Web Application Security & the Mobile World

OWASP

Mikko Saario
Founder of Finland (Helsinki) OWASP Chapter

Copyright © The OWASP Foundation
Permission is granted to copy, distribute and/or modify this document under the terms of the OWASP License.

The OWASP Foundation
http://www.owasp.org
Why Mobile Web Security?

- Where’s the beef?
- History & background
- Converging landscape
- The road ahead
OWASP in Finland

- Chapter created late 2006
- Vibrant community
  - Co-operation with e.g. general IT & development communities
  - Networking
- One corporate member (😊)

Contents [hide]

1 OWASP Helsinki Local Chapter
2 Local News
3 Current Activities
4 OWASP Helsinki Chapter Meeting #9: April 2009
5 OWASP Helsinki Chapter Meeting #8: March 2009
6 OWASP Introduction to startup firms: Thursday, January 15th 2009
7 OWASP Helsinki Meeting #7: Tuesday, November 11th 2008
8 OWASP Helsinki Web Hacking Workshop, Tuesday, September 10th 2008
9 OWASP Goes! CERT-FI, Thursday, June 12th 2008
10 OWASP Helsinki Chapter meeting/Get Together #6 Tuesday, May 13th 2008
11 OWASP Helsinki Chapter meeting #5 Tuesday, March 11th 2008
12 OWASP Helsinki & RWSUG Seminar Tuesday, January 29th 2008
13 OWASP Helsinki Introduction to ISACA Finland Thursday, January 24th 2008
14 OWASP Helsinki meeting #4 Fall 2007 with Mark Curfey, Tuesday, October 2, 2007
17 OWASP Helsinki meeting #1, Tuesday, Dec 12, 2006 at Ernst & Young

A1 - Cross Site Scripting (XSS)
(Haitallisten komentojen välittäminen käyttäjän selaimen toisen sivuston kautta)

Johdanto

Huom! Tämä sivu on täysin työn alla. Tämän jälkeen osallistua kaikki.

Tavoitteet

OWASP Top 10:n tärkein tavoite on kouluttaa sovelluskehittäjiä, suunnitellakseen ja toteuttamaan tarjotoja perusmenetelmät nälöä haavoittuvuusilta suojautumiseen - loistava etu.

A2 - Injektio-ongelmat
(Haitallisen komennon välittäminen syötteessä osaksi taustajärjestelmäkyselyä)

Injektio-ongelmat, erityisesti SQL-injektio, ovat tyyppillisä web-sovelluksissa. Injektio onnistuu, kun käyttäjän antama syöte istutetaan suoraan osaksi taustajärjestelmäkomentoja tai tietokantakyselyä. Tämä mahdollistaa hyökkääjälle komentojen ajamisen tai tiedon muuttamisen taustajärjestelmässä.
Vive la Différence!

- Battery power
  - Limits CPU, screen
- Physical size & form
  - Input
  - Screen
  - Battery
  - Features
    - CPU+GPU, GPS, Wi-Fi, Camera, Networks, FM Radio ...
- Connectivity

Where’s the Beef?

- Personal information
  - Calendar, contacts
  - Phone calls, text messages, bdays & shopping lists
- Financial information
  - ‘Wallet’, credit card or other such data
- Location information
  - GPS, Wi-Fi, BT, etc tracking
- Your Company Files
  - Locally stored data
  - Remotely accessed data
- Device has a value of its own
Somebody’s Always Trying

Fake WAP browser (Trojan)

Malware: Records voice & SMS, access via a www service

Telnet to root

http://www.talkandroid.com/364-g1-root-access-pterminal/

1. Turn on your phone’s WiFi. This gives your phone an IP you can reach it at.
2. Get to a command prompt on your device by using the PTerminal application from the Android Market. (adb shell does not seem to work with these instructions, telnetd does not start up)
3. cd system
4. cd bin
5. telnetd
6. netstat (get your phones IP)
7. telnet into your phone’s IP from your computer
8. you now have root!

http://www.f-secure.com/v-descs/flexispy_a.shtml
Mobile Threats: Data

- Trojans, worms, viruses
  - Distribution via SMS/MMS messages, bluetooth etc.
    - Malicious apps calling to premium phone numbers
    - Stealing information
  - Typically may require user consent
- Platform/system attacks
  - Typically some type of privilege elevation
  - Cracking into something
  - Do stuff someone does not want you to do
Mobile Threats: Usage

■ User
  ▸ Malicious apps / Trojans

■ Small is beautiful
  ▸ Power management
    ▪ Does the app keep polling something?
  ▸ Cost management
    ▪ Pay per use connections
  ▸ Memory management
    ▪ Execution & storage sizes
      – Content, Caches, etc.

