Unlocking the Toolkit

Attacking Google Web Toolkit Applications

Ron Gutierrez

Gotham Digital Science

Introduction

- Ron Gutierrez
- Security Consultant @ GDS
- rgutierrez@gdssecurity.com
- http://www.gdssecurity.com/l/b/

Purpose of Presentation

- Why is black-boxing GWT apps difficult?
- Discuss tools and techniques for testing GWT apps
- This presentation is not about finding flaws in GWT but rather finding flaws in the underlying application built using GWT

MOTIVATION

- Google’s new web application bug bounty
- Pays $500 – $3,133.70 for bugs
- Google uses GWT to create some of their web applications (Adwords, Wave, etc)
- http://googleonlinesecurity.blogspot.com/2010/11/rewarding-web-application-security.html

AGENDA

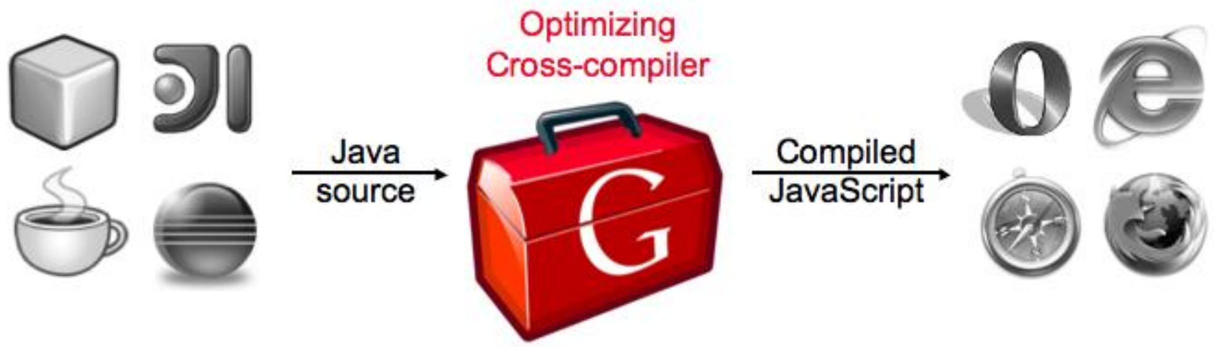
- The Overview
  - GWT Introduction
  - Testing Difficulties
- The Reconnaissance
  - GWT service and method enumeration
  - Unlocking UI Features
- The Attack
  - What common web app vulnerabilities apply?
  - GWT RPC parsing and fuzzing

Unlocking the Toolkit: Attacking Google Web Toolkit (GWT)

THE OVERVIEW

Google Web Toolkit (GWT)

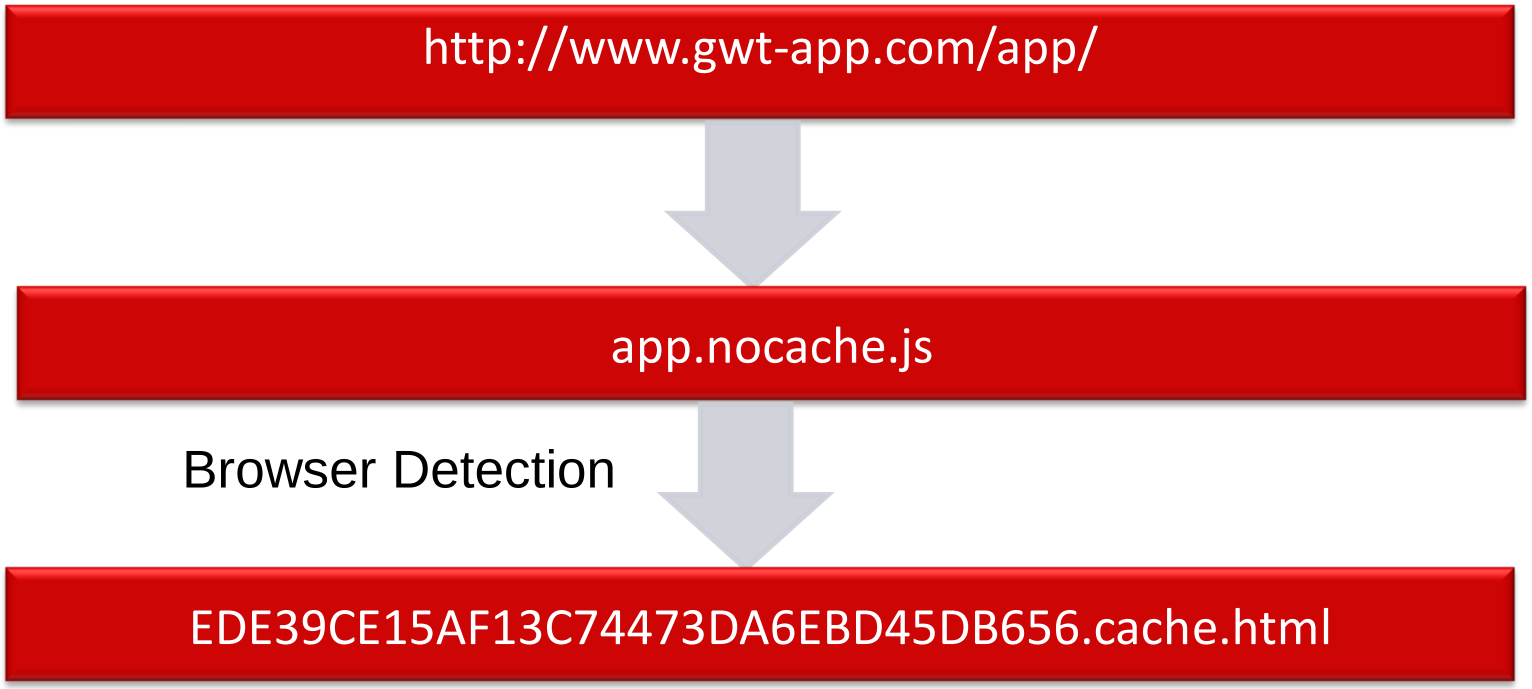
- Open source Java framework used to create Rich Internet Applications
- Both server and front-end are written in Java
- Java-to-Javascript compiler



