Cracking the Code on the Mobile Software Supply Chain

A Closer Look at the Android Kernel

David Maxwell, Open Source Strategist

March 1, 2011

Coverity Scan Initiative

Started in 2006 under contract with the US Department of Homeland Security

291 open source projects

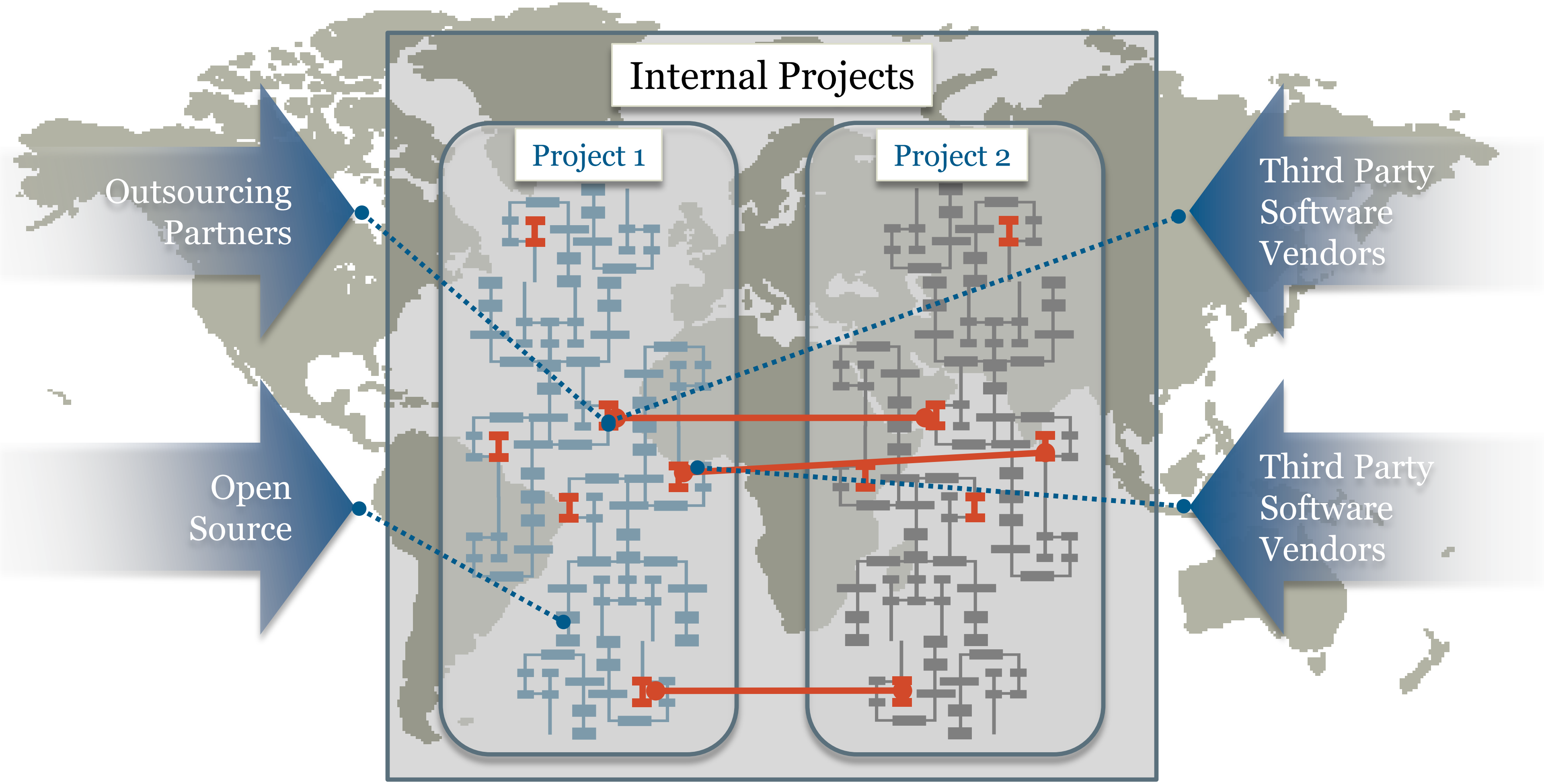
61 million lines of code tested

49,654 defects identified

15,278 defects fixed

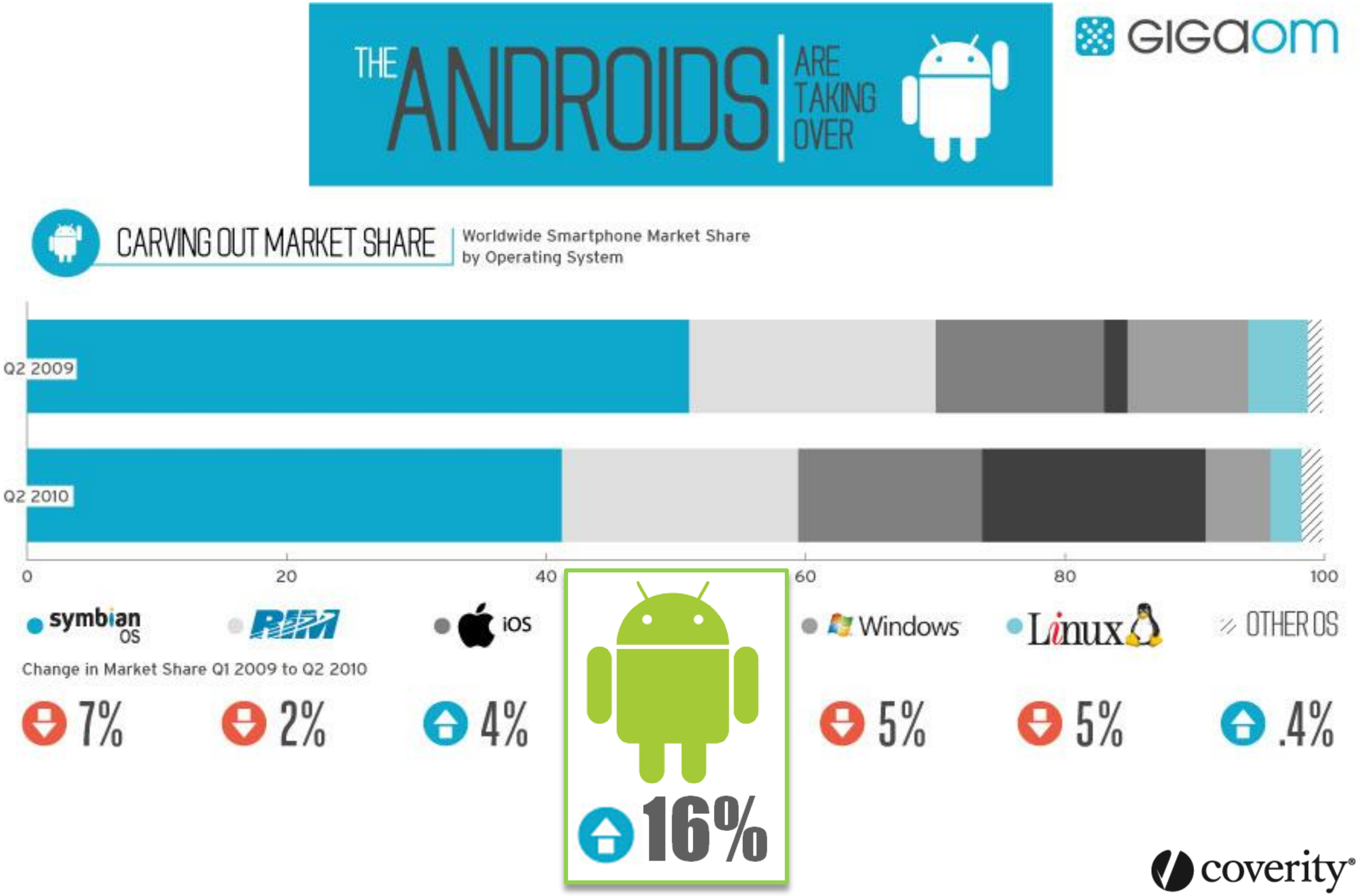
Supply Chain Increases Complexity

Geographically distributed teams and third party suppliers require new levels of visibility and control



