



# **The Web Application Hacker's Handbook**

**Discovering and Exploiting Security Flaws**

Dafydd Stuttard  
Marcus Pinto





# Contents

<b>Acknowledgments</b>	<b>xxiii</b>
<b>Introduction</b>	<b>xxv</b>
<b>Chapter 1 Web Application (In)security</b>	<b>1</b>
The Evolution of Web Applications	2
Common Web Application Functions	3
Benefits of Web Applications	4
Web Application Security	5
“This Site Is Secure”	6
The Core Security Problem: Users Can Submit Arbitrary Input	8
Key Problem Factors	9
Immature Security Awareness	9
In-House Development	9
Deceptive Simplicity	9
Rapidly Evolving Threat Profile	10
Resource and Time Constraints	10
Overextended Technologies	10
The New Security Perimeter	10
The Future of Web Application Security	12
Chapter Summary	13
<b>Chapter 2 Core Defense Mechanisms</b>	<b>15</b>
Handling User Access	16
Authentication	16
Session Management	17
Access Control	18
Handling User Input	19
Varieties of Input	20
Approaches to Input Handling	21

"Reject Known Bad"	21
"Accept Known Good"	21
Sanitization	22
Safe Data Handling	22
Semantic Checks	23
Boundary Validation	23
Multistep Validation and Canonicalization	26
Handling Attackers	27
Handling Errors	27
Maintaining Audit Logs	29
Alerting Administrators	30
Reacting to Attacks	31
Managing the Application	32
Chapter Summary	33
Questions	34
<b>Chapter 3</b>	<b>35</b>
<b>Web Application Technologies</b>	<b>35</b>
The HTTP Protocol	35
HTTP Requests	36
HTTP Responses	37
HTTP Methods	38
URLs	40
HTTP Headers	41
General Headers	41
Request Headers	41
Response Headers	42
Cookies	43
Status Codes	44
HTTPS	45
HTTP Proxies	46
HTTP Authentication	47
Web Functionality	47
Server-Side Functionality	48
The Java Platform	49
ASP.NET	50
PHP	50
Client-Side Functionality	51
HTML	51
Hyperlinks	51
Forms	52
JavaScript	54
Thick Client Components	54
State and Sessions	55
Encoding Schemes	56
URL Encoding	56
Unicode Encoding	57

HTML Encoding	57
Base64 Encoding	58
Hex Encoding	59
Next Steps	59
Questions	59
<b>Chapter 4    Mapping the Application</b>	<b>61</b>
Enumerating Content and Functionality	62
Web Spidering	62
User-Directed Spidering	65
Discovering Hidden Content	67
Brute-Force Techniques	67
Inference from Published Content	70
Use of Public Information	72
Leveraging the Web Server	75
Application Pages vs. Functional Paths	76
Discovering Hidden Parameters	79
Analyzing the Application	79
Identifying Entry Points for User Input	80
Identifying Server-Side Technologies	82
Banner Grabbing	82
HTTP Fingerprinting	82
File Extensions	84
Directory Names	86
Session Tokens	86
Third-Party Code Components	87
Identifying Server-Side Functionality	88
Dissecting Requests	88
Extrapolating Application Behavior	90
Mapping the Attack Surface	91
Chapter Summary	92
Questions	93
<b>Chapter 5    Bypassing Client-Side Controls</b>	<b>95</b>
Transmitting Data via the Client	95
Hidden Form Fields	96
HTTP Cookies	99
URL Parameters	99
The Referer Header	100
Opaque Data	101
The ASP.NET ViewState	102
Capturing User Data: HTML Forms	106
Length Limits	106
Script-Based Validation	108
Disabled Elements	110
Capturing User Data: Thick-Client Components	111
Java Applets	112