Benefits

- Code re-use between server and client
- Provides Remote Procedure Call (RPC) mechanism for client-server communication
- Complex and visually appealing front-ends without the cross-browser headaches
- Lots of widgets and extensions freely available
- No browser plugin required

BOOTSTRAP SEQUENCE


http://www.gwt-app.com/app/
app.nocache.js
Browser Detection
EDE39CE15AF13C74473DA6EBD45DB656.cache.html

Each {HEX}.cache.html is browser specific

CLIENT SIDE CODE

- In Expression Languages (i.e. JSF or Struts), presentation logic is run on the server
- With GWT, all front-end logic is compiled into a Javascript equivalent
- All client-side code is downloaded to the user’s browser

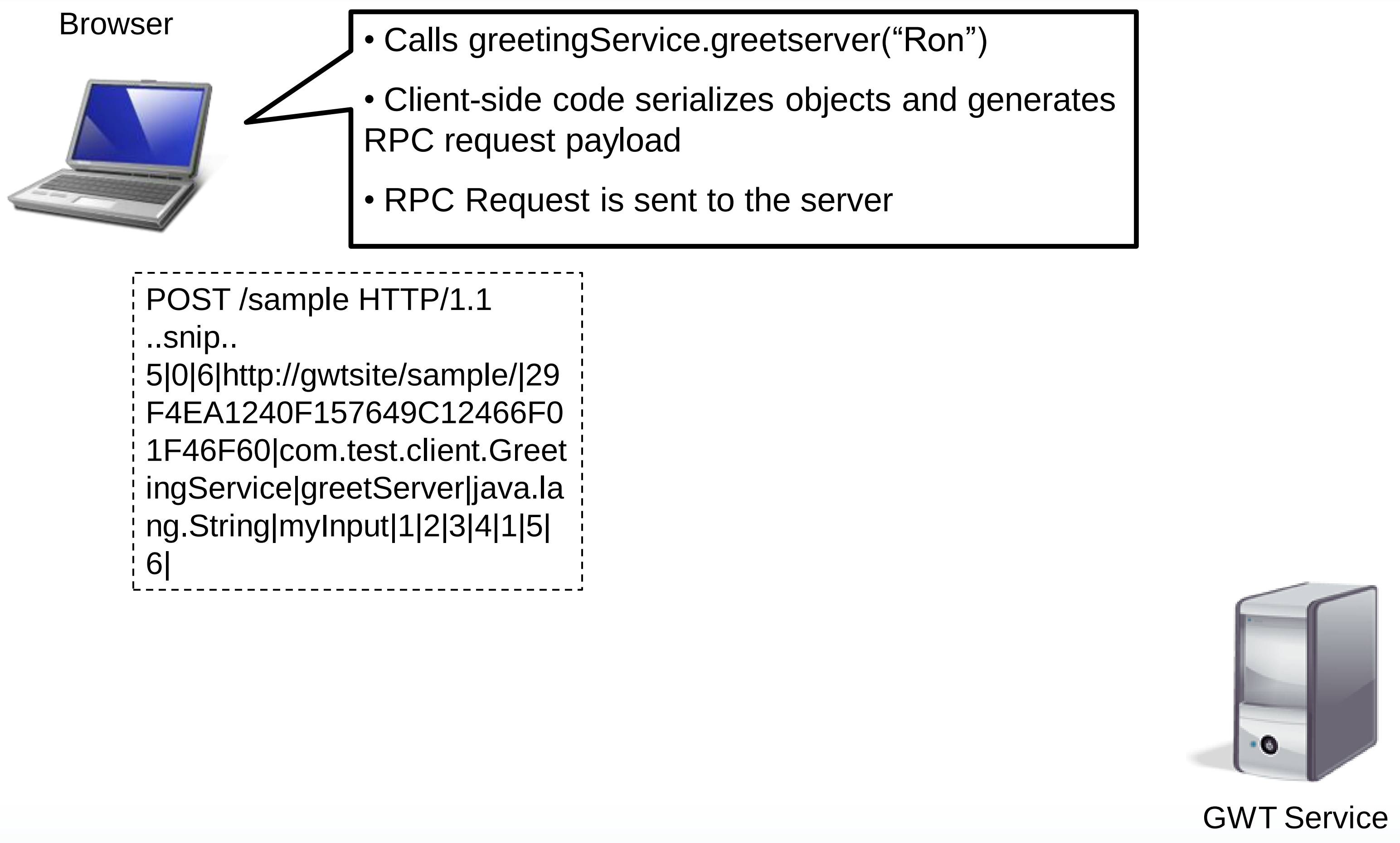
CLIENT SIDE CODE

- Javascript code is protected with obfuscation
- Contains some valuable information
