Software Vulnerability Assessment & Secure Storage
Software Vulnerability Assessment

- **Vulnerability assessment** is the process of identifying flaws that reside in an OS, application software or devices’ configuration files.

- **Software Analyzing Techniques:**
  - **Black-Box Analysis:** The inner workings of the target are hidden from the tester
    - Testing a web server
  - **White-Box Analysis:** Source codes, network topology, or other additional resources are available to the auditor

While both black and white box testing can determine if the system is doing what it is supposed to do, only white box testing is effective at determining if the "how" part of the equation is correct.
Software Vulnerability Assessment

- **Gray-Box Analysis**: Carefully selected subset of details are available to the software auditor.
  - Critical components of the system

- **Static and Dynamic Analysis**
  - **Static Analysis** refers to the process of analyzing a system without actually executing the targeted software
Static and Dynamic Analysis

• **Static Analysis Techniques:**
  - **Source-Code Auditing** is the process of carefully examining the source code of a target application in the attempt to uncover security vulnerabilities
    - Take place during the software development cycle
    - Take place in response to security flaws found in the system
    - Finding software vulnerabilities through source scanning tool
  - **Binary Auditing** is the process of examining the binary code of the target application where the source code is unavailable to the auditor
    - The auditor use a disassembler to convert the binary code to a high level machine language called *assembly code*
Static and Dynamic Analysis

- **Dynamic Analysis** is a method that involve actual live execution of the target application to uncover flaws
  - Dynamic analysis is done with the assistance of a debugger
  - Debugger allows an auditor to control the execution of a program and step through the program execution one instruction at a time
  - Virtual machine tool is used to discover vulnerabilities in new released OS
Dynamic Analysis

• **Fuzzing** is the process of automating the process of injecting unexpected input into target application and recording responses from the application. Responses includes:
  - Crashes
  - Error message generated by the target application
  - Fuzzer are developed for specific network program of file format
Administration and Auditing

• Security Techniques Used by System Administrator:
  • **User Policies**: Each user or system component must be assigned the least privilege for smooth operation of a secure system
    • A User access control must be implemented to establish a set of access rules
    • **Strong password** must be implemented to authenticate user into the system
  • **System Policies** software patching is important to prevent compromise due software vulnerabilities
    • Clear policies must be implemented that monitor and apply software update in response to security announcements
Administration and Auditing

- **Network Policies** System administrator should minimize the attack surface of their network by implementing the following methods:
  - *Deploying firewall* that allow the bare minimum of network traffic
  - **Network Segmentation**: Machines that provides services to external user should be placed in a DMZ
    - Machine for interval usage should be placed behind a firewall
    - Machine that do not need access to the internet should be isolated

![Diagram](image.png)

*Figure 9.7: Example of segmentation of a network into regions.*
What Is a Penetration Testing?

• Testing the security of systems and architectures from the point of view of an attacker (hacker, cracker ...)

• A “simulated attack” with a predetermined goal that has to be obtained within a fixed time
Technique – Penetration Testing

1) Gather Information
2) Scan IP addresses
3) Fingerprinting
4) Identify vulnerable services
5) Exploit vulnerability (with care!)
6) Fix problems?
Lost Laptops

• Lost and stolen laptops are a common occurrence
  • Estimated occurrences in US airports every week: 12,000

• Average cost of a lost laptop for a corporation is $50K
  • Costs include data breach, intellectual property loss, forensics, lost productivity, legal and regulatory expenses
  • Data breach much more serious than hardware loss
  • Encryption decreases cost by $20K
  • The existence of a full backup increases cost
Password-Based File Encryption

• **Microsoft Office 97/2003**
  - 40-bit encryption key
  - Guaranteed cracking in two weeks with standard PC

• **Microsoft Office 2007**
  - AES encryption
  - Default 128-bit key size can be increased to 256
  - Secret key derived from password by iteratively hashing salted password 50,000 times with SHA-1

• **Adobe Acrobat 9**
  - AES encryption
  - 256-bit keys
  - Secret key derived from password by hashing salted password once with SHA-256, which is faster than SHA-1 ...

