IDS14-J. Do not trust the contents of hidden form fields

HTML allows fields in a web form to be visible or hidden. Hidden fields supply values to a web server but do not provide the user with a mechanism to modify their contents. However, there are techniques that attackers can use to modify these contents anyway. A web servlet that uses a `GET` form to obtain parameters can also accept these parameters through a URL. URLs allow a user to specify any parameter names and values in the web request. Consequently, hidden form fields should not be considered any more trustworthy than visible form fields.

Noncompliant Code Example

The following noncompliant code example demonstrates a servlet that accepts a visible field and a hidden field, and echoes them back to the user. The visible parameter is sanitized before being passed to the browser, but the hidden field is not.

```
public class SampleServlet extends HttpServlet {

  public void doGet(HttpServletRequest request, HttpServletResponse response)
    throws IOException, ServletException {
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();
    out.println("<html>");

    String visible = request.getParameter("visible");
    String hidden = request.getParameter("hidden");

    if (visible != null || hidden != null) {
      out.println("Visible Parameter:");
      out.println( sanitize(visible));
      out.println("<br>Hidden Parameter:");
      out.println(hidden);
    } else {
      out.println("<p>");
      out.print("<form action=\"");
      out.print("SampleServlet\" ");
      out.println("method=POST>");
      out.println("Parameter:");
      out.println("<input type=text size=20 name=visible>");
      out.println("<br>");

      out.println("<input type=hidden name=hidden value=\'a benign value\'>");
      out.println("<input type=submit>");
      out.println("</form>");
    }
  }

  public void doPost(HttpServletRequest request, HttpServletResponse response)
    throws IOException, ServletException {
    doGet(request, response);
  }

  // Filter the specified message string for characters
  // that are sensitive in HTML.
  public static String sanitize(String message) {
    // ...
  }
}
```

When fed the parameter `param1`, the web page displays the following:

> *Visible Parameter: param1*
> *Hidden Parameter: a benign value*

However, an attacker can easily supply a value to the hidden parameter by encoding it in the URL as follows:

> *http://localhost:8080/sample/SampleServlet?visible=dummy&hidden=%3Cfont%20color=red%3ESurprise%3C/font%3E!!!*

When this URL is provided to the browser, the browser displays:

> *Visible Parameter: dummy*
> *Hidden Parameter: Surprise!!!*

Compliant Solution

This compliant solution applies the same sanitization to the hidden parameter as is applied to the visible parameter:

```
public class SampleServlet extends HttpServlet {
  public void doGet(HttpServletRequest request, HttpServletResponse response)
    throws IOException, ServletException {
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();
    out.println("<html>");

    String visible = request.getParameter("visible");
    String hidden = request.getParameter("hidden");

    if (visible != null || hidden != null) {
      out.println("Visible Parameter:");
      out.println( sanitize(visible));
      out.println("<br>Hidden Parameter:");
      out.println( sanitize(hidden));          // Hidden variable sanitized
    } else {
      out.println("<p>");
      out.print("<form action=\"");
      out.print("SampleServlet\" ");
      out.println("method=POST>");
      out.println("Parameter:");
      out.println("<input type=text size=20 name=visible>");
      out.println("<br>");

      out.println("<input type=hidden name=hidden value=\'a benign value\'>");
      out.println("<input type=submit>");
      out.println("</form>");
    }
  }

  public void doPost(HttpServletRequest request, HttpServletResponse response)
    throws IOException, ServletException {
    doGet(request, response);
  }

  // Filter the specified message string for characters
  // that are sensitive in HTML.
  public static String sanitize(String message) {
    // ...
  }
}
```

Consequently, when the malicious URL is entered into a browser, the servlet produces the following:

> *Visible Parameter: dummy*
> *Hidden Parameter: <font color=red>Surprise</font>!!!*

Risk Assessment

Trusting the contents of hidden form fields may lead to all sorts of nasty problems.

Rule	Severity	Likelihood	Remediation Cost	Priority	Level
IDS14-J	High	Probable	High	P6	L2

Automated Detection

Tool	Version	Checker	Description
The Checker Framework	2.1.3	**Tainting Checker**	Trust and security errors (see Chapter 8)
Fortify	6.10.0120	**Hidden_Field**	Implemented

Bibliography

[Fortify 2014]	Fortify Diagnostic