  - GWT-RPC service endpoints
  - Custom object structures
  - Restricted or hidden UI functionality
- The obfuscation is one of the obstacles we hope to solve during this presentation

GWT-RPC

- Built-in Remote Procedure Call (RPC) framework
- Uses a serialization protocol to call remote methods
  - Sends Java data types and objects as parameters from client to server.
- GWT RPC methods return objects serialized using JSON

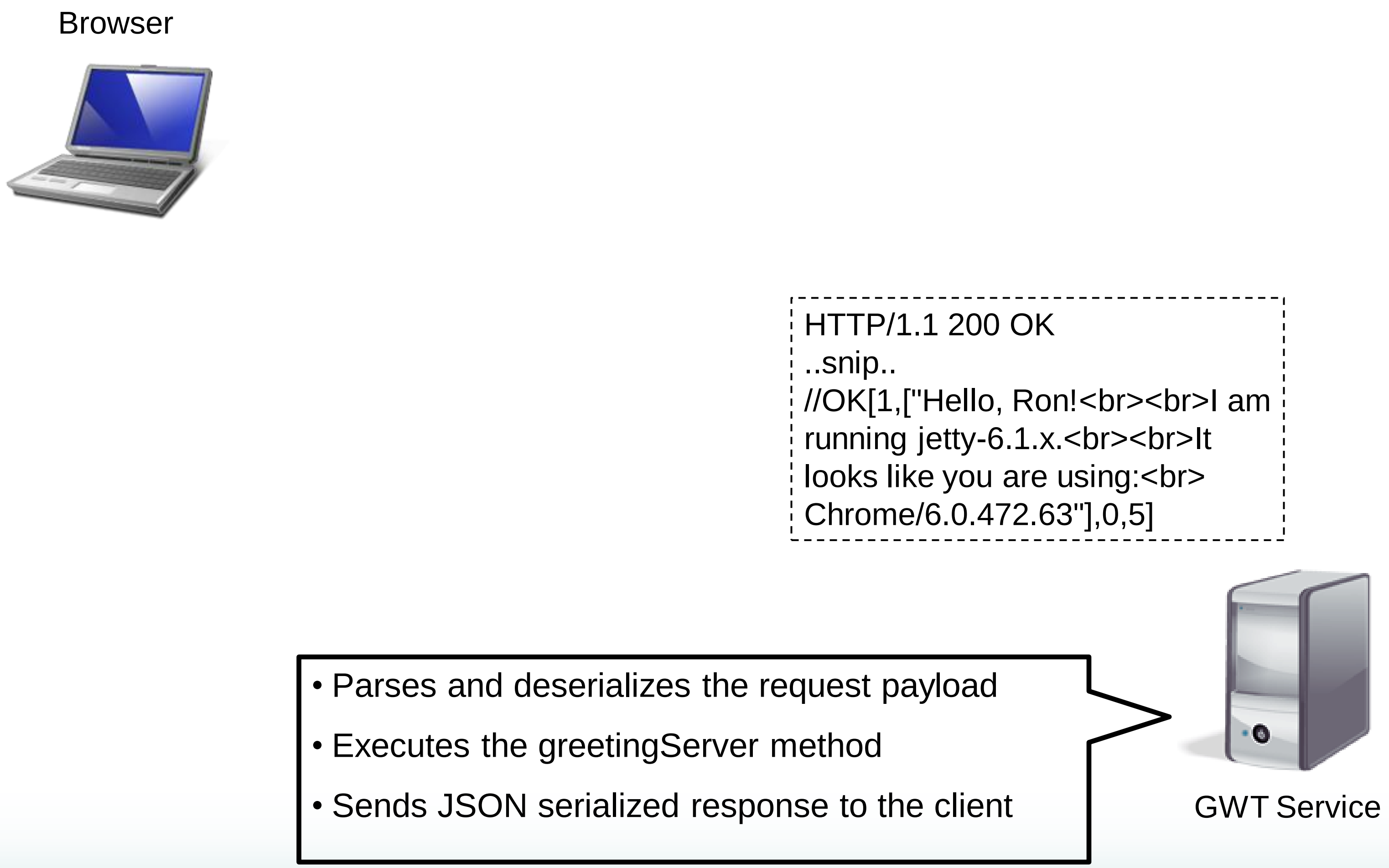
GWT-RPC DIAGRAM



- Calls greetingService.greetserver(“Ron”)
- Client-side code serializes objects and generates RPC request payload
- RPC Request is sent to the server

```
POST /sample HTTP/1.1
..snip..
5|0|6|http://gwtsite/sample/|29
F4EA1240F157649C12466F0
1F46F60|com.test.client.Greet
ingService|greetServer|java.la
ng.String|myInput|1|2|3|4|1|5|
6|
```