coverity®

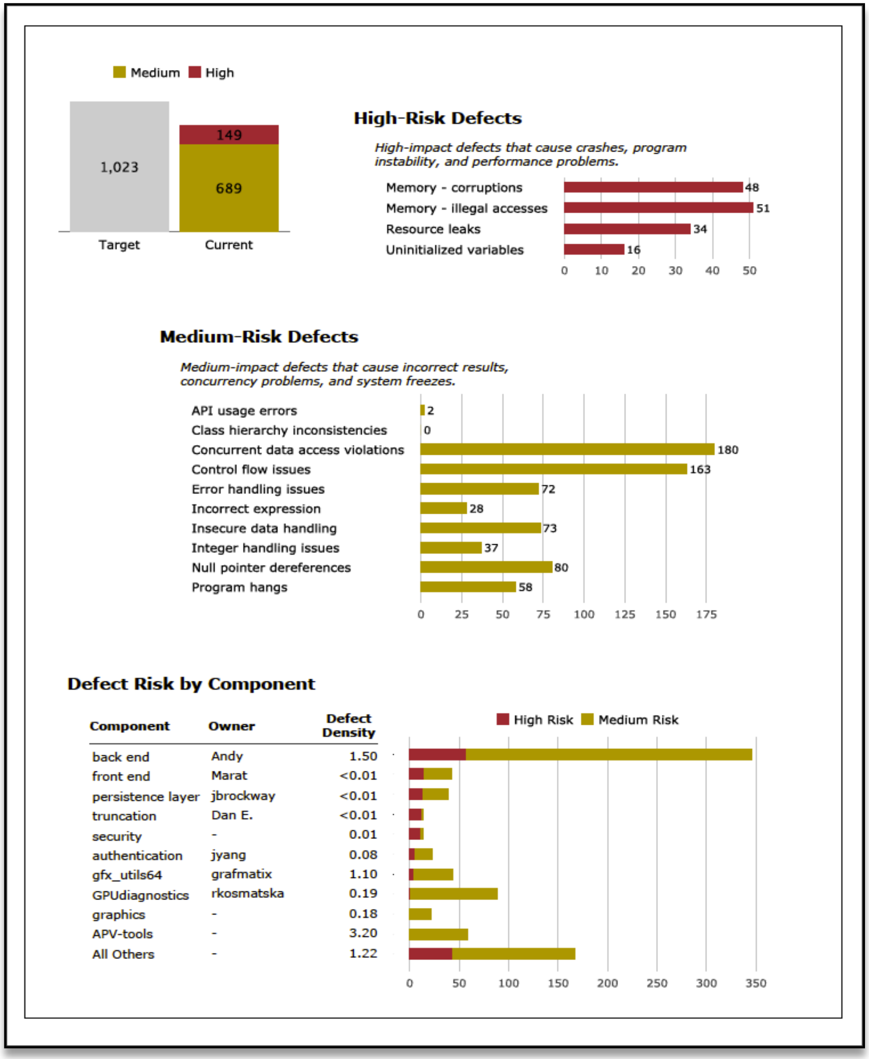
The Growth of Android



2010 Report Highlights

Results from our test of the Android kernel 2.6.32 shipping in the HTC Droid Incredible phone:

- Better than average quality: .47 defect density
- 359 defects identified
- 88 high risk defects



High-Risk Defects

High-impact defects that cause crashes, program instability, and performance problems.

