Как уязвимости в реализациях криптоалгоритмов могут свести защищенность криптосредств к нулю

Алексеев Евгений Константинович,
начальник отдела криптографических исследований
ООО «КРИПТО-ПРО»

КРИПТОПРО

XI симпозиум
«Современные тенденции в криптографии»
CTCrypt 2022

В чем проблема?

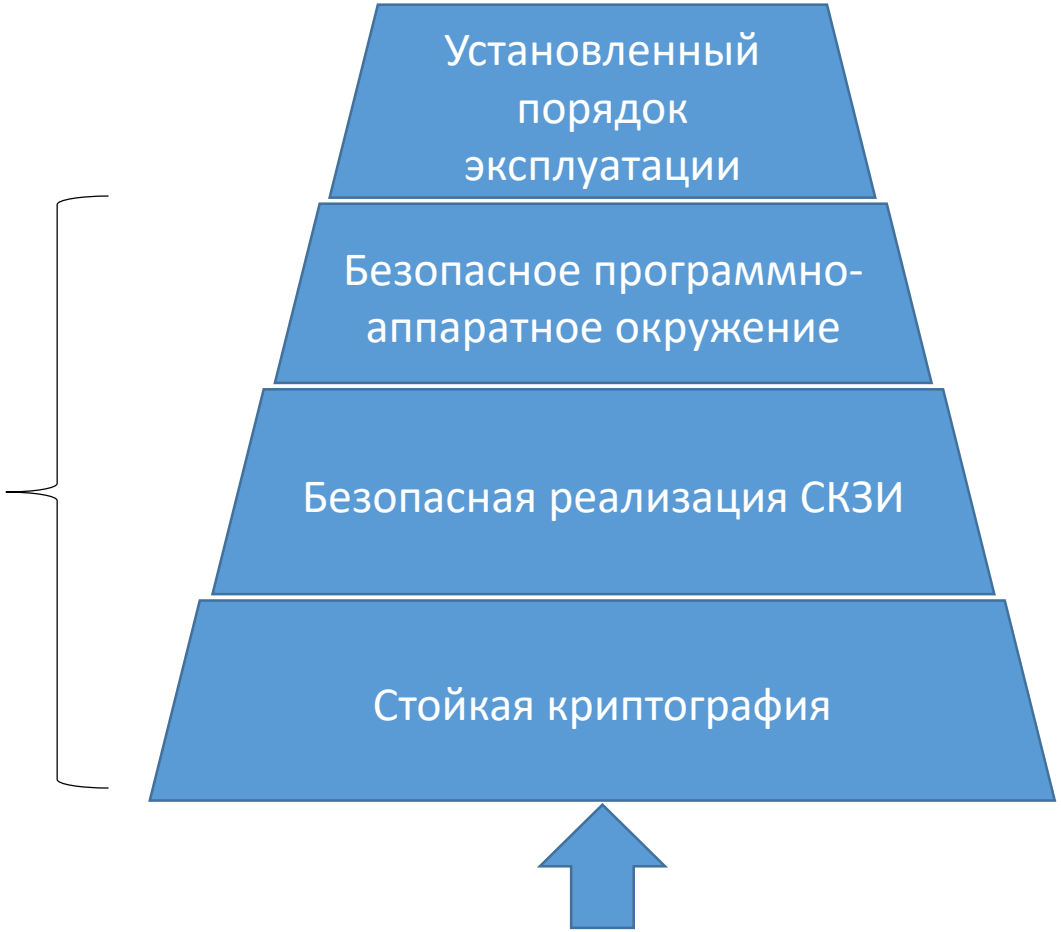
- Требование об использовании сертифицированных СКЗИ
- Частые мнения и вопросы:
  - ✓ Это дает лишь безопасность на бумаге, а не в реальной жизни
  - ✓ За границей все используют Open Source или собственные реализации и все хорошо
- Далее рассмотрим примеры того, как отсутствие достаточного образования и экспертизы в области криптографии приводило к реальным взломам

КРИПТОПРО

Архитектура безопасности информационных систем

Тематические исследования СКЗИ:

- Оценки при конкретных параметрах
- Фиксация доступного диапазона параметров с учетом целевых характеристик и уровней стойкости
- Безопасность реализации самого СКЗИ
- Безопасность определенной части программно-аппаратного окружения
- Требования к порядку использования



