Dmytro Sotnyk

Hadoop Security Project

Symmetric Encryption Algorithms overview

2016

Introduction

“Cryptanalysis always gets better. It never gets worse”
-- NSA

Goal of this investigation is to choose safe symmetric encryption algorithms, analyze known vulnerabilities and estimate risks.

We will overview 3 most popular and used algorithms: AES, 3DES and RC4.

AES

AES using block-size 128 bits, but three different key lengths: 128, 192 and 256 bits.

The key size used for an AES cipher specifies the number of repetitions of transformation rounds that convert the input, called the plaintext, into the final output, called the ciphertext. The number of cycles of repetition are as follows:
10 cycles of repetition for 128-bit keys.
12 cycles of repetition for 192-bit keys.
14 cycles of repetition for 256-bit keys.

Known attacks

By 2006, the best known attacks were on 7 rounds for 128-bit keys, 8 rounds for 192-bit keys, and 9 rounds for 256-bit keys

In June **2003**, the U.S. Government announced that AES could be used to protect classified information: “The design and strength of all key lengths of the AES algorithm (i.e., 128, 192 and 256) are sufficient to protect classified information up to the **SECRET** level. **TOP SECRET** information will require use of either the **192** or **256** key lengths. The implementation of AES in products intended to protect national security systems and/or information must be reviewed and certified by NSA prior to their acquisition and use”

First news in 2002 about broken AES with XSL attack was declared as unworkable because of limited applicability to block ciphers. See corresponding explanation in Schneier’s blog.

July 1, 2009, Bruce Schneier blogged about a related-key attack on the 192-bit and 256-bit versions of AES, discovered by Alex Biryukov and Dmitry Khovratovich, which exploits AES's

somewhat simple key schedule and has a complexity of 2^119. In December 2009 it was improved to 2^99.5. This is a follow-up to an attack discovered earlier in 2009 by Alex Biryukov, Dmitry Khovratovich, and Ivica Nikolić, with a complexity of 2^96 for one out of every 2^35 keys. However, related-key attacks are not of concern in any properly designed cryptographic protocol, as properly designed software will not use related-keys.

In the same time, Schneier declared that "*And for new applications I suggest that people don't use AES-256. AES-128 provides more than enough security margin for the forseeable future. But if you're already using AES-256, there's no reason to change.*"
Also Schneier reported, that "*In an e-mail, the authors wrote: We also expect that a careful analysis may reduce the complexities. As a preliminary result, we think that the complexity of the attack on AES-256 can be lowered from 2^119 to about 2^110.5 data and time*."

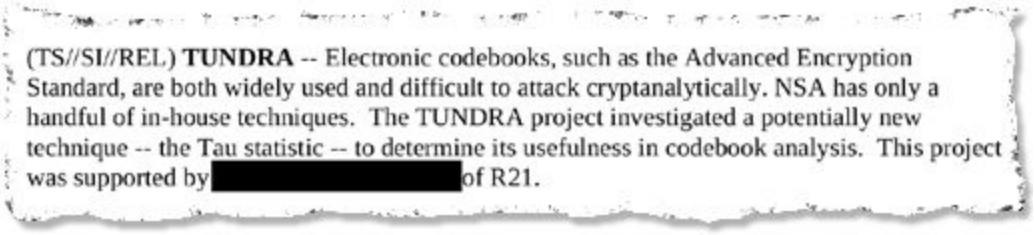
In November 2009, the first known-key distinguishing attack against a reduced 8-round version of AES-128 was released as a preprint. This known-key distinguishing attack is an improvement of the rebound or the start-from-the-middle attacks for AES-like permutations, which view two consecutive rounds of permutation as the application of a so-called Super-Sbox. It works on the 8-round version of AES-128, with a time complexity of 2^48, and a memory complexity of 2^32. 128-bit AES uses 10 rounds, so this attack isn't effective against full AES-128.

