Demonstrating the Absence of Malice in Commodity Software

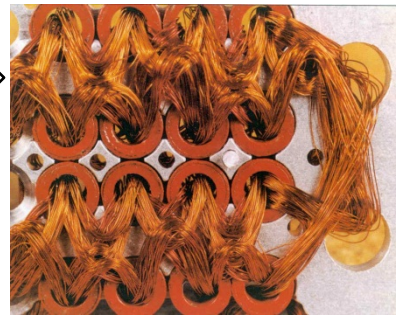
Tim Fraser
Program Manager
Information Innovation Office (I2O)
DARPA

6 May 2015

The Software Supply Chain

1965
Apollo Guidance Computer

24KB software

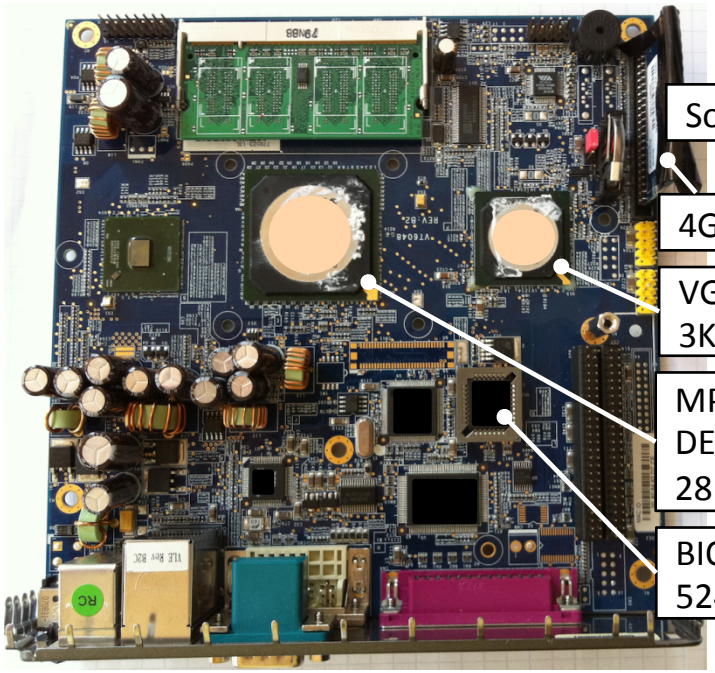


From Hall, Journey to the Moon, AIAA 1996

- Custom hardware and software
- USG has a complete spec
- Programmed and assembled in US

2012
Thin Client from U.S. Supplier*

International



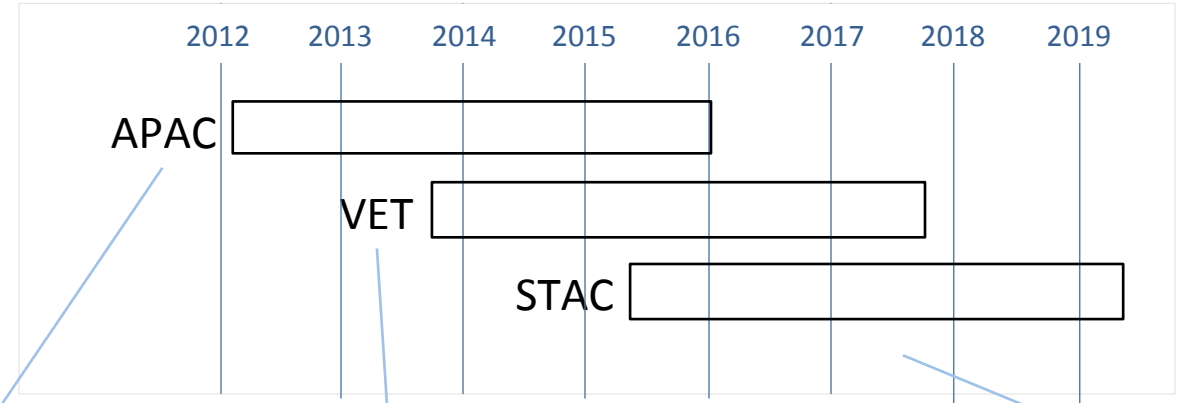
- Commodity hardware and software
- USG has no spec; is a small customer
- Programmed and assembled overseas

DoD, USG, Defense Industrial Base, US Commercial Industry relies on much commodity IT equipment. How can we use these devices with confidence?