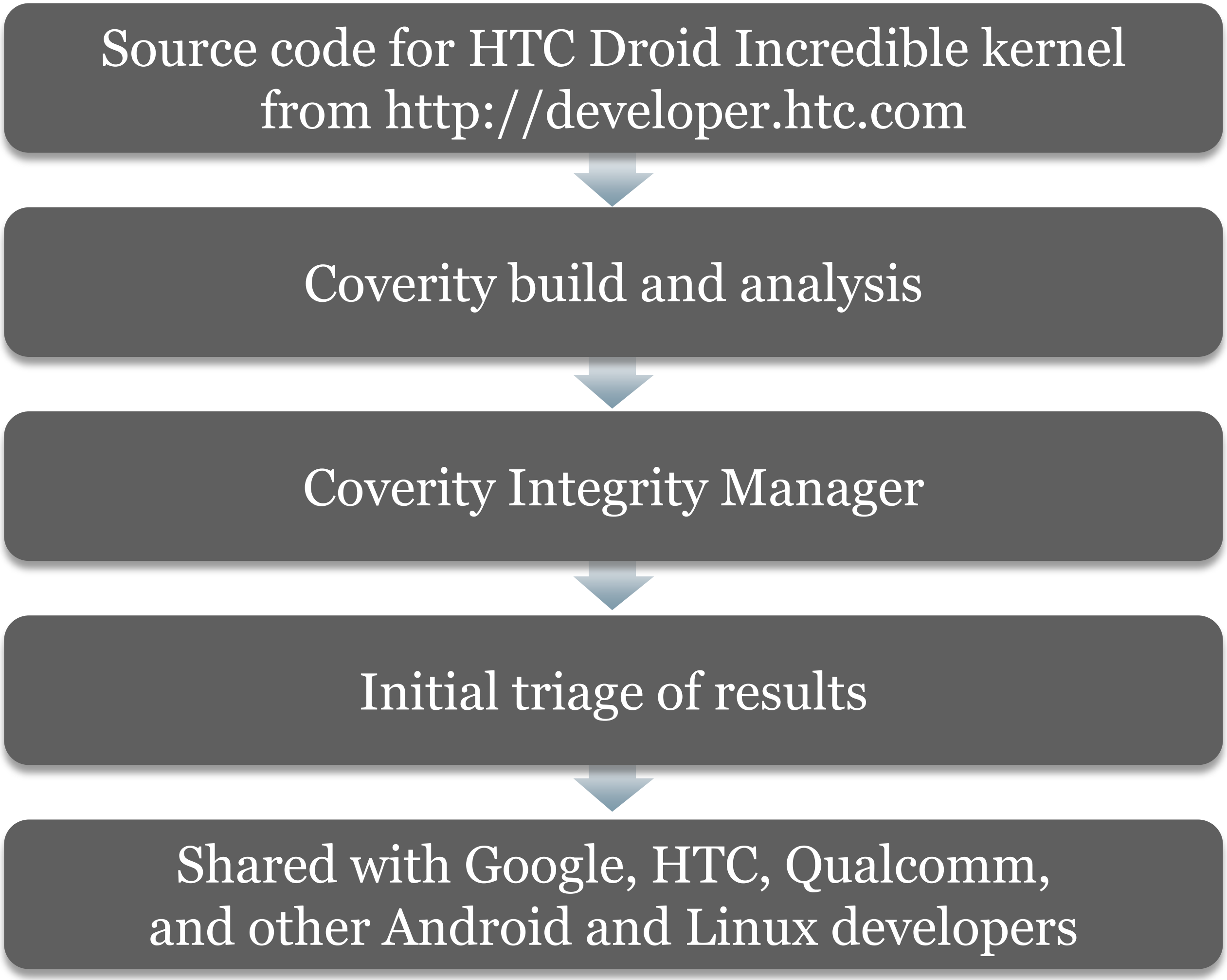
Medium-Risk Defects

Medium-impact defects that cause incorrect results, concurrency problems, and system freezes.

Defect Risk by Component

Component	Owner	Defect Density
back end	Andy	1.50
front end	Marat	<0.01
persistence layer	jbrockway	<0.01
truncation	Dan E.	<0.01
security	-	0.01
authentication	jyang	0.08
gfx_utils64	grafmatix	1.10
GPUdiagnostics	rkosmatska	0.19
graphics	-	0.18
APV-tools	-	3.20
All Others	-	1.22

