ESAPI: A GUIDED TOUR

Joe Combs
OWASP Cincinnati
SEI - Cincinnati, LLC
jcombs@sysev.com

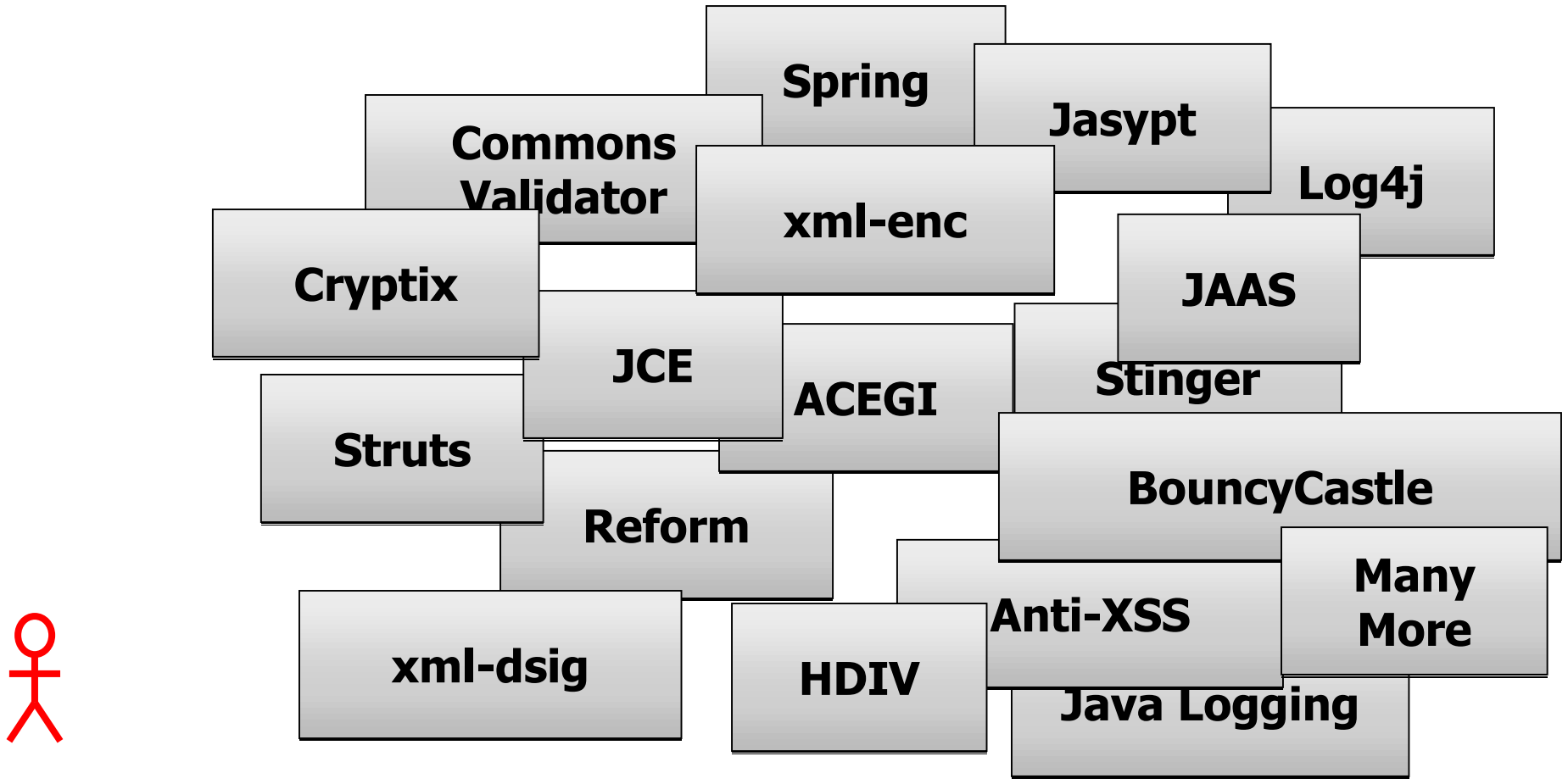
OWASP
26 August 2008

The OWASP Foundation
http://www.owasp.org

AGENDA

- What Problem Are We Trying To Solve?
- What Is ESAPI?
- Architecture
- Example Usage
- Future Direction

The Problem Defined



The Problem Defined

- Getting security controls right is hard stuff
  - texts, articles and tutorials are overflowing with examples of bad security
  - developers copy from other existing code
- Nearly 1/3 of all security code reviewed contains security flaws
- Most developers shouldn't build security controls

The Problem Defined

- In many enterprise environments, security controls evolve over time in a reactive fashion.
- Most enterprises need the same set of calls in most (or all!) of their applications
- A common approach to security controls can aid static analysis

