Defense Guided by Experience

A tale of mobile threats

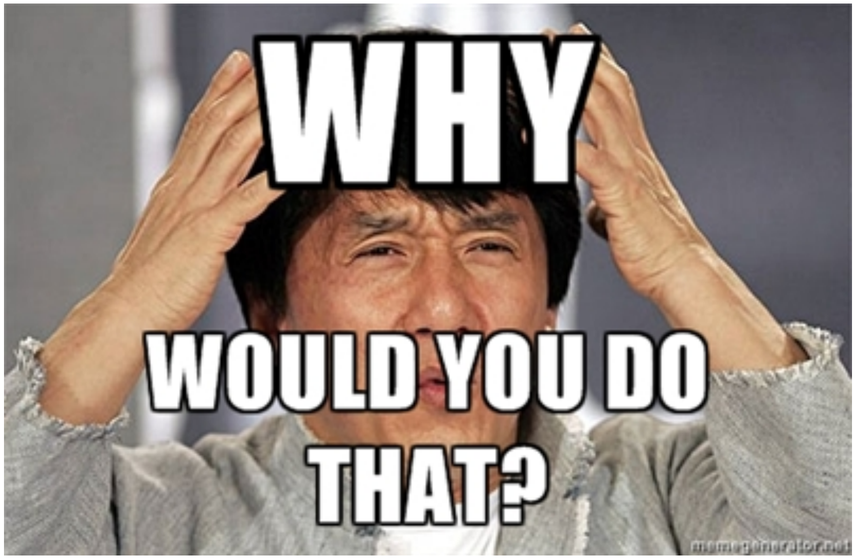
Vincenzo Iozzo

Director of Security Engineering
Trail of Bits, Inc

Part 1

In which I blame people

That’s how we deal with mobile


WHY
WOULD YOU DO
THAT?
memegenerator.net

How does offense work?

- Attacker’s mindset
- Gaining access
- Keeping access/stealing data

First problem: spot the difference

Black swans? What's that?

A very interesting research result that is unlikely to happen in real life

Why black swans exist?

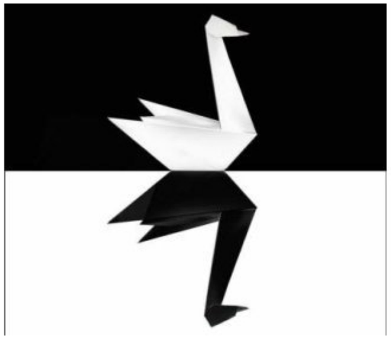
"Machines can remain vulnerable longer than you can remain sane"

The security community is fixated on persistance

A lot of people forget the mantra: "whoever scores is right"

