Enterprise Application Programming

Security

Security Topics

What is Security?

What does Security mean to software development?

Common Client/Server attacks

The Security Policy Document

HBGary case study

- http://arstechnica.com/tech-policy/2011/02/anonymous-speaks-the-inside-story-of-the-hbgary-hack/
- HBGary: security company
- CEO was about to "expose" key figures of anonymous
- hbgaryfederal.com was running a CMS susceptible to SQL injection
  - http://www.hbgaryfederal.com/pages.php?pageNav=2&page=27.
  - attackers got the list of usernames, e-mail addresses, and password hashes for the HBGary employees authorized to make changes to the CMS
  - two HBGary Federal employees—CEO Aaron Barr and COO Ted Vera—used passwords that were very simple; each was just six lower case letters and two numbers.
  - It stored only hashed passwords, no salts, which hackers reversed with rainbow tables

HBGary (cont'd)

- same passwords used across e-mail, Twitter accounts, and LinkedIn
- attackers logged into support.hbgary.com with COO's password
- used privilege escalation flaw to become root
  - deleted HBGary's backups
- CEO's password used to access email system, for which he was administrator
- Got access to another key employee's email (Greg Hoglund)
- In that email was One: the root password to the machine running Greg's rootkit.com site was either "88j4bb3rw0cky88" or "88Scr3am3r88". Two: Jussi Jaakonaho, "Chief Security Specialist" at Nokia, had root access. Vandalizing the website stored on the machine was now within reach.

HBGary (cont'd)

- Couldn't login to rootkit.com as root (did one thing right?)
- Emailed administrator using Gregs email: fake Greg appeared to know the root password and, well, the e-mails were coming from Greg's own e-mail address
- administrator gave attackers the real root password for rootkit.com, a special port for ssh, reset the password on real Greg's account, and told attackers what real Greg's account name was
- attackers now had remote root access on rootkit.com

apache.org case study

- http://blog.sucuri.net/2010/03/apache-org-defaced-security-archive-case-study.html
- http://www.dataloss.nl/papers/how.defaced.apache.org.txt

What is Security?

- Security can be defined as the ***protection of personnel, physical resources, data, communication assets and associated infrastructure from loss, damage, modification or observation***.

- Why do we care so much about security? People post all their information on Facebook anyway, who cares if the NSA is listening?

Remember WarGames?

- WarGames is a movie in which a teenager hacks into a military computer and almost starts WWIII
- WarGames is just science fiction, or Hollywood drama, right?
  - By the way… Peter Schwartz from SRI (Stanford Research Institute) was a consultant in the development of the movie WarGames

By the way…

- In 1983, the same year WarGames came out, Stanislav Petrov was the duty officer at the command center for the Oko nuclear early warning system when the system reported a small launch from the United States. Petrov judged that the report was a **false alarm**, later indicating the influences in this decision included the following:
  - that he was informed a U.S. strike would be all-out, so five missiles seemed an illogical start,
  - that the launch detection system was new and, in his view, not yet wholly trustworthy, and
  - that ground radars failed to pick up corroborative evidence, even after minutes of delay.

- But that's just a story about a launch detection system. In real life, how much of a threat are teenagers?

By the way…

- http://www.cybercrime.gov/usamay2001_7.htm

- A juvenile in Massachusetts pleads guilty to charges he disabled a key telephone company computer servicing the Worcester airport control tower, thereby disabling both the main radio transmitter, as well as a circuit which enabled aircraft on approach to send signals activating the runway lights.

By the way…

- A 16-year-old from Florida pleads guilty and is sentenced to six months in a detention facility for intercepting electronic communications on military computer networks and for illegally obtaining information from a NASA computer network.

By the way…

- A 16-year-old in Virginia pleads guilty to computer trespassing after hacking into a Massachusetts Internet service provider's (ISPs) computer system, causing $20,000 in damages.

Military Security

- After all these years, with large security budgets and great advances in computer technology, and with obvious motivation to solve the security problem well, how are “the military” doing?