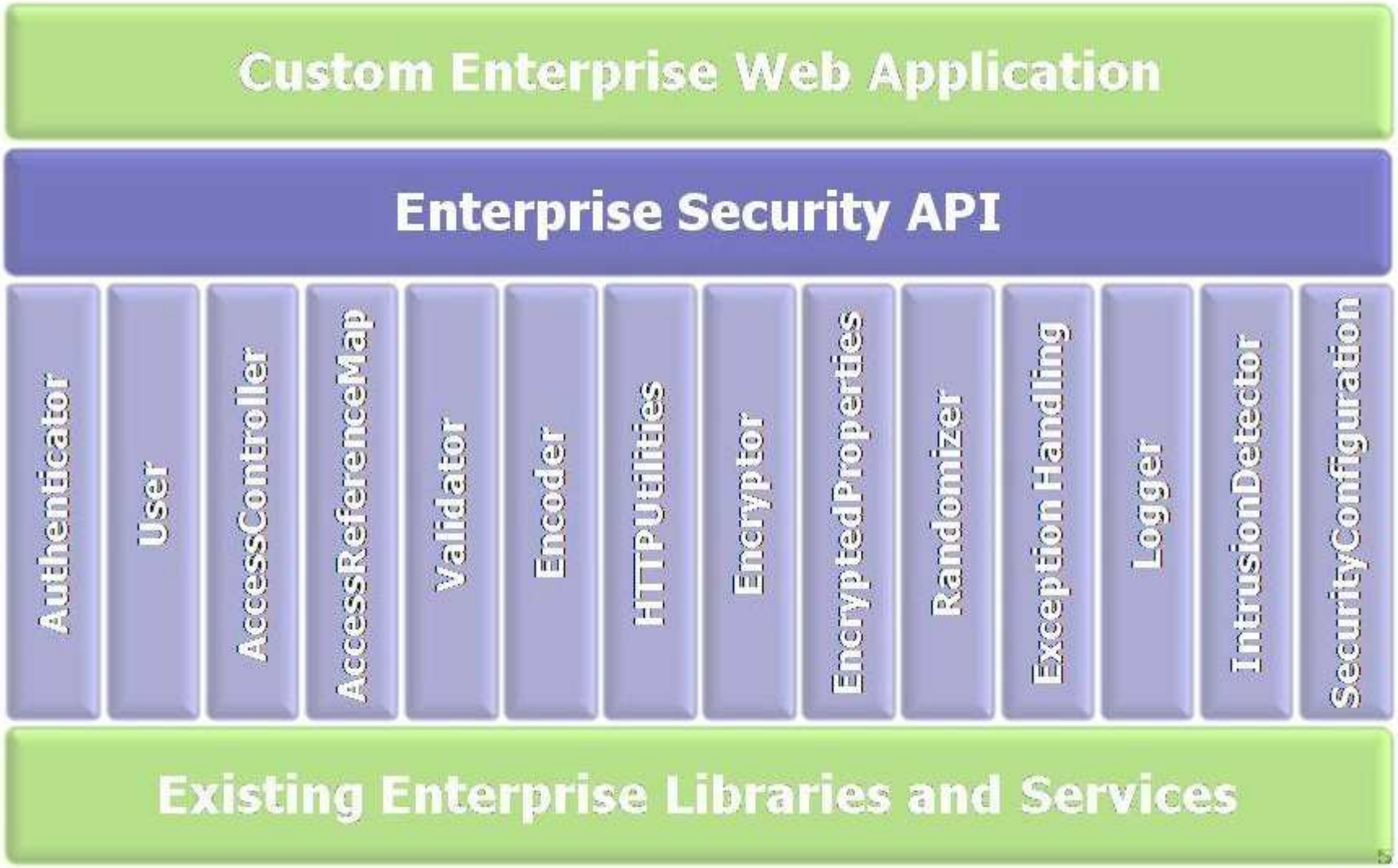
What is ESAPI?

- Enterprise Security API
- Open Source - BSD license
- Create a standardized mechanism for Java EE applications to address security concerns
- ESAPI is NOT a framework. It's a set of well defined interfaces and a reference implementation of the "right" way to do security controls
- Not a silver bullet