Technical elegance is highly valued

Black swans and attacker math

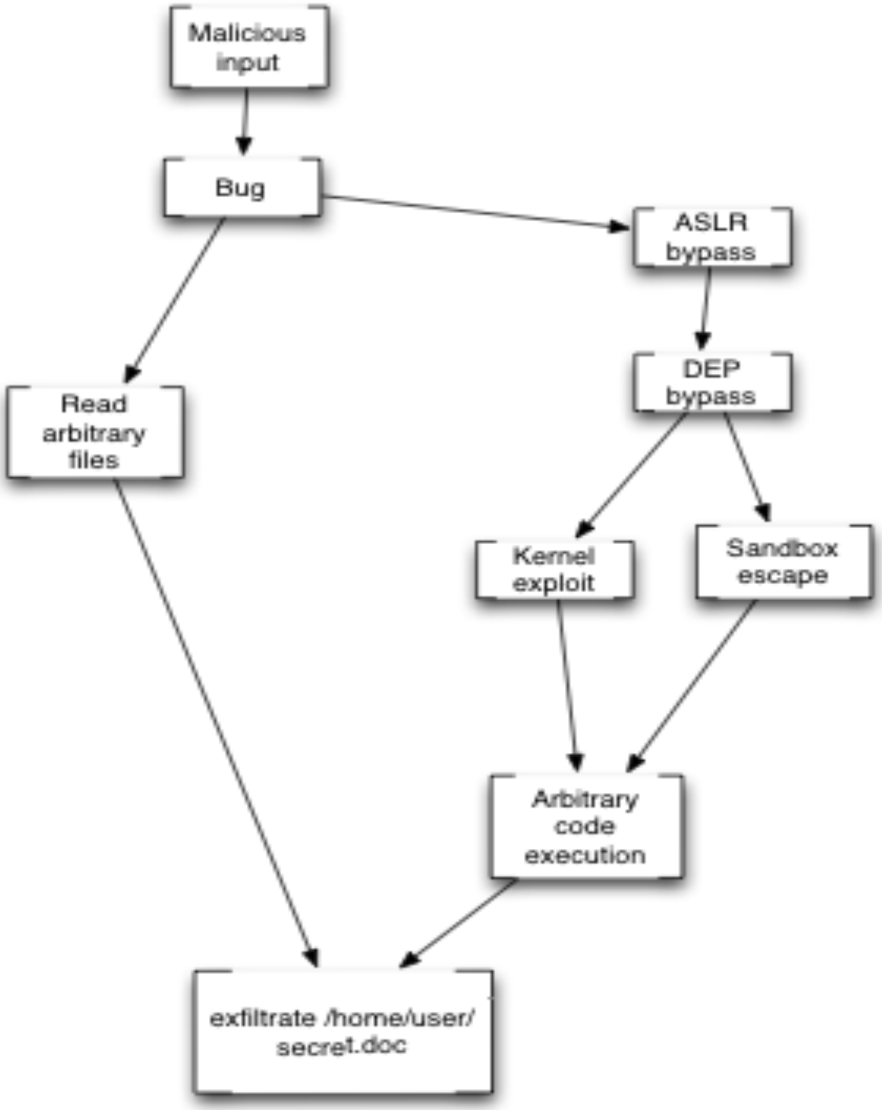


Attackers are resource-constrained: "The Exploit Intelligence Project" (Dan Guido)

