PHONEGAP DAY US - JANUARY 28, 2016

Adobe PhoneGap

PHONEGAP

PRIVACY AND SECURITY

TOMMY-CARLOS WILLIAMS (AKA: DEVGEEKS)

- Snr Computer Scientist at Adobe
- @theRealDevgeeks
- https://github.com/devgeeks
- http://blog.devgeeks.org
- tommy@devgeeks.org

LET ME JUST START BY SAYING:

HAPPY DATA PRIVACY DAY!

AND ON THAT NOTE:

What is the difference between “Security” and “Privacy”?

SECURITY AND PRIVACY GO HAND IN HAND 👬

- One does not always guarantee the other
- Good security does not always keep data private
- Privacy is also a security issue (attack vectors, etc)

STRONG PRIVACY NEEDS SECURITY

- Without good security, Privacy cannot be protected from those with malicious intent

TO QUOTE TROY HUNT (VIA JEREMIAH GROSSMAN):

"HACK YOURSELF FIRST"

Fun resource: http://hackyourselffirst.troyhunt.com

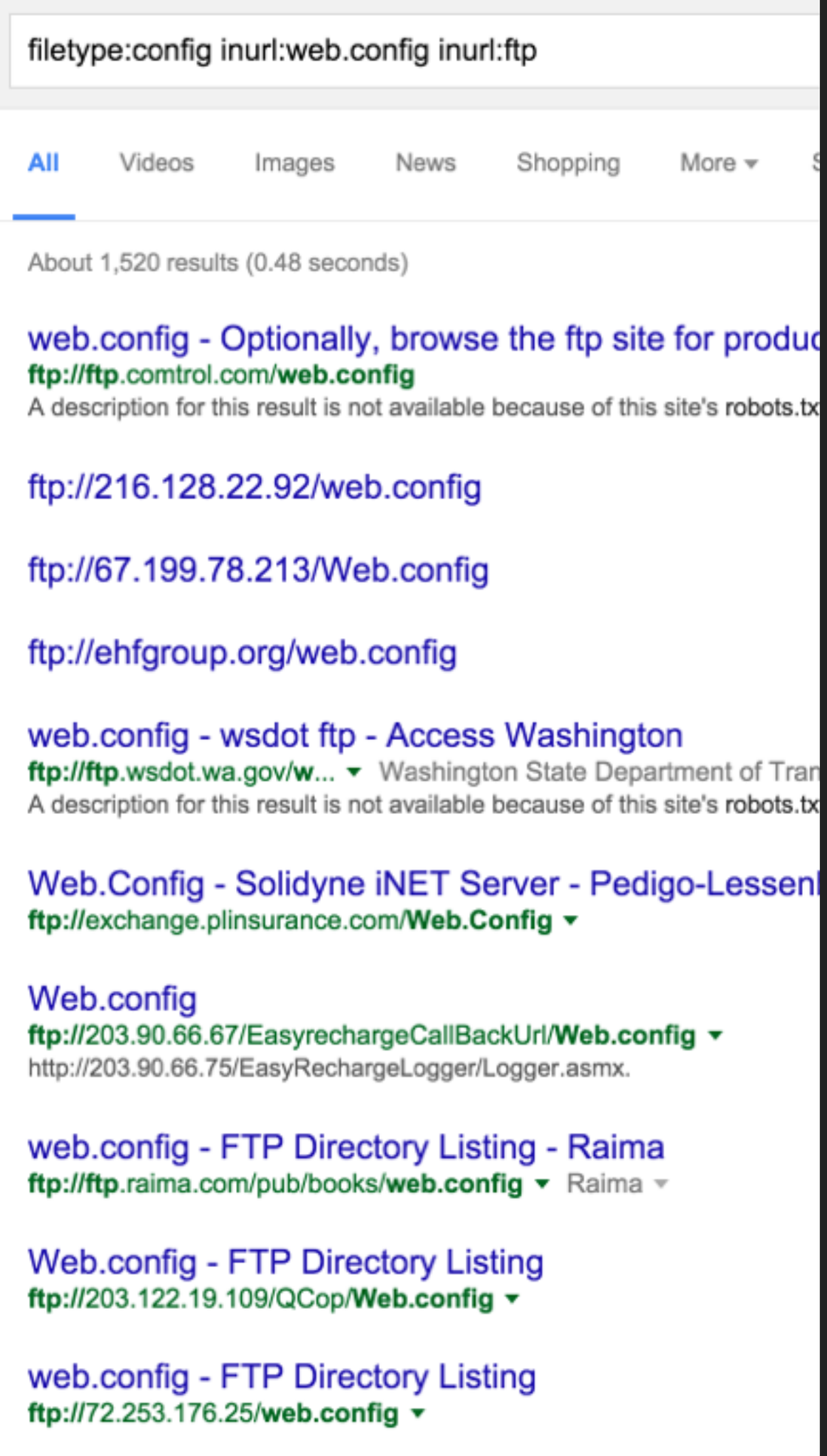