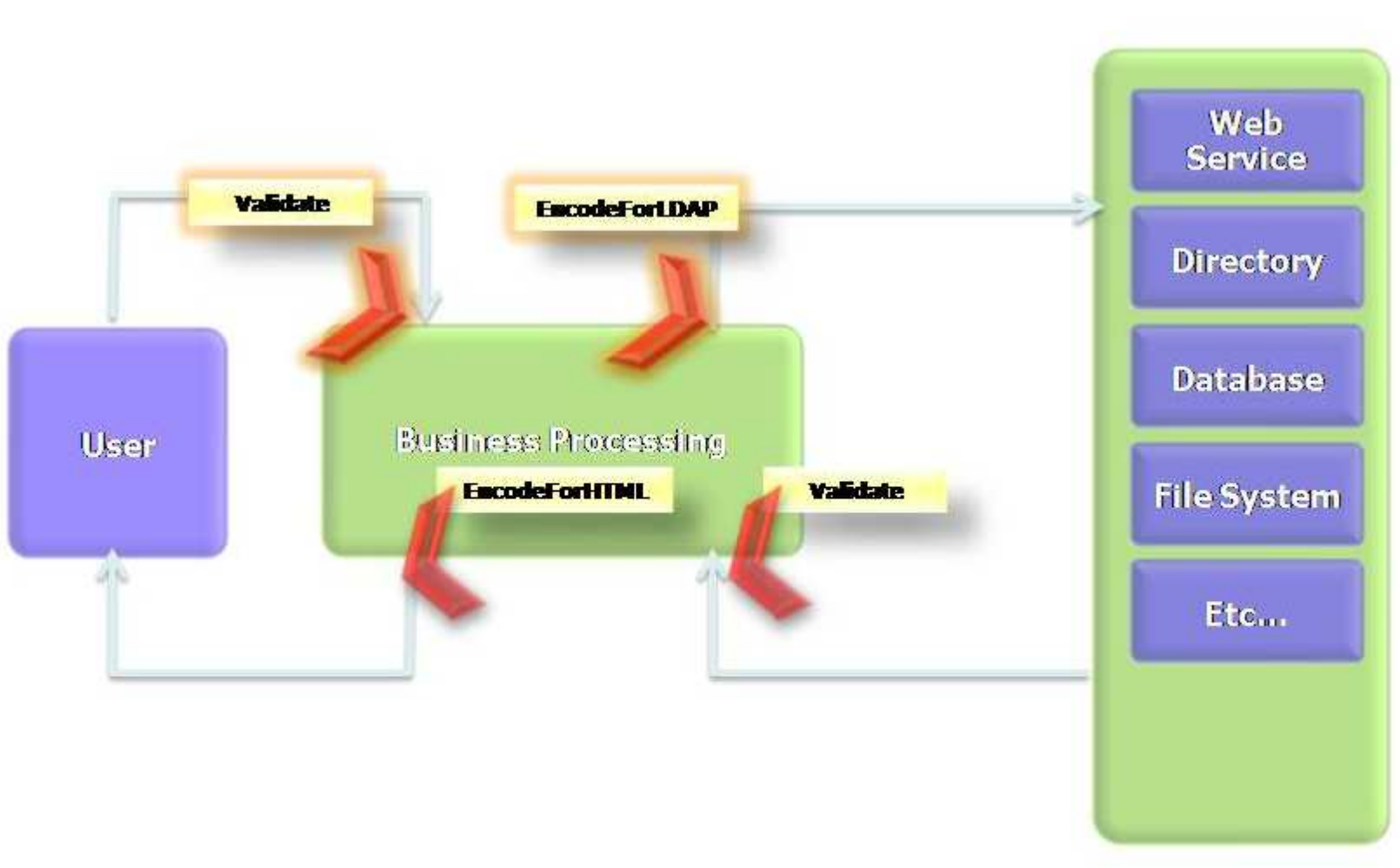
Getting Started

- Pull the latest code from SVN – http://code.google.com/p/owasp-esapi-java/
  - the jar offered as a courtesy download is HORRIBLY out of date
- Set up a resources directory and put a copy of ESAPI.properties inside it
  - Change the master password
  - Make this location "safe"
- Set up user accounts
  - java -Dorg.owasp.esapi.resources="c:\resources" -classpath owasp-esapi-java-1.1.1.jar org.owasp.esapi.Authenticator yourname yourpass admin
- Build, deploy and run the test app, also found in SVN