- From CBC news (2011): “An unprecedented cyberattack on the Canadian government also targeted Defence Research and Development Canada, making it the third key department compromised by hackers, CBC News has learned.The attack, apparently from China, also gave foreign hackers access to highly classified federal information and also forced the Finance Department and Treasury Board — the federal government's two main economic nerve centres — off the internet.”
  - http://www.cbc.ca/news/politics/story/2011/02/16/pol-weston-hacking.html

Military Security (cont'd)

- http://www.ctv.ca/CTVNews/TopStories/20100219/forbes_defense_100221/

- The Pentagon's forensics-focused Cyber Crime Center, where [Steven] Shirley is executive director, found that between August 2007 and August 2009, 71 government agencies, contractors, universities and think tanks with connections to the U.S. military had been penetrated by foreign hackers, in some cases multiple times. In total, Shirley told Forbes, the center performed 116 investigations following spying breaches and found that in all but 14 of those cases the intruders had gained complete administrator-level access to the victim's network. Read more: http://www.ctv.ca/CTVNews/TopStories/20100219/forbes_defense_100221/#ixzz1I2SU9IMA

Who's been compromised?

- Pentagon deems major cyber attacks as acts of war (May 2011)
- Many systems have been compromised:
  - U.S.:
    - Defence Department
    - U.S. Dept of Commerce
  - Canadian:
    - Defence Research and Development Canada
    - Finance Dept
    - Treasury Board
  - Sony
  - ValuJet
  - NASA
  - CIA
  - Malaysian Government
  - Six Flags
  - NY Times
  - FOX
  - *Etc… Adobe, Google?, Microsoft?, Apple?...*

The Security Problem

- Optimistic view: From a Cisco textbook:
  - “New research from Gartner indicates that most information technology security breaches take advantage of known, patchable flaws that exist because of **poor enterprise security practices** and **lack of investment** in system protection.”
  - “Gartner places the blame primarily on **poor security practices** and IT departments that are overworked and lacking in trained security professionals”
- On the other hand…
  - “I don’t believe **any** system is totally secure” -- David Lightman (WarGames movie)

Why is security so difficult

- layers: How about Transport Layer Security? (SSL)
  - Debian Security AdvisoryDSA-1571-1
  - openssl -- predictable random number generator
  - Date Reported:13 May 2008 (Since September 2006)
  - Affected Packages:openssl
  - Vulnerable:Yes
  - Security database references:In Mitre's CVE dictionary: CVE-2008-0166.
  - More information:Luciano Bello discovered that the random number generator in Debian's openssl package is predictable.
  - This is caused by an incorrect Debian-specific change to the openssl package (CVE-2008-0166). As a result, cryptographic key material may be guessable.This is a Debian-specific vulnerability which does not affect other operating systems which are not based on Debian. However, other systems can be indirectly affected if weak keys are imported into them.It is strongly recommended that all cryptographic key material which has been generated by OpenSSL versions starting with 0.9.8c-1 on Debian systems is recreated from scratch. Furthermore, all DSA keys ever used on affected Debian systems for signing or authentication purposes should be considered compromised; the Digital Signature Algorithm relies on a secret random value used during signature generation.
- See also: http://www.schneier.com/blog/archives/2008/05/random_number_b.html

TCP/IP attacks

- TCP/IP, the protocol on which the WWW depends, is vulnerable to attack
- IP Address Spoofing
- ICMP abuse
  - carrying secret information in an ICMP packet payload
- IP Fragment Attacks
  - attackers can artificially fragment packets to mislead routers/firewalls
  - ping of death: create fragments that when assembled at the other end, are larger than the maximum permissible length
- TCP Flags
  - illegal combinations can be used in a DOS attack
- SYN Flood
  - can try to have many open connections and disable a platform
- Closing a connection by FIN
  - spoofed FIN packet to kill a connection
- Connection Hijacking
  - take over a connection with spoofed Seq numbers