In July 2010 Vincent Rijmen published a paper on "chosen-key-relations-in-the-middle" attacks on AES-128 which states 2^32 solution for AES-128 for related case and corner cases which states that "*the entropy of the key is reduced from 128 to 32 bits without making a single query to the encryption oracle*"

The first Biclique key-recovery attacks on full AES were due to Andrey Bogdanov, Dmitry Khovratovich, and Christian Rechberger were published in 2011. The attack is a biclique attack and is faster than brute force by a factor of about four. It requires 2^126.1 operations to recover an AES-128 key. For AES-192 and AES-256, 2^189.7 and 2^254.4 operations are needed, respectively. This is a very small gain, as a 126-bit key (instead of 128-bits) would still take billions of years. Also, the authors calculate the best attack using their technique on AES with a 128 bit key requires storing 2^88 bits of data. That works out to about 38 trillion terabytes of data, which is more than all the data stored on all the computers on the planet. As such this is a theoretical attack that has no practical implication on AES security. See also analyze in Schneier's blog.

Classified sources

According to the Snowden's documents, the NSA is doing research on whether a cryptographic attack based on tau statistic may help to break AES



(TS//SI//REL) **TUNDRA** -- Electronic codebooks, such as the Advanced Encryption Standard, are both widely used and difficult to attack cryptanalytically. NSA has only a handful of in-house techniques. The TUNDRA project investigated a potentially new technique -- the Tau statistic -- to determine its usefulness in codebook analysis. This project was supported by [redacted] of R21.

Special note on related-keys attacks

Related-key attacks are not a problem when the encryption algorithm is used for encryption, because they work only when the victim uses several distinct keys, such that the differences (bitwise XOR) between the keys are known to the attacker and follow a very definite pattern.

This is not the kind of thing which often occurs in protocols where AES is used; correspondingly, resistance to related-key attacks was not a design criterion for the AES competition.

Related-key attacks can be troublesome when we try to reuse the block cipher as a building block for something else, e.g. a hash function.

More information in this discussion.

Summary of the known attacks

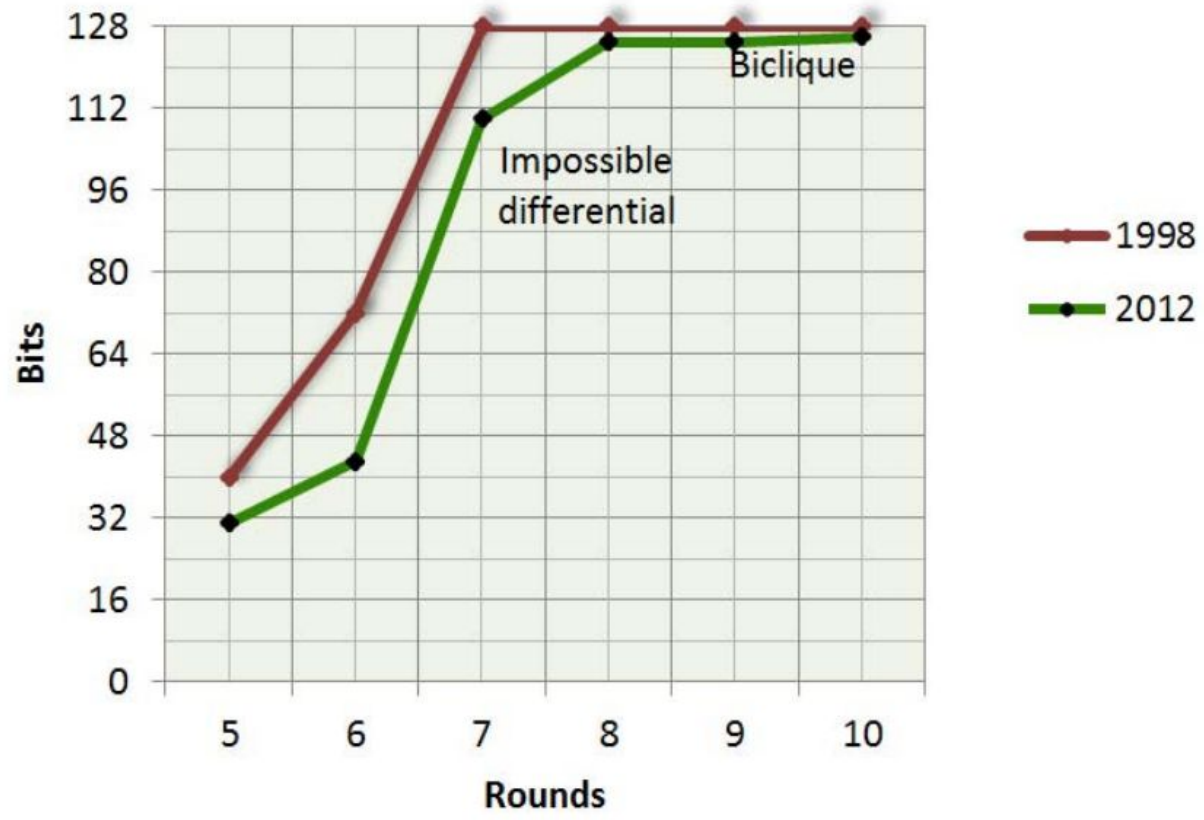
Here is the brief summary of known attacks to AES

