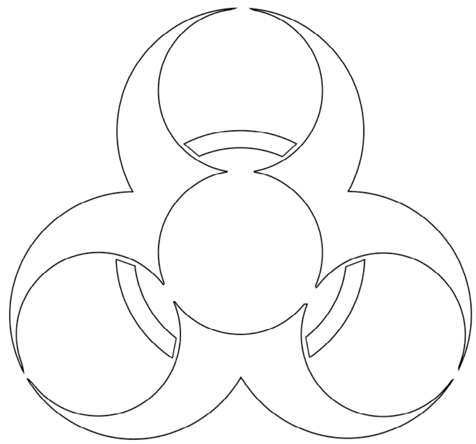
Building Robust Backdoors in Secret Symmetric Ciphers



Adam L. Young MITRE Corporation
joint work with Moti Yung Columbia Univ./RSA Labs

BlackHat 2005

Types of Backdoors

- Most of the time, when we hear about a “backdoor” it is a:
  - Self-contained malware program that resides on a victim’s machine

  or 

  - A Trojan Horse that was appended to or built into a program (e.g., passwd).

- A very interesting class of backdoors are those that are built into an algorithm (e.g., encryption algorithm).
  - These conform to public I/O specifications, but leak secret keys to the attacker.

Early Concern About Backdoors - US Air Force

- The concept of multi-user operating systems grew out of the need to make efficient use of expensive computing machinery.
- Prior to this, physical controls were used to maintain the security of batch processing machines.
- Effectiveness of such controls began to wane as soon as programs from different users began sharing memory on a single computer.
- In 1967 Petersen and Turn addressed computer subversion in an article that was published in the proceedings of the AFIPS Conference [PT67].

The Anderson Report

- The actual phrase “Trojan horse” appeared in a 1972 computer security technology planning study that was prepared for the USAF by James P. Anderson [An72].
- The Anderson report identifies the threat of a single user that operates as a hostile agent.
- It is noted that such an agent can simply modify an operating system to by-pass or suspend security controls.

The Data Encryption Standard (DES)

- 64-bit block size
- 56-bit key size
- Developed by IBM (for the most part)
- First published in the *Federal Register* on March 17, 1975
- Contains substitution box constants (S-Boxes)
- Contains permutation box constants (P-Boxes)
- Designed with the threat of *differential cryptanalysis* in mind

Backdoor in DES?

- Design criteria of S-boxes are not completely known [St95].
- Several people have suggested that the S-boxes might contain hidden “trapdoors” which would allow NSA to decrypt messages while maintaining that DES is “secure” [St95].
- S-boxes sent to Washington and when they came back they were all different [Sch96 – Konheim].

- Current wisdom is to choose cipher constants using:
  - A cryptographic hash function (publish the pre-image)
  - Universal constants (π, *e*, etc.)

The Digital Signature Standard (DSS)

- Published in the *Federal Register* on May 19, 1994.
- Adopted as a standard on December 1, 1994.
- Based on the Discrete-Logarithm problem in a prime order subgroup.
- Related to Schnorr & ElGamal
- Uses parameters (g,p,q) where:
  - p is a prime, |p| is any multiple of 64 between 512 and 1024 bits inclusive
  - q is a 160-bit prime such that q | p-1
  - g < p has order q modulo p
- DSS hash function = f(m) = SHA1(m) mod q
  - m is message being signed
- Produces 160 + 160 = 320-bit digital signatures

Backdoor in DSS?

- Some criticisms of DSS were put forward in 1991. Here is a non-exhaustive list [St95,Sch96]:
- Criticism 1: NIST selection process was not public. Standard developed by NSA without input from US Industry.

- Criticism 2: p fixed at 512 bits.
  - NIST altered standard to let p be larger.
- Criticism 3: (Lenstra-Haber) Certain moduli are easier to crack than others.
  - In FIPS PUB 186, May 1994, NIST recommended an algorithm for generating (p,q) using SHA1. This algorithm outputs (p,q,s,c) where s is a 160-bit seed and c ≥ 0 is a counter value.
  - In some sense, SHA1 “chooses” most of the bits in (p,q).

Approaches to Casting Doubt

1) Discover (reverse-engineer) a backdoor in an existing cryptographic algorithm (e.g., DES, DSS).