Trust

- “**Trust** is the likelihood that people will act how you expect them to act” (De Laet and Schauwers)
  - Certificates and Public Key Infrastructure (PKI) allow us to trust strangers
  - most trusted: the internal network that only a few well-known people have access to
  - less trusted: internal users and authenticated remote users
  - least trusted: the Internet (Internet servers and remote unauthenticated users)
- Who can we trust?
- What can we trust?
- What parts of a computer system can we trust?
  - http://www.securityweek.com/los-alamos-pulls-networking-gear-because-it-was-created-china

Trust Frameworks

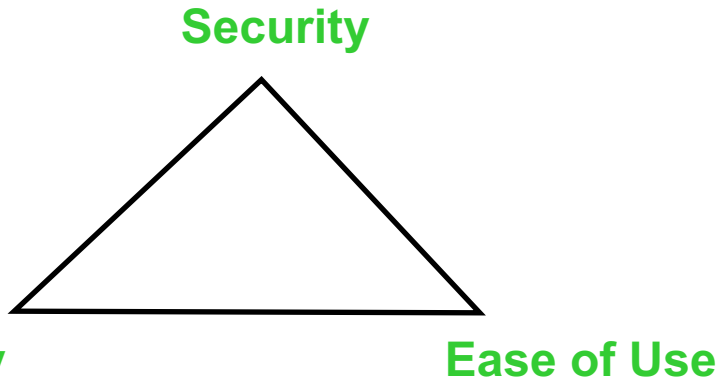
- PKI
  - Certificate Authorities issue certificates (basically signed public keys) for organizations
  - When you browse (https) to a site of one of these organizations, your browser uses its copy of the public key of the CA to verify the signature in the certificate
  - If the certificate is valid, the organization was "vetted" by the CA
  - You can trust the organization if you trust the CA (and your browser)
- Web of Trust
  - Everyone can generate a Private/Public key pair
  - When you validate that a Public key really belongs to the actual person (not a bad guy), you can sign that Public key
  - When someone has signed your Public key, they can opt to trust Public keys that you have signed
  - This forms a Web of Trust
  - You should not sign public keys without verifying them! (Key signing party)

Solution?

- By now, hopefully you're convinced that the Web Security problem is difficult to solve in general, and the current situation in the real world (not just movies) can be considered quite bad.
- What can be done?
- What are these "security practices" that Gartner refers to?
- We can't have perfect security, but we can implement best practices
- In a nutshell: Security Policy and The Security Policy Document
  - The goal is to formalize and document
    1. The Security Policy that will result in the desired level of Safety at a tolerable level of inconvenience
    2. Security Policy Compliance: enforcement, monitoring, logging, auditing
    3. Security Policy Evolution: Security Policy is a moving target
  - Analogous to writing a good computer program, writing and implementing a good Security Policy Document takes knowledge, experience, and skill

Security Goals

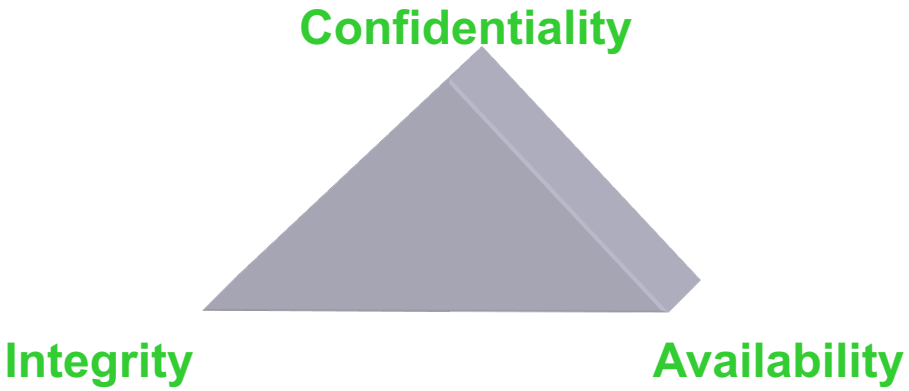
- In this day and age of "*customer satisfaction*", Security is considered to be a balancing act between:
  - **Security Concerns**
  - **Functionality**
  - **Ease of Use**