Decompiling Java Bytecode	114	
Coping with Bytecode Obfuscation	117	
ActiveX Controls	119	
Reverse Engineering	120	
Manipulating Exported Functions	122	
Fixing Inputs Processed by Controls	123	
Decompiling Managed Code	124	
Shockwave Flash Objects	124	
Handling Client-Side Data Securely	128	
Transmitting Data via the Client	128	
Validating Client-Generated Data	129	
Logging and Alerting	131	
Chapter Summary	131	
Questions	132	
<b>Chapter 6</b>	<b>Attacking Authentication</b>	<b>133</b>
	Authentication Technologies	134
	Design Flaws in Authentication Mechanisms	135
	Bad Passwords	135
	Brute-Forcible Login	136
	Verbose Failure Messages	139
	Vulnerable Transmission of Credentials	142
	Password Change Functionality	144
	Forgotten Password Functionality	145
	“Remember Me” Functionality	148
	User Impersonation Functionality	149
	Incomplete Validation of Credentials	152
	Non-Unique Usernames	152
	Predictable Usernames	154
	Predictable Initial Passwords	154
	Insecure Distribution of Credentials	155
	Implementation Flaws in Authentication	156
	Fail-Open Login Mechanisms	156
	Defects in Multistage Login Mechanisms	157
	Insecure Storage of Credentials	161
	Securing Authentication	162
	Use Strong Credentials	162
	Handle Credentials Secretively	163
	Validate Credentials Properly	164
	Prevent Information Leakage	166
	Prevent Brute-Force Attacks	167
	Prevent Misuse of the Password Change Function	170
	Prevent Misuse of the Account Recovery Function	170
	Log, Monitor, and Notify	172
	Chapter Summary	172

<b>Chapter 7</b>	<b>Attacking Session Management</b>	<b>175</b>
	The Need for State	176
	Alternatives to Sessions	178
	Weaknesses in Session Token Generation	180
	Meaningful Tokens	180
	Predictable Tokens	182
	Concealed Sequences	184
	Time Dependency	185
	Weak Random Number Generation	187
	Weaknesses in Session Token Handling	191
	Disclosure of Tokens on the Network	192
	Disclosure of Tokens in Logs	196
	Vulnerable Mapping of Tokens to Sessions	198
	Vulnerable Session Termination	200
	Client Exposure to Token Hijacking	201
	Liberal Cookie Scope	203
	Cookie Domain Restrictions	203
	Cookie Path Restrictions	205
	Securing Session Management	206
	Generate Strong Tokens	206
	Protect Tokens throughout Their Lifecycle	208
	Per-Page Tokens	211
	Log, Monitor, and Alert	212
	Reactive Session Termination	212
	Chapter Summary	213
	Questions	214
<b>Chapter 8</b>	<b>Attacking Access Controls</b>	<b>217</b>
	Common Vulnerabilities	218
	Completely Unprotected Functionality	219
	Identifier-Based Functions	220
	Multistage Functions	222
	Static Files	222
	Insecure Access Control Methods	223
	Attacking Access Controls	224
	Securing Access Controls	228
	A Multi-Layered Privilege Model	231
	Chapter Summary	234
	Questions	235
<b>Chapter 9</b>	<b>Injecting Code</b>	<b>237</b>
	Injecting into Interpreted Languages	238
	Injecting into SQL	240
	Exploiting a Basic Vulnerability	241
	Bypassing a Login	243
	Finding SQL Injection Bugs	244
	Injecting into Different Statement Types	247

The UNION Operator	251
Fingerprinting the Database	255
Extracting Useful Data	256
An Oracle Hack	257
An MS-SQL Hack	260
Exploiting ODBC Error Messages (MS-SQL Only)	262
Enumerating Table and Column Names	263
Extracting Arbitrary Data	265
Using Recursion	266
Bypassing Filters	267
Second-Order SQL Injection	271
Advanced Exploitation	272
Retrieving Data as Numbers	273
Using an Out-of-Band Channel	274
Using Inference: Conditional Responses	277
Beyond SQL Injection: Escalating the Database Attack	285
MS-SQL	286
Oracle	288
MySQL	288
SQL Syntax and Error Reference	289
SQL Syntax	290
SQL Error Messages	292
Preventing SQL Injection	296
Partially Effective Measures	296
Parameterized Queries	297
Defense in Depth	299
Injecting OS Commands	300
Example 1: Injecting via Perl	300
Example 2: Injecting via ASP	302
Finding OS Command Injection Flaws	304
Preventing OS Command Injection	307
Injecting into Web Scripting Languages	307
Dynamic Execution Vulnerabilities	307
Dynamic Execution in PHP	308
Dynamic Execution in ASP	308
Finding Dynamic Execution Vulnerabilities	309
File Inclusion Vulnerabilities	310
Remote File Inclusion	310
Local File Inclusion	311
Finding File Inclusion Vulnerabilities	312
Preventing Script Injection Vulnerabilities	312
Injecting into SOAP	313
Finding and Exploiting SOAP Injection	315
Preventing SOAP Injection	316
Injecting into XPath	316
Subverting Application Logic	317