PRETTY MUCH THE SAME AS THOSE USED TO ATTACK BROWSER APPS

- Direct API or Server Access
- Cross Site Scripting (XSS)
- Cross Site Request Forgery (CSRF)
- SQL Injection
- MitM
- Broken Auth and Session Management
- User Exploitation (Phishing, etc)
- Etc…

IS HACKING HARD? DOES IT TAKE MAD SKILLZ?

AS WE SAY IN AUSTRALIA: "YEAH, NAH. . . "*

* SADLY AND CONFUSINGLY, THAT MEANS "NO"



GOOGLE DORKS

WHO ARE YOU CALLING A DORK…
YA DORK?

NOT GOING TO BOTHER DEMOING THIS, YOU GET THE IDEA

PLENTY OF HACKING TOOLS

DON'T REQUIRE ADVANCED KNOWLEDGE

OK, SO WHAT ARE WE GOING TO HACK ON TODAY?

PIXFOR
(THE VULNERABLE EDITION)

RUNNING THE APP

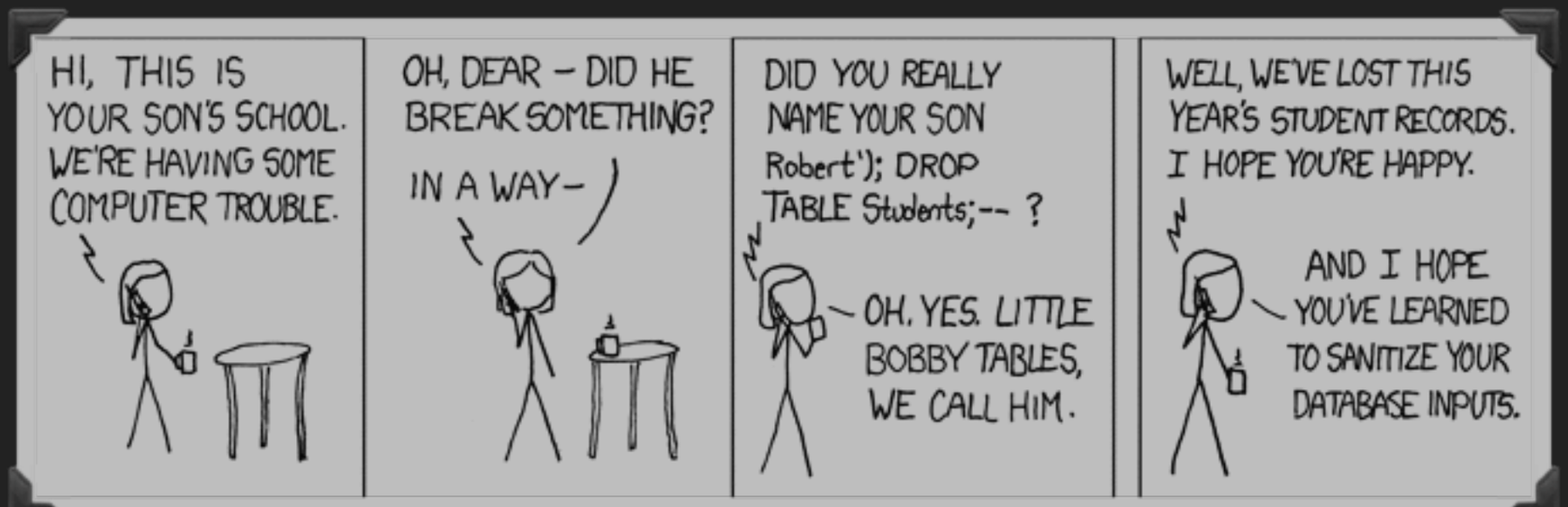
- $ git clone https://github.com/devgeeks/pixfor-vulnerable
- $ cd pixfor-vulnerable
- $ npm install
- $ phonegap serve (and use the PhoneGap Developer App), or
- $ phonegap run [ios|android] [--device]

