BTI 4202: Threat Landscape

Christian Grothoff

BFH

23.4.2021

Learning Objectives

Cyber attacks and actors

Software vulnerabilities

What are Cryptographic Protocols?

Example: Protocol vulnerability

Review: Cipher modes

Security definitions: IND-CPA

Example: Attack on CBC Stateful IV

Part I: Cyber Attacks and Actors

Attacker origins

- Insider
- Ex-insider (“disgruntled former employee”)
- Competitor
- Hacktivist
- Criminal
- State actor
- *Researcher*

Attacker objectives

- Fame
- Stealing information (business secrets, credentials)
- Modifying information (e.g. bank transactions)
- Abusing infected systems (e.g. spamming)
- Attacking other systems (origin obfuscation)
- Hiding (avoid detection, achieve long-term persistence)
- Contact command and control (C2) for instructions

Vulnerability origins

- Hardware (host, network)
- Software (host, network)
- Humans
- Environment

Attack strategies

- Large scale attack: attack a large, untargeted population. Even if the success rate is low, the absolute number of infections and the resulting revenue can be high. ("cyber crime")
- Targeted attack: attack a few, selected users or their machines. Select high-value target first, then learn about it as much as possible for a precision strike ("Advanced persistent threat")

Defense strategies

- Access control (physical, logical)
- Deterrance (legal, counter-attacks, auditing, accounting)
- Redundancy
- Obfuscation
- Comprehension (simplification, transparency, education)
- Monkey wrench / havoc
- Defense-in-depth

Part II: Software vulnerabilities

Technical vulnerabilities

There are many types of technical vulnerabilities in various parts of an IT system:

- Misconfigured firewalls
- Hardware bugs
- Automatically executed software from CD/USB stick on old W32 systems
- etc.

The probabily most important class of technical vulnerabilities are software bugs.

Typical bugs

Software is often used to display data obtained over the network:

1. User downloads file (PDF, MP4, etc.)
2. User selects software to open file
3. Software parses file
4. Bug $\Rightarrow$ malicious code execution

Common bugs include problems in the parsing or rendering logic, or scripting functionality supported by the document format in combination with an interpreter that is insufficiently sandboxed.

Data and code

The central goal for an attack is to turn data into code. Memory of a process contains data and code! Thus:

- Existing code may interpret the data (intentionally or unintentionally), thereby allowing certain code sequences to be executed.
- Existing code may be caused to jump to the data (once data page is set to executable).
- Execution may be passed to another program (shell, interpreter) that will parse and run it.

Example exploit: SQL injection

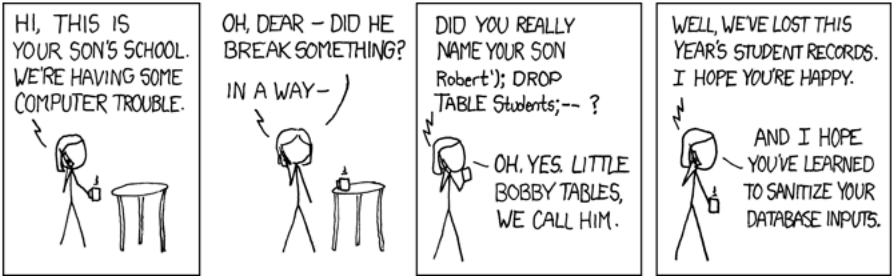
In a PHP script, hopefully far, far away:

```
SELECT (user, first_name, last_name)
FROM students
WHERE (user == '$user');
```

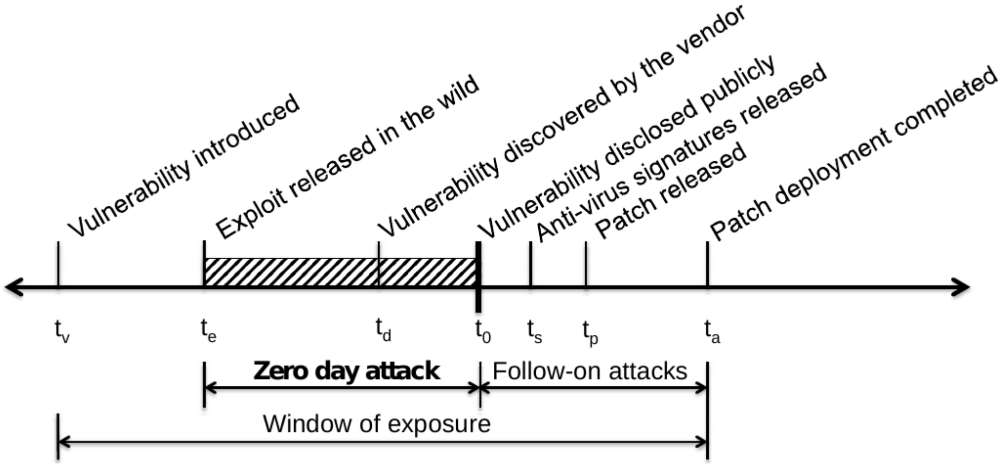
Input:

```
Robert'); DROP TABLE students;--
```