- One of the main reasons organizations have security issues is that, as you increase security, you decrease functionality
  - Functionality is what keeps companies in business
  - However, these days, lack of security will help to ensure a company doesn't STAY in business

The CIA Triad

- CIA helps to define what you are trying to protect using 3 elements
  - **Confidentiality**
  - **Integrity**
  - **Availability**



- All 3 elements are important, but there is usually one that's more important than the rest for a given situation.
  - *The balancing act involves moving the target within the triangle*
  - ***CIA² adds Accountability to the equation***

The Hacker's Triad

- Hackers have created their own version of the triad called the DAD
  - **Disclosure**
    - Attempts to defeat confidentiality
  - **Alteration**
    - Attempts to defeat integrity
  - **Destruction**
    - Attempts to defeat availability
- The security conundrum:
  - ***If I don't protect it, they can get to it.***
  - ***But if I protect it, it might identify something worth getting to !***

Security Services

- To support the 3 elements of the CIA Triad, security professionals require mechanisms to perform:
  - Identification
    - Mechanism to provide identity (*without validation*)
  - Authentication
    - Proving you are who you say you are
    - Based on something unique you know, have or are
  - Authorization
    - What you're allowed to do once you've been authenticated
  - Accountability (***CIA²***)
    - Keeping track of what user(s) do across systems

What is Software Security?

- A balance between Safety and Usability?
- Privacy and Access?
- In practice, practitioners (ie programmers) must manage vulnerabilities
- There are always vulnerabilities in Software
  - During its development
  - During its deployment
  - During its operation
  - During its maintenance

Development Vulnerabilities

- The original coder intentionally introduces a vulnerability
  - **Back Door**: a mechanism where access can be obtained through bypassing normal security mechanisms

- The original coder unintentionally introduces a vulnerability
  - Buffer Overflow possibilities
  - Poorly implemented security mechanisms
  - General Bugs leading to indeterminate (and insecure) behaviors

Deployment Vulnerabilities

- Distribution
  - Distribution Package tampering
  - Distribution Channel tampering during transmission
  - Intentional (disgruntled employee?) or Accidental
- Installation
  - Target platform not locked down (host server vulnerable to attack)
  - Configuration not done properly (insecure configuration choices)
  - Intentional or Accidental

Operation Vulnerabilities

- Bugs discovered and slow to be fixed (cf Windows,Linux,OSX)
- Bugs fixed but slow to be deployed
- The older the version, the more vulnerable that version gets
- Changes in load or stress cause untrustworthy behaviors
- Operation environment different in a key way from Validation/Verification environment
- Any networked system is vulnerable
  - Public or private network?
  - Internet connectivity?
- Users, Administrators, Operators may be malicious
- Backups not done (valuable data go missing, no disaster recovery)
- Backups not carefully managed (tapes, disks “go missing”?)
- Backup system itself operates at a very high privilege level (target)

Operation Vulnerabilities (cont'd)

- Centralized logging and monitoring
  - Can be a vehicle for back doors
- User account administration
  - If sloppy, introduces back doors
- Administrative remote access
  - Can be a vehicle for back doors

Maintenance Vulnerabilities

- Software's maintainer may be malicious
  - Embed code or remove/circumvent fixes for security flaws
- Configuration changes can undo security
  - Intentional changes
  - Unintentional changes
- Patches or Updates are delayed
  - Many patches and updates directly address vulnerability exploits
  - Old version vulnerabilities get worse over time (more and more known)
- A large part of security is to keep up-to-date with security patches

The problem is recursive! (it applies to the fixes too)

- Correcting vulnerabilities early in the software development lifecycle is cheaper (cost-effective) than developing and releasing frequent patches
- The patches themselves are candidates for the same types of vulnerabilities (Development, Deployment, Operation, Maintenance)
- security is a necessary property from the beginning of the system's life cycle (i.e., needs and requirements definition) to its end (retirement).

Key Aspects of Development and Deployment

- **development principles and practices:** The practices used to develop the software and the principles that governed its development are expressly intended to encourage and support the consideration and evaluation of security in every phase of the software's development life cycle.
- **development tools:** The programming language(s), libraries, and development tools used to design and implement the software are evaluated and selected for their ability to avoid security vulnerabilities and to support secure development practices and principles.
- **testing practices and tools:** The software is expressly tested to verify its security, using tools that assist in such testing.