Architecture



Input Validation & Encoding



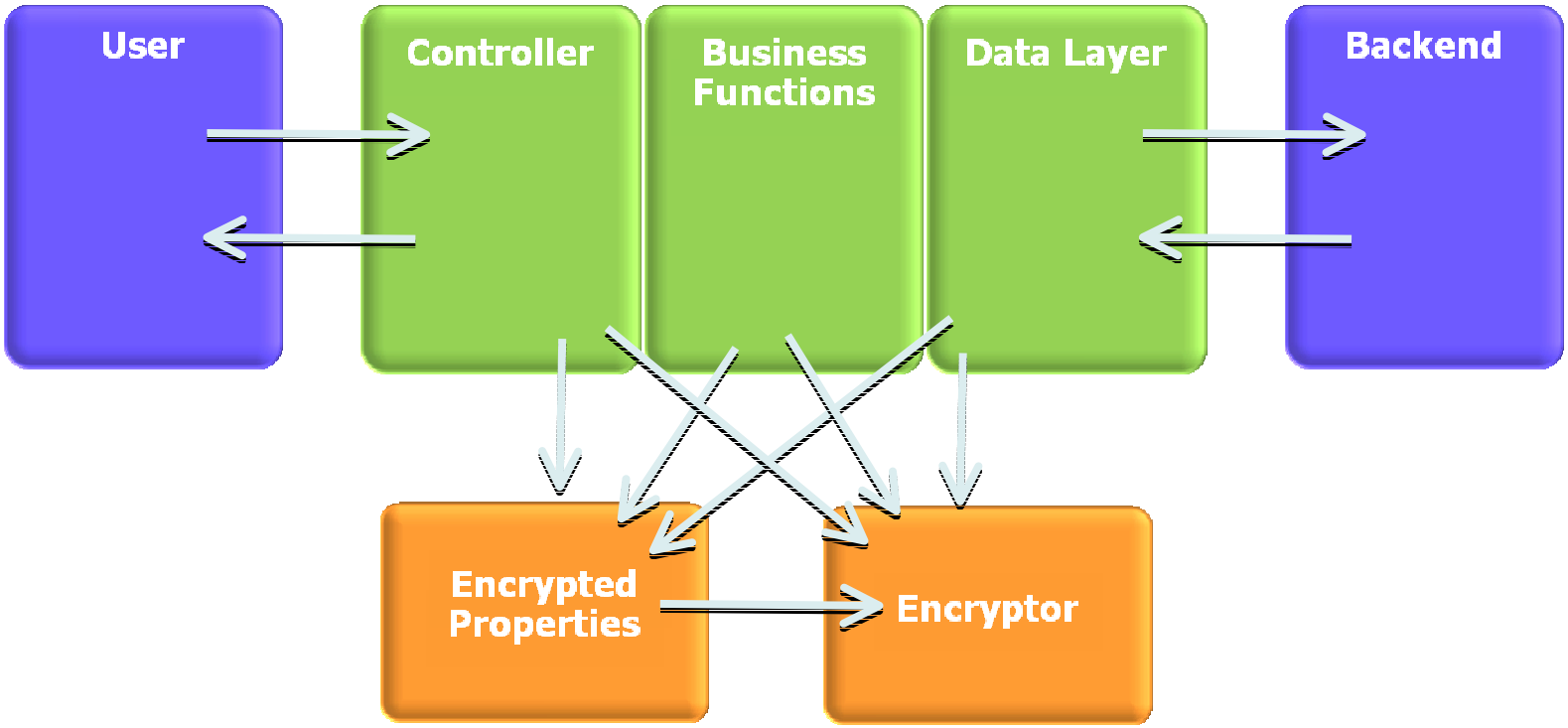
Input Validation & Encoding

- Canonicalizing - reducing a possibly encoded string down to its simplest form.
  - Double encoding is not something a user does so generally regarded as an attack
- Encoding - various methods for different destinations. Whitelist acceptable characters and encode those that don't pass muster
- Validating - after canonicalizing, ensures data is of the correct type, in acceptable ranges, etc.

Input Validation & Encoding

- Supports either boolean returns or throwing exceptions "to allow maximum flexibility because not all validation errors are security problems"
- safeReadLine() to prevent DoS attack
- File name and directory path validation
- Basic credit card validation
- AntiSamy protection

