Abstract

- This talk will present two arguments against performing input validation testing at the HTTP request level (using proxies) and suggest a new approach: performing input validation testing directly in the user interface.

- It will also introduce Groundspeed, an open-source add-on for Mozilla Firefox that allows you to modify, on the fly, the forms and form fields in the page loaded in the browser.

- Groundspeed is available at:
  http://groundspeed.wobot.org
1. Introduction
An Answer with no Question

- In the “Hitchhiker's Guide to the Galaxy”, some super intelligent beings build a super computer to answer “the ultimate question of life the universe and everything”
- After seven and a half million years, the computer gives its answer: “42”
- Puzzled and disappointed, they ask the computer what the question was
- But the computer is not powerful enough to respond that
Information Needs Context to be Useful

- In the book, the joke is in the fact that the answer “42” makes no sense without the question.
- In order to understand the information in the answer, we need to know the question because the question provides context to the answer.
- This is an unusual problem when people try to understand a piece of data, and it affects us more than we first imagine.
- Let’s consider a normal example of interaction with a web application.
Example: User Submits a Form

- User loads a web page with a form
- User types a value in a form field and submits
- Client side logic validation is executed
- Browser creates an HTTP with the input data in a parameter in the GET or POST request
- Browser sends the request to the server
- Server reads the HTTP request and handles data to application
- Application processes the data and responds
The Life Cycle of Input Data

User

Types input data in form field

User Interface

Name: “John Smith”

custname.value = “John Smith”;

Client Side Logic

... &custname="John Smith"& ...

HTTP Request

To the server
Interface Labels Provide Context to Humans

- The form label next to the textbox field (for example “Name: ”) is the question being asked to the user.
- The value typed inside the textbox is the response to that question.
- As data travels down through the layers, the answer is separated from the question.
- When considered from the perspective of the HTTP request, the data is no longer in its original context (the labels provided context).
Argument 1: The Mapping Problem

- Usually this is not a big problem because we can use the HTTP parameter names as “pseudo” labels.

- But parameter names do not need to match the labels, they are meant for server-side code not for humans so it could be any arbitrary value.

- In order to compensate for the lack of context, we need to solve this “mapping problem” (mapping label to parameter name), and this makes the test a little more difficult.
Filling the Gaps

- When performing input validation with a proxy, the tester has to solve the mapping problem.
- We probably figured out some tricks to help with this process:
  - Typing input data that explain what each field is.
  - Using the source code of the page to map parameters and label names.
- But these tricks only work well for simple web applications.
A Problem That Will Get Harder to Solve

The problem is that in order for the tricks to work, we have to assume two conditions:

- That one form will translate to one HTTP request, form fields matching HTTP parameters one to one
- That submitting one form will cause one HTTP request to be sent, at the time of the click

But those conditions are not true all the time:

- As client-side logic gets more complex, data can be consolidated, pre-processed, encoded on the client
- AJAX (no synchronicity of the user actions and HTTP)

And things will only get more complicated
Argument 2: Working at Two Places

- Another problem of working at the HTTP level is that it forces the tester to switch between two worlds (user interface and HTTP).
- Ideally, manipulation of the input data should happen in the user interface, but testers have to work at the HTTP level to bypass browser limitations.
- This switching has a cost associated to it.
- Let’s consider the task of performing one input validation test on one form field.
Task Breakdown

- In the browser, fill in the field to be tested
- Submit the form
- Select the proxy window
- Scan through the HTTP request to find where the request parameters are located
- Scan through the parameters to find the one that needs to be modified
- Insert the attack string
- Send the request to the server and select the browser window to see the result
Hard Problems

- Even considering a simple task (1 test on 1 field) there are a lot of steps to complete
- Some steps are distracting
- But some steps create problems
  - Scanning through text requires “parsing”
  - Find the right parameter requires “mapping”
- There are even more tasks if some fields are encoded or if there is form validation
- There are a lot of secondary steps that do not contribute to the main goal: test input validation
Test Friction

- Consider now all the tests we have to perform on all the parameters of the web application
- There will be a lot of secondary tasks, this is not an optimal process
- The secondary steps are like energy that is lost in things that do not contribute to the goal
  - We could think about them as “test friction”
- This is a user experience problem not a usability problem
2. A New Approach
Why do We Work at the HTTP Level?