■ Denial of Service
  ▸ Nasty stuff like malformed SMS/Email message that kills the phone
    ▪ Or a jpg doing the same thing
Mobile Threats: Device

- SIMLock cracks
  - Remove the operator lockdown
- Alter IMEI identification code
  - Easier to resell stolen or unlocked device
- Theft
  - $$$
- Breakage
  - Who do you allow to fix it?
Mobile Threats: Offline Attacks

- Have the ‘capabilities’ you can’t have
  - Might be able to read/write/modify

- Similar to booting a Win machine with Linux
  - Which security model?

- Compromising
  - Confidentiality
  - Integrity
Mobile Security: System

(Ok ok, a bit of a Symbian-skewed view)

- Secure HW base + boot
  - Identity

- Trusted Computing Base: OS core
  - User has no/limited access to the core OS
    - Caging system data
  - Trust models limiting API & file access
    - capabilities [Symbian-speak] / privileges

(By Regan Coleman, Dr. Dobb's Journal)
Mobile Security: Usage

- Tightly controlled installation of applications
  - Controlled availability of signing certificates
    - ‘Manufacturer’, ‘Developer’, etc. certificates
  - Controlled availability of rights in the system

- Tightly controlled usage
  - SIM locks
    - Operator substitutions
  - Potentially limited application availability or access
    - Access only to certain services or apps
“Sometimes, it’s not that techy”

protected, using a very simple trick gives anyone full access to your cellphone private information in Mail, SMS, Contacts, and even Safari. The two-step trick is even simpler to the one used in the past to gain access to the phone to install unlocking cards or jailbreak.

First, password protect your phone and lock it. Then slide to unlock and do this:

1. Tap emergency call.

2. Double tap the home button.

Done. You are now in your favorites. This seems like a feature,

If you click in a mail address, it will give you full access to the Mail application. All your mail will be exposed.

If there's a URL in your contact (or in a mail message) you can click on it and have full access to Safari.

If you click on send text message in a contact, it will give you full access to all your SMS.
What Do They Have in Common?

Pictures are from respective vendor websites.
But, first, a quick look back!
Web Access Creeping In (Early Days)

- There was this thing called WAP + WML
  - Wireless Application Protocol + Wireless Markup Language
    - Index.html + index.wml
  - ‘Limited Success’ (to be politically correct)
  - Most ‘notable’ security issue was the Wap Gap

- First mobile browsers
  - Limited by
    - Compatibility
    - Speed and cost of network access
    - System resources and capabilities
  - Very small target base
Web Technology Usage Increases

- Started with GPRS, but a hit with 3G and WLAN
  - Run 10Gb at home and 64kb on mobile...? Naah
- HTTP, XML, Web Services utilized by client apps
  - Apps talking to a backend via web technology
  - Native or java clients
- Lots of existing applications
  - Web browser is a standard, basic app
- Pretty much any mobile now
  - Even many of the ‘low-end’ devices
- A whole lot of new users are entering the web
  - First user experience may come via a mobile
...And Related Security Issues

- Attacks against platform
  - Still *quite* proprietary
    - Substitution attacks
    - Various elevation of privilege attacks
    - Protocol weaknesses (e.g. file format)
    - Java implementation etc.

- Attacks against the application or the backend server
  - Protocol implementation (e.g. http, roap)
  - Intercepting and modifying XML contents

- Web attacks more difficult than in PC
  - Tightly controlled platform & HW
  - Proxy usage may differ from PC
And Once You Have a Fix

- You need to distribute it
  - Over The Air
  - Firmware updates, reflash the device
  - Package management, linux-based OS
  - Manually installable updated apps

- User awareness & ability
Adding Web Runtime & Widgets

■ WRT engine
  ▪ Browser-less support for web technologies
    ▪ XHTML, JavaScript, CSS, ...

■ Widget
  ▪ Requires a widget host engine (app), or
  ▪ Standalone offline local app

■ Easy to create
  ▪ Lots of web developers

■ A variety of widget engines exist
  ▪ Lack of firm standardization

http://www.slideshare.net/dap@quist/nokia-webruntime-presentation-phong-vu-presentation
A Happy Marriage?

http://www.slideshare.net/dappelquist/nokia-webruntime-presentation-phong-vu-presentation
Another Way of ‘Webification’

- Nokia Mobile Web Server
  - Personal ‘Mobisites’
- Apache web server ported to Symbian & S60 OS
  - Run your own web server from the mobile
- Designed for a gateway service
- So, do you need a waf for the phone as well? 😊

Original: [www.modsecurity.org](http://www.modsecurity.org) (hope Ivan won’t get upset); strictly ‘gimpware’
‘Webifying’ the Mobile Eco-System

- Use web technology to build the operating system
  - WebKit-based framework for web & apps

- Upcoming Palm Pre
  - Full ‘WebOS’
    - Utilizes web technology in the OS UI level
      - HTML5, JavaScript, CSS
      - ‘Native’ apps
    - You still e.g. need to sign apps
User Perspective: Browsing vs. Widget

- The User is likely comfortable with browsing the internet
  - But problems even understanding URLs
- Widget is locally-installed
  - More ‘personal’
  - Hidden connections
Web Code Accessing Device APIs