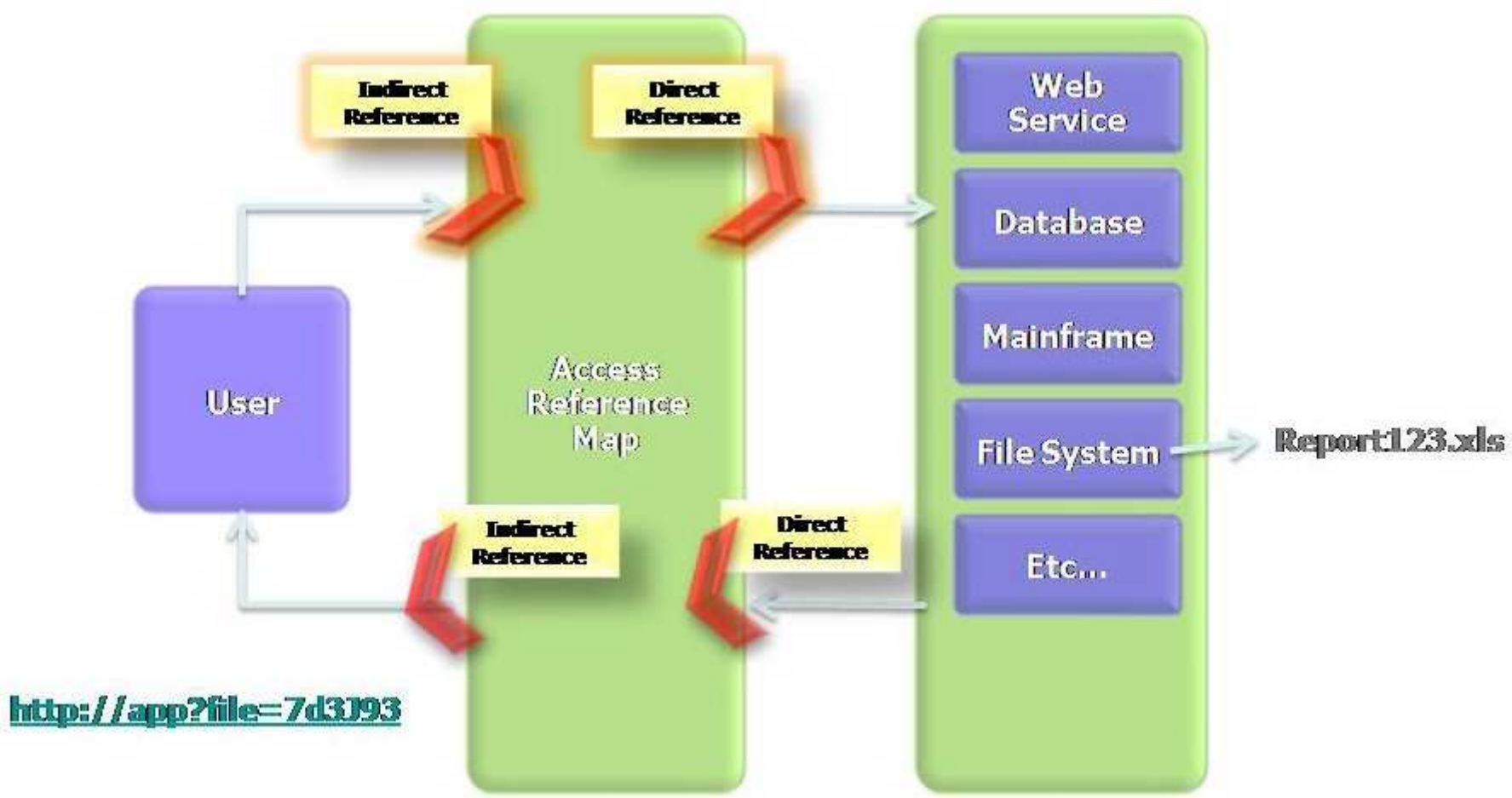
Encryption


User
Controller
Business Functions
Data Layer
Backend
Encrypted Properties
Encryptor

Encryption

- Reference implementation utilizes Java Cryptography Extension (JCE)
- Ensure strong salt and password values are used - takes away the chance for developers to make poor choices for these crucial values
- Algorithms configurable via properties
- Seal - encrypts data and includes an expiration timestamp

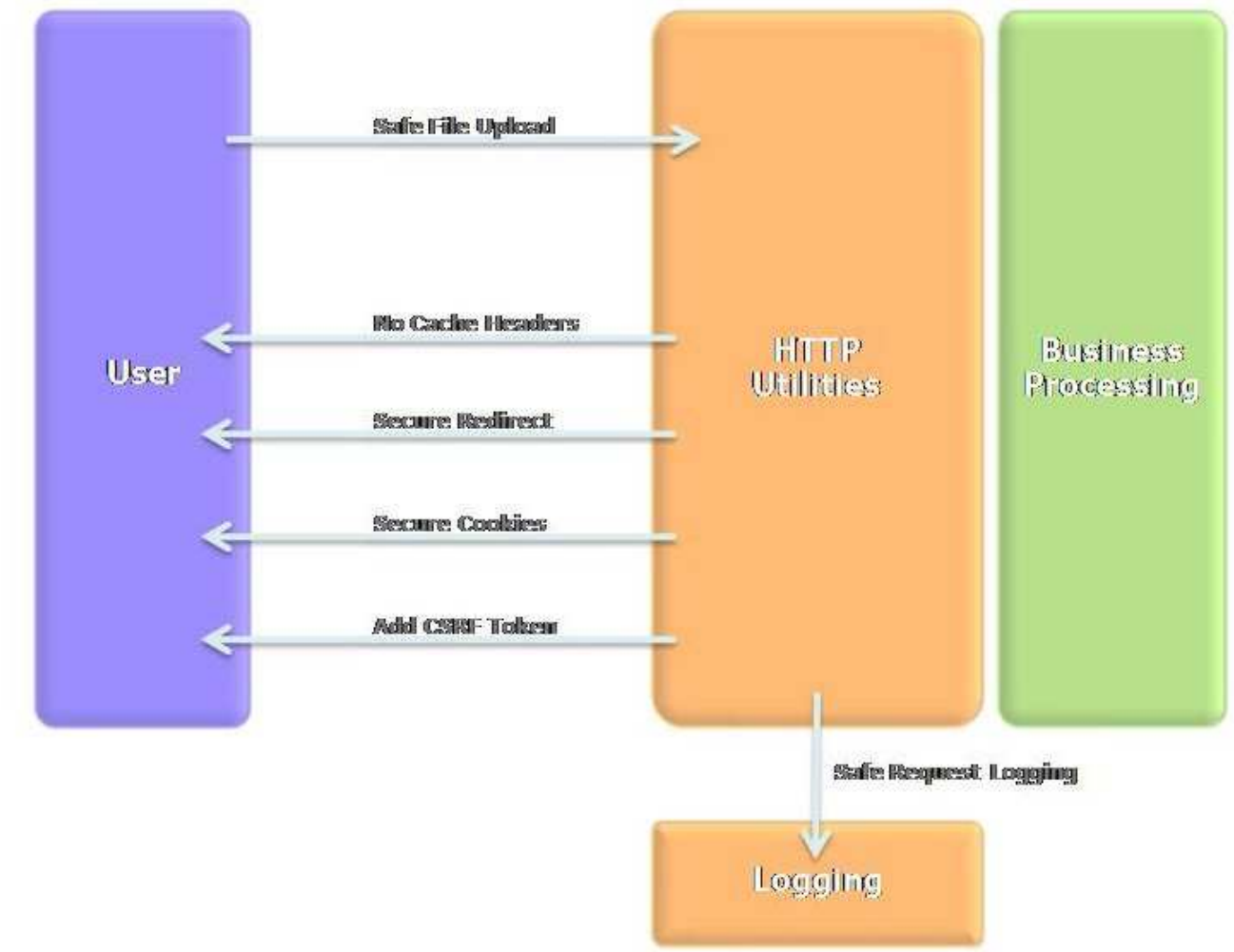
Indirect Object References



Indirect Object References

- Reference implementation includes 2 concrete classes: integer based and random strings
- Defeats parameter tampering attacks
- Can help combat CSRF if per-user access reference maps are used