* Image of a WYSE (DELL) VX0 Thin Client mainboard. LOC estimates based on counts for latest coreboot as of 2012-10-03, libmpeg2-0.5.1, and VGABIOS 0.7a.
Specific software and hardware examples are for illustration only and are not meant to imply product vulnerabilities.

Research Goal

Goal: demonstrate practical methods for demonstrating the absence of malice in as-built commodity software.



Build tools to enable analysts to identify Trojan horse mobile applications faster and more accurately.

Broaden scope to firmware.

What malice should analysts look for and where?

How to run diagnostics on devices rigged to lie?

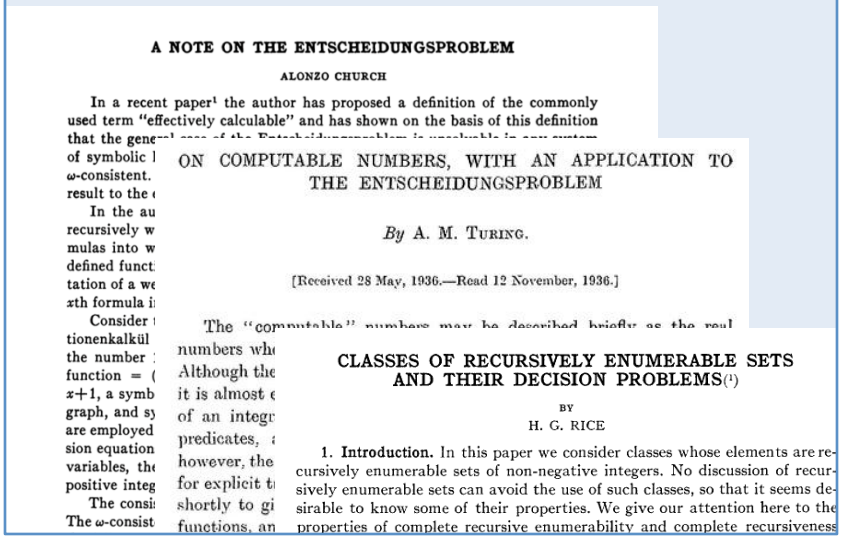
Reason about the next generation of flaws: vulnerabilities to algorithmic complexity and side channel attacks.

Major Technical Challenges, Enabling Factors

Major Technical Challenges

The field of malice appears limited only by the imagination of our adversaries.

Program analysis is fundamentally hard. Static analyses will have false alarms, missed detections, or both.



A NOTE ON THE ENTSCHEIDUNGSPROBLEM

ALONZO CHURCH

In a recent paper[1] the author has proposed a definition of the commonly used term "effectively calculable" and has shown on the basis of this definition that the gene... of symbolic l... ω-consistent. result to the ... In the au... recursively w... mulas into w... defined funct... tation of a we... xth formula i... Consider ... tionenkalkül ... the number ... function = ... x+1, a symb... graph, and s... are employed ... sion equation ... variables, th... positive integ... The consi... The ω-consist...

ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM

By A. M. TURING.

[Received 28 May, 1936.—Read 12 November, 1936.]

The "computable" numbers may be described briefly as the real... numbers wh... Although the... it is almost ... of an integr... predicates, ... however, the... for explicit t... shortly to gi... functions, an...

CLASSES OF RECURSIVELY ENUMERABLE SETS AND THEIR DECISION PROBLEMS(1)

BY

H. G. RICE

1. Introduction. In this paper we consider classes whose elements are recursively enumerable sets of non-negative integers. No discussion of recursively enumerable sets can avoid the use of such classes, so that it seems desirable to know some of their properties. We give our attention here to the properties of complete recursive enumerability and complete recursiveness

Enabling Factors

Insight: this is a contest between humans.

Although there can be no perfect automated program analysis tool for this problem, imperfect but practically useful tools can provide an edge.

Vetting as-built software reverses the traditional offense/defense dynamic: the adversary must present software for analysis and hope it evades an unknown set of analyses.

What APAC is trying to do:

Third-party developers submit mobile apps to DoD.