Стандартизация:
Обеспечения совместимости решений +
Многоэтапное и всестороннее исследование:

- параметризованные оценки стойкости
- практическая реализуемость условий эксплуатации
- актуальность целевых свойств механизма

КРИПТОПРО

Ошибки в арифметике

- 2011 год: «Practical realization and elimination of an ECC-related software bug attack»
  - ✓ Ошибка в реализации арифметических операций OpenSSL version 0.9.8g
  - ✓ Восстановление долговременного закрытого ключа сервера за 633 запроса
  - ✓ При использовании кривой NIST P-256
  - ✓ Уязвимы криптонаборы со статическим ECDH-ключом
  - ✓ Уязвимы криптонаборы с эфемерным ECDH-ключом при применении ephemeral-static ECDH-оптимизации (один эфемерный ключ на много сессий)

Bug-атака на OpenSSL+P-256

Причины и идея атаки:

- Допущена ошибка в процедуре приведения по модулю p
- При случайном тестировании (в OpenSSL именно так) ошибка появляется с вероятностью не большей, чем $10 \cdot 2^{-29}$
- Серверу, использующему один и тот же ключ d для вычисления общего ключа $d \cdot Q_{eph}$, адаптивно подаются точки Q_{eph} и по тому, верно ли он вычисляет общий ключ, принимается решение о значении новой «части» значения d

Bug-атака на OpenSSL+P-256

Причины и идея атаки:

- Допущена ошибка в процедуре приведения по модулю p
- При случайном тестировании (в OpenSSL именно так) ошибка появляется с вероятностью не большей, чем $10 \cdot 2^{-29}$
- Серверу, использующему один и тот же ключ d для вычисления общего ключа $d \cdot Q_{eph}$, адаптивно подаются точки Q_{eph} и по тому, верно ли он вычисляет общий ключ, принимается решение о значении новой «части» значения d

Вывод авторов работы:

«… the skills gap between cryptography and engineering can be problematic.»

КРИПТОПРО

Отсутствие необходимых проверок

- 2022 год: CVE-2022-21449: «Psychic Signatures in Java»
- Ошибки в реализации проверки подписи ECDSA (валидна любая подпись вида (0,0))
- Уязвимые версии: Java 15, Java 16, Java 17, Java 18

```
|  Welcome to JShell -- Version 17.0.1
|  For an introduction type: /help intro
jshell> import java.security.*
jshell> var keys = KeyPairGenerator.getInstance("EC").generateKeyPair()
keys ==> java.security.KeyPair@626b2d4a
jshell> var blankSignature = new byte[64]
blankSignature ==> byte[64] { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ... ,
0, 0, 0, 0, 0, 0, 0, 0 }
jshell> var sig = Signature.getInstance("SHA256WithECDSAInP1363Format")
sig ==> Signature object: SHA256WithECDSAInP1363Format<not initialized>
jshell> sig.initVerify(keys.getPublic())
jshell> sig.update("Hello, World".getBytes())
jshell> sig.verify(blankSignature)
$8 ==> true
// Oops, that shouldn't have verified...
```

КРИПТОПРО

Отсутствие необходимых проверок

Проверка подписи ECDSA (ANS X9.62 – 1998):

5.4.2 Modular Computations

1. If r' is not an integer in the interval [1, n-1], then reject the signature.
2. If s' is not an integer in the interval [1, n-1], then reject the signature.
3. Compute $c = (s')^{-1} \bmod n$. (See Annex D.1.2.)
4. Compute $u_1 = e'c \bmod n$ and $u_2 = r'c \bmod n$.

5.4.3 Elliptic Curve Computations

1. Compute the elliptic curve point $(x_1, y_1) = u_1G + u_2Q$ (see Annex D.3.2). (If $u_1G + u_2Q$ is the point at infinity, then reject the signature.)

5.4.4 Signature Checking

1. Convert the field element x_1 to an integer $\bar{x}_1$, as described in Section 4.3.5.
2. Compute $v = \bar{x}_1 \bmod n$.
3. If $r' = v$, then the signature is verified, and the verifier has a high level of confidence that the received message was sent by the party holding the secret key d corresponding to Q.