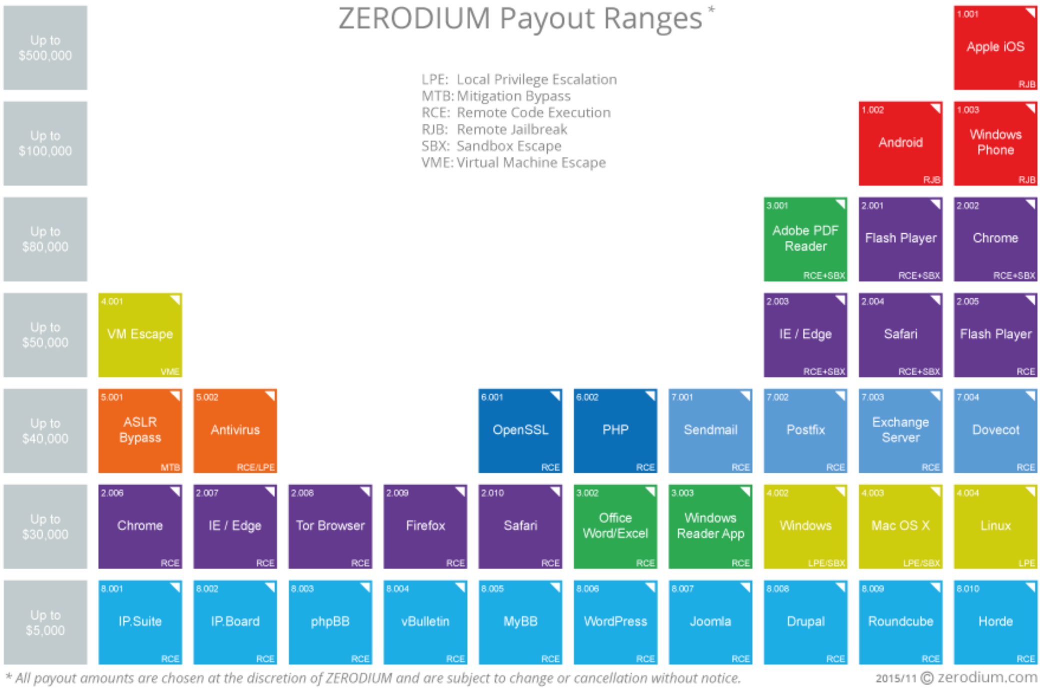
XKCD


HI, THIS IS YOUR SON'S SCHOOL. WE'RE HAVING SOME COMPUTER TROUBLE.
OH, DEAR – DID HE BREAK SOMETHING?
IN A WAY–
DID YOU REALLY NAME YOUR SON Robert'); DROP TABLE Students;-- ?
OH. YES. LITTLE BOBBY TABLES, WE CALL HIM.
WELL, WE'VE LOST THIS YEAR'S STUDENT RECORDS. I HOPE YOU'RE HAPPY.
AND I HOPE YOU'VE LEARNED TO SANITIZE YOUR DATABASE INPUTS.