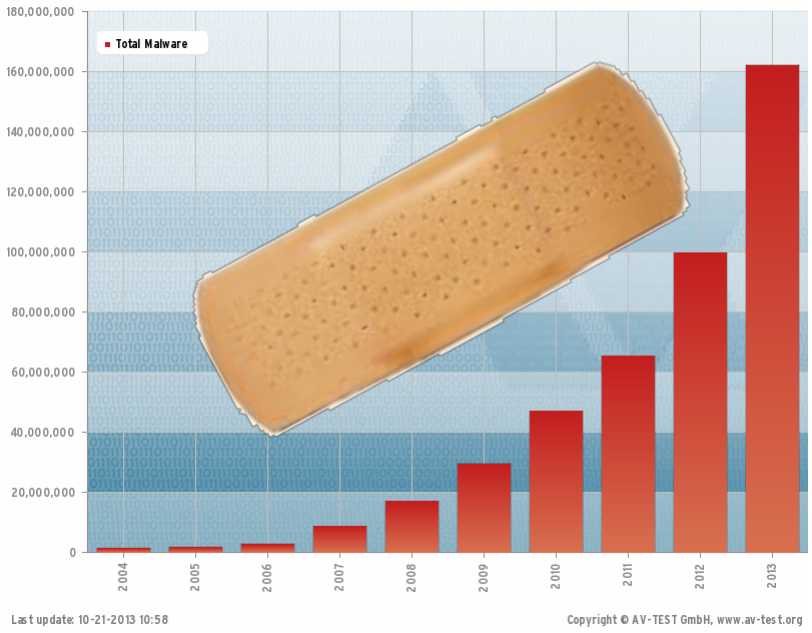
Some contain hidden malicious functionality

DoD analysts keep mobile app store free of malicious apps.

Goal: enable analysts to detect hidden malice in apps fast enough and accurately enough to keep a DoD app store malware-free.

How Malice is Detected in Mobile Apps Today:

PC-STYLE ANTI-MALWARE PRODUCTS



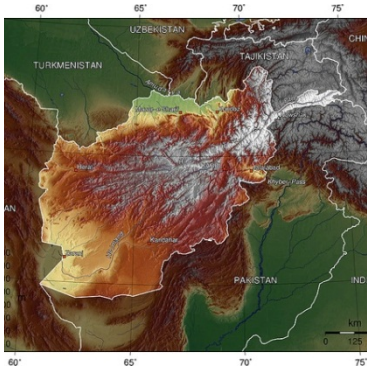
- Ineffective against novel malware.
- Malware with 10K's victims get signatures first.

TODAY'S PROGRAM ANALYSIS PRODUCTS

SUDOKU

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

ANDROIDMAP



If apps get my physical location and use the Network, which is malware?

Today's tools are insufficient to detect malice hidden at APAC's level of sophistication.

ANDROIDMAP was not used in operations in Afghanistan.