Informed XPath Injection	318
Blind XPath Injection	319
Finding XPath Injection Flaws	320
Preventing XPath Injection	321
Injecting into SMTP	321
Email Header Manipulation	322
SMTP Command Injection	323
Finding SMTP Injection Flaws	324
Preventing SMTP Injection	326
Injecting into LDAP	326
Injecting Query Attributes	327
Modifying the Search Filter	328
Finding LDAP Injection Flaws	329
Preventing LDAP Injection	330
Chapter Summary	331
Questions	331
<b>Chapter 10 Exploiting Path Traversal</b>	<b>333</b>
Common Vulnerabilities	333
Finding and Exploiting Path Traversal Vulnerabilities	335
Locating Targets for Attack	335
Detecting Path Traversal Vulnerabilities	336
Circumventing Obstacles to Traversal Attacks	339
Coping with Custom Encoding	342
Exploiting Traversal Vulnerabilities	344
Preventing Path Traversal Vulnerabilities	344
Chapter Summary	346
Questions	346
<b>Chapter 11 Attacking Application Logic</b>	<b>349</b>
The Nature of Logic Flaws	350
Real-World Logic Flaws	350
Example 1: Fooling a Password Change Function	351
The Functionality	351
The Assumption	351
The Attack	352
Example 2: Proceeding to Checkout	352
The Functionality	352
The Assumption	353
The Attack	353
Example 3: Rolling Your Own Insurance	354
The Functionality	354
The Assumption	354
The Attack	355
Example 4: Breaking the Bank	356
The Functionality	356
The Assumption	357
The Attack	358



Example 5: Erasing an Audit Trail	359
The Functionality	359
The Assumption	359
The Attack	359
Example 6: Beating a Business Limit	360
The Functionality	360
The Assumption	361
The Attack	361
Example 7: Cheating on Bulk Discounts	362
The Functionality	362
The Assumption	362
The Attack	362
Example 8: Escaping from Escaping	363
The Functionality	363
The Assumption	364
The Attack	364
Example 9: Abusing a Search Function	365
The Functionality	365
The Assumption	365
The Attack	365
Example 10: Snarfing Debug Messages	366
The Functionality	366
The Assumption	367
The Attack	367
Example 11: Racing against the Login	368
The Functionality	368
The Assumption	368
The Attack	368
Avoiding Logic Flaws	370
Chapter Summary	372
Questions	372
<b>Chapter 12 Attacking Other Users</b>	<b>375</b>
Cross-Site Scripting	376
Reflected XSS Vulnerabilities	377
Exploiting the Vulnerability	379
Stored XSS Vulnerabilities	383
Storing XSS in Uploaded Files	385
DOM-Based XSS Vulnerabilities	386
Real-World XSS Attacks	388
Chaining XSS and Other Attacks	390
Payloads for XSS Attacks	391
Virtual Defacement	391
Injecting Trojan Functionality	392
Inducing User Actions	394
Exploiting Any Trust Relationships	394
Escalating the Client-Side Attack	396

Delivery Mechanisms for XSS Attacks	399
Delivering Reflected and DOM-Based XSS Attacks	399
Delivering Stored XSS Attacks	400
Finding and Exploiting XSS Vulnerabilities	401
Finding and Exploiting Reflected XSS Vulnerabilities	402
Finding and Exploiting Stored XSS Vulnerabilities	415
Finding and Exploiting DOM-Based XSS Vulnerabilities	417
HttpOnly Cookies and Cross-Site Tracing	421
Preventing XSS Attacks	423
Preventing Reflected and Stored XSS	423
Preventing DOM-Based XSS	427
Preventing XST	428
Redirection Attacks	428
Finding and Exploiting Redirection Vulnerabilities	429
Circumventing Obstacles to Attack	431
Preventing Redirection Vulnerabilities	433
HTTP Header Injection	434
Exploiting Header Injection Vulnerabilities	434
Injecting Cookies	435
Delivering Other Attacks	436
HTTP Response Splitting	436
Preventing Header Injection Vulnerabilities	438
Frame Injection	438
Exploiting Frame Injection	439
Preventing Frame Injection	440
Request Forgery	440
On-Site Request Forgery	441
Cross-Site Request Forgery	442
Exploiting XSRF Flaws	443
Preventing XSRF Flaws	444
JSON Hijacking	446
JSON	446
Attacks against JSON	447
Overriding the Array Constructor	447
Implementing a Callback Function	448
Finding JSON Hijacking Vulnerabilities	449
Preventing JSON Hijacking	450
Session Fixation	450
Finding and Exploiting Session Fixation Vulnerabilities	452
Preventing Session Fixation Vulnerabilities	453
Attacking ActiveX Controls	454
Finding ActiveX Vulnerabilities	455
Preventing ActiveX Vulnerabilities	456
Local Privacy Attacks	458
Persistent Cookies	458
Cached Web Content	458