How to read: “70 on 11r” means complexity of attack in 2^{70} for 11 rounds; “176” means 2^{176} for full set of rounds (10 rounds for 128-bit keys, 12 for 192-bit, 14 for 256-bit).

	AES-128	AES-192	AES-256
Related-key attack, 2009		176	119 70 on 11r 45 on 10r 39 on 9r
Known-key distinguishing attack, 2009	48 on 8r		

"Chosen-key-relations-in-the-middle" for related keys, 2010	32		
Biclique attack by Andrey Bogdanov, Dmitry Khovratovich, and Christian Rechberger on full AES, 2011	126.1 125.4 on 8r	189.7	254.4
Boomerang on Related Keys	97 on 7r	169	99.5
Rectangle on Related Keys		182 on 10r 143 on 9r	173 on 10r
Differential on Related Keys			131
Subkey difference			49 on 10r
Multiset	70 on 6r		
Partial sum	120 on 7r 44 on 6r		
Boomerang	71 on 6r		
Impossible diff	117.2 on 7r		

Here is the overview of strength of AES-128, depending on rounds and related keys usage



So, as we can see, the major weakness of AES is a result of:

- Related keys
- Less rounds than recommended by NIST in Federal Information Processing Standards Publication : Announcing the ADVANCED ENCRYPTION STANDARD (AES)

So, the only known attack which can be used against properly implemented AES (full amount of rounds) with no related keys is Biclique attack, which reduce complexity to:

- AES-128 to 2^126.1
- AES-196 to 2^189.7
- AES-256 to 2^254.4

As such this is a theoretical attack that has no practical implication on AES security.

Estimate of brute-force speed applied to AES

...brute force attacks against 256-bit keys will be infeasible until computers are built from something other than matter and occupy something other than space".
-- Bruce Schneier in Applied Cryptography

Modern super-computer, China's Tianhe-2 was ranked the world's fastest with a record of 33.86 petaFLOPS, what means 2^55 FLOPS.

Even if every FLOPS will result in one key attempt (but for every round we need hundreds FLOPS and we have 10-14 rounds), with known attacks, properly used AES-128 will be decrypted in 2^(126.1-55-1) = 2^70.1 or 1.26*10^21 seconds. **This is 4*10^13 years, which is ~ 28 500 times more than age of our universe.**

Today's practical limit is near 2^90 combinations (key bits), which corresponds to 1000 Tianhe-2 array which will work 1 year. By the way, it's for optimistic case, when 1 FLOPS - one key.

AES Modes

*This part of the overview is **copied** from Block cipher mode of operation and How to choose an AES encryption mode (CBC ECB CTR OCB CFB)?*

Also see modes, proposed by NIST.

