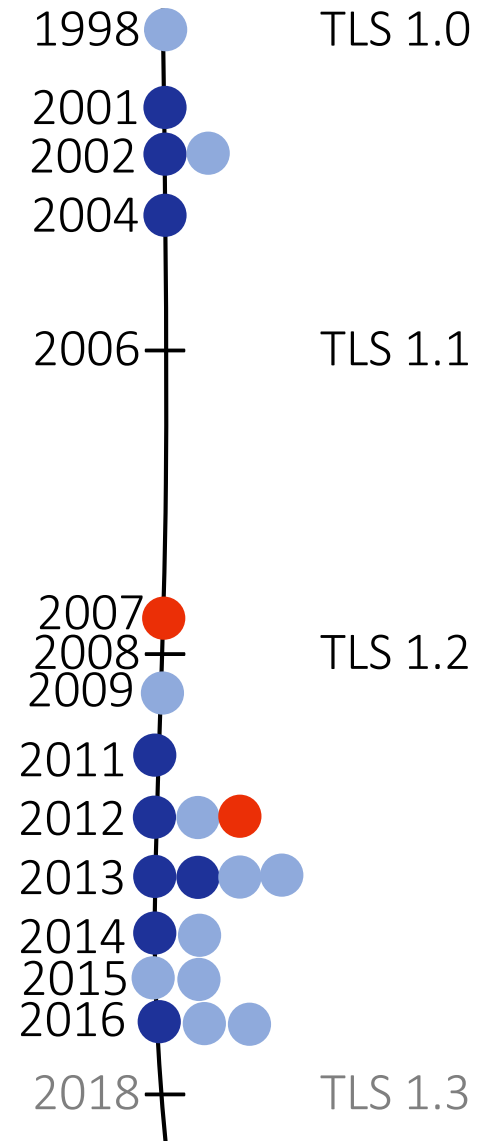
Атака SLOTH основывается на нестойкости распространенных хэш-функций MD5 и SHA-1 к построению коллизий различных видов.



**Alert**

# Прочие атаки

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



**ИТОГ**

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

## 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
```

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.



# 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.

**Спасибо за внимание!**

# 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 АСРКМ	1 Кбайт	4 Кбайт
Максимальная длина записи	$2^{14} + 8$ Байт	$2^{14} + 16$ Байт

# TLS/ Record/ External re-keying (TLSTREE)

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

Как рассчитывать константы  $C_1, C_2, C_3$ :

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

# Handshake/ Атака Марша Рэя

