

Date: 2015-05-11 20:36  
  
Project: SKF

# Table of contents

Introduction

Submit forms - pattern

Access-control/Login systems - pattern

XSLT injections

# Introduction

The security knowledge framework is composed by means of the highest security standards currently available and is designed to maintain the integrity of your application, so you and your costumers sensitive data is protected against hackers. This document is provided with a checklist in which the programmers of your application had to run through in order to provide a secure product.

In this part of security knowledge framework, al the parameters and variables are audited by means of the information given by the programmer such as the processing techniques. Each of these techniques contain different types of vulnerabilities when implemented in a improper fashion. This document will raise awareness about these vulnerabilities, as well as presenting solutions for the right implementation.

# Knowledge-Base: Submit forms - pattern

## Technology: Submit forms

Description:  
Whenever a user can submit a form in your system you should consider implementing  
the following defence mechanism in order to ensure high level security.  
  
Single user input validation controls and Audit logs  
CSRF tokens  
Principle of least privilege  
GET/POST requests  
  
  
Solution:  
Here are the steps described briefly.  
For more detailed information you should look into these items in the knowledge base.  
First, you should create a single user input validation control class which should   
validate the expected input values in order to verify if the user is not tampering data   
or is injecting malicious code into your application. All infringements should be logged  
and repercussions should be taken whenever these infringements are frequent.   
Second, whenever an authenticated user is submitting the form always ensure the forms  
contain CSRF tokens in order to prevent cross site request forgery.  
Third, Whenever there are authenticated users with different roles/privileges you should  
enforce restrictions on the server side upon your form submits/processing in order   
to prevent privilege escalation. You should apply the principle of least privilege in   
order to ensure higher level of security.  
Fourth, Whenever the application is sending sensitive data through the form submit  
this data must always be send through an POST variable instead of an GET since  
a GET will leak this data through the url by example the referer header.

# Knowledge-Base: Access-control/Login systems - pattern

## Technology: Access controls or Login systems

Description:  
For successful access control/login functionality there are a lot of things to take into  
consideration before you start implementing these functions.  
  
Solution:  
If you design your access/login system with these items into consideration it will save you a   
lot of time not having to implement them afterwards.  
  
Audit logs  
Principle of least privilege (Privilege based authentication system)  
Passwords must be encrypted, salted and stretched  
Security flags(HttpOnly, secure flag, idle timeout, absolute timeout)  
Cross-Site Request Forgery (CSRF for authenticated forms)  
  
First, you must implement audit logs in order to trace user requests for forensic purposes whenever  
an attacker abuses your system.  
Second, your users table in your database should contain some extra tables and rows for handling  
the privilege based authentication system, as well as implementing a system for user lock-out when  
your logging system detects a possible attacker actively attacking your system.  
After these steps we build a login system with strong password encryption, PDO prepared statements and set security flags  
for the sessions in order to protect them against XSS and enforce them to be transmitted over  
only encrypted data lines.  
And last, now that all the protection layers are in place we want to secure al our data transactions by means of   
CSRF tokens.  
note: As soon as the user hits your application you want to enforce him using a https protected   
connection this can be done by including the Strict-Transport-Security header which looks like:  
Strict-Transport-Security: max-age=31536000; includeSubDomains  
Also you should consider adding your application to a HSTS Preload list for enforcing a higher  
level of security.

# Knowledge-Base: XSLT injections

## Technology: XSLT input and output

Description:  
The vulnerability occurs when a XSL file is loaded from a source controlled by an attacker.  
When the attacker is given the opportunity to specify the source of the included XSL file  
he could include a file which contains malicious code to be parsed on the target application.  
This could lead to, code execution, reading arbitrary files and many more  
vulnerabilities such as XSS.  
  
Solution:  
To protect against such vulnerability one needs to make sure that he does not use   
user-supplied input in the XSL filename.   
The best solution would be to define a list of permitted filenames and   
only accept XSL filenames from that list.