Key Aspects of Development and Deployment (cont'd)

- **acquired components:** Commercial off-the-shelf (COTS) and Operations System Support (OSS) components are evaluated to determine whether they contain vulnerabilities, and if so whether the vulnerabilities can be remediated through integration to minimize the risk they pose to the software system.
  - Cannot just trust it because you bought it
- **deployment configuration:** The installation configuration of the software minimizes the exposure of any residual vulnerabilities it contains.
  - Turning off javascript, turning off cookies – balance between safety and usability
- **execution environment:** Protections are provided by the execution environment that can be leveraged to protect the higher level software that operates in that environment.
  - Operating system protections, user management, etc.

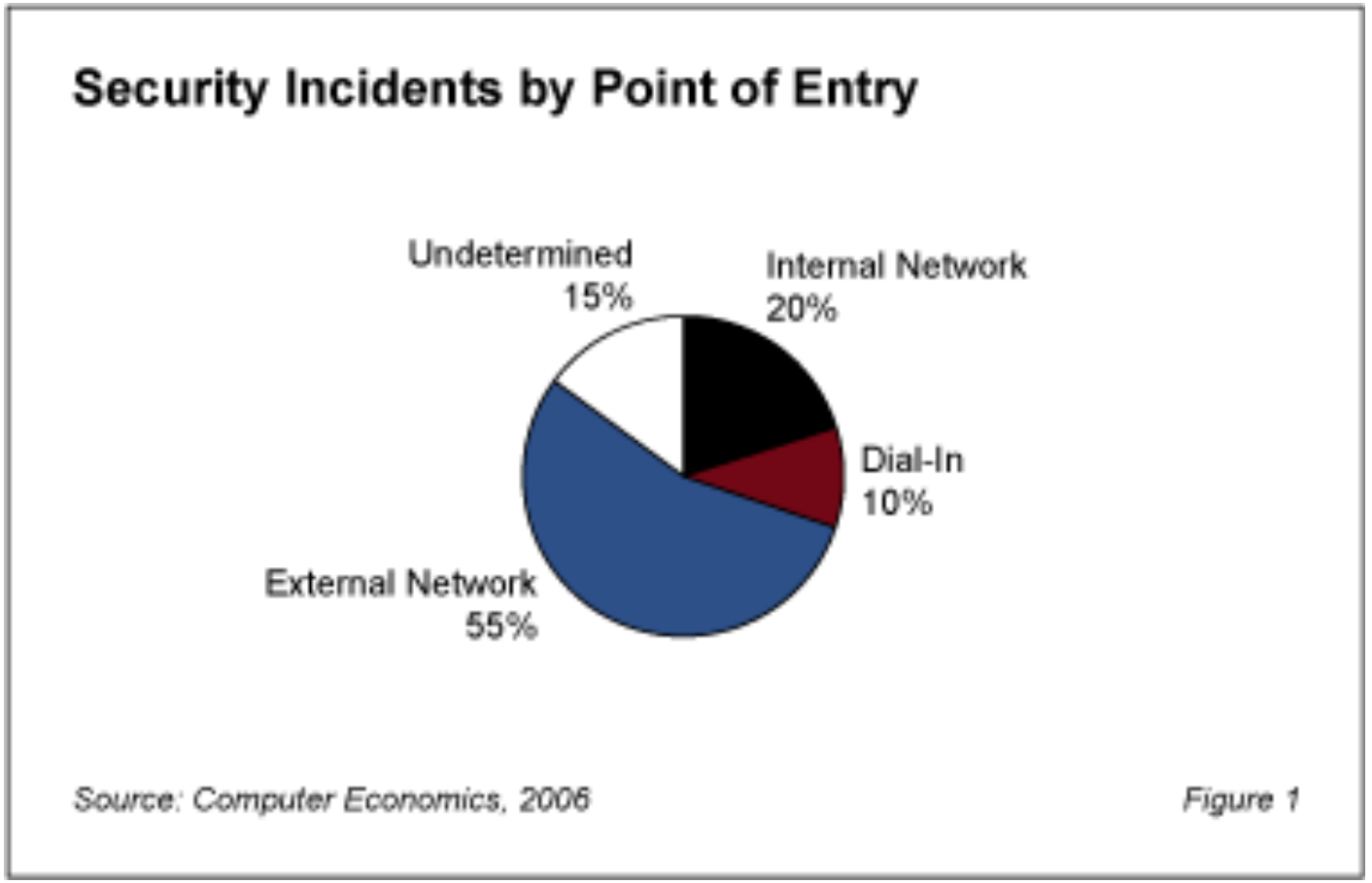
Key Aspects of Development and Deployment (cont'd)