Vulnerability timeline



Capitalism


ZERODIUM Payout Ranges *
LPE: Local Privilege Escalation
MTB: Mitigation Bypass
RCE: Remote Code Execution
RJB: Remote Jailbreak
SBX: Sandbox Escape
VME: Virtual Machine Escape
Up to $500,000
1.001 Apple iOS RJB
Up to $100,000
1.002 Android RJB
1.003 Windows Phone RJB
Up to $80,000
3.001 Adobe PDF Reader RCE+SBX
2.001 Flash Player RCE+SBX
2.002 Chrome RCE+SBX
Up to $50,000
4.001 VM Escape VME
2.003 IE / Edge RCE+SBX
2.004 Safari RCE+SBX
2.005 Flash Player RCE
Up to $40,000
5.001 ASLR Bypass MTB
5.002 Antivirus RCE/LPE
6.001 OpenSSL RCE
6.002 PHP RCE
7.001 Sendmail RCE
7.002 Postfix RCE
7.003 Exchange Server RCE
7.004 Dovecot RCE
Up to $30,000
2.006 Chrome RCE
2.007 IE / Edge RCE
2.008 Tor Browser RCE
2.009 Firefox RCE
2.010 Safari RCE
3.002 Office Word/Excel RCE
3.003 Windows Reader App RCE
4.002 Windows LPE/SBX
4.003 Mac OS X LPE/SBX
4.004 Linux LPE
Up to $5,000
8.001 IP.Suite RCE
8.002 IP.Board RCE
8.003 phpBB RCE
8.004 vBulletin RCE
8.005 MyBB RCE
8.006 WordPress RCE
8.007 Joomla RCE
8.008 Drupal RCE
8.009 Roundcube RCE
8.010 Horde RCE
* All payout amounts are chosen at the discretion of ZERODIUM and are subject to change or cancellation without notice.
2015/11 © zerodium.com

Part III: Cryptographic Protocols

Protocols

- "A **protocol** is a series of steps, involving two or more parties, designed to accomplish a task."
- Everyone involved must know the steps in advance and agree to follow it.
- The protocol must be complete and unambiguous.
- For cryptographic protocols, it should not be possible to do more or learn more than *what is specified in the protocol*.

Dramatis Personae

- Alice, Bob, Carol and Dave
- Eve – Eavesdropper
- Mallory – Malicious active attacker
- Trent – Trusted arbitrator
- Walter – Warden
- Peggy – Prover
- Victor – Verifier

Attack Personae

- Eavesdroppers
- Passive cheaters
- Active cheaters
- Real-world adversaries – Mallory

Efficiency

- Number of steps in protocol
- Size of messages
- Conflict resolution cost:
  1. Involvement of trusted party (arbitrated protocols)
  2. Resolution by trusted party on dispute (adjudicated protocols)
  3. Self-enforcing protocols

Example: Symmetric Cryptography

1. Alice and Bob agree on a cryptosystem
2. Alice and Bob agree on a key
3. Alice encrypts plaintext with key
4. Alice sends ciphertext to Bob
5. Bob decrypts ciphertext and reads it

Problem

Alice has an item x, and Bob has a set of five distinct items $y_1, \ldots, y_5$. Design a protocol through which Alice (but not Bob) finds out whether her x equals any of Bob's five items; Alice should not find out anything other than the answer ("Yes" or "No") to the above question, and Bob should not know that answer. Your solution must always be correct, not just with high probability.

Break

Part IV: Example: Protocol vulnerability

Guiding questions “Making the Theoretical Possible”

- What is the root cause of the vulnerability exploited in the attack?
- What does the attack achieve?
- Summarize the attack (how does it work?, capture every step!)
- Comment on the different “levels” of breaking a hash function (i.e. what is achieved in the attack that goes beyond finding an arbitrary collision).