2) Design (forward-engineer) a backdoor in a new cryptographic algorithm that is designed according to "current wisdom".

current wisdom - acceptable key/block sizes, standard I/O specifications, security against chosen plaintext attacks, etc.

- The cryptographic research community has utilized approach (2) to investigate the possibility of deploying robust backdoors in block ciphers (robust = backdoor that can only be used by designer).
- This is evidenced by [RP97, PG97,YY98,YY03,YY04].

Reasons for Researching Backdoors

- We study ways of stealing keys using *cryptotrojans* so that we can better safeguard them in the future.
- We also study ways of stealing keys so that we can establish an appropriate level of faith in black-box cryptosystems (e.g., tamper-resistant microchips) or lack thereof.

Motivation 1 – RC4/Skipjack

- RC4
  - Symmetric Stream Cipher – variable key size
  - Designed by RSA Data Security Inc.
  - Originally a trade-secret (supposedly reverse-engineered and is now public)
- Skipjack
  - Symmetric Block Cipher – 64-bit block, 80-bit key
  - Designed by US Federal Government
  - Originally secret (now declassified)
- Can secret ciphers be trusted for commercial/personal use?
- Should secret ciphers be trusted for commercial/personal use?

Motivation 2 – DRM and Code Obfuscation

- Digital Rights Management efforts have sought to utilize code obfuscation and hardware to manage digital content ([CEJO02] obfuscated a cipher).
- Boneh et al. described a technique akin to differential fault analysis to cryptanalyze a simplified version of a software obfuscation package [JBF02].
- Boneh et al. suggested the possibility of using a secret design prior to obfuscating the code.
- These recent efforts show that the danger of using secret symmetric ciphers is still not being observed.

Backdoor in a Public Block Cipher

- Problem: Informally, the problem is to design a backdoor within a **published** block cipher such that the cipher:

  1) (indistinguishability) Utilizes a sound design paradigm that does not immediately reveal the presence of the backdoor.
     - This implicitly implies that the cipher algorithm does not expose (e.g.,contain) the master key. Hence, any such solution will necessarily be *asymmetric*.

  2) (completeness – user) Enables the sender and receiver to always be able to encipher/decipher messages correctly.

  3) (confidentiality) Constitutes a secure block cipher w.r.t. everyone except the designer (e.g., secure against chosen plaintext attacks).

  4) (completeness - designer) Permits the malicious designer to efficiently learn plaintext information of users of the cipher using a secret “master key”.

Approaches to Building Backdoors in Public Block Ciphers…and their Cryptanalysis

- Rijmen and Preneel presented a construction [RP97].
- It was cryptanalyzed in Asiacrypt ’98 by Wu, Bao, Deng, and Ye [WBDY98].

- Patarin and Goubin presented a construction [PG97].
- It was cryptanalyzed in Crypto ’99 by Ding-Feng et al. [DKZ99].
- It was also cryptanalyzed in Eurocrypt ’00 by E. Biham [Bi00].

- This is a hard problem.
- New solutions may involve new intractability assumptions and might therefore be questionable.

Backdoor in a Secret Block Cipher

- Problem: Informally, the problem is to design a backdoor within a secret block cipher such that the cipher:

   1) (indistinguishability) Utilizes a sound design paradigm that does not reveal the presence of the backdoor via black-box queries.

   2) (completeness – user) Enables the sender and receiver to always be able to encipher/decipher messages correctly.

   3) (confidentiality) Constitutes a secure block cipher w.r.t. a successful reverse-engineer that learns the design, but is not secure w.r.t. the designer.
      - Note that **security w.r.t. the reverse-engineer implies security against the user** (that has even less information).

   4) (completeness - designer) Permits the malicious designer to efficiently learn plaintext information of users of the cipher using a secret "master key".

Master key Cryptosystems

- It has been shown that a backdoor block cipher is equivalent to a public key cryptosystem [BFL95].

1) Bob gives Alice a backdoor block cipher that he designs.

2) Alice chooses symmetric key k randomly.

3) Alice encrypts messages with k and sends ciphertexts to Bob.

4) Bob uses the master key (backdoor) to read her messages.