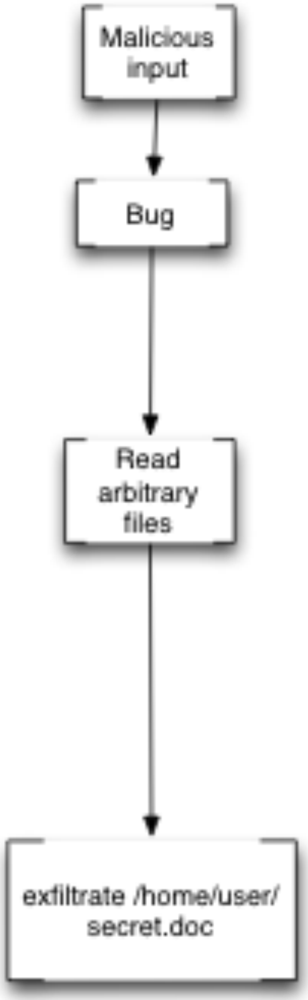
Attackers are rational human beings

Attackers will take a given exploitation path IFF no cheaper paths are available

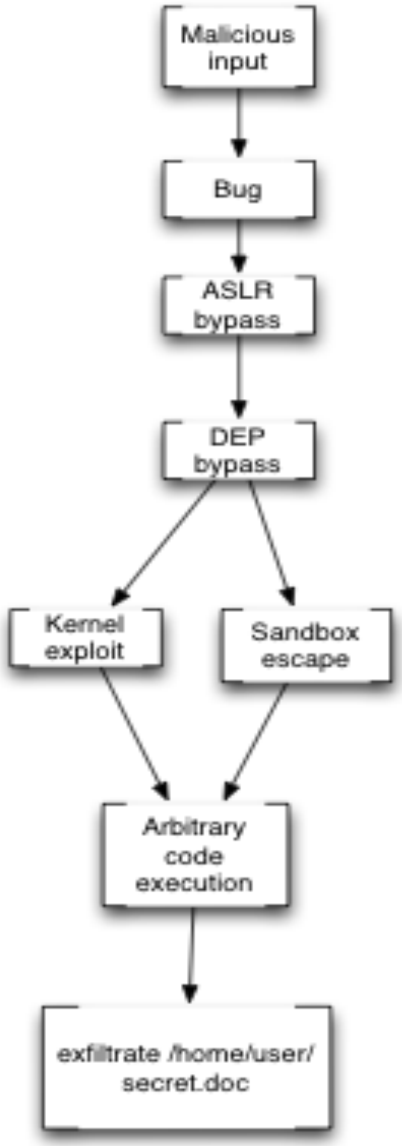
Exploitation paths



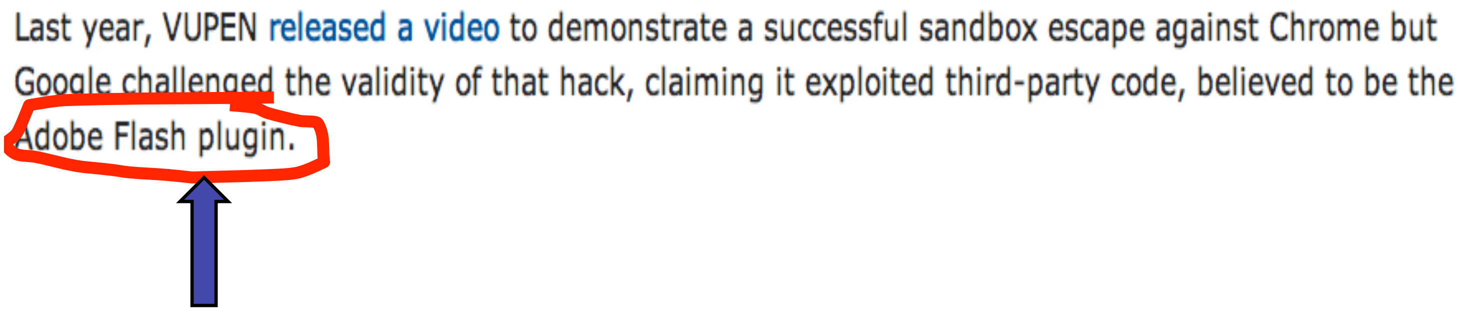
A rational attacker



A black swan

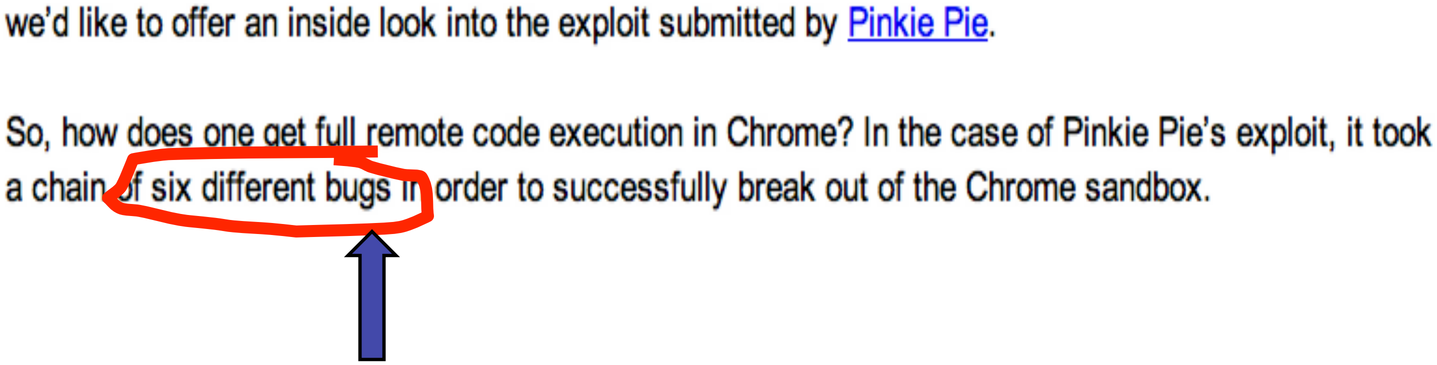

Malicious input
Bug
ASLR bypass
DEP bypass
Kernel exploit
Sandbox escape
Arbitrary code execution
exfiltrate /home/user/secret.doc

Practical example



Last year, VUPEN released a video to demonstrate a successful sandbox escape against Chrome but Google challenged the validity of that hack, claiming it exploited third-party code, believed to be the Adobe Flash plugin.

A rational attacker



we'd like to offer an inside look into the exploit submitted by Pinkie Pie.

So, how does one get full remote code execution in Chrome? In the case of Pinkie Pie's exploit, it took a chain of six different bugs in order to successfully break out of the Chrome sandbox.

A black swan (AKA: are you nuts?)

So…

1

VS

0

Unless..

The ROI on a black swan is higher, for some definition of “return”

Flame md5 collision attack comes to mind

Therefore our graph is weighted

Weight function

That’s very hard to calculate in the general case

Some examples in “Attacker Math 101” (Dino Dai Zovi)