- **practitioner knowledge:** The software's analysts, designers, developers, testers, and maintainers are provided with the necessary information (e.g., through training and education) to give them sufficient security awareness and knowledge to understand, appreciate, and effectively adopt the principles and practices that will enable them to produce secure software.
  - Constant vigilance
  - Don't say, hmm, "that's strange, … probably nothing"
  - Resolve suspicious situations

Secure Software Development

- Minimize the high-consequence components
  - The Fewer Critical and Trusted components the better
- Hide the high-consequence components
  - Private and isolated Critical and Trusted components
- No anonymous accesses
  - Detailed logging of accesses and activities (beware remote logging)
- Always assume the worst
  - Unlikely is not the same as impossible
  - What are the chances someone will… (over time, the chances increase)
  - Do not rely on a firewall to provide protection (layers are good)
- Do not assume "Security Software" is secure (it's software too)
  - firewalls
  - Virus scanners
  - Intrusion detection software

Security Incidents Origins



The Client

- Clients are under the control of the user
- A malicious user can tamper with all data that's stored on the Client
- Any sensitive information or implementation details that pass through the client are discoverable
- Validation codes can be bypassed
- All network traffic from the client must be treated as untrustworthy
  - The user can tamper with data after it leaves, for example, a browser
  - The user can create malicious client traffic to be sent to the server

The Network

- Network traffic can be intercepted and tampered with when it is transmitted in plain text
- Even HTTPS data may have been tampered with at the client end before it's encrypted
- Denial of Service Attacks take advantage of the network's ubiquity
  - Distributed Denial of Service Attacks don't come from any particular place – they come from "everywhere"

HTTPS: TLS/SSL

- HTTP with Transport Layer Security
- Creates a secure channel over an insecure network
- Provides protection from eavesdroppers and Man-In-The-Middle if…
  - The server certificate is validated and **trusted**
  - The trust is provided by major certificate authorities
  - We must trust the certificate authorities, because we ask them in effect to **tell us** who to trust
- How does it work?
  - Browser visits an https URL
  - Server provides a Certificate to the browser
  - Browser validates the Certificate based on its trusted authorities
    - Versign, Digicert, Globalsign, and many others
  - Browser ensures URL/site “matches” certificate
  - Browser and Server use the TLS/SSL encryption layer (which is trusted)

Common Attacks

- Cross-Site Scripting (XSS)
- Scripts are entered into URLs or form fields of a vulnerable site
- One user enters a script that is executed by another user
- Message systems, book reviews, guestbook entries, blog comments
- Maliciously Emailed URLs with tampered parameters

Common Attacks

- SQL injection
- User supplied fields end up in database queries on the server
- Attacker arranges for unintended changes to the logic
  - Ex WHERE id = <userid> and password = <userpass>
    - If userid is entered as “ceo’ --” this can become
    - WHERE id = ‘ceo’ --’ and password = <whocares>
    - Notice that everything after ‘ceo’ is commented out by comment --

Common Attacks

- Buffer Overflows
- Happen when size of input is not checked
- When size is too large, it overflows into other variables/fields
- Example field: the return address of the function executing
- Very famous attack, and most good software accounts for it

Web Attacks

- NULL string
- Input validation code used to detect other attacks can be fooled by the null string character ‘\0’ or %00
- May find <script> but fail to find <%00script>