GWT-RPC DIAGRAM



```
HTTP/1.1 200 OK
..snip..
//OK[1,["Hello, Ron!<br><br>I am
running jetty-6.1.x.<br><br>It
looks like you are using:<br>
Chrome/6.0.472.63"],0,5]
```

- Parses and deserializes the request payload
- Executes the greetingServer method
- Sends JSON serialized response to the client

TESTING OBSTACLES

- Client-side code is obfuscated by default
- RPC request payloads are serialized in a custom format
- Dynamic web app scanners do not correctly test GWT-RPC requests

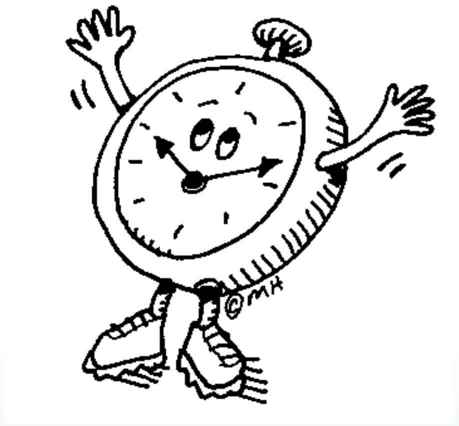
TESTING OBSTACLES

```
5|0|8|http://tester:8888/testapp/|9E4CB3
D5635C548906BFB576DD18C710|com.test.app.
client.GreetingService|greetServer|[Ljav
a.lang.String;/2600011424|hi|there|blah|
1|2|3|4|1|5|5|3|6|7|8|%26ping%20-
n%2020%20127.0.0.1%26
```

WEB SCANNER EXAMPLE

TESTING OBSTACLES

- Security assessments must be finished within a short time frame
- Researching GWT and trying to overcoming it’s obstacles on your own is time lost in actual SECURITY TESTING



Unlocking the Toolkit: Attacking Google Web Toolkit (GWT)

THE RECONNAISSANCE

What kind of recon?

- Enumerating all GWT-RPCs available on the client
  - We want full application coverage (All Services, Methods and Parameter Values)
- Unlocking hidden/restricted functionality available in the UI

Unlocking the Toolkit: Attacking Google Web Toolkit (GWT)

THE RECONNAISSANCE: ENUMERATION

GWT Compilation Modes

- Obfuscated: Javascript is obfuscated and minified to reduce the size of generated files. (Default Option)
- Pretty: Generated Javascript is human readable
- Detailed: Javascript contains even more detail, such as verbose variable names

OBFUSCATED JAVASCRIPT

- Functions and variable names are renamed
- White space removed
- Functions re-ordered based on size
- String values are stored as global variables towards the end of the file

PRETTY GWT-RPC CALL

```
function $UserMethod1(this$static, str1, str2, i, callback){
    [..snip..]
    !!$stats && $stats({moduleName:$moduleName, sessionId:$sessionId,
    subSystem:'rpc', evtGroup:requestId,
    method:'UserConsoleService_Proxy.UserMethod1', millis:(new Date).getTime(),
    type:'begin'});

    streamWriter = $createStreamWriter(this$static);

  try {
     append(streamWriter.encodeBuffer, '' + $addString(streamWriter,
                  'com.gwtsample.client.UserConsoleService'));
     append(streamWriter.encodeBuffer, '' + $addString(streamWriter,
                  'UserMethod1'));
     append(streamWriter.encodeBuffer, '3');
    [..snip..]
    payload = $toString_3(streamWriter);
     [..snip..]
     $doInvoke(this$static, ($clinit_136() ,
                          'UserConsoleService_Proxy.UserMethod1'), requestId,
    payload, callback);
  }
```

Obfuscated GWT-RPC Call

```
function jy(b,c,d,e,f){
    [..snip..]
    !!$stats&&$stats({moduleName:$moduleName,sessionId:$sessionId,subSystem:T
    G,evtGroup:j,method:oI,millis:(new Date).getTime(),type:WH});
    k=vr(b);
    try{
                    lr(k.b,oF+Oq(k,pI));
                    lr(k.b,oF+Oq(k,qI));
                    lr(k.b,ZH);
                    lr(k.b,oF+Oq(k,$H));
                    lr(k.b,oF+Oq(k,$H));
                    lr(k.b,oF+Oq(k,rI));
                    lr(k.b,oF+Oq(k,c));
                    lr(k.b,oF+Oq(k,d));
                    lr(k.b,oF+e);
                    i=jr(k);

                    [..snip..]
                    wr(b,(cs(),oI),j,i,f)
    }
```

