Application Security Overview

D. Vanderbist 19/05/2013

Types of security

Confidentiality:

- Confidentiality means that sensitive information should not be readable by unauthorized persons. For a part, confidentiality can be achieved by authentication and authorization. In this chapter, we discuss other confidentiality measures you can take to keep private information private.
- SSL + Cryptography, browser settings: disable browser features, directory listing on web-server, cached data + cookies

Integrity

- Integrity means that data should not be able to be changed during data transmission. If data is tampered with, this should be detected.
- SSL + Cryptography

Authentication

- Authentication means we should assure that only legitimate persons are able to access the application. During authentication, the identity of a user is established, and his credentials are checked.
- Authentication + Access Control, caching authentication tokens, account locked out after retries, hard-coded credentials in code, difficult and encrypted session ID's, limit number of sessions, session timeout, no default passwords, secure kept open channels, secure administrative interfaces, apply patches, secure DB connections

Authorization

- Authorization means we should control access to application resources by authenticated users or code.
- Code-access security, role based security, secured DB accounts, secure DB admins to avoid direct data manipulation bypassing the system

Non-repudiation

- Non-repudiation, in its simplest form, means that proof should be provided about the fact that a certain message has been sent by a certain party. Non-repudiation of this kind can be obtained by using cryptography: digital signatures assure that a certain message is sent by the owner of a certain private key, and that the message has not been tampered with.
- SSL + Cryptography

Availability

- Availability means that we should prevent attacks that try to make the application unavailable for legitimate users of the application.
- Input Validation, HTTP request validation, input length validation, unsupported characters, character set validation

Types of security

Availability	Non-repudiation	Authorization	Authentication	Integrity	Confidentiality	
Check for harmful user input	Use appropriate encryption technology Use SSL	Use code-access security	Use a proper authentication scheme Configure client browser settings	Use appropriate encryption technology Use SSL	Use appropriate encryption technology Use SSL Configure client browser settings	**Client tier**
Check for harmful user input Check the entire HTTP request Check input data length Decide on case sensitivity Check unicode value Use a single point-of-entry Use code-access security	Use appropriate encryption technology Use SSL	Use role-based security Use code-access security Securely configure runtime environment	Use a proper authentication scheme Account management Avoid hardcoded credentials No implicit component trust Make session IDs hard to guess Encrypt session IDs Make sessions time out Limit the number of sessions Secure open channels Secure services Secure public administrative interfaces Change default passwords Secure sample code and demos Apply patches, but carefully	Use appropriate encryption technology Use SSL	Use appropriate encryption technology Use SSL Disable directory listing on webserver	**Web tier**
Check for harmful user input Check input data length Decide on case sensitivity Use code-access security No implicit component trust		Use role-based security Use code-access security Securely configure runtime environment				**Business tier**
Check for harmful user input Use secure database users		Use role-based security Use code-access security Securely configure runtime environment Data segregation Use secure database users	Avoid hardcoded credentials Use secure database users			**EIS tier**

Techniques: CAS Code Access Security

Permission class	Description
DirectoryServicesPermission	Ability to access Active Directory through the System.DirectoryServices classes
DnsPermission	Ability to use the TCP/IP Domain Name System (DNS)
EnvironmentPermission	Ability to read and write environment variables
EventLogPermission	Ability to read and write to the event log
FileDialogPermission	Ability to access files that have been selected by the user in the Open dialog box
FileIOPermission	Ability to work with files (reading, writing and appending to files, as well as creating and altering folders)
IsolatedStorageFilePermission	Ability to access private isolated storage. Isolated storage is a special facility provided by .NET for storing data even when no file access is allowed. An assembly is given access to a segregated storage on disk. No access to other data is allowed, and isolated storage is available only to the specific assembly it was created.
MessageQueuePermission	Ability to use message queues through the Microsoft Message Queue
OleDbPermission	Ability to access databases through the ADO.NET OleDb managed provider
PerformanceCounterPermission	Ability to make use of performance counters
PrintingPermission	Ability to print
ReflectionPermission	Ability to discover information about a type at runtime using System.Reflection
RegistryPermission	Ability to read, write, create or delete registry keys and values
SecurityPermission	Ability to execute, assert permissions, call into unmanaged code, skip verification and other security related rights
ServiceControllerPermission	Ability to access Windows services
SocketPermission	Ability to make or accept TCP/IP connections on a transport layer address
SqlClientPermission	Ability to access SQLServer through the ADO.NET SqlClient managed provider
UIPermission	Ability to access the user interface
WebPermission	Ability to make or accept connections to/from the web

Identity Permission class	Identity represented
PublisherIdentityPermission	The software publishers digital signature
SiteIdentityPermission	The location of the website from which the assembly originated
StrongNameIdentityPermission	The strong name of the assembly
URLIdentityPermission	The URL from which the assembly originates (including protocol prefix)
ZondeIdentityPermission	The zone from which the assembly originates

Permission Set	Description
FullTrust	No permission restrictions
Execution	The ability to run, but not to access any protected resources
Nothing	No permissions and unable to execute
LocalIntranet	A subset of the full set of permissions. This is the default permission set for the local intranet.
Internet	This is the default permission set for code of unknown origin.
Everything	All the standard permissions, except the permission to skip code verification.

Techniques: Cryptography

- symmetric cryptography
- asymmetric cryptography
- one-way hashing
- random number generation

Techniques: End-to-End Security

Types

- Network security
    - *Network security* involves the design of the network over which the tiers of the software system communicate. Typical issues in network security are communication protocols, firewalls, routing, and so on.
- Host security
    - *Host security* involves the installation and configuration of the hardware and software used by the software system. Typical points of interest are the operating system and the services it is running.
- Application security
    - *Application security* is concerned with the security built into the software system itself. How does the design of the system tackle security issues?

Hacker Approach

- Information gathering
    - Published leaks
    - Newsgroups
- Target probing
    - OS
    - Open ports
    - Services listening on ports

Attack Types

- DOS
- Modification & Destruction of Data
- Confidential Data

Attack Counter-Measures

- Password sniffing: SSL / IPSec
- Try default passwords: change defaults
- Account enumeration: lock after retries
- Read confidential data: turn of directory listing
- Cross-site scripting: user input validation: input validation
- Parameter poisoning: SQL injection, URL encoding, meta characters, null, buffer overrun
- Session hijacking/Replay attacks: difficult session id's, session timeouts, encrypt session id, limit number of sessions, SSL
- Abuse system configuration: services, open ports, class path variable, config files access, no hardcoded credentials, secure config environment
- Abuse file permissions
- Abuse sample code
- Abuse public interfaces
- Retrieve system information: log files, code comments, emails, compiled code, browser chance, browser history, translate error code
- Brute force: lock account after retries, secure session (see above)
- Abuse implicit component trust: code access security
- Replace password controls: control replaced by hackers control, SSL
- Abuse vendor patches: patch
- Break weak encryption: cryptography
- Abuse debug entry points: remove debug access
- Bypass client-side validation: double check client and server side because hacker can bypass the client and go directly to the server
- Use badly segregated data: user get access to not belonging to him
- Abuse database misconfiguration: secure DB users

Impersonation:
In some situations, it might be necessary to impersonate another user. This means that you temporarily have to associate the Identity and Principal object of another user with the current thread, probably because it has more permissions.