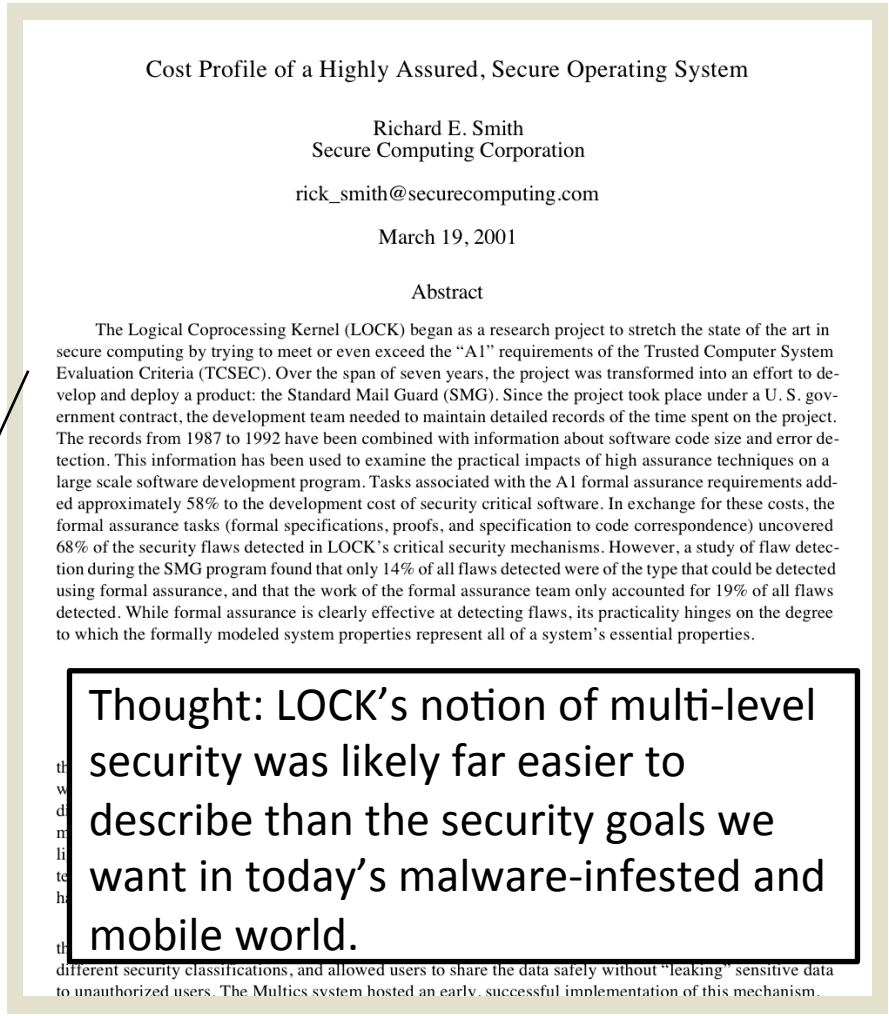
It's Hard to Define What "Secure" Means

Challenge #1: Can you define "not malicious" in terms of properties you can demonstrate with a tool?

LOCK/Standard Mail Guard effort (1987-1992):

Mostly manual formal methods uncovered only 68% of the security flaws found. Why not 100%?

"While formal assurance is clearly effective at detecting flaws, its practicality hinges on the degree to which the formally modeled system properties represent all of a system's essential properties."



Cost Profile of a Highly Assured, Secure Operating System

Richard E. Smith
Secure Computing Corporation

rick_smith@securecomputing.com

March 19, 2001

Abstract

The Logical Coprocessing Kernel (LOCK) began as a research project to stretch the state of the art in secure computing by trying to meet or even exceed the "A1" requirements of the Trusted Computer System Evaluation Criteria (TCSEC). Over the span of seven years, the project was transformed into an effort to develop and deploy a product: the Standard Mail Guard (SMG). Since the project took place under a U. S. government contract, the development team needed to maintain detailed records of the time spent on the project. The records from 1987 to 1992 have been combined with information about software code size and error detection. This information has been used to examine the practical impacts of high assurance techniques on a large scale software development program. Tasks associated with the A1 formal assurance requirements added approximately 58% to the development cost of security critical software. In exchange for these costs, the formal assurance tasks (formal specifications, proofs, and specification to code correspondence) uncovered 68% of the security flaws detected in LOCK's critical security mechanisms. However, a study of flaw detection during the SMG program found that only 14% of all flaws detected were of the type that could be detected using formal assurance, and that the work of the formal assurance team only accounted for 19% of all flaws detected. While formal assurance is clearly effective at detecting flaws, its practicality hinges on the degree to which the formally modeled system properties represent all of a system's essential properties.

Thought: LOCK's notion of multi-level security was likely far easier to describe than the security goals we want in today's malware-infested and mobile world.

different security classifications, and allowed users to share the data safely without "leaking" sensitive data to unauthorized users. The Multics system hosted an early, successful implementation of this mechanism.

Examples of APAC Trojan Horse Apps

Conference Monitor:

15K SLOC **E5A, March 2015**

Stated purpose: record and replay DTMF tones to rapidly dial into teleconference bridges.

Malice: prepends tones to the recording that dial the adversary's phone first. Adversary can then act as a man-in-the-middle.

Trick: spotting malice depends on understanding subtleties of Android PendingIntents.

Noiz2:

10K SLOC **E5A, March 2015**

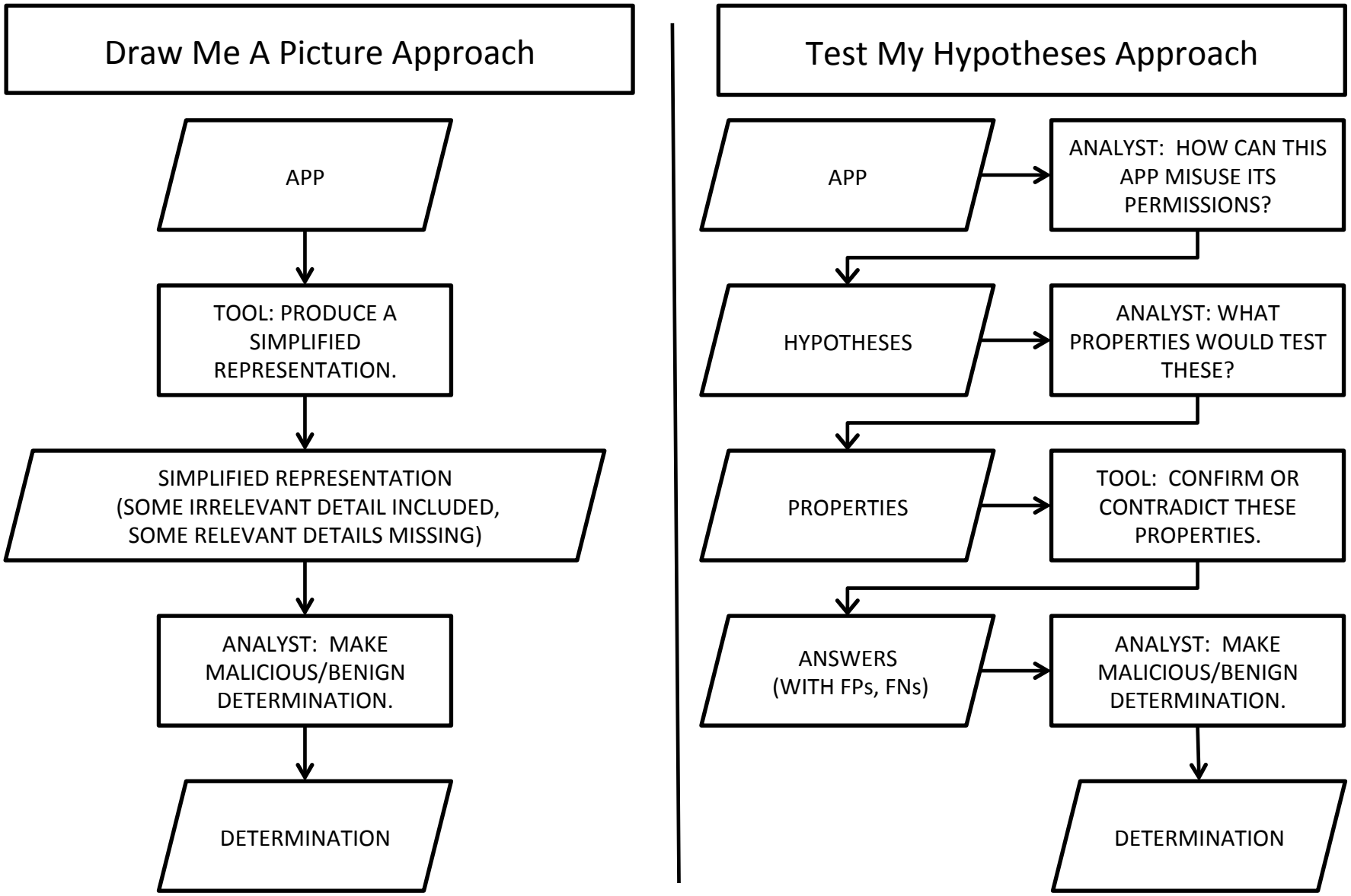
Stated purpose: video game that uses mDNS to do 30-second search LAN for other players.

Malice: trigger causes it to do 8-hour network recon search – long enough for user to carry the device into their private enclave.