Browsing History	459
Autocomplete	460
Preventing Local Privacy Attacks	460
Advanced Exploitation Techniques	461
Leveraging Ajax	461
Making Asynchronous Off-Site Requests	463
Anti-DNS Pinning	464
A Hypothetical Attack	465
DNS Pinning	466
Attacks against DNS Pinning	466
Browser Exploitation Frameworks	467
Chapter Summary	469
Questions	469
<b>Chapter 13 Automating Bespoke Attacks</b>	<b>471</b>
Uses for Bespoke Automation	472
Enumerating Valid Identifiers	473
The Basic Approach	474
Detecting Hits	474
HTTP Status Code	474
Response Length	475
Response Body	475
Location Header	475
Set-cookie Header	475
Time Delays	476
Scripting the Attack	476
JAttack	477
Harvesting Useful Data	484
Fuzzing for Common Vulnerabilities	487
Putting It All Together: Burp Intruder	491
Positioning Payloads	492
Choosing Payloads	493
Configuring Response Analysis	494
Attack 1: Enumerating Identifiers	495
Attack 2: Harvesting Information	498
Attack 3: Application Fuzzing	500
Chapter Summary	502
Questions	502
<b>Chapter 14 Exploiting Information Disclosure</b>	<b>505</b>
Exploiting Error Messages	505
Script Error Messages	506
Stack Traces	507
Informative Debug Messages	508
Server and Database Messages	509
Using Public Information	511
Engineering Informative Error Messages	512

Gathering Published Information	513
Using Inference	514
Preventing Information Leakage	516
Use Generic Error Messages	516
Protect Sensitive Information	517
Minimize Client-Side Information Leakage	517
Chapter Summary	518
Questions	518
<b>Chapter 15 Attacking Compiled Applications</b>	<b>521</b>
Buffer Overflow Vulnerabilities	522
Stack Overflows	522
Heap Overflows	523
“Off-by-One” Vulnerabilities	524
Detecting Buffer Overflow Vulnerabilities	527
Integer Vulnerabilities	529
Integer Overflows	529
Signedness Errors	529
Detecting Integer Vulnerabilities	530
Format String Vulnerabilities	531
Detecting Format String Vulnerabilities	532
Chapter Summary	533
Questions	534
<b>Chapter 16 Attacking Application Architecture</b>	<b>535</b>
Tiered Architectures	535
Attacking Tiered Architectures	536
Exploiting Trust Relationships between Tiers	537
Subverting Other Tiers	538
Attacking Other Tiers	539
Securing Tiered Architectures	540
Minimize Trust Relationships	540
Segregate Different Components	541
Apply Defense in Depth	542
Shared Hosting and Application Service Providers	542
Virtual Hosting	543
Shared Application Services	543
Attacking Shared Environments	544
Attacks against Access Mechanisms	545
Attacks between Applications	546
Securing Shared Environments	549
Secure Customer Access	549
Segregate Customer Functionality	550
Segregate Components in a Shared Application	551
Chapter Summary	551
Questions	551