Common Attacks

- Denial of Service
- Flood a server with requests, overwhelming it
- Browser should not make more than 2 connections to a webserver
- A custom program (modified web browser) can violate that
- Distributed Denial of Service Attacks come from many places at once (botnets)
- Sony, VISA, etc, attacked recently

Security Policy Document

- One of the benefits that a service company like IBM can provide to its enterprise customers: a Security Policy Document
- Security Policy is then formalized, and can be enforced (partially, to some extent) with server monitoring tools
- Security Policy Document has three aspects:
  - Write down the Security Policy precisely
  - Write down how to verify the policy is being followed
  - Write down the process for keeping policy up to date

Securing JEE Web Apps

- RESOURCE: https://docs.oracle.com/javaee/7/tutorial/security-intro.htm

Java Security Mechanisms

- **Java Authentication and Authorization Service (JAAS)** is a set of APIs that enable services to authenticate and enforce access controls upon users. JAAS provides a pluggable and extensible framework for programmatic user authentication and authorization. JAAS is a core Java SE API and is an underlying technology for Java EE security mechanisms.
- **Java Generic Security Services (Java GSS-API)** is a token-based API used to securely exchange messages between communicating applications. The GSS-API offers application programmers uniform access to security services atop a variety of underlying security mechanisms, including Kerberos.

Java Security Mechanisms (cont'd)

- **Java Cryptography Extension (JCE)** provides a framework and implementations for encryption, key generation and key agreement, and Message Authentication Code (MAC) algorithms. Support for encryption includes symmetric, asymmetric, block, and stream ciphers. Block ciphers operate on groups of bytes; stream ciphers operate on one byte at a time. The software also supports secure streams and sealed objects.

Java Security Mechanisms (cont'd)

- **Java Secure Sockets Extension (JSSE)** provides a framework and an implementation for a Java version of the Secure Sockets Layer (SSL) and Transport Layer Security (TLS) protocols and includes functionality for data encryption, server authentication, message integrity, and optional client authentication to enable secure Internet communications.
- **Simple Authentication and Security Layer (SASL)** is an Internet standard (RFC 2222) that specifies a protocol for authentication and optional establishment of a security layer between client and server applications. SASL defines how authentication data is to be exchanged but does not itself specify the contents of that data. SASL is a framework into which specific authentication mechanisms that specify the contents and semantics of the authentication data can fit.

JEE Security

- Application Layer Security
  - Component containers
  - What about data traveling across network?
- Transport Layer Security
  - TLS (was SSL)
  - https:// urls
  - Involves on trusted Certificate Authorities
  - Not sufficient on its own – need Application Layer Security also
- Message-Layer Security
  - SOAP related

Login Security

- Login security is based on Realms, Users, Groups, and Roles
- See https://docs.oracle.com/javaee/7/tutorial/security-intro005.htm

Realms

- `file` Realm: user credentials are stored locally in a `keyfile`
  - `file` realm is what we added our test users to
- `certificate` Realm: server stores user credentials in a certificate database
  - X.509 certificates
- `admin-realm` Realm
  - Also a file realm
- `jdbc` Realm: usernames, passwords, groups stored in DB
  - Good for us because we can add users programmatically

Groups

- Can be used to group users that have traits in common
- Cannot be used in Certificate Realm
- A group is designated for the whole Glassfish Server

Roles

- A Role entails permission to access a set of resources
- Users and groups are mapped to roles (giving those users permissions)
  - Done in `glassfish-web.xml`
- The default Role mapping is Groupname = Rolename
  - Today we'll use the default mapping

Steps to secure logins to our apps

- Define and add a JDBC Realm
  - Specifies datasource
  - Which table contains the username and password columns
  - Which table contains the group column
  - We'll do this in pom.xml
- Create a java:global/DataSourceName datasource
  - We do this in web.xml
- Specify which paths of the webapp are protected, and by which realm
  - We do this in web.xml