Encryption only

Requires padding

Padding can generally be dangerous because it opens up the possibility of padding oracle attacks. The easiest defense is to authenticate every message before decryption (see MAC and HMAC section below).

ECB, electronic codebook, encrypts each block of data independently and the same plaintext block will result in the same ciphertext block. Take a look at the ECB encrypted Tux image on the ECB Wikipedia page to see why this is a serious problem. I don't know of any use case where ECB would be acceptable.

CBC (and **PCBC**) is a block cipher mode of operation, it has an IV and thus needs randomness every time a message is encrypted, changing a part of the message requires re-encrypting

everything after the change, transmission errors in one ciphertext block completely destroy the plaintext and change the decryption of the next block, decryption can be parallelized / encryption can't, the plaintext is malleable to a certain degree - this can be a problem.

Stream cipher modes

These modes generate a pseudo random stream of data that may or may not depend the plaintext. Similarly to stream ciphers generally, the generated pseudo random stream is XORed with the plaintext to generate the ciphertext. As you can use as many bits of the random stream as you like you don't need padding at all.

Disadvantage of this simplicity is that the encryption is completely malleable, meaning that the decryption can be changed by an attacker in any way he likes as for a plaintext p1, a ciphertext c1 and a pseudo random stream r and attacker can choose a difference d such that the decryption of a ciphertext c2=c1⊕d is p2 = p1⊕d, as p2 = c2⊕r = (c1 ⊕ d) ⊕ r = d ⊕ (c1 ⊕ r).

Also the same pseudo random stream must never be used twice as for two ciphertexts c1=p1⊕r and c2=p2⊕r, an attacker can compute the xor of the two plaintexts as c1⊕c2=p1⊕r⊕p2⊕r=p1⊕p2. That also means that changing the message requires complete reencryption, if the original message could have been obtained by an attacker.

All of the following steam cipher modes only need the encryption operation of the block cipher, so depending on the cipher this might save some (silicon or machine code) space in extremely constricted environments.

CTR (counter) is simple, it creates a pseudo random stream that is independent of the plaintext, different pseudo random streams are obtained by counting up from different nonces/IVs which are multiplied by a maximum message length so that overlap is prevented, using nonces message encryption is possible without per message randomness, decryption and encryption are completed parallelizable, transmission errors only effect the wrong bits and nothing more

OFB (output feedback) also creates a pseudo random stream independent of the plaintext, different pseudo random streams are obtained by starting with a different nonce or random IV for every message, as with CTR using nonces message encryption is possible without per message randomness, decryption is parallelizable / encryption is not, as with CTR transmission errors only effect the wrong bits and nothing more

CFB's (cipher feedback) pseudo random stream depends on the plaintext, a different nonce or random IV is needed for every message, like with CTR and OFB using nonces message encryption is possible without per message randomness, decryption is parallelizable / encryption is not, transmission errors completely destroy the following block, but only effect the wrong bits in the current block

Disk encryption modes

Disk encryption modes specialized to encrypt data below the file system abstraction. For efficiency reasons changing some data on the disc must only require the rewrite of at most one disc block (512 bytes or 4kib). They are out of scope of this overview. Don't use them for anything except block level disc encryption.

Tweakable narrow-block encryption modes (**LRW**, **XEX**, and **XTS**) and wide-block encryption modes (**CMC** and **EME**) are designed to securely encrypt sectors of a disk.

Authenticated encryption

To prevent padding oracle attacks and changes to the ciphertext, one can compute a message authentication code (MAC) on the ciphertext and only decrypt it if it has not been tampered with. This is called encrypt-then-mac and should be preferred to any other order. Except for very few use cases authenticity is as important as confidentiality (the latter of which is the aim of encryption).

A number of modes of operation have been designed to combine secrecy and authentication in a single cryptographic primitive, these are called AEAD (Authenticated-Encryption with Associated-Data) schemes. Authenticated encryption modes are classified as single pass modes or double pass modes. For example, EAX mode is a double pass AEAD scheme while OCB mode is single pass.