Roadmap for Talk

- An overview will be given of the "Monkey" backdoor block cipher (from [YY98])
  - Presents 2 key ideas:
    - Utility of known-plaintext to extract backdoor info
    - Asymmetric encryption of symmetric key as shared string
  - Leaks a plaintext bit to reverse-engineer (quasi-setup attack)
- An overview will be given of the "Black-Rugose" backdoor block cipher (from [YY03]).
  - Presents 2 key ideas:
    - Enlarges block size (just like in AES) to yield more "elbow room"
    - Uses data compression to create a subliminal channel
  - Backdoor not accessible with high-entropy plaintexts.
- A detailed description will be given of the "Insignis" backdoor block cipher (from [YY04])
  - Solves all previous problems

17 of 50

Subliminal Channels

- A subliminal channel is a communications channel out of (or into) a cryptosystem that can be used to transmit information in an undetectable fashion.
- Prisoners' Problem: Simmons' original formulation [Si84] has two prisoner's communicating using a subliminal channel in digital signatures.
- The prisoners are allowed to sign data but not encrypt data.
- The *warden* verifies signatures but cannot detect the use of the channel even when he tries.

18 of 50

Channels in Probabilistic Cryptosystems

- The vast majority of subliminal channels utilize the random tape used by the host cryptosystem (the cryptosystem that is being subverted).
- These subliminal channel attacks take control of the randomness source in the host cryptosystem and use acceptance/rejection on the input random stream to obtain a bit sequence that enables secret data to be leaked securely and subliminally in the normal outputs of the cryptosystem.
- This type of acceptance/rejection does not appear to be possible in a deterministic block encryption function (without ruining indistinguishability, completeness w.r.t. the user, etc.).
- So building a channel into a deterministic encryption function is non-trivial.

19 of 50

Key Idea #1 - Virtual Pseudo-One Time Pad

- Q: How can a channel be built into a deterministic function?
- Q: How can a channel be built into a deterministic block cipher where the length of the output in bits equals the length of the input?
- A: we "appropriate" bandwidth by assuming that the attacker will be able to obtain a set of plaintext/ciphertext pairs under one symmetric key k [sec. 3.2 of RP97,YY98].
- The plaintext block m and corresponding ciphertext block c_k define a virtual encryption "pad" r_k. Through the use of the block encryption algorithm, k leads to $c_k = m \oplus r_k$.
- The "virtual pad" r_k "carries" the subliminal information that constitutes m_s.

20 of 50

Key Idea #2 - Deterministic Asymmetric Encryption

- **Fact 1**: A deterministic asymmetric encryption function always produces the same ciphertext for a given plaintext.
- Example: The RSA encryption function $E_n(m)$ is deterministic (public key is (e,n)).
- **Fact 2**: The block encryption function ENC and decryption function DEC of a symmetric cipher are given a shared string k (the secret symmetric key). So ENC and DEC both "share" k.
- Observation: When a cryptotrojan is planted into ENC and DEC, they both "share" the common string $E_n(k)$.
- Key Idea #2: Somehow "display" a bit of $E_n(k)$ in each ciphertext that is output. Key Idea #1 tells us this may be possible. Note that $E_n(k)$ in no way compromises k even if $E_n(\cdot)$ becomes public.

The Monkey Cipher

- Tailored after Skipjack.
- Plaintext m is 64-bits (64-bit block size)
- Symmetric key k is 80-bits
- Goal: Leak the 80-bit key securely and subliminally to the designer using known plaintext attack.
- Leaks one bit of the asymmetric ciphertext of k in each ciphertext.
- Recall that this is a deterministic asymmetric cipher.

Monkey Overview

- Backdoor attack uses *pseudorandom functions*.
- The 1-bit leak is successful provided that the most significant plaintext bit is known for the given ciphertext.
- A 63-bit block cipher simply encrypts the 63 lower order bits.
- The most significant bit is XOR encrypted.
- The XOR is of a pad bit, the most significant plaintext bit, and a pseudorandomly selected bit from the asymmetric ciphertext.
- The 1-bit pad is pseudorandomly chosen.
- This omits certain details (pre and post processing, etc.)
- Attacker collects the bits of the asymmetric ciphertext and then reassembles them. The attacker then decrypts it (using the private key that only he knows).