HTTP Utilities



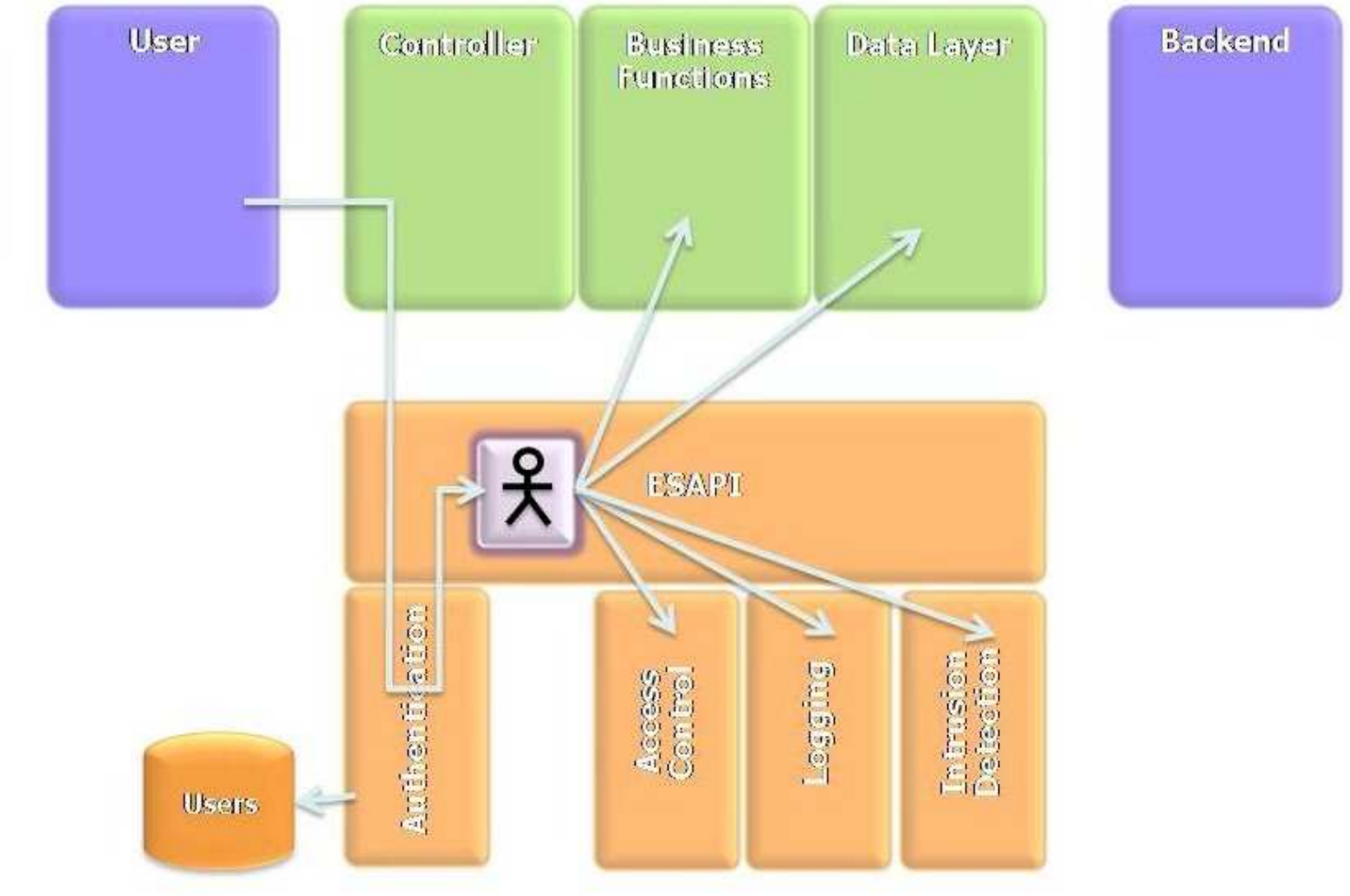
HTTP Utilities

- Provides useful methods relating to request, response, cookies, sessions, etc.
- addCSRFToken()
- changeSessionIdentifier()
- encrypt/decrypt fields
- safeXXX() methods for adding headers, sending forwards & redirects to ensure character encoding