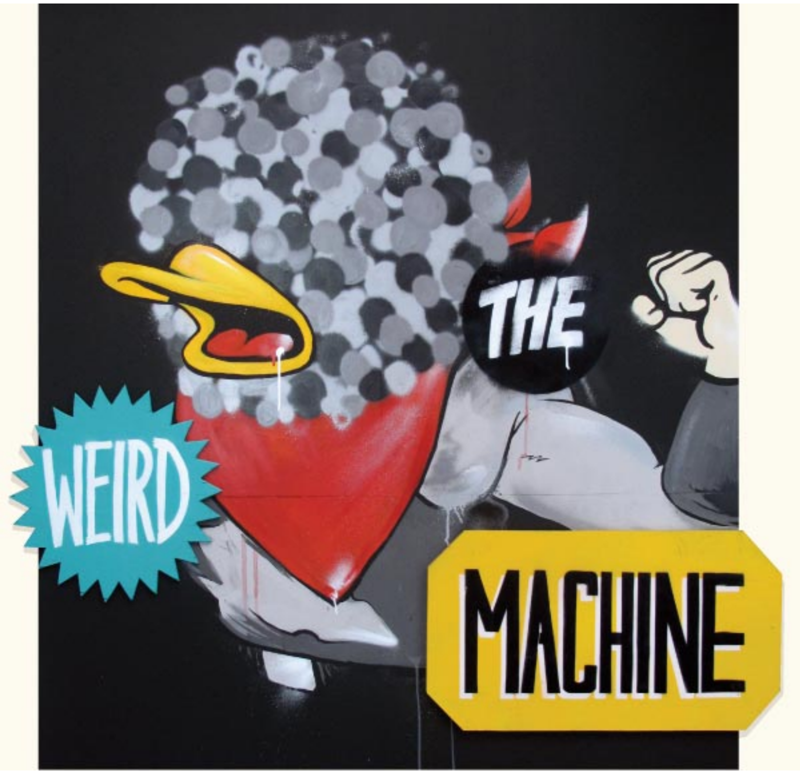
A bit out of scope here

But we can usually draw a line easily

What if two paths are equally cost effective?

Gaining access..

It's all about programming a "weird machine" (Sergey Bratus et al.)



The weird machine

In short: "a machine that executes an unexpected series of instructions"

BOB NEWHART
I shouldn't Even Be Doing This!
and Other Things that Strike Me as Funny

By examples

- ROP
- JIT Spraying – Dion Blazakis
- SpiderMonkey Bytecode Hijacking – Thomas Dullien
- JIT code hijacking – Chris Rohlf and Yan Ivnitskiy
- …

Exploitation

Exploitation is setting up, instantiating, and programming the weird machine - Thomas Dullien

Controlling the machine

- You need write primitives
- You need infoleaks/memleaks

For both you need some degree of control over the application.

It’s either pure data or you can directly influence the application state (eg: through an interpreter of some kind)

Me no like exploits

This process is challenged in a few ways:

- Negate the initialization (fix bugs)
- Make the setup hard (heap/stack mitigations, ASLR)
- Make it hard to put together ‘weird instructions’ (ASLR, DEP, JIT hardening)
- Reduce/Neutralize the effects of a running weird machine (sandboxing, code signing)
- More to come in the future..

Get to the data/persistence

- How hard is to get your code on a target?
- How far away is the data you care for from you?

For future reference..

So here’s the thing:

- In a few years everything an attacker cares for will be inside a browser/mobile app

Do sandboxes help with that? *NO*

Let's wrap up

Attacker's mindset: take the most cost-effective path

When it comes to exploitation the most cost-effective path is:

1) As close as possible to your data

2) Reduces as much as possible the need for multiple bugs/exploits

3) Reduces maintenance cost

Part 2

In which I actually talk about mobile

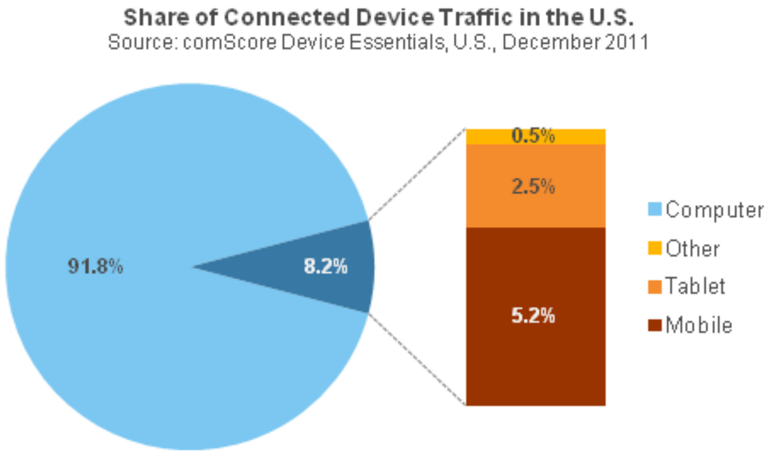
Drive-bys

Mobile Town

ING SEYF S0543 [RF] © www.visualphotos.com

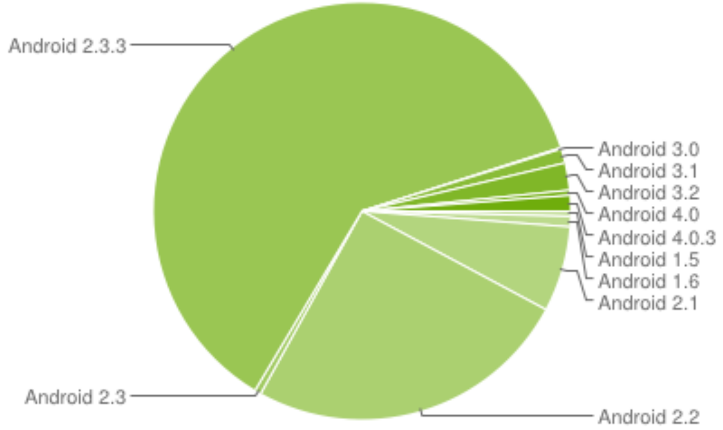
Desktop City

Too few and too many


Share of Connected Device Traffic in the U.S.