- Reliable app identification (AuthN, AuthZ)
  - Installation & execution
- Discovering the intent
  - Granting permissions
- Reliable isolation from others
  - Processes, APIs, stored or cached data
- Reaction to misbehavior
  - What can be done
Example APIs: BONDI

- ‘Under construction’
  - Least common denominator
- APIs (JavaScript)
  - Application Invocation
  - Messaging
  - User Interaction
  - File IO
  - Gallery
  - Device Status
  - System Events
  - Application Settings
  - Location
  - Camera
  - Communications Log
  - PIM
Governing the Permissions

- Separate authentication & authorization
- Moving towards a policy approach
  - Separate Policy (Permissions) from Mechanism (e.g. API)
    - Some access has required e.g. manufacturer / operator signing
  - Keep the framework as a framework
    - And put permissions in definable policies
- Certain dimensions missing from ‘PC’
  - Operators & variants
Governing the Permissions

- AuthN: e.g. origin-based vs. signature-based
- AuthZ: e.g. allow only trusted services privileged access?
  - If coming from URI ‘n’, allow access to API ‘z’?
  - If ‘x’-signed code (DN=x) and SSL cert ‘y’, allow...
- Declaring required permissions
  - a la AndroidManifest.xml
    - android.permission.CALL_EMERGENCY_NUMBERS
Governing the Permissions

- Permission lifespan could be
  - One-time (‘consume’ immediately)
  - Session
  - Permanent

- And managed at
  - Installation-only (e.g. Android)
  - During process execution
    - E.g. by prompting user, via policy
Multiple/Cross-Origin

Need to run code or retrieve content from multiple sources (cross-origin)
  - UI integration of various data sources

Managing the trust
  - How does the web app handle trust
    - Trust everything coming from a ‘trusted’ source?

Cross-document messaging
  - Communication between documents

Implementors are urged to take extra care in the implementation of this feature. It allows authors to transmit information from one domain to another domain, which is normally disallowed for security reasons. It also requires that UAs be careful to allow access to certain properties but not others.

http://dev.w3.org/html5/spec/Overview.html
Offline Web & Persistent Storage

- Application cache (HTML5)
  - Offline content
- Persistent data storage (e.g. HTML5, Gears)
  - Beyond cookies
    - Session, local, DB
  - Large db sizes
Could These Things Happen IRL?

Black Hat: Google Gears Offline Data Vulnerable

Google defends its product after a demonstration of a Web service-based attack using a cross-site scripting vulnerability.

By Thomas Claburn
InformationWeek
February 19, 2009 05:05 PM

The emergence of Web applications that function offline through technologies like Google Gears brings with it new risks: server-side attacks that can access client-side data.

In a presentation at the Black Hat conference in Washington, D.C., on Wednesday, Michael Sutton, VP of search research for Zscaler, demonstrated how a Google Gears-enabled Web service called Paymo.biz could be attacked using a cross-site scripting (XSS) vulnerability so that data stored in a user’s local Google (NASDAQ: GOOG) Gears database could be accessed or altered.

And no matter how responsive Web sites are to security problems that get reported, the overall problem remains. "Both Gears and HTML5 Database Storage leverage client-side JavaScript to create and interact with local databases," Sutton said in a blog post on Thursday. "Therefore, if an XSS vulnerability is present, it's all too easy for an attacker to compromise the confidentiality and integrity of locally stored data by reading from or writing to the local database."

• Implementing a secure technology on an insecure site invalidates the built in protections

System requirements
- Windows XP/Vista
- Firefox 1.5+ and Internet Explorer 6.0+

Gears is available for Windows, Windows Mobile (IE Mobile, Opera Mobile), Mac (Firefox, Safari), Linux, and Android.


http://gears.google.com/

The Road Ahead

- How to satisfy both the Mobile & PC-based needs?
- Common framework(s) in the works
  - Robust implementation
- W3C has a various working groups
What About Mobile Web App Developers?

- Most PC apps are web-based or web-enabled
  - Toolkits and platforms more mature
- Security in mobiles is not a new thing
  - Security understanding
- Shift in technologies
  - The implementation is changing
  - Different types of (& emerging) attack vectors
    - E.g. client side SQL injection, but even XSS
      - Anyone utilizing e.g. various local storage features
  - Test efforts need to evolve & keep up
- Reusing common, well-known standards
A Common Approach

- Initiatives
  - Such as, OMTP (BONDI), OpenAJAX, W3C, ...
- OWASP is key source for web app security
  - De facto, but obviously PC/app-minded
  - Intrinsic Security Working Group
Why Is All This Important?

- Unified approach to:
  - Protect the Consumer / User
    - While allowing a good consumer experience
    - Keeping personal property safe
  - Protect the Business Ecosystem
    - Operators
    - Manufacturers
    - Content providers
    - Service providers
  - Protect the Enterprise
    - Increased usage for business purposes
Thank You! A Dank!

OWASP