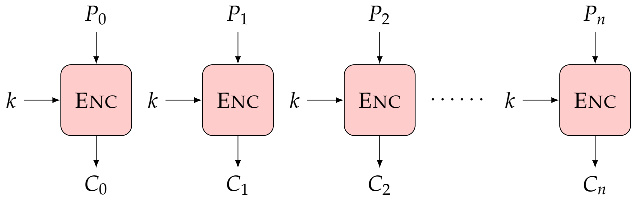
MD5: Making the Theoretical Possible

25c3, 2008

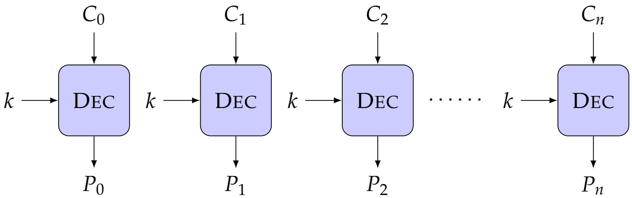
Break

Part V: Review: Cipher modes

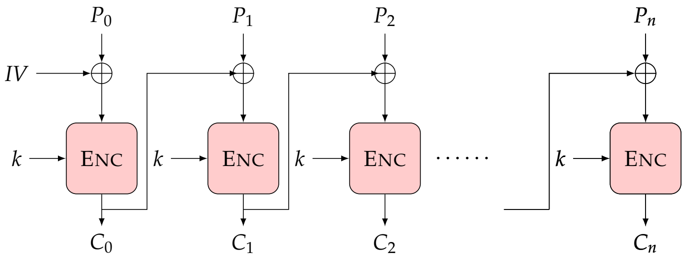
ECB encryption



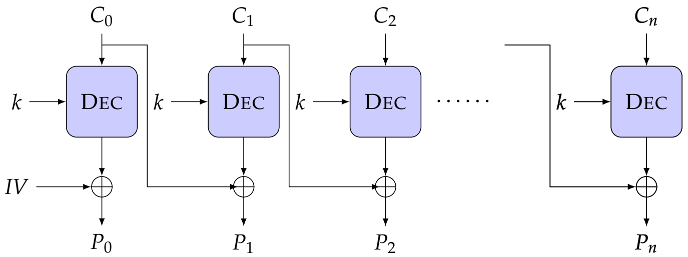
ECB decryption



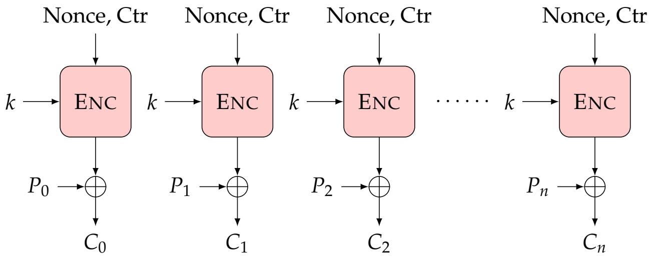
CBC encryption



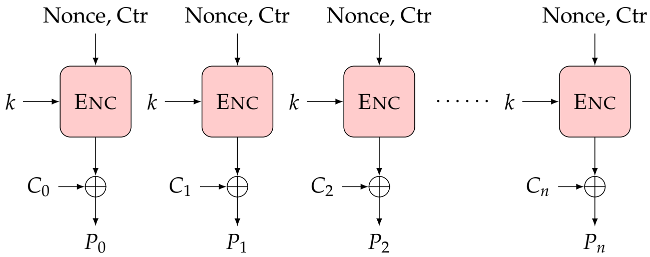
CBC decryption



CTR encryption



CTR decryption



Problem

Which mode is secure?

Problem

Which mode is secure?

How to prove it?

Security Definitions for Symmetric Encryption

Simplistic security definitions would be:

1. It must be impossible for an adversary to find the key from ciphertexts.
2. It must be impossible for an adversary to find the plaintext from a ciphertext.

Security Definitions for Symmetric Encryption

Simplistic security definitions would be:

1. It must be impossible for an adversary to find the key from ciphertexts.
2. It must be impossible for an adversary to find the plaintext from a ciphertext.

These are insufficient as, for example, they do not capture the insecurity of the ECB mode!

Problem

We need a precise, succinct and comprehensive security definition!

Subtle Corner Cases

Given n stocks, the message $m := m_1||m_2||m3||\ldots||m_n$ tells your broker to buy i-th stock if $m_i = 1$ or to sell if $m_i = 0$. Suppose m is encrypted and sent to your broker. We would consider the encryption to have failed if an adversary can even just compute *one bit* of the message to learn whether you want to buy or sell stock i.

Subtle Corner Cases

Given n stocks, the message $m := m_1||m_2||m3||\dots||m_n$ tells your broker to buy i-th stock if $m_i = 1$ or to sell if $m_i = 0$. Suppose m is encrypted and sent to your broker. We would consider the encryption to have failed if an adversary can even just compute *one bit* of the message to learn whether you want to buy or sell stock i.

Even partial information leakage about a message is problematic.