Monkey Security

- The reverse-engineer will learn all the seeds to the pseudorandom functions.
- The reverse-engineer will, after seeing enough Monkey ciphertexts under a particular key k, reassemble the asymmetric ciphertext of k in it's entirety.
- The reverse-engineer will always know the pad bit and the bit position of the asymmetric ciphertext bit for each Monkey ciphertext c.
- This means that the reverse-engineer will know the most significant plaintext bit of every Monkey ciphertext.
- The attacker, however, learns all plaintext bits…

Key Idea #3 - Exploit Low Entropy Plaintexts

- Observation 1: Often **significant numbers of plaintexts are low-entropy**.
- Observation 2: The plaintext space $\{0,1\}^{192}$ is huge and **a user will not be able to collectively encrypt every plaintext** in any reasonable length of time.
- Idea #3:
  - Compress plaintexts that can be compressed and in the free space leak the following:
    1) Securely encoded piece of the user's symmetric key.
    2) Cryptographic checksum that allows the designer to distinguish between "normal" encryptions and "attacked" encryptions.
  - Collisions can occur that cause decryption to fail.
- Observation 2 suggests that collisions can be made to be very rare.

25 of 50

The Black-Rugose Cipher

- Has block size of 192-bits (tailored after Rijndael).
- Symmetric key k is 192-bits
- Goal: Leak the 192-bit key securely and subliminally to the designer whenever the plaintexts are "sufficiently" redundant.
- For each sufficiently redundant plaintext:
  - Rugose leaks M pseudorandomly chosen asymmetric ciphertext bits (of the encryption of k).
- Recall that this is a deterministic asymmetric cipher.
- *coupon collector's problem* [Fe57] tells how many Rugose block ciphertexts you would expect to need in order to reassemble the asymmetric ciphertext (this is covered in more detail for Insignis).

26 of 50

Black-Rugose Overview

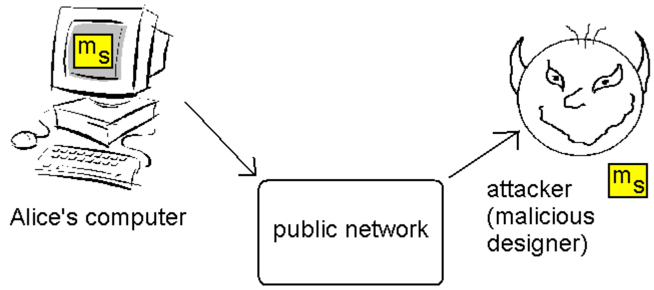
- Encryption algorithm encrypts normally if message won't Huffman compress to 96 bits or less.
- If message compresses then a tiny Huffman tree t is computed.
- The tree and compressed plaintext are concatenated and then symmetrically encrypted using a (128-M)-bit block cipher.
- M bit positions are pseudorandomly selected from asymmetric ciphertext.
- The corresponding asymmetric ciphertext bits are XOR encrypted with M pseudorandom pad bits.
- The resulting XOR ciphertext is concatenated with a 64-bit cryptographic checksum
- 64 + M + (128 - M) = 192 bits
- M can be, say, 11

Black-Rugose Security

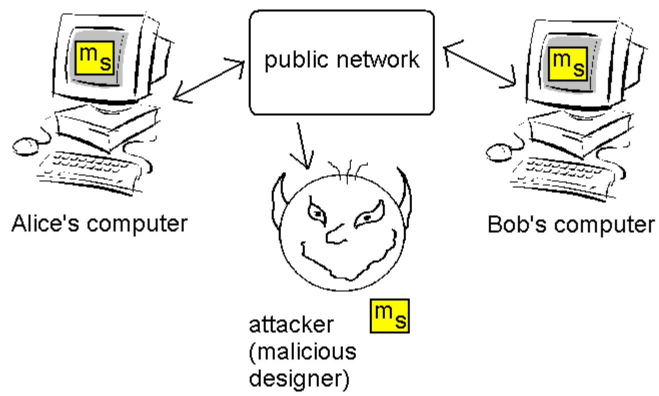
- Security analysis bounds from above the probability that decryption fails.
  - This is necessary for indistinguishability.
- Security analysis shows that the reverse-engineer learns at most a bound (upper or lower) on the entropy of each Black-Rugose ciphertext.
- See the proceedings of ACISP '03 for details.