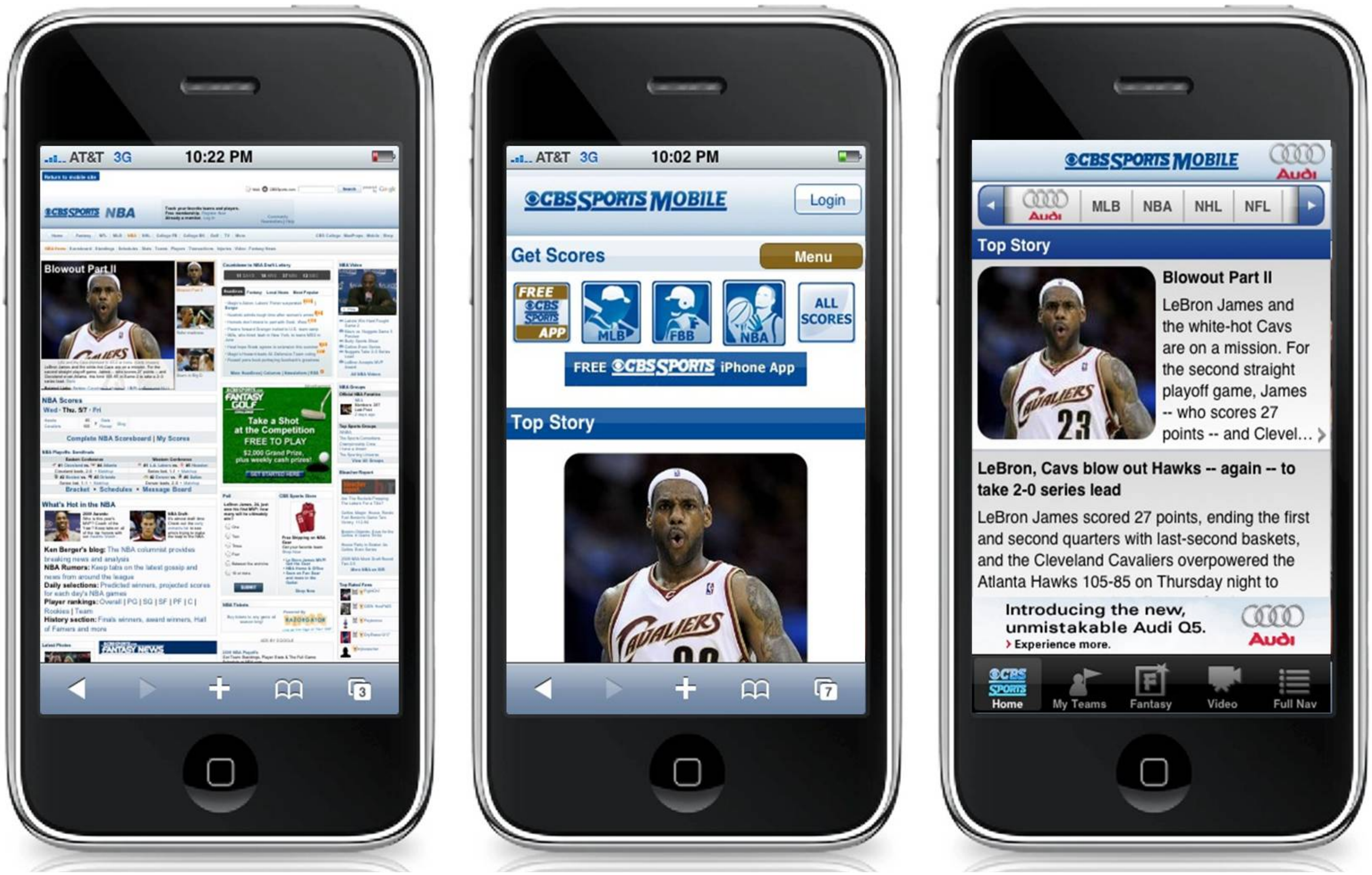
Source: comScore Device Essentials, U.S., December 2011

~8% of total web traffic comes from mobile devices



Breakdown by version / features (+ varying rates of feature support)

Like Facebook..