   If r' does not equal v, then the message may have been modified, the message may have been incorrectly signed by the signatory, or the message may have been signed by an impostor. The message shall be considered invalid.

Отсутствие необходимых проверок

Проверка подписи ECDSA (ANS X9.62 – 1998):

5.4.2 Modular Computations

1. If r' is not an integer in the interval [1, n-1], then reject the signature.
2. If s' is not an integer in the interval [1, n-1], then reject the signature.
3. Compute $c = (s')^{-1} \bmod n$. (See Annex D.1.2.)
4. Compute $u_1 = e'c \bmod n$ and $u_2 = r'c \bmod n$.

5.4.3 Elliptic Curve Computations

1. Compute the elliptic curve point $(x_1, y_1) = u_1G + u_2Q$ (see Annex D.3.2). (If $u_1G + u_2Q$ is the point at infinity, then reject the signature.)

5.4.4 Signature Checking

1. Convert the field element x_1 to an integer $\bar{x}_1$, as described in Section 4.3.5.
2. Compute $v = \bar{x}_1 \bmod n$.
3. If $r' = v$, then the signature is verified, and the verifier has a high level of confidence that the received message was sent by the party holding the secret key d corresponding to Q.

   If r' does not equal v, then the message may have been modified, the message may have been incorrectly signed by the signatory, or the message may have been signed by an impostor. The message shall be considered invalid.

Проверки не производятся:

подпись (r=0,s=0) будет проверяться дальше

КРИПТОПРО

Отсутствие необходимых проверок

Проверка подписи ECDSA (ANS X9.62 – 1998):

5.4.2 Modular Computations

1. If r' is not an integer in the interval [1, n-1], then reject the signature.
2. If s' is not an integer in the interval [1, n-1], then reject the signature.
3. Compute $c = (s')^{-1} \bmod n$. (See Annex D.1.2.)
4. Compute $u_1 = e'c \bmod n$ and $u_2 = r'c \bmod n$.

5.4.3 Elliptic Curve Computations

1. Compute the elliptic curve point $(x_1, y_1) = u_1G + u_2Q$ (see Annex D.3.2). (If $u_1G + u_2Q$ is the point at infinity, then reject the signature.)

5.4.4 Signature Checking

1. Convert the field element x_1 to an integer $\bar{x}_1$, as described in Section 4.3.5.
2. Compute $v = \bar{x}_1 \bmod n$.
3. If $r' = v$, then the signature is verified, and the verifier has a high level of confidence that the received message was sent by the party holding the secret key d corresponding to Q.

   If r' does not equal v, then the message may have been modified, the message may have been incorrectly signed by the signatory, or the message may have been signed by an impostor. The message shall be considered invalid.

Должно было бы «сломаться» здесь!

Но нет – обратный вычисляется по формуле $s^{n-2} \bmod n$

КРИПТОПРО

Отсутствие необходимых проверок

Проверка подписи ECDSA (ANS X9.62 – 1998):

5.4.2 Modular Computations

1. If r' is not an integer in the interval [1, n-1], then reject the signature.
2. If s' is not an integer in the interval [1, n-1], then reject the signature.
3. Compute $c = (s')^{-1} \bmod n$. (See Annex D.1.2.)
4. Compute $u_1 = e'c \bmod n$ and $u_2 = r'c \bmod n$.

5.4.3 Elliptic Curve Computations

1. Compute the elliptic curve point $(x_1, y_1) = u_1G + u_2Q$ (see Annex D.3.2). (If $u_1G + u_2Q$ is the point at infinity, then reject the signature.)

5.4.4 Signature Checking

1. Convert the field element x_1 to an integer $\bar{x}_1$, as described in Section 4.3.5.
2. Compute $v = \bar{x}_1 \bmod n$.
3. If $r' = v$, then the signature is verified, and the verifier has a high level of confidence that the received message was sent by the party holding the secret key d corresponding to Q.

   If r' does not equal v, then the message may have been modified, the message may have been incorrectly signed by the signatory, or the message may have been signed by an impostor. The message shall be considered invalid.