Under the Hood…
Background on the Testing Process



Source code for HTC Droid Incredible kernel from http://developer.htc.com

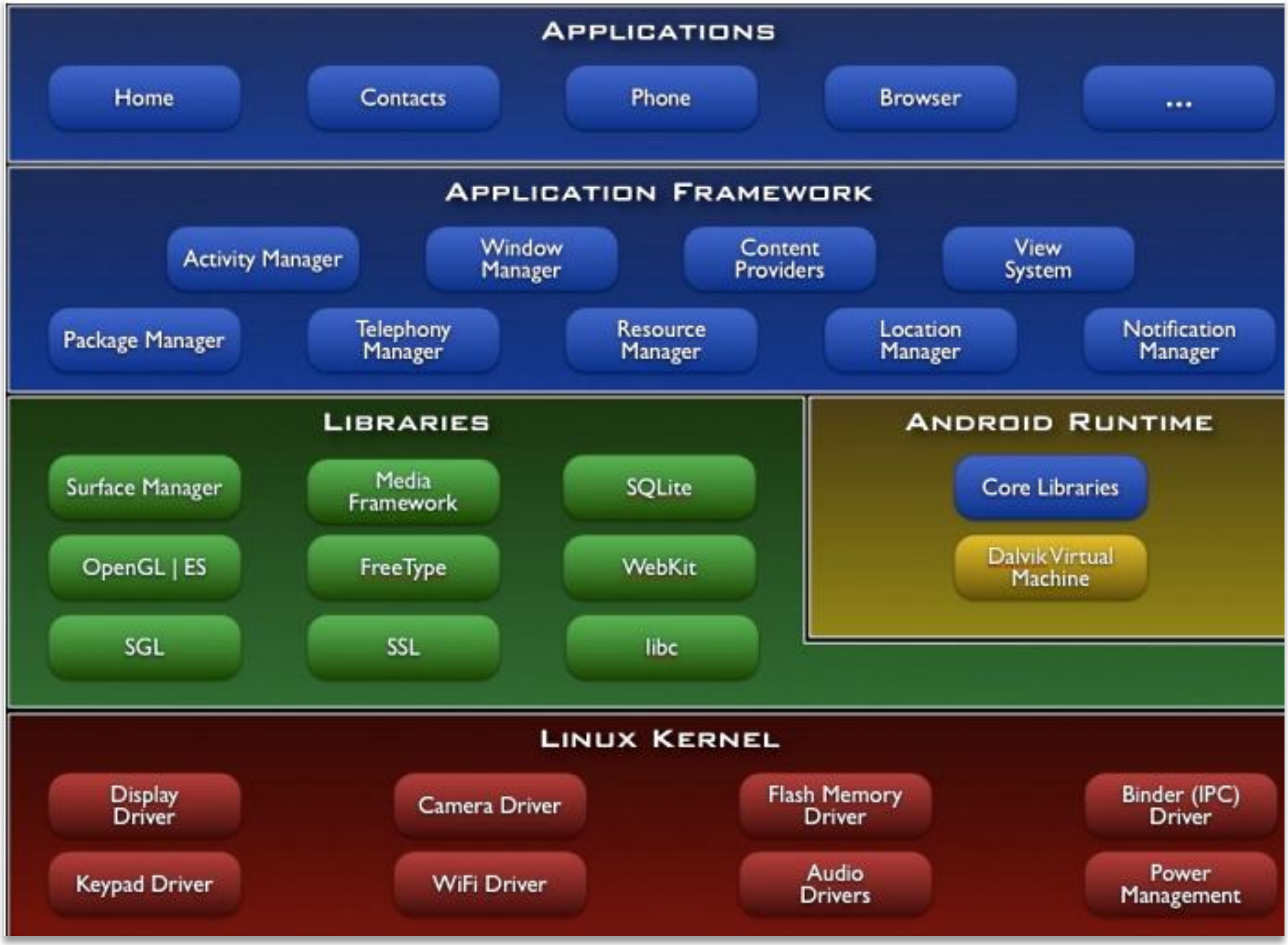
Coverity build and analysis

Coverity Integrity Manager

Initial triage of results

Shared with Google, HTC, Qualcomm, and other Android and Linux developers

The Android Stack



APPLICATIONS

Home, Contacts, Phone, Browser, ...

APPLICATION FRAMEWORK

Activity Manager, Window Manager, Content Providers, View System

Package Manager, Telephony Manager, Resource Manager, Location Manager, Notification Manager

LIBRARIES

Surface Manager, Media Framework, SQLite

OpenGL | ES, FreeType, WebKit

SGL, SSL, libc

ANDROID RUNTIME

Core Libraries

Dalvik Virtual Machine

LINUX KERNEL

Display Driver, Camera Driver, Flash Memory Driver, Binder (IPC) Driver

Keypad Driver, WiFi Driver, Audio Drivers, Power Management

Source: http://developer.android.com/guide/basics/what-is-android.html

Types of High Risk Defects Identified

Type of Defect	Real-world Impact
Memory corruptions	Program or system crash
Illegal memory accesses	Unpredictable behavior
Resource leaks	General software reliability problems
Uninitialized variables	Denial of service conditions/ unexpected control flow modifications

Progress Since November 2010

- Android Kernel in HTC Droid Incredible 2.6.32
  - Reduced total defect count from 359 to 222
  - Reduced high risk defect count from 88 to 55