Subtle Corner Cases

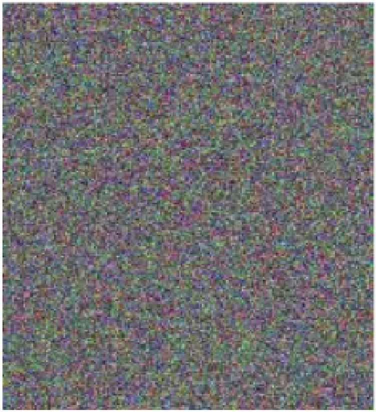
Given n stocks, the message $m := m_1||m_2||m3||\ldots||m_n$ tells your broker to buy i-th stock if $m_i = 1$ or to sell if $m_i = 0$. Suppose m is encrypted and sent to your broker. We would consider the encryption to have failed if an adversary can even just compute *one bit* of the message to learn whether you want to buy or sell stock i.

Even partial information leakage about a message is problematic.

In fact, even *probabilistic* leakage is a problem: an adversary that can tell that with probability of 90% whether you are buying or selling might be a problem!

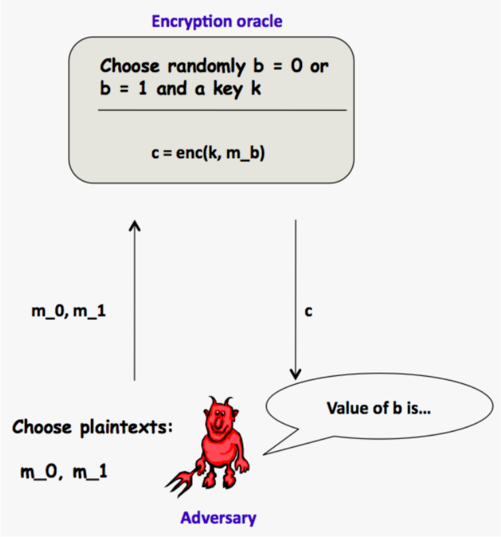
What we want

Our goal is to formalize the intuitive notion of secure encryption shown here:



The picture shows that an adversary does not learn any useful information about a plaintext from a ciphertext.

Indistinguishability under Chosen Plaintext Attacks (IND-CPA)



Indistinguishability under Chosen Plaintext Attacks (IND-CPA)

Security Game: Adversary chooses m_1 and m_2. Defender chooses key k and $b \in \{0, 1\}$. Defender computes $c := \texttt{enc}(k, m_b)$ and gives c to the adversary.

Definition: A symmetric encryption scheme `enc()` is *IND-CPA secure*, if it is impossible for all possible adversaries to tell whether $b = 0$ or $b = 1$. That is, the adversary wins if they can determine the correct b.

Problem

The above definition is incomplete: What if the adversary wins 60% of the time?

Cryptographic Games

An *oracle* is a party in a game that the adversary can call upon to indirectly access information that is otherwise hidden from it. **IND-CPA** can then be formalized like this:

Setup: Generate random key k, select $b \in \{0,1\}$ for $i \in \{1,\ldots,q\}$.

Oracle: Given M_0 and M_1 (of same length), return $C := \texttt{enc}(k, M_b)$.

The adversary wins, if it can guess b with probability greater than $\frac{1}{2} + \epsilon(\kappa)$ where $\epsilon(\kappa)$ is a negligible function in the security parameter κ.

Restrictions on Oracle use

Many schemes break after an large number of messages. Thus, restrictions are generally imposed on the use of the Oracle by the adversary:

- Best known attack on AES uses birthday attack, 2^{64} queries
- $\Rightarrow$ limit oracle use to say 2^{30} queries of some maximum length, say 2^{13} (1 kB).

Then the resulting *advantage* of the adversary remains "small".

IND-CPA

IND-CPA is a widely accepted definition of secure symmetric encryption.

Practically relevant symmetric encryption schemes (i.e. AES in CTR or CBC mode) are considered IND-CPA secure.

Examples for IND-CPA Insecure Schemes

- Schemes where the plaintext can be recovered from the ciphertext ...
- Schemes where the key can be recovered from the ciphertext ...
- ECB mode encryption ...
- Schemes where the n-th plaintext bit can be recovered from ciphertext ...

... are all IND-CPA insecure.

Examples for IND-CPA Insecure Schemes

- Any deterministic, stateless encryption scheme is insecure.
- CBC stateful IV mode (where IV is *predictable* because, for example, sender determines next IV by incrementing previous IV) is IND-CPA insecure