OBFUSCATED GWT-RPC CALL

```
function jy(b,c,d,e,f){
    [..snip..]
    !!$stats&&$stats({moduleName:$moduleName,sessionId:$sessionId,subSystem:T
    G,evtGroup:j,method:oI,millis:(new Date).getTime(),type:WH});
    k=vr(b);
    try{
                    lr(k.b,oF+Oq(k,pI));
                    lr(k.b,oF+Oq(k,qI));
                    lr(k.b,ZH);
                    lr(k.b,oF+Oq(k,$H));
                    lr(k.b,oF+Oq(k,$H));
                    lr(k.b,oF+Oq(k,rI));
                    lr(k.b,oF+Oq(k,c));
                    lr(k.b,oF+Oq(k,d));
                    lr(k.b,oF+e);
                    i=jr(k);

                    [..snip..]
                    wr(b,(cs(),oI),j,i,f)
    }
```

```
oI='UserConsoleService_Proxy.UserMethod1'
```

```
pI='com.gwtsample.client.UserConsoleSer
vice'
```

```
ZH='3'
```

```
qI='UserMethod1'
```

GWTEnum

- Python script that automates the GWT-RPC enumeration
- Downloads a {HEX}.cache.html file and uses regular expressions to enumerate all methods
- Source @ github.com/rongutierrez

GWTENUM

```
Usage: gwtenum.py [options]

A tool for enumerating GWT RPC methods

Options:
  --version             show program's version number and exit
  -h, --help            show this help message and exit
  -p PROXY, --proxy=PROXY
                        Proxy Host and Port (ie. -p
                        "http://proxy.internet.net:8080")
  -b, --basicauth       User Basic Authentication ( Will be prompted for creds )
  -k COOKIES, --cookies=COOKIES
                        Cookies to use when requesting the GWT Javascript
                        Files (ie. -c "JSESSIONID=AAAAAA")
  -u URL, --url=URL     Required: GWT Application Entrypoint Javascript File
                        (ie. *.nocache.js )
```

GWTENUM

- What do we look for in the results?
  - Admin methods
  - Un-called methods
  - Unauthenticated access to methods

GWTEnum

- Are the {HEX}.cache.html files accessible by unauthenticated users?
- Is the login functionality implemented using GWT RPC?
- If yes, the {HEX}.cache.html files are leaking out information to <u>unauthenticated users</u>!

GWTENUM DEMO

CREATING A GWT RPC CLIENT

- SyncProxy by gdevelop
  - Can invoke GWT Service methods from pure Java
  - http://code.google.com/p/gwt-syncproxy/
- Cookie support is lacking
- Use SyncProxy to generate the RPC request and then you can capture the request through a proxy

SyncProxy

```
private static GreetingService rpcService =
   SyncProxy.newProxyInstance(GreetingService.class,
   ‘http://example.com/helloApp’, ‘greet’);

..snip..

String result =
   rpcService.greetServer(‘SyncProxy’, ‘A String’, 1);
```

Unlocking the Toolkit: Attacking Google Web Toolkit (GWT)

THE RECONNAISANCE: UNLOCKING UI FEATURES

UNLOCKING UI FEATURES

- <u>ALL</u> client-side code is compiled and loaded by the browser
- What if the front-end displays different functionality based on a user's role?