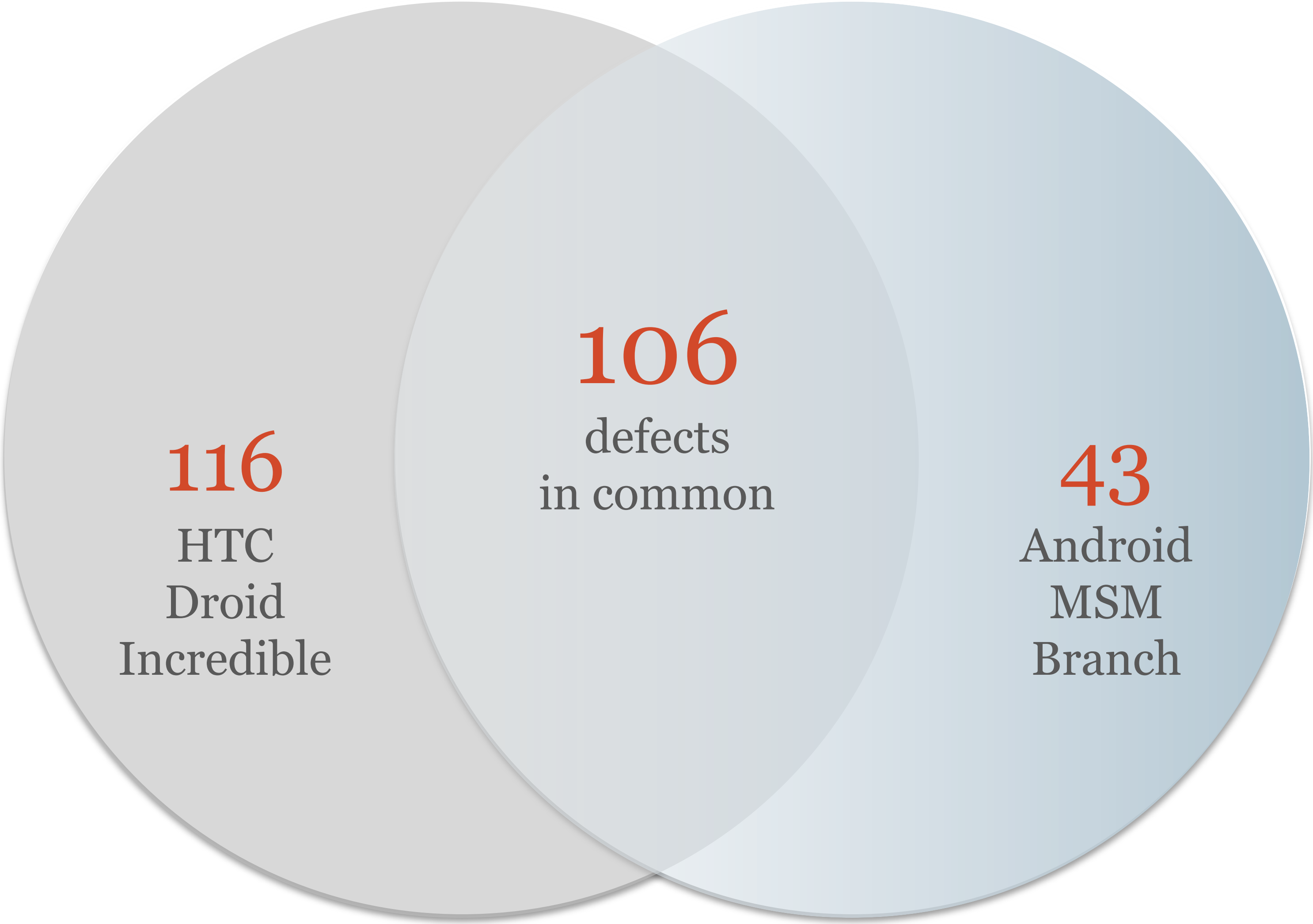
- Extended our research to look at additional code bases—starting with Android development branch
  - The version of the Android kernel in the HTC Droid Incredible phone was approximately 1 year old relative to the latest development branch
  - Certain source code was modified or added specifically to the HTC phone that was not part of the standard Android kernel

Progress Since November 2010

- Google Android MSM Development Branch 2.6.35
  - Total defect count: 149
  - High risk defect count: 32
    - 16 memory corruption or out of bounds access issues
    - 6 resource leaks
    - 10 uninitialized variables

- Provided by Google to reference an active and more recent Android Kernel
- Release details not known for this branch