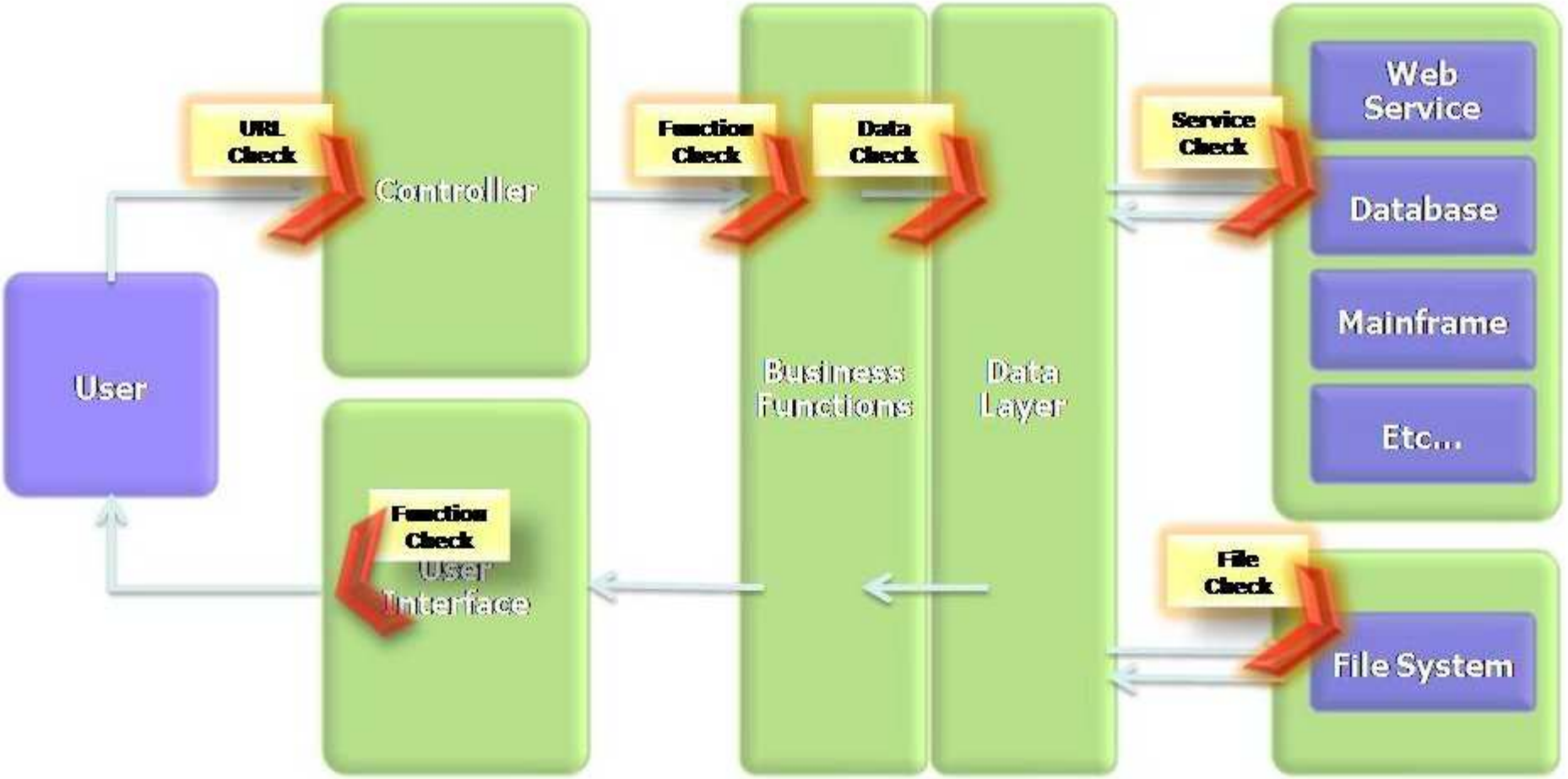
HTTP Utilities

- Reference implementation utilizes Apache Commons FileUploader
- Reference implementation relies on current request & response being stored in ThreadLocal variables - means you have to utilize the ESAPI authenticator or explicitly call ESAPI.authenticator().setCurrentHTTP()