~ or ~

- Open the PhoneGap Developer App and point it at:
  http://take.pixfor.me:8888
- Source on the USB sticks the TAs have…

SQL INJECTION

' OR 1=1; --



LITTLE BOBBY TABLES

ROBERT'; DROP TABLE STUDENTS; --

STOP!

DEMO TIME

SQL INJECTION

WHAT CAN WE DO?

IF NOTHING ELSE

- Avoid SQL injection by use of parameterisation
  - This keeps untrusted input from breaking out of the parameter context
- And of course, listen to Bobby's mom and sanitise your inputs

CROSS SITE SCRIPTING

XSS

BUT IS XSS THAT BIG OF A DEAL?

HTTPS://WWW.SEANCASSIDY.ME/LOSTPASS.HTML

ಠ_ಠ

TWO PRIMARY CONCERNS

- What are the sources of input?
- Where is this data going (the target)?

TYPES OF SOURCES

- Location and URL Sources
- Cookies
- Referrer (less so in a PhoneGap app)
- Window Name
- Indirect Sources (Client side db such as sqlite/pouch/etc)
- Other Objects (Post Message/Intents/etc)

TYPES OF TARGETS

- Execution Target
- HTML Element Target
- Set Location Target
- Control Flow Target
- ...and more

EXECUTION TARGET

- eval()
- onclick, onsubmit, ...
- Function()
- script.src
- setTimeout()
- script.text
- setInterval()
- script.textContent
- setImmediate()
- script.innerText
- etc...

HTML ELEMENT TARGET

- document.write
- document.writeln
- element.innerHTML
- element.insertAdjacentHTML
- Range.createContextualFragment
- HTMLButton.value
- etc…

SET LOCATION TARGET

- window.location

CONTROL FLOW TARGET

- this[foo](bar)

(seems contrived, but allows arbitrary script execution if input `foo` is untrusted)

IT'S PROBABLY TIME FOR A

DEMO

A QUICK NOTE ABOUT

FRAMEWORKS

FRAMEWORKS

- jQuery
- Angular
- React

FRAMEWORKS – JQUERY

- jQuery is practically a target in and of itself (mostly fixed)
  - be aware that old versions might be targets
- Choc-full of HTML element targets:
  element.add(userContent)
  element.append(userContent)
  element.after(userContent)
  element.html(userContent)
  etc…

FRAMEWORKS – ANGULAR

- Old versions should be avoided and updated
  - 1.1.5?

```
<div class=“ng-app">
  {{constructor.constructor('alert(1)')()}}
</div>
```

- Fixed now, but beware older versions

FRAMEWORKS – REACT

- Gives a clue in the function name: dangerouslySetInnerHTML(), but devs still use it (static site generators, etc)
- Not using it also doesn’t alleviate the need to sanitise your inputs

PROBLEMS WITH FRAMEWORKS

- Add complexity
- Abstract away targets (innerHTML, etc)
- Add syntactic sugar
- Add loopholes to browser security controls
- The frameworks and practices you use should (attempt to) be secure by default
- If your frameworks add syntactic sugar, be aware of the implications

STAY. UP. TO. DATE.

- This applies to PhoneGap/Cordova and its plugins as much as your front-end JavaScript frameworks

XSS

WHAT CAN WE DO?

MINIMISE ATTACK SURFACE

- Avoid converting strings to scripts
- Avoid innerHTML wherever possible!
- Don't write your own HTML sanitiser (srsly)
- Whitelist*
- Content Security Policy (CSP)*

* we'll get to these in a bit

AVOID CONVERTING STRINGS TO SCRIPTS

- eval, Function.apply, setTimeout("string"), etc
- inline event handlers like onclick="string", etc

AVOID INNERHTML WHEREVER POSSIBLE!