Defects in Common



116
HTC
Droid
Incredible

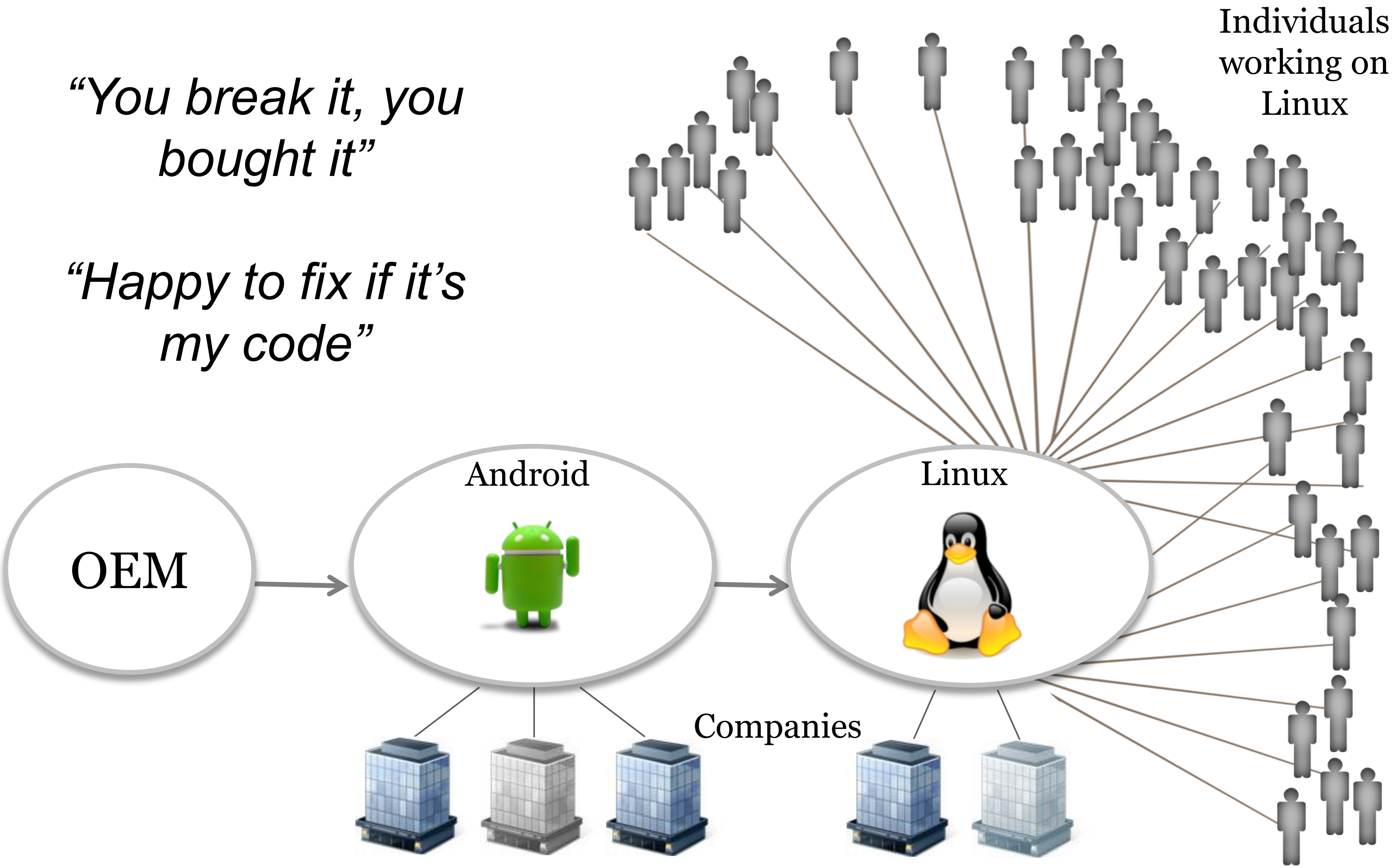
106
defects
in common

43
Android
MSM
Branch

Responsibility to Fix Defects Is Fragmented: Android Supply Chain Complexity

“You break it, you bought it”

“Happy to fix if it’s my code”



Android Lessons Learned

1. Even code with an above-average level of quality still has high risk defects…and the problems don’t necessarily disappear from one version to the next.

2. Responsibility and standards for quality are fragmented due to the complexity of the supply chain and number of contributors to Android.