- The browser is a tool designed for normal users, so it does not provide all the functionality that web app testers need.
- A few years ago, to modify the browser was too hard (closed source) and to create a browser from scratch was too much work.
- Working outside the browser, at the HTTP level was the best alternative:
  - The mapping problem was easy to solve in the old web.
  - The user experience problem was a reasonable price to pay.
Now Things are Different

- Because of the richer web apps, mapping parameters to interface labels is getting harder.
- The mapping problem might still be tolerable but will only get worse as apps get more complex.
- Working at the HTTP level forces the tester has to switch between two worlds: user interface (browser) and HTTP (proxy) adding “friction” to the process.
- Another important change: browsers are now easier to modify (open source and add-on systems).
Suggesting a New Approach

- We should reconsider the strategy for input validation tests
  - It makes sense to keep some tests at the HTTP level
  - But most could benefit from work at the interface level
- It is easier and more efficient to manipulate data from the user interface instead of the HTTP level
  - That would eliminate the mapping problem
  - Data makes more sense at the user interface level (bigger chance of business logic exploitation)
  - Improves user experience for tester (reduces friction)
3. Introducing Groundspeed
Introducing Groundspeed

Groundspeed is an open-source add-on for Mozilla Firefox (http://groundspeed.wobot.org)

Groundspeed allows a tester to perform input validation testing from the web app user interface.

Groundspeed modify the browser in order to adapt it to the needs of web app testers:
  - Manipulate the application’s user interface
  - Remove client side validation and other limitations
Download and Install Groundspeed

- Visit http://groundspeed.wobot.org

- The following screenshots were taken on demo forms available at groundspeed.wobot.org/demo
How to Start Groundspeed

In order to start working with groundspeed, open the sidebar using:

Tools – Groundspeed

Once groundspeed is open, load the form information from the current page using the “Reload” button
How to Start Groundspeed

1. Click on the 'View' menu.
2. Select 'Ground Speed' and press 'Alt + R'.
3. Click on the 'Reload' button.
Select a Form or a Form Field

- The groundspeed sidebar has two sections
  - The top section lists the forms and fields in the page
  - The bottom section lists the HTML attributes of the selected form or form element

- Select one form or field in the list in the top section, and groundspeed will highlight that element in the page and display its attributes in the lower part of the sidebar
View the Selected Form and its Attributes

Login to access your information
Please authenticate using your username and password in order to access the web application.

Username
Password
Submit

Form Attributes:
- action =
- method = post
- class = appntrn
- id = form_102846
View the Selected Field and its Attributes

Find a location near you!
We are everywhere! With more than a thousand stores, you will have no problems finding a convenient location!

ZIP code

Distance

Submit

Generated by pForm
View Hidden Fields

- If you select a hidden field from the list of forms and form fields, the hidden field becomes visible in the page.
View Hidden Fields

Select form_id (HIDDEN)

Result:

form_id = 103746
Edit Attributes of Forms and Fields

- Groundspeed lets you modify, add and remove HTML attributes on the form and the form fields.
- In the upper list, select the form or the form field that you want to change the attributes.
- Right-click on the attribute in the lower list and select from the context menu.
- A double-click on a selected attribute will edit the current value.
Edit Attributes of Forms and Fields

1. Select the form or field you want to edit.
2. Right-click on the selected option to open the context menu.
3. Choose the 'Edit' option to modify the attributes of the selected form or field.
Change the Type of any Form Field

- Groundspeed allows you to change any form field into a textbox with 2 clicks:
  - Right-click on the element you want to convert
  - Choose the “Transform into Text Input”
- After the change, you can edit the contents of the field directly in the page
- Groundspeed will create a label that matches the field “name” or “id” attribute
- For “Select” elements, groundspeed adds a “cheat-sheet” with the options that existed in the drop down list
Change the Type of any Form Field

right click
form_id (HIDDEN)

select
Transform into Text Input

result
form_id (originally HIDDEN)
103746
Submit

Find a location near you!
We are everywhere! With more than a thousand stores, you will have no problem finding a location!

ZIP code
Distance

Generated by pForm
Changing a Select into textbox

right click  
select  
result
Other Manipulations in Forms

Other useful things you can do in a form:

- Add a new form fields
- Make the form submit in a new window (so you don’t have to manipulate the interface all over again)
- Change all hidden fields into textboxes
- Save all form field values (and clear the saved values)
- Remove all JavaScript event handlers (in the form and its fields)

These are all options available when right clicking the form in the list
Other Manipulations in Forms
Other Manipulations in Fields

Other useful things you can do in a text or hidden field:

- Remove length limits (so you can add more data)
- Modify size (so the field looks bigger in the page)
- Encode and decode the contents (Base64, Hex, HTML Entities, Unicode, URL Encode)
- Hash the contents of the field (MD5, SHA1)
- Remove all JavaScript Event Handlers

These options are available when you right-click on a field in the list
Other field manipulation options
Groundspeed

- Adapts the web application interface to fit the needs of the security tester
  - What you need, where you need: no friction
  - Eliminates the complex secondary tasks
- Allows input validation to be performed from the interface
  - Eliminates the mapping problem
Groundspeed Limitations and Bugs

- Groundspeed does not work with poorly designed pages (where layout is made with HTML tables) because of DOM hierarchy
- Groundspeed does not see JavaScript event handlers that are set programmatically using JavaScript