- .textContent
- $(el).text()
- document.createElement/setAttribute
- Use a template system with escaping
  - HOWEVER!! location targets are not as protected
    i.e.: `<a href="{{value}}">...</a>`

DON'T WRITE YOUR OWN HTML SANITISER

If you MUST...

- Whitelist, NOT blacklist
- fail conservatively, better to fail to display nicely than to be insecure
- instead consider: DOMPurify, Angular's $sanitize, Bleach,js (for workers?), etc...

TL;DR

- Avoid eval & innerHTML
- Use a template lang* with escaping, but be careful with attributes
- filter HTML input conservatively
- Whitelist / CSP

* However, a lot of the tempting langs don’t play well CSP, but we’ll get to that…

NEXT, MAYBE SOME FUN TO MAKE SURE

YOU ARE AWAKE

DO YOU RECOGNISE ANY OF THESE?

#SFO FREE WIFI, 13WestMyrtle, 2WIRE012, 5099251212, @Hyatt-WiFi, @yvrairport, ACU, ADBEEmp2014, ADO, ATT2yrd6rC, AccessDenied, Admirals_Club, AdobeCorp, AdobeGuest, Aer_Lingus_WIFI, Amanda's iPhone, AmtrakConnect, AndroidAP, Avatar Hotel, Avcenter, BDLpublic, BELL647, BELL_WIFI, BERNIES CAFE, BWW-PUBLIC, Best Western Park Place, BestBuy, Boingo Hotspot, Boyd's iPhone 6 Plus, Bycen, CAFE ZUPAS, CORTECH_Guest, CSWireless5ghz, Cafe 300, CapNet, CenturyLink1499, Cl-Wireless, CoJPublic, CoxWiFi, D&B_Guest, DIMTER, DIRECT-6bM2020 Series, DVG-N5402SP-212017, Detroit Airport Wi-Fi, DevMountain, DevMountainApt7, Douglas Guest, DrupalCon, ETS, El Mexsal, EmployeeHotspot, Engedi, EuropaCoffeehouse, FairPublic, Fly-Fi, FourSeasons Guest, Frahmbo's iPhone, Free PHX Boingo WiFi, FullCircle, Fusion-IO Guest, Google Starbucks, GoogleGuest, HI Express Richfield, HI Express Richfield , HOME-6CF2, HOME-C4C8, Handlery_San_Francisco, Happy Campers, HarborLink - Buffalo Wild Wings, Hope Alliance, Hyatt, IMEG-GUEST, ITGUEST, Jerk Grill, Joss, Kimpton, KingMaint, LYNDA-GUEST, Learntoprogram, MATHIS, MATHIS2, MH_Network, MMM_WiFi_Guest, MPLS, MiFi4620L Jetpack B472 Secure, MokiGuest, Mothership-guest, NETGEAR-Guest, NETGEAR53, NETGEAR82, NS FCCLA, OMA-Free-WiFi, OPTUSDN368CFC, OceanWiFi Very Hotspot, Ocho, Oscars, PCMC_Ice_Free_Wifi, PDI-Guest, PIGS, PPS, Park City Ice Patron Wifi, PhoneGap, Public, Quality, Quantum, Rain-Guest, Rangle.io WiFi, ReactWeek, Reclaim_EC, Redtail, Rogers, SDC2014, SEATAC-FREE-WIFI, SFUNET-SECURE, SGMC, SIN, SKYHARBOR PUBLIC WLAN, Seabay, Selnate, Shazron's iPhone 5, Shopify Guests, Solid Attendees, SouthwestWiFi, SpencerWireless, SpringOne2GX, Starbucks WiFi, Stratus018222, Streamyx Mobility, T-Mobile Broadband 67, TJR 008, TPCG0, Taco Bell, The Hotel Collection Guest WiFi, Two Jacks Pizza, U Street Cafe Wifi, UConnect, UNITE-295E, Virgin Hotels, WL, WebsterRec, Willowcreek, Wolverine-WiFi, WorkbarGuest, activehit, appigo47, attwifi, attwifibn, bPhone, baker, bycen, cabinsusa, dd-wrt, dlink, doraemon, duece, elive, ethostream, fazidin2, gogoinflight, hhonors, hhonors lobby, hhonors_lobby, hills, houseofnuts, iPhone (2), indiapalace, intermountain_guest, iviejuicebar, jexus, jucienjava, juicenjava, lhm-open, loganwifi, mikes, mywifi, nortel-wlan, qds_office, rangleio Extension, raspberry-pi, reSETguest, rogers, silvermtnsuites6, surveillance vehicle 2, testnet, video2g, wguest, xfinitywifi, zulkefley13@unifi