Ken Berger's blog: The NBA columnist provides breaking news and analysis
NBA Rumors: Keep tabs on the latest gossip and news from around the league
Daily selections: Predicted winners, projected scores for each day's NBA games
Player rankings: Overall | PG | SG | SF | PF | C | Rookies | Team
History section: Finals winners, award winners, Hall of Famers and more

Blowout Part II

LeBron James and the white-hot Cavs are on a mission. For the second straight playoff game, James -- who scores 27 points -- and Clevel…

LeBron, Cavs blow out Hawks -- again -- to take 2-0 series lead

LeBron James scored 27 points, ending the first and second quarters with last-second baskets, and the Cleveland Cavaliers overpowered the Atlanta Hawks 105-85 on Thursday night to

Introducing the new, unmistakable Audi Q5.
› Experience more.
Audi

Takeaway

Drive-bys don't matter and realistically never will

Hard to get anything useful (contrary to dekstops) out of them

Hard to run the attack in the first place

The web is the future of the desktop, apps are the future of mobile = attackers behave accordingly

Malware

1

Apple App Store

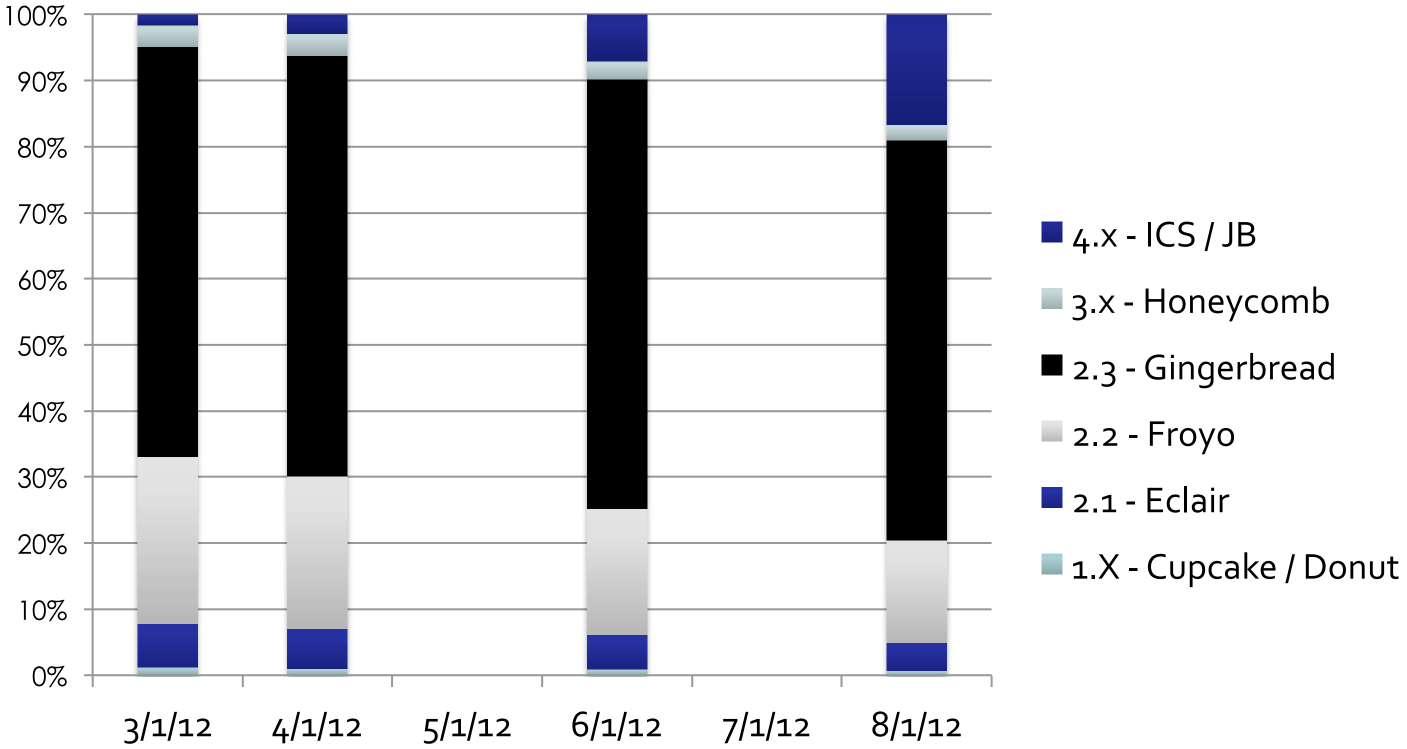
31

Google Marketplace

One of the reasons

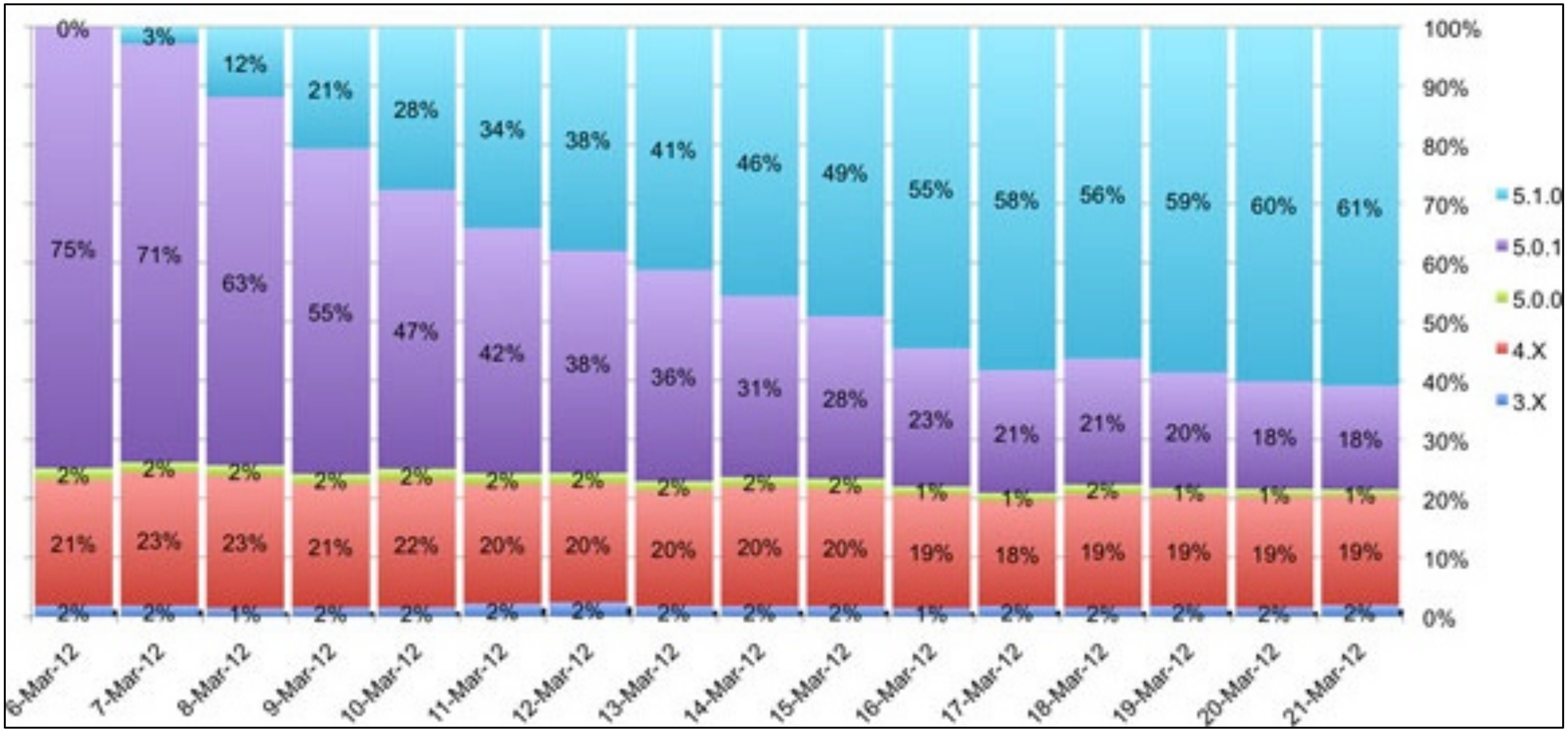
Savage Chickens
by Doug Savage
YOU'VE GOT A PROBLEM WITH AVOIDING PERSONAL ACCOUNTABILITY
YA, AND WHOSE FAULT IS THAT?
© 2006 BY DOUG SAVAGE
www.savagechickens.com

Malware lasts long on Android



Android Exploit	Time to Patch 50%
Exploid (2.1)	294 days
RageAgainstTheCage (2.2.1)	> 240 days

Not so much on iOS



Vulnerability	Exploit	Patch Availability
Malformed CFF	Star (JailbreakMe 2.0)	10 days
T1 Font Int Overflow	Saffron (JailbreakMe 3.0)	9 days

AppStore vs Google Play