Все закончилось бы здесь… но эта проверка тоже не выполняется☹

КРИПТОПРО

Некорректная интерпретация спецификации

- 2019 год: атака на систему передачи данных CROC (CVE-2021-31603)

Система передачи данных CROC

Установка защищенного канала с помощью криптографического протокола выработки общего ключа на основе пароля (PAKE)

⇓

Передача файлов по установленному защищенному каналу

КРИПТОПРО

Некорректная интерпретация спецификации

Система CROC использует
криптографический протокол выработки общего ключа SPAKE2

Draft RFC, описывающий спецификацию протокола SPAKE2:

- В рамках процедуры Setup вырабатываются точки эллиптической кривой *M*, *N*
- Описаны алгоритмы генерации точек *M*, *N*
- Раздел “Security Considerations” содержит замечание:

  Выбор *M* и *N* имеет крайне важное значение для безопасности протокола. Описанные методы их генерации позволяют помочь избежать проблем, связанных со знанием дискретных логарифмов *M* и *N*

Если для точки M известен дискретный логарифм, то возможна атака, позволяющая восстановить ключ шифрования

КРИПТОПРО

Некорректная интерпретация спецификации

- 2019 год: атака на систему передачи данных CROC (CVE-2021-31603)

В системе CROC (в части реализации протокола SPAKE2)
точку *M* генерирует одна из сторон протокола

⇓

Нарушитель, который является легитимным участником системы,
может выбрать такую точку *M*, для которой он знает дискретный логарифм

⇓

Нарушитель может **восстанавливать ключ шифрования**

Авторы атаки (https://redrocket.club/posts/croc/) приводят демонстрацию, где наиболее существенная часть атаки требует 17 секунд

КРИПТОПРО

Отсутствие необходимых ограничений

- 2016 год: SWEET32 (CVE-2016-2183)
- «On the Practical (In-)Security of 64-bit Block Ciphers»
- Атаки на HTTP+OpenVPN и на HTTPS
- Проблема в использовании блочных шифров с размером блока 64 бита в режиме CBC

SWEET32

- Предыстория:
  - ✓ 2000 год. Фундаментальная оценка стойкости режима CBC:

Theorem 17 [**Security of** CBC **using a pseudorandom permutation**] *Suppose F is a PRP family with length l. Then, for any t, q_e and $\mu_e = ql$,*

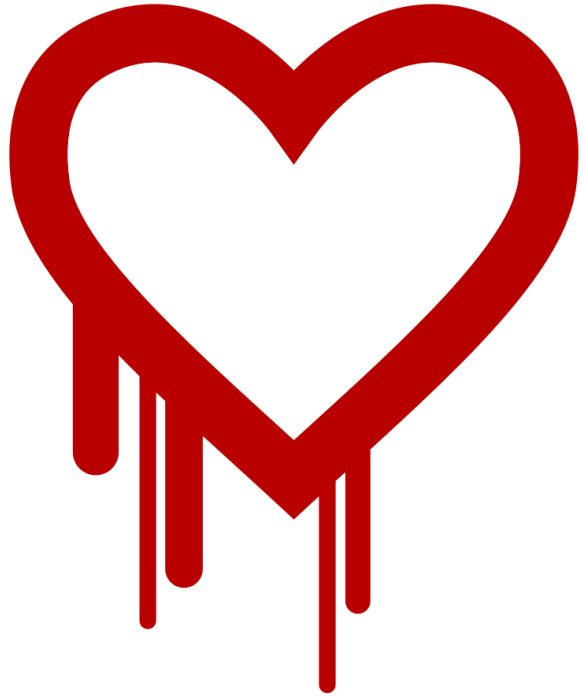
$$\mathbf{Adv}^{\text{lor-cpa}}_{\text{CBC}[F]}(\cdot, t, q_e, \mu_e) \leq 2 \cdot \mathbf{Adv}^{\text{prp}}_{F}(t, q) + q^2 2^{-l-1} + \left(\frac{\mu_e^2}{l^2} - \frac{\mu_e}{l}\right) \cdot 2^{-l} \; . \blacksquare$$

  - ✓ Сами по себе прикладные протоколы не ограничивают нагрузку на ключ
  - ✓ Учет нагрузки отдан на откуп конкретным реализациям