YOUR DEVICES ARE ALWAYS LOOKING FOR WIFI THEY KNOW

- These are called “Probe Requests”
- If the WiFi was unencrypted (no WEP/WPA, etc), then another device could see these probe requests and simply pretend to be that WiFi (i.e.: A WiFi Pineapple, etc)
- Most devices would then happily join and start sending traffic through the malicious WiFi spot
- Even easier is a “honey pot” like a WiFi called “FREE WIFI”, etc. Everyone likes free WiFi.

THESE ARE JUST SOME OF THE MANY WAYS TO BECOME A:

MAN IN THE MIDDLE (MITM)

I THINK THIS CALLS FOR ANOTHER

DEMO

MITM

WHAT CAN WE DO?

YES

ALL OF THEM

TOO HARD? TOO EXPENSIVE?

LET'S ENCRYPT!

CONTENT SECURITY POLICY

CSP

CSP IS AWESOME

- What is it?
  - It’s a whitelist of content sources
  - http://www.html5rocks.com/en/tutorials/security/content-security-policy
- Cordova / PhoneGap “hello World” templates include a CSP

HELLO WORLD

```
<meta http-equiv="Content-Security-Policy"

content="default-src 'self' data: gap:
https://ssl.gstatic.com;

style-src 'self' 'unsafe-inline';

media-src *">
```

OK, LET'S GET MORE

PHONEGAP SPECIFIC

CORDOVA-PLUGIN-WHITELIST

WHITELIST

READ AND UNDERSTAND THE WHITELIST GUIDE

- http://cordova.apache.org/docs/en/latest/guide/appdev/whitelist/index.html
- <access origin="https://*.mydomain.com" />
- On iOS:
  - Application Transport Security (ATS)
  - <access origin="https://*.mydomain.com" minimum-tls-version="TLSv1.1" requires-forward-secrecy="false" />

THE PROBLEM WITH

IFRAMES

ACCESS TO THE "BRIDGE"

- If content is served in an iframe from a whitelisted domain, that domain will have access to the native Cordova bridge
- This means that if you whitelist a third-party advertising network and serve those ads through an iframe, it is possible that a malicious ad will be able to break out of the iframe and perform malicious actions
- Be careful what you whitelist

GETTING HARDCORE WITH

CERTIFICATE PINNING

(LIMITED) OPTIONS FOR CERT PINNING

- Cordova does not support true certificate pinning
- There are ways to approximate certificate pinning
  - TOFU (yum)
  - EddyVerbruggen/SSLCertificateChecker-PhoneGap-Plugin
- True certificate pinning for some platforms
  - wymsee/cordova-HTTP

FOR DEVELOPMENT ONLY!

SELF-SIGNED CERTIFICATES

SOME MORE ADVICE FROM THE SECURITY GUIDE

- Do not use Android Gingerbread (2.3)!
- Use InAppBrowser for outside links
- Validate all user input (worth repeating)
- Do not cache sensitive data
- Don't use eval() unless you know what you're doing
- Do not assume that your source code is secure

A FEW RESOURCES AS YOU GO FORWARD

- OWASP Top 10 - http://www.veracode.com/directory/owasp-top-10
- SQL Injection Myths and Fallacies - http://www.slideshare.net/billkarwin/sql-injection-myths-and-fallacies
- PhoneGap Platform Security wiki - https://github.com/phonegap/phonegap/wiki/Platform-Security
- Online Security Confs - http://www.tunnelsup.com/online-security-conferences
- HTML4 Security Cheat Cheet - https://html5sec.org/

THANKS, AND

LET'S STAY SAFE OUT THERE