Apple enforces accountability

Sandbox escape: Android > iOS

Fragmented user-base = the investment lasts longer

On Android privesc are enough to cause troubles

That being said: jailbroken iOS = Android

Malware - takeaway

- Does only matter on Android and jailbroken iOS

- It scales, it's easy and it lasts

- Can this be fixed? Yes, Apple did

App specific

Android NDK can open up this attack surface a lot

Interesting because applications are likely less audited than system code

But more importantly: interesting data will be inside the app. Why go anywhere else?

Expect them in the future!

More "smart" in phone?

Enter baseband

A few words on it

- Most of the code in there is old (1990 old)
- Based on the assumption that the actors are trusted
- Most of the research has been done by Ralf Philipp Weinmann
- His research led to bug fixing and some mitigations

Baseband weird machines

Increased attention being paid to bugs in there

Still a very big surface with few (known) actors

Big state machine based on a giant interface, so hard to fuzz

Need profound knowledge to find certain bugs

Baseband weird machine 2

Very few mitigations in place

Still most of the heap metadata exploitation is possible (eg: write4 primitives on Infineon)

No ASLR, no "sandboxes"

Remote: control through data only

Local: "interpreter" (AT commands)

Baseband - persistence

Good luck with forensics/IR

Depending on how the App processor interacts with the BB it might lead to full-device compromise

Regardless: access to phone calls, SMS and data

Attack scenarios

- Remote exploit to steal/alter/make sms/ data/phone calls
- App remote-> BB local rootkit
- BB remote -> BB local rootkit
- DDos in case of crisis?