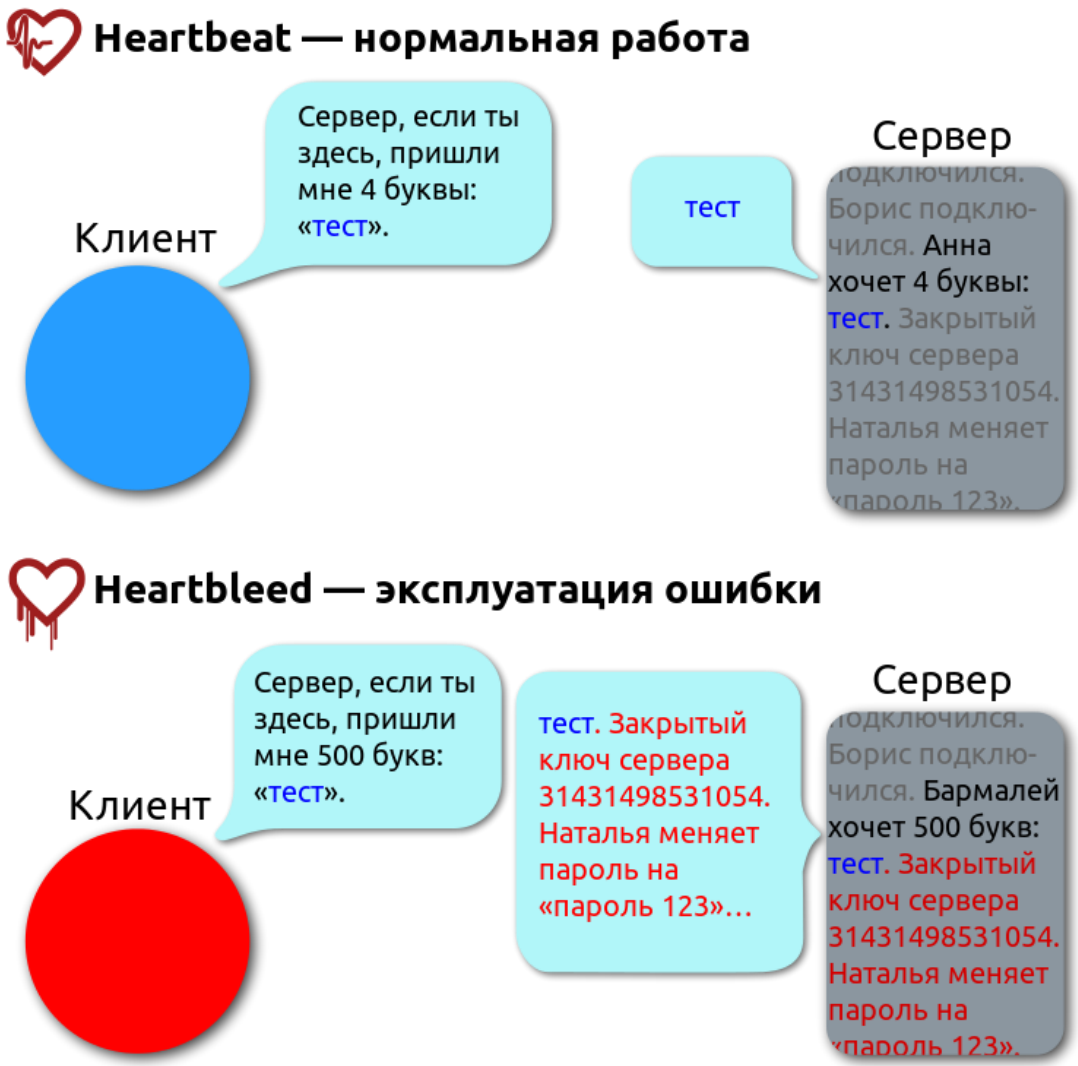
- Детали атаки:
  - ✓ Восстановление секретных 16-байтовых значений BasicAuth и cookie
  - ✓ Нужно набрать 800 ГБ трафика (19 часов для OpenVPN и 38 часов для HTTPS)

Некорректная обработка формата

- 2014 год: Heartbleed (CVE-2014-0160)
- Уязвимость в модуле Heartbeat библиотеки OpenSSL
- Некорректная обработка специального запроса для оптимизации поддержания активности сеанса
- Восстановление долговременных ключей, секретных cookie или конфиденциальной информации других соединений



Heartbleed



КРИПТОПРО

Вопросы

Спасибо за внимание!
Вопросы?