• **Elcomsoft** markets password-recovery tools
  - Crack attempts per second: 5K Office 2007 vs. 75M for Acrobat 9
Encryption of File Systems

- **Disk encryption**
  - Block-level encryption
  - Encryption of physical or logical drive
  - **BitLocker** in Windows Vista and 7
  - **TrueCrypt** open source software

- **File system encryption**
  - File-level encryption
  - **Encrypting File System (EFS)** in Windows
Sharing Encrypted Files

• Solution A
  • Encrypt file with symmetric key K
  • Share K with authorized users
  • Users need to keep many keys
  • User revocation requires redistributing new key

• Solution B
  • Different symmetric keys $K_1, \ldots, K_n$ for authorized users
  • Encrypt file multiple times with $K_1, \ldots, K_n$
  • Inefficient in terms of space and computing time

• Solution C
  • Encrypt file with single symmetric key K
  • Encrypt K with public keys of authorized users $PK_1, \ldots, PK_n$
  • Store with file $E_{PK_1}(K), \ldots, E_{PK_n}(K)$
Encrypting File System (EFS)

- Available in Windows since Windows 2000
- Features
  - Work transparently by providing automatic encryption/decryption of files in specified folders
  - Protects file content but not file name and other metadata
  - Supports sharing of encrypted files
  - Keys unlocked on successful user login
  - Latest version uses RSA, SHA-256, and AES
- Issues
  - Protection only local to file system
    - File copied to another file system is decrypted
    - Email attachment sent decrypted
  - File content may be leaked to unprotected temporary files
  - Key management is cumbersome
EFS Keys

- Users have public-private key pairs
- Each file is encrypted with a different symmetric **File Encryption Key** (FEK)
- FEK is encrypted with public key of file owner and other authorized users
- **Data Decryption Fields** (DDF) stored in file header (metadata)
  - ID of authorized user
  - FEK encrypted with public key of user
- **Data Recovery Fields** (DRFs) provide additional encrypted FEKs, associated with recovery agents

<table>
<thead>
<tr>
<th>ID1</th>
<th>$E_{PK1}(FEK)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID2</td>
<td>$E_{PK2}(FEK)$</td>
</tr>
<tr>
<td>ID3</td>
<td>$E_{PK3}(FEK)$</td>
</tr>
</tbody>
</table>

$E_{FEK}$(file contents)
TrueCrypt Hidden Volume

A standard TrueCrypt volume

- Space Occupied by Files
- Free Space

Header of the Standard Volume

The standard TrueCrypt volume after a hidden volume was created within it

- Data Area of the Hidden Volume
- Header of the Hidden Volume

Padded with random bits
TrueCrypt Hidden Volume

A standard TrueCrypt volume

- Space Occupied by Files
- Free Space
- Header of the Standard Volume

The standard TrueCrypt volume after a hidden volume was created within it

- Data Area of the Hidden Volume
- Header of the Hidden Volume

Padded with random bits

Inside the standard TrueCrypt volume are still random bits
TrueCrypt Hidden Volume

A standard TrueCrypt volume

Space Occupied by Files

Header of the Standard Volume

The standard TrueCrypt volume after a hidden volume was created within it

Password (PA) standard volume

Password (PB) hidden volume

Data Area of the Hidden Volume

Header of the Hidden Volume

Padded with random bits

Inside the standard TrueCrypt volume are still random bits

PA ≠ PB
BitLocker

- Encrypts NTFS volumes
- All disk sectors encrypted with *symmetric encryption* method
- *Key* can be provided by user at boot time
  - Passphrase
  - Hardware token
- Key can be stored in special cryptographic chip that releases it after checking the integrity of the system
  - *Trusted Platform Module* (TPM)
BitLocker Architecture

• Volumes
  • Small unencrypted boot volume
  • Large encrypted volume storing rest of OS and user files

• Keys
  • Volume Master Key (VMK)
    • Unlocked through authentication procedure
  • Full Volume Encryption Key
    • Used to encrypt sectors of encrypted volume
    • Stored on boot volume encrypted with VMK
    • Kept in memory and never written unencrypted to disk
Trusted Platform Module (TPM)

- Crypto processor
  - Mounted on motherboard
  - Tamper-resistant
  - Holds root key $K$ that is never released
  - Has several platform configuration registers (PCRs), with fixed value at power up

- Operation **seal**
  - Encrypts with $K$ supplied plaintext $p$ and associates it with a PCR $i$
  - Returns ciphertext $c = E_K(p)$ and MAC $m = MAC(K, PCR[i])$

- Operation **unseal**
  - Input is a ciphertext $c$, PCR index $i$, and claimed MAC $m$
  - Decrypts ciphertext $c$ and returns $D_K(c)$ if $MAC(K, PCR[i]) = m$

- Operation **extend**
  - Only operation supported on PCRs
  - Input is a data item $x$ and PCR index $i$
  - Computes step of hash chain: $PCR[i] = h(PCR[i], x)$
Booting with a TPM

• Multi-level integrity checking
• Allows BitLocker authentication without user intervention

• Initialization
  • PCR extended with layers of trusted OS code (BIOS, boot loader, kernel, etc.)
  • Volume master key sealed to PCR

• Trusted boot
  • Tamper-proof BIOS associated with TPM
  • Each code layer extends PCR with next layer
  • If integrity is not verified, PCR is extended with random value
  • Execution is transferred to next code layer
  • VMK can be unsealed only if the integrity of all layers has been successfully verified