Unfortunately for the cryptographic user community, many of the single pass authenticated encryption algorithms (such as OCB mode) are patent encumbered.
Such combine of the two part process of encryption and authentication into one block cipher mode that also produces an authentication tag in the process. In most cases this results in speed improvement

CCM is a simple combination of CTR mode and a CBC-MAC. Using two block cipher encryptions per block it is very slow.

OCB is faster but encumbered by patents. For free (as in freedom) or non-military software the patent holder has granted a free license, though.

GCM is a very fast but arguably complex combination of CTR mode and GHASH, a MAC over the Galois field with 2^128 elements. Its wide use in important network standards like TLS 1.2 is reflected by a special instruction Intel has introduced to speed up the calculation of GHASH.

Also see **XCBC**, **IACBC**, **IAPM**, **EAX**, **CWC**

Conclusion

AES is recommended by U.S. government and there is no known attack to it, if AES implemented properly (full rounds set) and used properly (no related keys).

Worst complexity is
- AES-128 to 2^126.1
- AES-196 to 2^189.7
- AES-256 to 2^254.4

which is enough (2^36 better than current practical limit) even for AES-128

So, currently AES have no known approaches to break. AES-128 is strong enough, but, if possible, accordingly to recommendations from U.S. government, AES-256 preferred.

Additional materials

1. Algebraic cryptanalysis of AES: An Overview, Harris Nover
2. An Overview of Cryptanalysis Research for the Advanced Encryption Standard
3. Attacks on Advanced Encryption Standard: Results and Perspectives, 2012
4. Cryptanalysis of the Full AES Using GPU-Like Special-Purpose Hardware, 2011
5. Key Recovery Attacks of Practical Complexity on AES Variants With Up To 10 Rounds
6. Federal Information Processing Standards Publication : Announcing the ADVANCED ENCRYPTION STANDARD (AES)
7. Computational Complexity: A Modern Approach
8. Block cipher mode of operation
9. How to choose an AES encryption mode (CBC ECB CTR OCB CFB)?

3DES

Triple DES uses a "key bundle" that comprises three DES keys, K1, K2 and K3, each of 56 bits

The standards define three keying options:
Keying option 1: All three keys are independent.
Keying option 2: K1 and K2 are independent, and K3 = K1.
Keying option 3: All three keys are identical, i.e. K1 = K2 = K3.

Known attacks

Key option 1, 168 bits

Keying option 1 is the strongest, with 3 × 56 = 168 independent key bits. Triple DES with three independent keys has a key length of 168 bits (three 56-bit DES keys), but due to the meet-in-the-middle attack, the effective security it provides is only 112 bits.

Key option 2, 112 bits

Keying option 2 provides less security, with 2 × 56 = 112 key bits. This option is stronger than simply DES encrypting twice, e.g. with K1 and K2, because it protects against meet-in-the-middle attacks.

However, this option is susceptible to certain chosen-plaintext or known-plaintext attacks, and thus, it is designated by NIST to have only 80 bits of security. Complexity of algorithm is estimated by the attack authors in **2^56**

Key option 3, 56 bits

Keying option 3 is equivalent to DES, with only **56 key bits**. This option provides backward compatibility with DES, because the first and second DES operations cancel out. It is no longer recommended by the National Institute of Standards and Technology (NIST) and is not supported by ISO/IEC 18033-3.

Classified sources

Accordingly to United States diplomatic cables leak

> *(7:55:26 AM) bradass87: DES / Triple DES... you're doomed in minutes*
> *(7:55:46 AM) bradass87: AES variants... take brute force*
> *(7:56:06 AM) bradass87: days to weeks to break*
> *(7:56:24 AM) bradass87: its about securing the keys, using complex enough keys...*
> *(7:56:42 AM) bradass87: and sticking to Rijndael variants*
> *- SNIP -*

> (7:58:06 AM) bradass87: RSA 1024 takes a few weeks... university of michigan finally broke it with a partial
> (7:59:00 AM) bradass87: 2048... never heard of it being broken publicly... NSA can feasibly do it, if they want to allocate national level "number-crunching" time to do it...