So ..

1) High ROI

2) Very few mitigations

3) Detection is hard

Great target for motivated attackers!

NFC

That’s complicated…

NFC - capabilities

Can potentially lead to device compromise through malformed packets at protocol level – device proximity

Can lead to device compromise at ‘application level’ – tag proximity

Steal data – roughly 1.5 meters with custom hardware

Auth bypass issues

First case

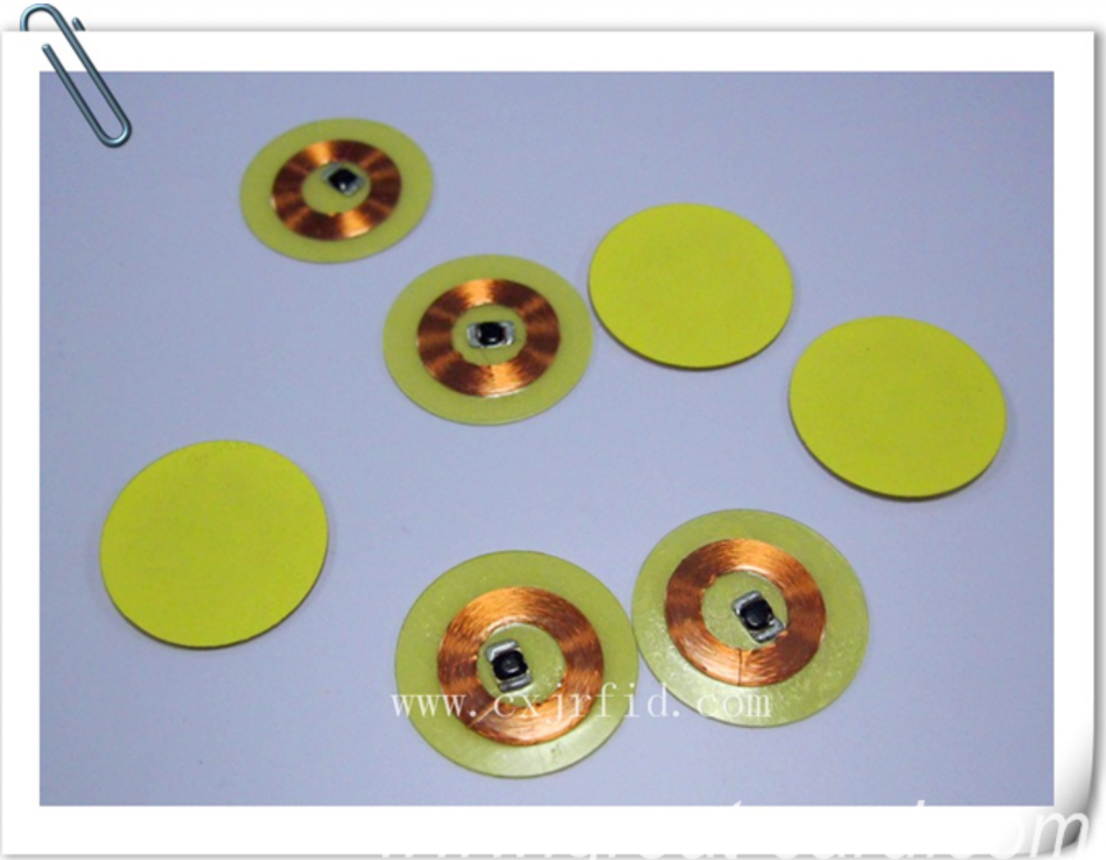
Not very viable..

On the flipside, you can potentially get huge access to the device

Most likely a black swan

Second case

You can compromise the device by using tags (simple stickers) -> do not need proximity



Second case

You can either run your exploit for browser and stuff (might require some kind of permission)

Compromise through tag parsing!

Mobile Pwn2own 2012 was won using this approach

This is more interesting! Rational black swan

Part 3

In which I make statements

Conclusion

If you don’t know *what* you’re protecting, you’ll fail

Likewise if you don’t know what you’re protecting *against*, you’ll fail

You don’t need a horde of code auditors & policy people, you need a CEO (chief exploitation officer)

Specific to mobile

Worry more about the “phone” than the “computer”

App sandboxes are great to make persistence hard, way less so for data exfiltration

Android is bad, you don’t want that in your company

NFC is/will be more a “physical” security issue than an Infosec one

Part 5

In which you can ask questions or insult me

Thanks!
Questions?

vincenzo@trailofbits.com