Define and add a JDBC Realm mysql

```
<plugin>
        <groupId>org.codehaus.mojo</groupId>
        <artifactId>exec-maven-plugin</artifactId>
        <version>1.2.1</version>
        <executions>
          <execution>
            <id>create-mariadb-realm</id>
            <phase>install</phase>
            <goals>
              <goal>exec</goal>
            </goals>
          </execution>
        </executions>
        <configuration>
          <skip>false</skip>
          <executable>C:/glassfish5/bin/asadmin.bat</executable>
          <arguments>
            <argument>create-auth-realm</argument>
            <argument>--classname</argument>
            <argument>com.sun.enterprise.security.auth.realm.jdbc.JDBCRealm</argument>
            <argument>--property</argument>
            <argument>jaas-context=jdbcRealm:datasource-jndi='java:global/MariaDataSource':user-table=USER_INFO:user-name-column=NAME:password-
column=PASSWORD:group-table=USER_INFO:group-name-column=GRPNAME:digest-algorithm=none</argument>
            <argument>mariaRealm</argument>
          </arguments>
          <successCodes>
            <successCode>0</successCode>
            <successCode>1</successCode>
          </successCodes>
        </configuration>
      </plugin>
```

Define and Add a JDBC Realm derby

```
<plugin>
        <groupId>org.codehaus.mojo</groupId>
        <artifactId>exec-maven-plugin</artifactId>
        <version>1.2.1</version>
        <executions>
          <execution>
            <id>create-derby-realm</id>
            <phase>install</phase>
            <goals>
              <goal>exec</goal>
            </goals>
          </execution>
        </executions>
        <configuration>
          <skip>false</skip>
          <executable>C:/glassfish5/bin/asadmin.bat</executable>
          <arguments>
            <argument>create-auth-realm</argument>
            <argument>--classname</argument>
            <argument>com.sun.enterprise.security.auth.realm.jdbc.JDBCRealm</argument>
            <argument>--property</argument>
            <argument>jaas-context=jdbcRealm:datasource-jndi='java:global/TutoringDatabase':user-table=app.USER_INFO:user-name-column=name:password-
column=password:group-table=app.USER_INFO:group-name-column=INGROUP:digest-algorithm=none</argument>
            <argument>moviedbRealm</argument>
          </arguments>
          <successCodes>
            <successCode>0</successCode>
            <successCode>1</successCode>
          </successCodes>
        </configuration>
      </plugin>
```

Define Datasource Derby (web.xml)

```
<data-source>
    <description>Todd's Tutoring data source.</description>
    <name>java:global/TgkDataSource</name>
    <class-name>org.apache.derby.jdbc.ClientDataSource</class-name>
    <server-name>localhost</server-name>
    <port-number>1527</port-number>
    <database-name>sun-appserv-samples</database-name>
    <user>app</user>
    <password>app</password>
    <property>
      <name>connectionAttributes</name>
      <value>;create=true</value>
    </property>
  </data-source>
```

Define DataSource MySQL (web.xml)

```
<data-source>
    <description>Todd's MariaDB data source.</description>
    <name>java:global/MariaDataSource</name>
    <class-name>com.mysql.jdbc.jdbc2.optional.MysqlDataSource</class-name>
    <server-name>192.168.124.134</server-name>
    <port-number>3306</port-number>
    <database-name>myusers</database-name>
    <user>tgk</user>
    <password>tgkpass</password>
    <property>
       <name>connectionAttributes</name>
       <value>;create=true</value>
    </property>
</data-source>
```

Add Constraint to paths

- Edit web.xml and select security tab; or put the following in web.xml

```
<security-constraint>
    <display-name>AdminConstraint</display-name>
    <web-resource-collection>
      <web-resource-name>moviedbadmin</web-resource-name>
      <description>Administration area</description>
      <url-pattern>/test/*</url-pattern>
    </web-resource-collection>
    <auth-constraint>
      <description>Basic Constraint</description>
      <role-name>Administrator</role-name>
    </auth-constraint>
  </security-constraint>
  <login-config>
    <auth-method>BASIC</auth-method>
    <realm-name>toddRealm</realm-name>
  </login-config>
```