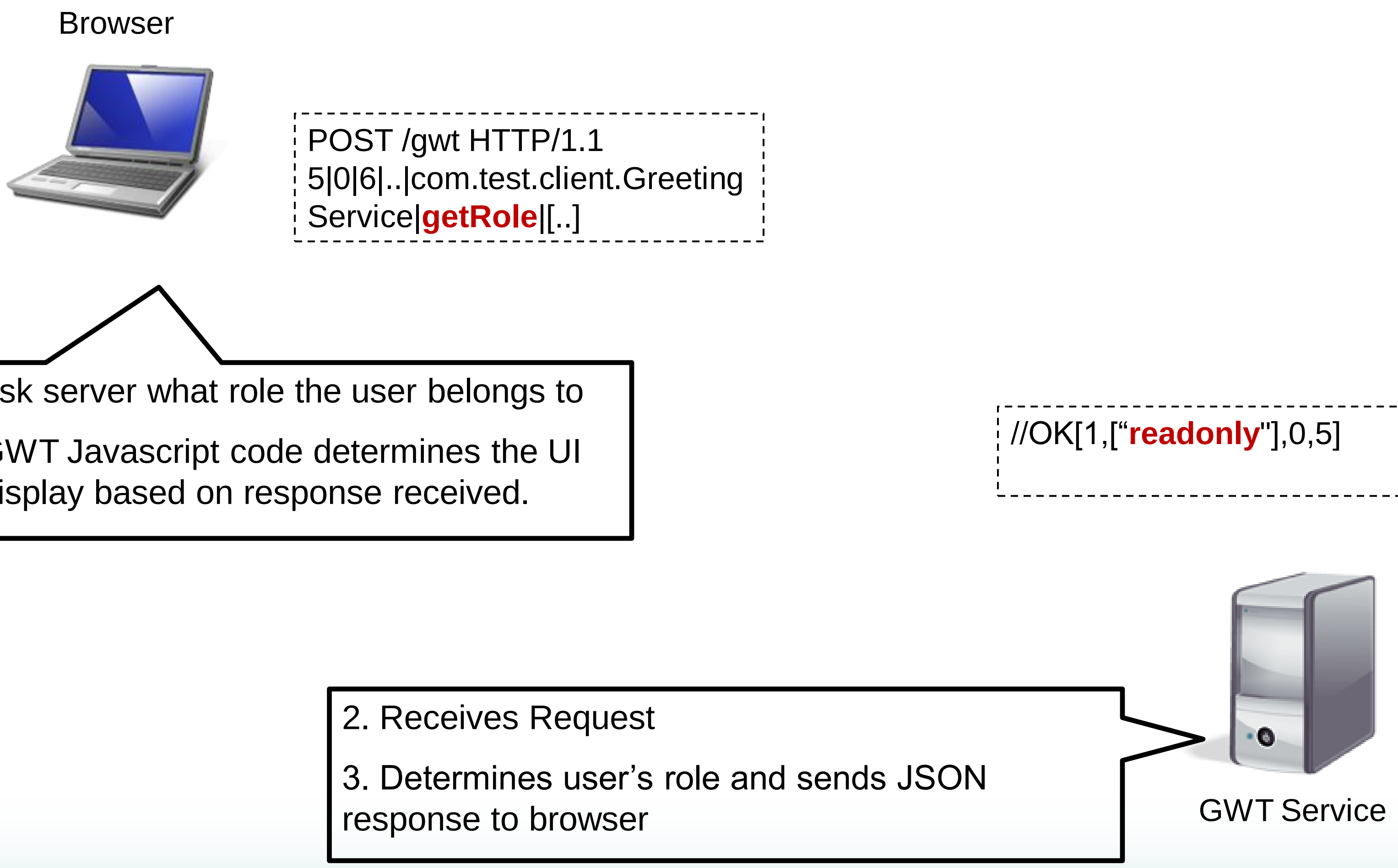
Typical Java Web Application

GET/app.jsp HTTP/1.1

HTTP 200 OK

- Receives Request
- Determines user’s role based on user’s session
- Executes presentation logic and returns UI based on user’s role.

GWT Web Application Example



```
POST /gwt HTTP/1.1
5|0|6|..|com.test.client.Greeting
Service|getRole|[..]
```

1. Ask server what role the user belongs to

4. GWT Javascript code determines the UI to display based on response received.

```
//OK[1,[“readonly"],0,5]
```

2. Receives Request

3. Determines user’s role and sends JSON response to browser

Unlocking UI Features

- Pay close attention to HTTP responses to see if the client is reading a user’s role or admin flag
- Roles can be manipulated in order to trick the client into displaying additional functionality
- Authorization testing is much easier when all application functionality readily available in the UI

UNLOCKING UI: IN REAL LIFE

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 2465

{accountExternalCustomerId:'[Omitted]',accountCustom
erId:' [Omitted]',emailAddress:' [Omitted]
',preBilling3:'false',canServiceAccounts:'false',obf
uscatedCustomerId:' [Omitted]
',defaultToCuesVersion2:'true`,
isInternalUser:'false',userCustomerId:' [Omitted]
',userExternalCustomerId:' [Omitted]
',CuesHeaderVersion:'2',capabilities:'
[{a:false,e:false},{a:false,e:false},{a:false,e:fals
e},{a:false,e:false},{a:false,e:false},{a:true,e:tru
e},{a:false,e:false},{a:false,e:false},{a:false,e:fa
lse},{a:false,e:false},{a:true,e:true},{a:true,e:tru
e},{a:true,e:true},{a:true,e:true},{a:false,e:false}
,{[..snip..]
```

THE ATTACK

ATTACKING GWT APPLICATIONS

- GWT RPC services are vulnerable to the same type of server-side vulnerabilities as typical web apps.
  - Ex. SQL Injection, Path Manipulation, etc
- How about vulnerabilities that affect the browser like Cross-Site Scripting?
- Do GWT requests contain CSRF protection?

GWT CSRF PROTECTION

- Requests include a “X-GWT-Permutation” HTTP header
- Provides sufficient protection because…
  - Form submissions cannot set headers
  - XmlHttpRequest can not make requests across domains because of Same Origin Policy (SOP)
  - Flash can set headers but it requires a mis-configured cross domain policy file..

CROSS-SITE SCRIPTING

- CSRF protection prevents most GWT applications from being vulnerable to reflected XSS
  - Can still be vulnerable if application is not using GWT RPC
- GWT applications are still be vulnerable to stored XSS
  - The GWT client API provide ways to render HTML within widgets (setInnerHTML, setHTML, and HTML constructor)

Unlocking the Toolkit: Attacking Google Web Toolkit (GWT)

THE ATTACK: REQUEST FUZZING

GWT-RPC Request Format

- Request payload is a plaintext, pipe-delimited serialized string
- Separated into three parts
  - Header
  - String table
  - Payload

GWT-RPC Request Format

```
5|0|8|http://localhost:8080/test/|168
78339F02B83818D264AE430C20468|com.tes
t.client.TestService|testMethod|java.
lang.String|java.lang.Integer|myInput
1|java.lang.Integer/3438268394|1|2|3|
4|2|5|6|7|8|1|
```

HEADER

GWT-RPC Request Format