<b>Chapter 17</b>	<b>Attacking the Web Server</b>	<b>553</b>
	Vulnerable Web Server Configuration	553
	Default Credentials	554
	Default Content	555
	Debug Functionality	555
	Sample Functionality	556
	Powerful Functions	557
	Directory Listings	559
	Dangerous HTTP Methods	560
	The Web Server as a Proxy	562
	Misconfigured Virtual Hosting	564
	Securing Web Server Configuration	565
	Vulnerable Web Server Software	566
	Buffer Overflow Vulnerabilities	566
	Microsoft IIS ISAPI Extensions	567
	Apache Chunked Encoding Overflow	567
	Microsoft IIS WebDav Overflow	567
	iPlanet Search Overflow	567
	Path Traversal Vulnerabilities	568
	Accipiter DirectServer	568
	Alibaba	568
	Cisco ACS Acme.server	568
	McAfee EPolicy Orcestrator	568
	Encoding and Canonicalization Vulnerabilities	568
	Allaire JRun Directory Listing Vulnerability	569
	Microsoft IIS Unicode Path Traversal Vulnerabilities	569
	Oracle PL/SQL Exclusion List Bypasses	570
	Finding Web Server Flaws	571
	Securing Web Server Software	572
	Choose Software with a Good Track Record	572
	Apply Vendor Patches	572
	Perform Security Hardening	573
	Monitor for New Vulnerabilities	573
	Use Defense-in-Depth	573
	Chapter Summary	574
	Questions	574
<b>Chapter 18</b>	<b>Finding Vulnerabilities in Source Code</b>	<b>577</b>
	Approaches to Code Review	578
	Black-Box vs. White-Box Testing	578
	Code Review Methodology	579
	Signatures of Common Vulnerabilities	580
	Cross-Site Scripting	580
	SQL Injection	581
	Path Traversal	582
	Arbitrary Redirection	583

OS Command Injection	584
Backdoor Passwords	584
Native Software Bugs	585
Buffer Overflow Vulnerabilities	585
Integer Vulnerabilities	586
Format String Vulnerabilities	586
Source Code Comments	586
The Java Platform	587
Identifying User-Supplied Data	587
Session Interaction	589
Potentially Dangerous APIs	589
File Access	589
Database Access	590
Dynamic Code Execution	591
OS Command Execution	591
URL Redirection	592
Sockets	592
Configuring the Java Environment	593
ASP.NET	594
Identifying User-Supplied Data	594
Session Interaction	595
Potentially Dangerous APIs	596
File Access	596
Database Access	597
Dynamic Code Execution	598
OS Command Execution	598
URL Redirection	599
Sockets	600
Configuring the ASP.NET Environment	600
PHP	601
Identifying User-Supplied Data	601
Session Interaction	603
Potentially Dangerous APIs	604
File Access	604
Database Access	606
Dynamic Code Execution	607
OS Command Execution	607
URL Redirection	608
Sockets	608
Configuring the PHP Environment	609
Register Globals	609
Safe Mode	610
Magic Quotes	610
Miscellaneous	611
Perl	611
Identifying User-Supplied Data	612

Session Interaction	613
Potentially Dangerous APIs	613
File Access	613
Database Access	613
Dynamic Code Execution	614
OS Command Execution	614
URL Redirection	615
Sockets	615
Configuring the Perl Environment	615
JavaScript	616
Database Code Components	617
SQL Injection	617
Calls to Dangerous Functions	618
Tools for Code Browsing	619
Chapter Summary	620
Questions	621
<b>Chapter 19 A Web Application Hacker's Toolkit</b>	<b>623</b>
Web Browsers	624
Internet Explorer	624
Firefox	624
Opera	626
Integrated Testing Suites	627
How the Tools Work	628
Intercepting Proxies	628
Web Application Spiders	633
Application Fuzzers and Scanners	636
Manual Request Tools	637
Feature Comparison	640
Burp Suite	643
Paros	644
WebScarab	645
Alternatives to the Intercepting Proxy	646
Tamper Data	647
TamperIE	647
Vulnerability Scanners	649
Vulnerabilities Detected by Scanners	649
Inherent Limitations of Scanners	651
Every Web Application Is Different	652
Scanners Operate on Syntax	652
Scanners Do Not Improvise	652
Scanners Are Not Intuitive	653
Technical Challenges Faced by Scanners	653
Authentication and Session Handling	653
Dangerous Effects	654
Individuating Functionality	655
Other Challenges to Automation	655