Authentication



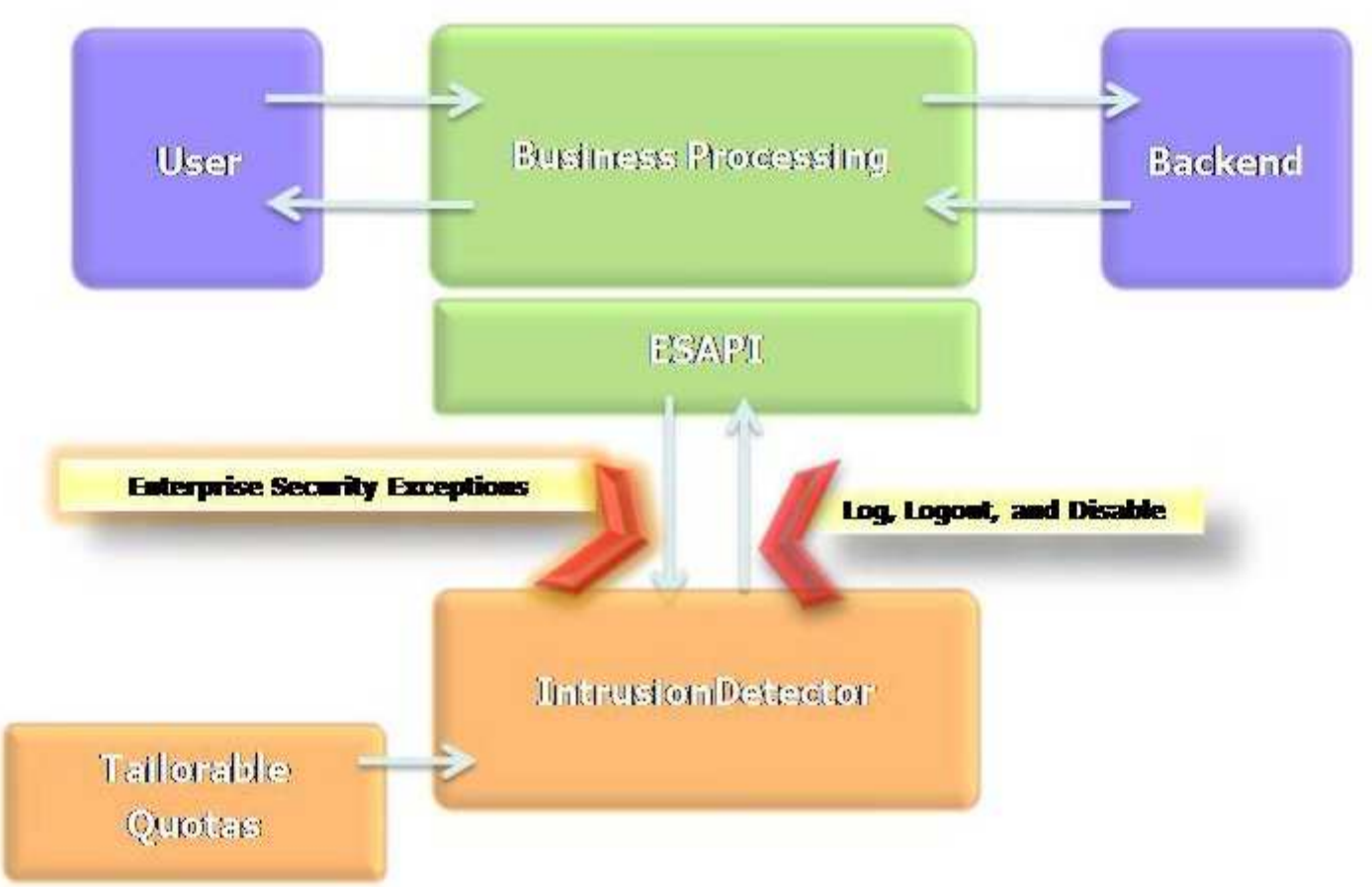
Access Control



Authentication & Access Control

- Reference implementation includes a file based authenticator.
- Provides login/logout capabilities, user authentication using hashed passwords
- Utility methods for password generation and to ensure account name and password strength
- Carries out some of the required setup for other components such as HTTPUtilities

Intrusion Detection



Intrusion Detection

- Reference implementation has a default detector that does rudimentary calculations of number of errors per time period

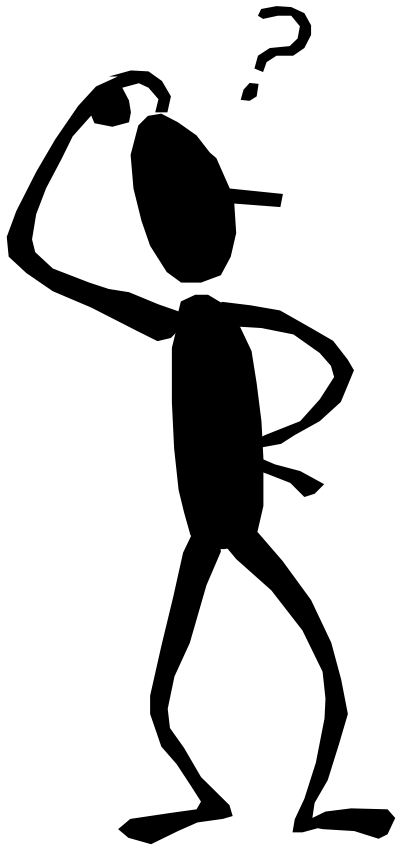
ESAPI Versus OWASP Top 10

OWASP Top Ten	OWASP ESAPI
A1. Cross Site Scripting (XSS)	Validator, Encoder
A2. Injection Flaws	Encoder
A3. Malicious File Execution	HTTPUtilities (upload)
A4. Insecure Direct Object Reference	AccessReferenceMap
A5. Cross Site Request Forgery (CSRF)	User (csrftoken)
A6. Leakage and Improper Error Handling	EnterpriseSecurityException, HTTPUtils
A7. Broken Authentication and Sessions	Authenticator, User, HTTPUtils
A8. Insecure Cryptographic Storage	Encryptor
A9. Insecure Communications	HTTPUtilities (secure cookie)
A10. Failure to Restrict URL Access	AccessController

What's Next?

- Java reference implementation still under active development (refactoring, adding a taglib, etc.)
- Porting to PHP and .NET has begun
- Is there need for a client-side security API?

Discussion



References

- http://www.aspectsecurity.com/documents/Aspect_File_Download_Injection.pdf
- http://msdn.microsoft.com/en-us/magazine/cc188938.aspx