3. OEMs should hold their suppliers accountable to the same standards they have in place for in-house developed code.

What We Are Doing Next

- Continuing to test Android code: expanding out to additional kernels and broadening the scope to include components further up the stack

- Testing and publishing more open source project results

- Helping OEMs get visibility by testing code across their entire software supply chain (internal, open source, third party)

- Working more closely with developers on open source projects to ensure defects are fixed

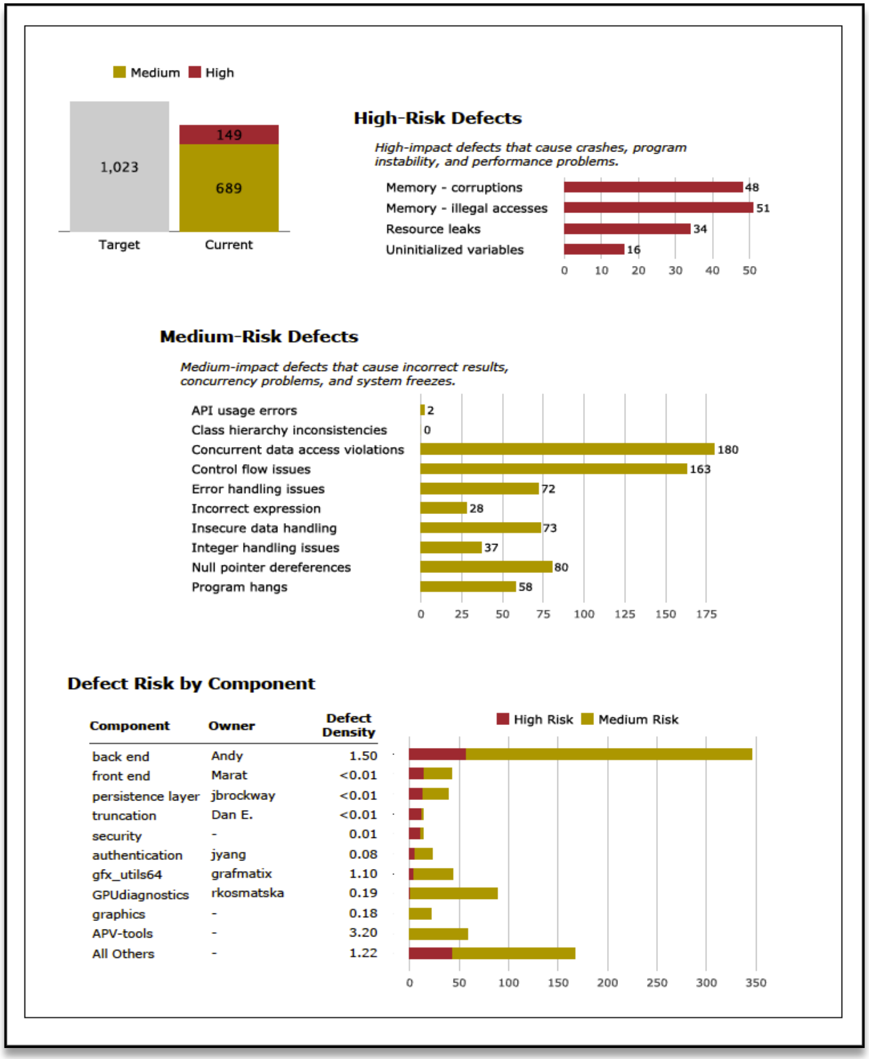
Open Source Integrity

Supply chain requirements every OEM must have:

- Consistent quality standards
- Visibility into quality of components
- Establish basic code testing methods

Dashboards - Software Integrity Report

- The same rules must apply to any code received across your supply chain.
- Have a consistent measure of quality and security to hold your suppliers accountable.
- Identify problems and gain confidence in the quality of the product you are about to ship (or are shipping).



High-Risk Defects

High-impact defects that cause crashes, program instability, and performance problems.

Medium-Risk Defects

Medium-impact defects that cause incorrect results, concurrency problems, and system freezes.

API usage errors 2
Class hierarchy inconsistencies 0
Concurrent data access violations 180
Control flow issues 163
Error handling issues 72
Incorrect expression 28
Insecure data handling 73
Integer handling issues 37
Null pointer dereferences 80
Program hangs 58
0
25
50
75
100
125
150
175

Defect Risk by Component

Component	Owner	Defect Density
back end	Andy	1.50
front end	Marat	<0.01
persistence layer	jbrockway	<0.01
truncation	Dan E.	<0.01
security	-	0.01
authentication	jyang	0.08
gfx_utils64	grafmatix	1.10
GPUdiagnostics	rkosmatska	0.19
graphics	-	0.18
APV-tools	-	3.20
All Others	-	1.22

Thank You

Questions

dmaxwell@coverity.com
android@coverity.com