Traditional Subliminal Channel

- Malicious designer puts backdoor into Alice’s black-box cryptosystem that compromises her keys through the subliminal message m_s.
- Alice uses her probabilistic black-box cryptosystem (e.g., RSA key generation algorithm, DSA signing alg. etc.) and publishes its output.
- Malicious designer obtains public outputs and extracts her keys from the subliminal channel

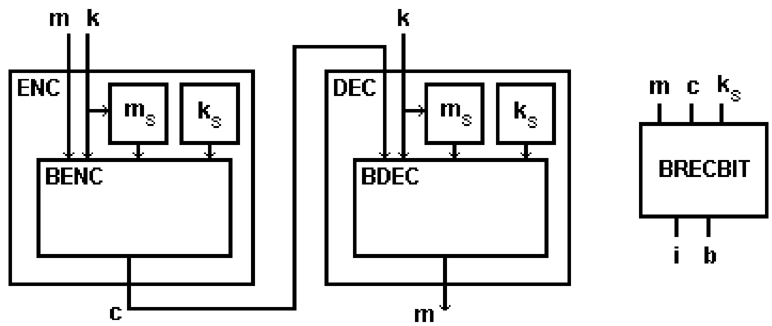


The Insignis Channel



- The new channel is a channel in a secret block cipher.
- The sending and receiving devices must know the subliminal message m_s.
- In our main application, $m_s = E(k)$, the asymmetric encryption of the symmetric key using the attacker’s public key.
- The message m_s is encoded in the ciphertexts output by the block cipher.
- The attacker must obtain a “sufficient” number of plaintext/ciphertext pairs under key k.
- From this the attacker can recover $m_s = E(k)$ and therefore $k = D(E(k))$.

Picture of Overall Attack



Block Cipher Notation

- A **block cipher** is a pair of algorithms (ENC,DEC) that is used to encrypt and decrypt plaintext messages m that are w bits in length.
- The encryption is performed using a symmetric key k such that for all m, the equality,

  $$m = DEC(k,ENC(k,m))$$

  holds.

Ideal Classic Cipher

- An **ideal classic cipher** implements a randomly chosen subset of all $2^w!$ permutations from the message space $\{0,1\}^w$ onto the ciphertext space $\{0,1\}^w$.
- They are secure against chosen-plaintext attacks.
- It is standard practice to make the cardinality of the key space exponential in some security parameter.
- A design principle for a block cipher is to make the cipher as close to an ideal classic cipher as possible.

The Parties

- The **designer** (oracle designer) is a malicious entity that is permitted to design and deploy the black-box device. The goal of the designer is to learn m_s.
- The **reverse-engineer** (oracle access) mounts a chosen-plaintext attack and wants to learn plaintext information (and m_s).
- The **inquirer** (oracle access) is a user (adversary) that tries to distinguish whether the oracle is a “good” or “bad” ideal classic cipher.
- The **sampler** (oracle access) is a user (adversary) that tries to choose a probability distribution that allows the reverse-engineer to violate the confidentiality of encryptions.

Definition of a Broadcast Block Cipher

(formal def.) A secure w-bit **broadcast block cipher** is a 3-tuple (BENC,BDEC,BREC) that satisfies the following:

1. (**inquirer indistinguishability**) It is computationally intractable for the inquirer to distinguish a black-box implementation of (ENC,DEC) from a randomly chosen ideal classic cipher.
2. (**completeness**) For all plaintexts m, for all symmetric keys k, for all subliminal messages m_s, and for all secret keys k_s, $m = BDEC(k,BENC(k,m,k_s,m_s),k_s,m_s)$.
3. (**reverse-engineer confidentiality**) After completing a chosen-plaintext attack, the reverse-engineer learns at most W pairs of random plaintexts and corresponding ciphertexts (computed using k) where W is bounded by a polynomial in the length of $((m_1,c_1),...,(m_\alpha,c_\alpha),k_s)$.
4. (**designer completeness**) For sufficiently large α, for all m_s, and for all distinct plaintexts $(m_1,m_2,...,m_\alpha)$, the subliminal message $m_s = BREC((m_1,m_2,...,m_\alpha,c_1,c_2,...,c_\alpha),k_s)$ with overwhelming probability where each ciphertext $c_i = BENC(k,m_i,k_s,m_s)$ for $i = 1,2,...,\alpha$. The probability is over the random choice of k and k_s.