Current Products	656
Using a Vulnerability Scanner	658
Other Tools	659
Nikto	660
Hydra	660
Custom Scripts	661
Wget	662
Curl	662
Netcat	663
Stunnel	663
Chapter Summary	664
<b>Chapter 20 A Web Application Hacker's Methodology</b>	<b>665</b>
General Guidelines	667
1. Map the Application's Content	669
1.1. Explore Visible Content	669
1.2. Consult Public Resources	670
1.3. Discover Hidden Content	670
1.4. Discover Default Content	671
1.5. Enumerate Identifier-Specified Functions	671
1.6. Test for Debug Parameters	672
2. Analyze the Application	672
2.1. Identify Functionality	673
2.2. Identify Data Entry Points	673
2.3. Identify the Technologies Used	673
2.4. Map the Attack Surface	674
3. Test Client-Side Controls	675
3.1. Test Transmission of Data via the Client	675
3.2. Test Client-Side Controls over User Input	676
3.3. Test Thick-Client Components	677
3.3.1. Test Java Applets	677
3.3.2. Test ActiveX controls	678
3.3.3. Test Shockwave Flash objects	678
4. Test the Authentication Mechanism	679
4.1. Understand the Mechanism	680
4.2. Test Password Quality	680
4.3. Test for Username Enumeration	680
4.4. Test Resilience to Password Guessing	681
4.5. Test Any Account Recovery Function	682
4.6. Test Any Remember Me Function	682
4.7. Test Any Impersonation Function	683
4.8. Test Username Uniqueness	683
4.9. Test Predictability of Auto-Generated Credentials	684
4.10. Check for Unsafe Transmission of Credentials	684
4.11. Check for Unsafe Distribution of Credentials	685



4.12. Test for Logic Flaws	685
4.12.1. Test for Fail-Open Conditions	685
4.12.2. Test Any Multistage Mechanisms	686
4.13. Exploit Any Vulnerabilities to Gain Unauthorized Access	687
5. Test the Session Management Mechanism	688
5.1. Understand the Mechanism	689
5.2. Test Tokens for Meaning	689
5.3. Test Tokens for Predictability	690
5.4. Check for Insecure Transmission of Tokens	691
5.5. Check for Disclosure of Tokens in Logs	692
5.6. Check Mapping of Tokens to Sessions	692
5.7. Test Session Termination	693
5.8. Check for Session Fixation	694
5.9. Check for XSRF	694
5.10. Check Cookie Scope	695
6. Test Access Controls	696
6.1. Understand the Access Control Requirements	696
6.2. Testing with Multiple Accounts	697
6.3. Testing with Limited Access	697
6.4. Test for Insecure Access Control Methods	698
7. Test for Input-Based Vulnerabilities	699
7.1. Fuzz All Request Parameters	699
7.2. Test for SQL Injection	702
7.3. Test for XSS and Other Response Injection	704
7.3.1. Identify Reflected Request Parameters	704
7.3.2. Test for Reflected XSS	705
7.3.3. Test for HTTP Header Injection	705
7.3.4. Test for Arbitrary Redirection	706
7.3.5. Test for Stored Attacks	706
7.4. Test for OS Command Injection	707
7.5. Test for Path Traversal	709
7.6. Test for Script Injection	711
7.7. Test for File Inclusion	711
8. Test for Function-Specific Input Vulnerabilities	712
8.1. Test for SMTP Injection	712
8.2. Test for Native Software Vulnerabilities	713
8.2.1. Test for Buffer Overflows	713
8.2.2. Test for Integer Vulnerabilities	714
8.2.3. Test for Format String Vulnerabilities	714
8.3. Test for SOAP Injection	715
8.4. Test for LDAP Injection	715
8.5. Test for XPath Injection	716
9. Test for Logic Flaws	717
9.1. Identify the Key Attack Surface	717
9.2. Test Multistage Processes	718
9.3. Test Handling of Incomplete Input	718

9.4. Test Trust Boundaries	719
9.5. Test Transaction Logic	719
10. Test for Shared Hosting Vulnerabilities	720
10.1. Test Segregation in Shared Infrastructures	720
10.2. Test Segregation between ASP-Hosted Applications	721
11. Test for Web Server Vulnerabilities	721
11.1. Test for Default Credentials	722
11.2. Test for Default Content	722
11.3. Test for Dangerous HTTP Methods	722
11.4. Test for Proxy Functionality	723
11.5. Test for Virtual Hosting Misconfiguration	723
11.6. Test for Web Server Software Bugs	723
12. Miscellaneous Checks	724
12.1. Check for DOM-Based Attacks	724
12.2. Check for Frame Injection	725
12.3. Check for Local Privacy Vulnerabilities	726
12.4. Follow Up Any Information Leakage	726
12.5. Check for Weak SSL Ciphers	727

**Index****729**