3des can be broken by NSA in minutes. **This means, that there are mathematical, undisclosed weakness, which may lead to ability to decrypt this algorithm, which can potentially be re-used by hackers**.

Conclusion

Accordingly to known attacks, only key option 1 (2^112) meets modern requirements and above modern theoretical brute-force limit (2^85).

This is not currently practical and NIST considers keying option 1 to be appropriate through 2030.

Also 3des with key-option 1 is one of 3 approved algorithms by NIST.

Nevertheless, classified sources marked this algorithm as potentially weak, so it should be avoided.

Also there is RFC Internet draft, created March 6, 2015 by MIT KIT, "Deprecate 3DES and RC4 in Kerberos"

Additional materials

1. Key-Schedule Cryptanalysis of IDEA, G-DES, GOST, SAFER, and Triple-DES
2. Manning-Lamo Chat Logs Revealed
3. Deprecate 3DES and RC4 in Kerberos

RC4

Known attacks

I'm not sure that I need to describe all known attacks on RC4 'cause this algorithm is completely compromised.

As of September, 2013, Edward Snowden suggest that the NSA can crack TLS/SSL connections, the widespread technology securing HTTPS websites and virtual private networks (VPNs) through RC4.

In the same year, group of security researchers at the Information Security Group at Royal Holloway, University of London reported an attack that can become effective using only 2^24 connections. While yet not a practical attack for most purposes, this result is sufficiently close to one that it has led to speculation that it is **plausible that some state cryptologic agencies may already have better attacks that render RC4 insecure**. See That earth-shattering NSA crypto-cracking: Have spooks smashed RC4? for more details.

In March 2015 researcher to Royal Holloway announced improvements to their attack, providing a 226 attack against passwords encrypted with RC4, as used in TLS

As of October 2014, Mozilla replaced RC4 with 3des in SSL/TLS, see bug 927045

As of February 2015, the IETF explicitly prohibits the use of RC4 in RFC 7465 because it has been proven that RC4 biases in the first 256 bytes of a cipherstream can be used to recover encrypted text. If the same data is encrypted a very large number of times, then an attacker can apply statistical analysis to the results and recover the encrypted text.

Accordingly to research, "*We also attack TLS as used by HTTPS, where we show how to decrypt a secure cookie with a success rate of 94% using 9*2^27 ciphertexts. This is done by injecting known data around the cookie, abusing this using Mantin's ABSAB bias, and brute-forcing the cookie by traversing the plaintext candidates. Using our traffic generation technique, we are able to execute the attack in merely 75 hours.*"

This publication was the crucial argument to deprecate RC4

List of known attacks can be found on Wikipedia, see "Fluhrer, Mantin and Shamir attack", "Klein's attack", "Royal Holloway attack", "Bar-mitzvah attack", "NOMORE attack" and All Your Biases Belong To Us: Breaking RC4 in WPA-TKIP and TLS, 2015 whitepaper.

Conclusion

RC4 must be avoided.

Additional materials

1. All Your Biases Belong To Us: Breaking RC4 in WPA-TKIP and TLS, 2015
2. RFC 7465 : Prohibiting RC4 Cipher Suites, 2015
3. RC4 crypto: Get RID of it already, say boffins, 2015
4. Killing RC4: The Long Goodbye
5. Schneier on Security : New RC4 Attack, 2015
6. Schneier on Security : New RC4 Attack, 2013
7. That earth-shattering NSA crypto-cracking: Have spooks smashed RC4?
8. HTTPS cookie crypto CRUMBLES AGAIN in hands of stats boffins, 2013
9. On the Security of RC4 in TLS and WPA paper, 2013
10. On the Security of RC4 in TLS and WPA report, 2013
11. Schneier on Security : Microsoft RC4 Flaw, 2005