35 of 50

Intuition Behind the Construction (Part 1)

- The channel transmits one pseudorandomly chosen bit of the subliminal message in each ciphertext block that is output.
- First, a large portion of the block is simply encrypted using a secure block cipher and part of the user's symmetric key.
- The resulting ciphertext is recoverable under a known plaintext attack by the reverse-engineer.
- This ciphertext is used as a **"public" string** that is input to a random function.
- This public input is also used to select a bit position randomly in the subliminal message.
- The problem is then to display the subliminal bit in this bit position **and** encrypt the remaining plaintext.
- But now we have a **"public" string** that is (with overwhelming probability—by birthday paradox) unique to this plaintext.

36 of 50

Intuition Behind the Construction (Part 2)

- The subliminal bit is embedded in the remaining plaintext data.
- To do so, the **"public" string** is supplied to a random function along with the user's symmetric key.
- The result is a random pad that is used to XOR encrypt all but the last remaining plaintext bit.
- The last plaintext bit is also XOR encrypted.
- This is accomplished by supplying the pad along with the public input to yet another random function to obtain a one bit pad.
- The 1-bit pad is XORed with the 1-bit plaintext which is XORed with the bit from m_s.
- IDEA: The *larger pad* is secret due to the secrecy of the user's symmetric key, and so this larger pad can be used to derive a *smaller pad* (1-bit) to XOR encrypt the final plaintext bit.
- An initial permutation and a final permutation are also performed for reasons that will become clear later on.

Building Blocks

- GetBit(s,i) returns the bit at position i of bit string s where $i \in \{0,1,2,\ldots,|s| - 1\}$. The bits are ordered from right to left starting with 0.

  If s = 0001 then GetBit(s,0) = 1
  GetBit(s,1) = 0
  GetBit(s,2) = 0
  GetBit(s,3) = 0

- Let $H_{\delta-1} : \{0,1\}^* \rightarrow \{0,1\}^{\delta-1}$ where δ is a constant
  $F_1 : \{0,1\}^* \rightarrow \{0,1\}$
  $GetRandPos_\theta : \{0,1\}^* \rightarrow \{0,1,2,\ldots,\theta - 1\}$
  be public random functions.
- (ENC1,DEC1) is a secret ideal classic cipher with a w-bit block size.
- (ENC2,DEC2) is a secret ideal classic cipher with a $(w-\delta)$-bit block size.

BENC Encryption Algorithm

$f(b, \beta_U, k_L) = H_{\delta-1}(k_L \;||\; \beta_U) \;||$
$\quad (b \text{ XOR } F_1(H_{\delta-1}(k_L \;||\; \beta_U) \;||\; \beta_U))$

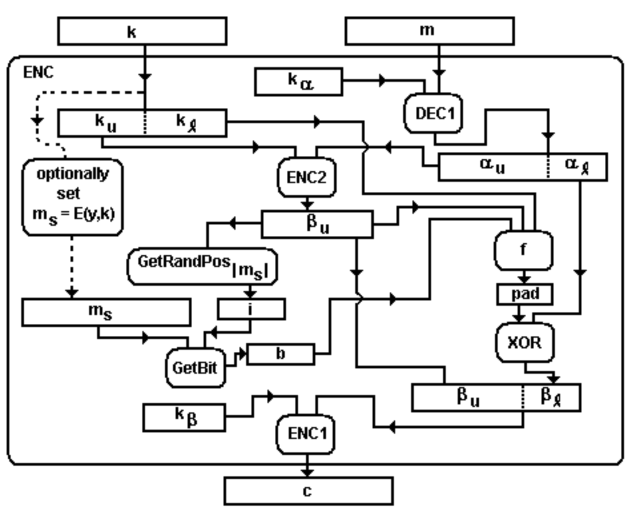
$\Pi_1(k, m, k_\alpha, m_s)$:

1. let k_U and k_L be strings such that
   $k = k_U \;||\; k_L$ and $|k_U| = |k_L|$
2. compute α = DEC1(k_α, m)
3. let α_U and α_L be strings such that
   $\alpha = \alpha_U \;||\; \alpha_L$ and $|\alpha_L| = \delta$
4. β_U = ENC2(k_U, α_U)
5. set $\theta = |m_s|$
6. set i = GetRandPos$_\theta$(β_U) and set b = GetBit(m_s, i)
7. set pad = f(b, β_U, k_L) and set β_L = pad XOR α_L
8. return $\beta = \beta_U \;||\; \beta_L$

BENC(k, m, (k_α, k_β), m_s):

1. $\beta = \Pi_1(k, m, k_\alpha, m_s)$
2. output c = ENC1(k_β, β)