```
5|0|8|http://localhost:8080/test/|16878339F02B83818D264AE430C20468|com.test.client.TestService|testMethod|java.lang.String|java.lang.Integer|myInput1|java.lang.Integer/3438268394|1|2|3|4|2|5|6|7|8|1|
```

STRING TABLE

GWT-RPC Request Format

```
5|0|8|http://localhost:8080/test/|168
78339F02B83818D264AE430C20468|com.tes
t.client.TestService|testMethod|java.
lang.String|java.lang.Integer|myInput
1|java.lang.Integer/3438268394|1|2|3|
4|2|5|6|7|8|1|
```

PAYLOAD

PARSING WALKTHROUGH

```
5|0|8|http://localhost:8080/test/|168
78339F02B83818D264AE430C20468|com.tes
t.client.TestService|testMethod|java.
lang.String|java.lang.Integer|myInput
1|java.lang.Integer/3438268394|1|2|3|
4|2|5|6|7|8|1|
```

SERIALIZATION VERSION

PARSING WALKTHROUGH

```
5|0|8|http://localhost:8080/test/|168
78339F02B83818D264AE430C20468|com.tes
t.client.TestService|testMethod|java.
lang.String|java.lang.Integer|myInput
1|java.lang.Integer/3438268394|1|2|3|
4|2|5|6|7|8|1|
```

STRING TABLE SIZE

PARSING WALKTHROUGH

- String table elements are referenced by the payload
- Payload reconstructs the method call, parameter types and values

1	http://localhost:8080/test/
2	16878339F02B83818D264AE430C20468
3	com.test.client.TestService
4	testMethod
5	java.lang.String
6	java.lang.Integer
7	myInput1
8	java.lang.Integer/3438268394

PARSING WALKTHROUGH

1 | 2 | 3 | 4 | 2 | 5 | 6 | 7 | 8 | 1 |

1	**http://localhost:8080/test/**
2	16878339F02B83818D264AE430C20468
3	com.test.client.TestService
4	testMethod
5	java.lang.String
6	java.lang.Integer
7	myInput1
8	java.lang.Integer/3438268394

SERVLET URL

PARSING WALKTHROUGH

1 | **2** | 3 | 4 | 2 | 5 | 6 | 7 | 8 | 1 |

1	http://localhost:8080/test/
2	**16878339F02B83818D264AE430C20468**
3	com.test.client.TestService
4	testMethod
5	java.lang.String
6	java.lang.Integer
7	myInput1
8	java.lang.Integer/3438268394

Not a CSRF Token

STRONG NAME

Parsing Walkthrough

1 | 2 | **3** | 4 | 2 | 5 | 6 | 7 | 8 | 1 |

1	http://localhost:8080/test/
2	16878339F02B83818D264AE430C20468
3	**com.test.client.TestService**
4	testMethod
5	java.lang.String
6	java.lang.Integer
7	myInput1
8	java.lang.Integer/3438268394

GWT SERVICE CLASS

PARSING WALKTHROUGH

1 | 2 | 3 | **4** | 2 | 5 | 6 | 7 | 8 | 1 |

1	http://localhost:8080/test/
2	16878339F02B83818D264AE430C20468
3	com.test.client.TestService
4	**testMethod**
5	java.lang.String
6	java.lang.Integer
7	myInput1
8	java.lang.Integer/3438268394

GWT SERVICE METHOD

Parsing Walkthrough

1 | 2 | 3 | 4 | **2** | 5 | 6 | 7 | 8 | 1 |

1	http://localhost:8080/test/
2	16878339F02B83818D264AE430C20468
3	com.test.client.TestService
4	testMethod
5	java.lang.String
6	java.lang.Integer
7	myInput1
8	java.lang.Integer/3438268394

# OF METHOD PARAMETERS

PARSING WALKTHROUGH

1 | 2 | 3 | 4 | 2 | **5** | **6** | 7 | 8 | 1 |

1	http://localhost:8080/test/
2	16878339F02B83818D264AE430C20468
3	com.test.client.TestService
4	testMethod
5	**java.lang.String**
6	**java.lang.Integer**
7	myInput1
8	java.lang.Integer/3438268394

PARAMETER TYPES

PARSING WALKTHROUGH

1 | 2 | 3 | 4 | 2 | 5 | 6 | **7** | 8 | 1 |

1	http://localhost:8080/test/
2	16878339F02B83818D264AE430C20468
3	com.test.client.TestService
4	testMethod
5	java.lang.String
6	java.lang.Integer
7	**myInput1**
8	java.lang.Integer/3438268394

Fuzzible

READ FIRST PARAM VAL

PARSING WALKTHROUGH

1 | 2 | 3 | 4 | 2 | 5 | 6 | 7 | **8** | 1 |

1	http://localhost:8080/test/
2	16878339F02B83818D264AE430C20468
3	com.test.client.TestService
4	testMethod
5	java.lang.String
6	java.lang.Integer
7	myInput1
8	**java.lang.Integer/3438268394**

READ AN INTEGER VALUE

PARSING WALKTHROUGH

1 | 2 | 3 | 4 | 2 | 5 | 6 | 7 | 8 | **1**

Numeric Fuzzible Value

1	http://localhost:8080/test/
2	16878339F02B83818D264AE430C20
3	com.test.client.TestService
4	testMethod
5	java.lang.String
6	java.lang.Integer
7	myInput1
8	java.lang.Integer/3438268394

READ SECOND PARAM VAL

Parsing Walkthrough

- That was a very simple example
- Different Java types can be serialized
  - Primitive Java types and Objects
  - Arrays, Lists, Vectors..etc
  - Maps
  - Custom Objects

PARSING WALKTHROUGH

What if the following request was sent?