The BENC Encryption Algorithm



BDEC Decryption Algorithm

$\Pi_{-1}(k,c,k_\alpha,m_s)$:

1. let k_U and k_L be strings such that $k = k_U \mid\mid k_L$ and $|k_U| = |k_L|$
2. compute $\beta = \mathrm{DEC1}(k_\beta,c)$
3. let β_U and β_L be strings such that $\beta = \beta_U \mid\mid \beta_L$ and $|\beta_L| = \delta$
4. $\alpha_U = \mathrm{DEC2}(k_U,\beta_U)$
5. set $\theta = |m_s|$
6. set $i = \mathrm{GetRandPos}_\theta(\beta_U)$ and set $b = \mathrm{GetBit}(m_s,i)$
7. set $pad = f(b,\beta_U,k_L)$ and set $\alpha_L = pad \text{ XOR } \beta_L$
8. return $\alpha = \alpha_U \mid\mid \alpha_L$

$\mathrm{BDEC}(k,c,(k_\alpha,k_\beta),m_s)$:

1. $\alpha = \Pi_{-1}(k,c,k_\alpha,m_s)$
2. output $m = \mathrm{ENC1}(k_\alpha,\alpha)$

BREC Recovery Algorithm

- BRECBIT recovers a single bit of m_s from a plaintext/ciphertext pair. BREC invokes this subroutine for each plaintext/ciphertext pair that it is given in order to recover m_s.

$\mathrm{BRECBIT}((k_\alpha,k_\beta),m,c)$:

1. compute $\alpha = \mathrm{DEC1}(k_\alpha,m)$
2. let α_U and α_L be strings such that $\alpha = \alpha_U \mid\mid \alpha_L$ and $|\alpha_L| = \delta$
3. compute $\beta = \mathrm{DEC1}(k_\beta,c)$
4. let β_U and β_L be strings such that $\beta = \beta_U \mid\mid \beta_L$ and $|\beta_L| = \delta$
5. compute $pad = \alpha_L \text{ XOR } \beta_L$
6. let z and t be strings such that $pad = z \mid\mid t$ and $|t| = 1$
7. compute $r = F_1(z \mid\mid \beta_U)$ and then set $b = t \text{ XOR } r$
8. set $\theta = |m_s|$
9. set $i = \mathrm{GetRandPos}_\theta(\beta_U)$ and then output (i,b)

Running Time

- Note that each bit is selected uniformly at random from the $|m_s|$ bit positions.
- So, the designer (and the reverse-engineer) can expect to have to obtain $O(|m_s| \log |m_s|)$ plaintext/ciphertext pairs under a common key k in order to recover m_s.
- This results from analyzing the first moment of the *coupon collector's problem* [Fe57].

Claims (Part 1)

- Claim 1: For all k, m_s, and k_α, $\pi_1(k,\bullet,k_\alpha,m_s)$ is a permutation over $\{0,1\}^w$.
  - Prove by contradiction
- Claim 2: A secret implementation of (ENC,DEC) is indistinguishable from an ideal classic cipher.
  - From Claim 1, the fact that the composition of two permutations is a permutation, and since k_β is secret…
- Corollary 1: With only oracle access, ENC appears like a randomly chosen invertible function.
  - From Claim 2 and the notion of security for an *ideal classic cipher*.

Claims (Part 2)

- Observe that in (ENC,DEC) there exist non-trivial distributions M_p that compromise plaintexts. These M_p's lead to a non-negligible probability of collision in β_U. A collision in β_U implies a collision in pad.
- So, it must be shown that the chances that the sampler compromises its own plaintexts is negligible.
- Define p_c to be the probability that two messages m_1 and m_2 that are chosen according to M_p lead to the same value for β_U in the corresponding encryptions c_1 and c_2.
- Claim 3: (random oracle model) If ENC2 is an ideal classic cipher and w-δ is sufficiently large then p_c is negligible.
  - From Corollary 1 and *Birthday Paradox*.
- Claim 4: (random oracle model) If p_c is negligible and k_L is secret then with overwhelming probability the values for pad that result (from the **sampler**'s choice of plaintexts) in the resulting ciphertexts are independently random and secret.