```
5|0|12|http://127.0.0.1:8888/gwt_test/|4E7583E4
BED25F58DDD5F1A1D675522A|com.gwttest.client.Tes
tService|testServer|java.util.ArrayList/3821976
829|com.gwttest.client.CustomObj/427743781|com.
gwttest.client.Person/2847577871|PersonName|jav
a.lang.Integer/3438268394|Joe
Shmoe|jshmoe@email.com|123456789|
1|2|3|4|2|5|6|5|2|7|200|8|7|200|8|6|9|200|10|11
|12|10|
```

FUZZING USER INPUT

- Fuzzing all pipe delimited values creates too much output
- The “smart” way to fuzz GWT requests is to identify user input and its data type
- Numeric values should not be tested for string related issues

GWTParse

- Python script which parses the GWT-RPC request and identifies user input
- Supports multiple forms of output so that results can be used with an existing fuzzer
- Source @ github.com/rongutierrez

GWTParse

```
Usage: gwtparse.py [options]

A tool for parsing GWT RPC Requests

Options:
  --version             show program's version number and exit
  -h, --help            show this help message and exit
  -p, --pretty          Output the GWT RPC Request in a human readable
format
  -s SURROUND_VALUE, --surround=SURROUND_VALUE
                        String used to surround all fuzzable values
  -r REPLACE_VALUE, --replace=REPLACE_VALUE
                        String used to replace all fuzzable values
  -b, --burp            Generates Burp Intruder Output
  -i RPC_REQUEST, --input=RPC_REQUEST
                        RPC Request Payload (Required)
  -w WRITE, --write=WRITE
                        Writes Fuzz String to a new output file
  -a APPEND, --append=APPEND
                        Appends Fuzz String to an existing output file
```

GWTParse Limitations

- Currently only supports the following types:
  - Primitive Java Types and Objects
  - Strings
  - Arrays, Arraylist, Vector, LinkedList
  - Custom Objects (to a limited extent)
- Only tested on serialization version 5

GWTPARSE + BURP INTRUDER DEMO

AUTOMATED FUZZING

The GWTParser class can be used as a stand-alone API

```
gwtparsed = GWTParser()

gwtparsed.deserialize( <GWT-RPC STRING> )

# Returns a list containing fuzzible indices in the GWT-RPC
String.
fuzzible_indices = gwtparsed.fuzzmarked
```

AUTOMATED FUZZING

- GDS Burp API by Marcin Wielgoszewski
- Parses Burp proxy logs into python objects
- Very useful for creating quick and dirty web application fuzzers
- Source @ mwielgoszewski.github.com/burpee

AUTOMATED FUZZING

- All GWT RPC requests are filtered from the Burp proxy log
- GWTParser identifies user input for fuzzing the filtered requests
- GWT requests can now be programmatically fuzzed.

AUTOMATED FUZZING

```
Usage: gwtfuzzer.py [options]

Automates the fuzzing of GWT RPC requests

Options:
  --version             show program's version number and exit
  -h, --help            show this help message and exit
  -b BURP, --burp=BURP  Burp logfile to fuzz
  -f FUZZFILE, --fuzzfile=FUZZFILE
                        File containing attack strings
  -e ERRORFILE, --errorfile=ERRORFILE
                        File containing error messages
  -o OUTPUT, --output=OUTPUT
                        Directory to store results
  -k COOKIES, --cookies=COOKIES
                        Cookies to use when requesting GWT RPC
                                        pages
  -p PROXY, --proxy=PROXY
                        Proxy Host and Port (e.g. -p
                        "http://proxy.internet.net:8080"
  -i IDRANGE, --idrange=IDRANGE
                        Range of decrements and increments to
                                        test parameter
manipulation with
```

GWTPARSE + GDS BURP API DEMO

FURTHER READING

- http://www.gdssecurity.com/l/b/2009/10/08/gwt-rpc-in-a-nutshell/
- http://www.gdssecurity.com/l/b/2010/05/06/fuzzing-gwt-rpc-requests/
- http://www.gdssecurity.com/l/b/2010/07/20/gwtenum-enumerating-gwt-rpc-method-calls/
- http://groups.google.com/group/Google-Web-Toolkit/web/security-for-gwt-applications?pli=1
- http://code.google.com/p/degwt/wiki/HowDeGWTWorks

Thanks for coming!

FIN