Known Plaintext Attack

- In a known plaintext attack the reverse-engineer queries an encryption oracle (the sampler) and receives $S = \{(m_1,c_1), (m_2,c_2),\ldots, (m_\gamma, c_\gamma)\}$.
- Consider the problem for the reverse-engineer to learn information relating to the plaintext in c where $c \neq c_i$ for all $i \in \{1,2,\ldots, \gamma\}$.
- This is possible in the following known plaintext attack:

1) The reverse-engineer computes pad, α_L, β_U, α_U, etc. used in (m_1,c_1).
2) The reverse-engineer begins to iterate through the possible values for α_L (there are 2^δ possible values in total).
3) The values $\alpha = \alpha_U \| \alpha_L$ are encrypted using k_α in the cipher ENC1.
4) This yields a set of new **RANDOM** plaintexts for ENC.
5) The reverse-engineer computes the corresponding ciphertexts in the same way that ENC would.

Security

- So, given that we don't have security against plaintext attacks (with respect to reverse-engineer only), what do we have?
- Since ENC1 is an ideal classic cipher, it follows that the new plaintexts that are learned are random.
- So, the reverse-engineer can only sample the new plaintext/ciphertext pairs *randomly*.
- With Claims 3 and 4 this shows that (BENC,BDEC,BREC) is a broadcast block cipher.

47 of 50

Conclusion

- We reviewed the history of backdoors in both operating systems/programs and ciphers.
- Specifically we touched on the controversy over DES and DSS.
- We showed 2 approaches to casting doubt on cipher designs (reverse-engineering and forward-engineering).
- We covered the notions of building a backdoor into a public cipher vs. a secret cipher.
- We covered backdoor cipher designs for the "secret" ciphers:
  - Monkey
  - Black-Rugose
  - Insignis

48 of 50

References 1

- [An72] Anderson, J. P. Computer Security Technology Planning Study. ESD-TR-73-51, vol. II, 1972.
- [Bi00] E. Biham, “Cryptanalysis of Patarin’s 2-Round Public Key System S Boxes (2R),” Eurocrypt ’00.
- [BFL95] M. Blaze, J. Feigenbaum, F. T. Leighton, “Master-Key Cryptosystems,” Crypto ’95 rump session.
- [CEJO02] S. Chow, P. Eisen, H. Johnson, P. C. van Oorschot. “A White-Box DES Implementation for DRM Applications,” Workshop on DRM, ACM, 2002.
- [DKZ99] Y. Ding-Feng, L. Kwok-Yan, D. Zong-Duo, “Cryptanalysis of the “2R” schemes,” Crypto ’99.
- [Fe57] W. Feller. An Introduction to Probability Theory and its Applications. John Wiley & Sons, Inc., pages 210-212, 1957.
- [JBF02] M. Jacob, D. Boneh, E. Felten, “Attacking an obfuscated cipher by injecting faults,” Workshop on DRM, ACM, 2002.
- [PG97] J. Patarin, L. Goubin. Asymmetric Cryptography with S-Boxes. In Proceedings of ICICS, pages 369-380, 1997.

49 of 50

References 2

- [PT67] Petersen, H. E., Turn, R. System Implications of Information Privacy. Proc. of the AFIPS Spring Joint Computer Conference, v. 30, pp. 291-300, 1967.
- [RP97] V. Rijmen, B. Preneel, A Family of Trapdoor Ciphers. Fast Software Encryption, pages 139-148, 1997.
- [Sch96] B. Schneier, *Applied Cryptography*, 2nd ed. Wiley, 1996.
- [Si84] G. J. Simmons, "The Prisoner’s Problem and the Subliminal Channel," In Proceedings of Crypto '83, 1984.
- [St95] D. Stinson, Cryptography – Theory and Practice, 1st Ed., CRC Press, 1995.
- [WBDY98] H. Wu, F. Bao, R. Deng, Q. Ye, “Cryptanalysis of Rijmen-Preneel Trapdoor Ciphiers,” Asiacrypt ’98.
- [YY98] A. Young, M. Yung, "Monkey: Black-Box Symmetric Ciphers Designed for monopolizing keys," In proceedings of Fast Software Encryption, 1998.
- [YY03] A. Young, M. Yung, "Backdoor Attacks on Black-Box Ciphers Exploiting Low-Entropy Plaintexts," In proceedings of ACISP, 2003.
- [YY04] A. Young, M. Yung, “A Subliminal Channel in Secret Block Ciphers,” SAC, 2004.

50 of 50