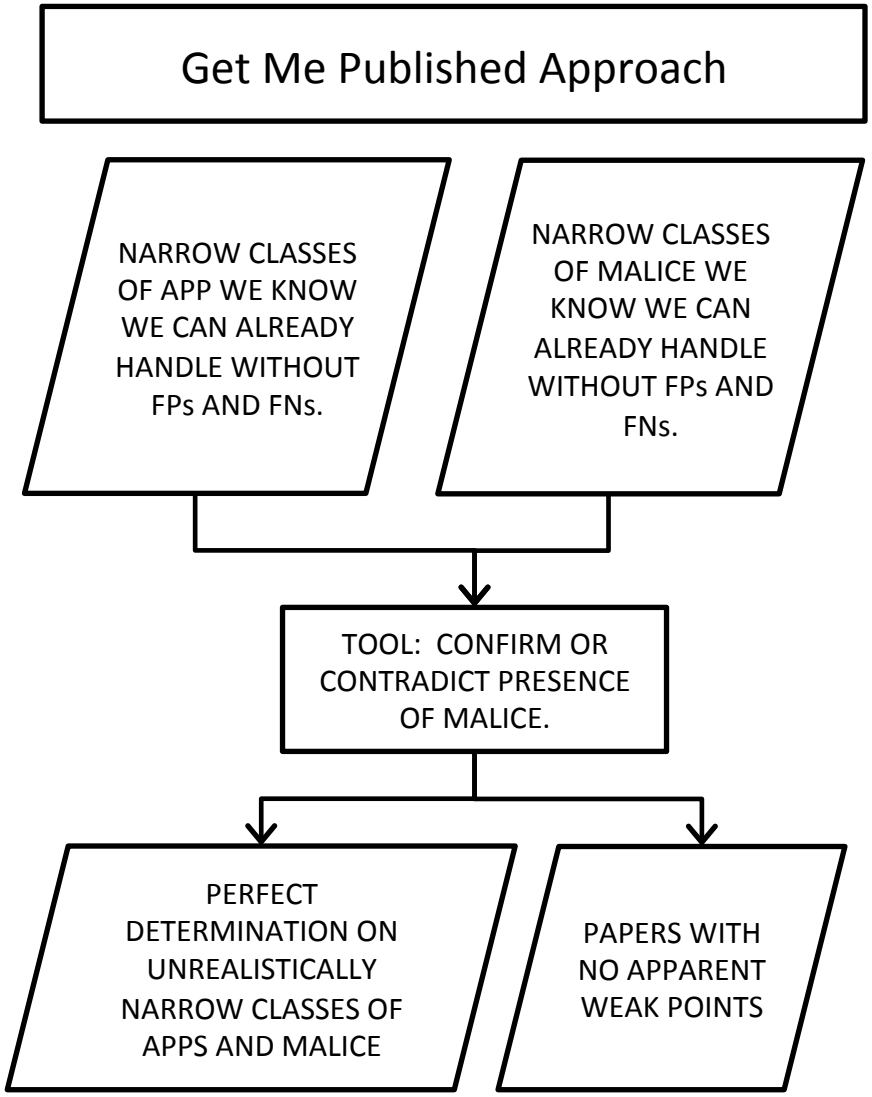
Trick: malicious functionality broken into 30-40 pieces spread throughout program.

Raytheon BBN - APAC Adversarial Challenge Team, April 2015.

Functional Schematics of APAC Approaches



The Need for the Adversarial Role in Research



The presence of Adversarial Challenge (AC) Teams discourages this approach.

The APAC AC Teams make Trojan apps with known malice in them.

"The adversary doesn't care about your carefully-worded definition of the problem."

Feedback on engagements at April 2014 PI Meet:

- MIT PI Gordon indicated that engagements forced MIT to increase precision beyond what was required to publish.
- Utah PI Might said participation in engagements "makes our lab stronger."

Example APAC Tool (UCSB's DarkDroid April 2014)

DarkDroid

Overview | Reports | Submit | About | Logout

ULTRACOOLMAP

APP MENU

- Overview
- Structure
- Data Flows
- Source Code

ANALYSIS

Success

DECISION

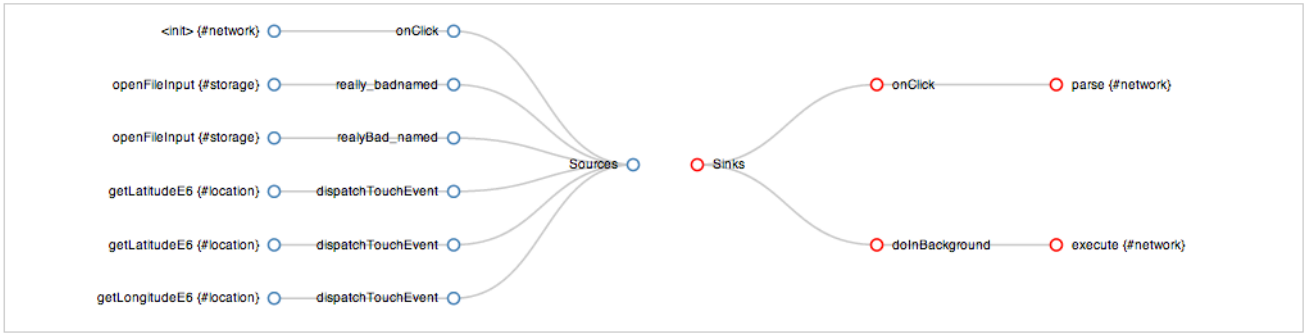
Malicious

FLOWS A9AC9F4A-75E9-40F5-8B10-903425DC0FDE

← Previous | Next →

{#NETWORK, #STORAGE, #LOCATION} → {#NETWORK}

1 / 2 FLOWS



DESCRIPTION

This data flow represents a transfer of information from `{#network, #storage, #location}` source(s) to `{#network}` sink(s). If this is unexpected behavior for this app, then the app should be marked as `malicious`.

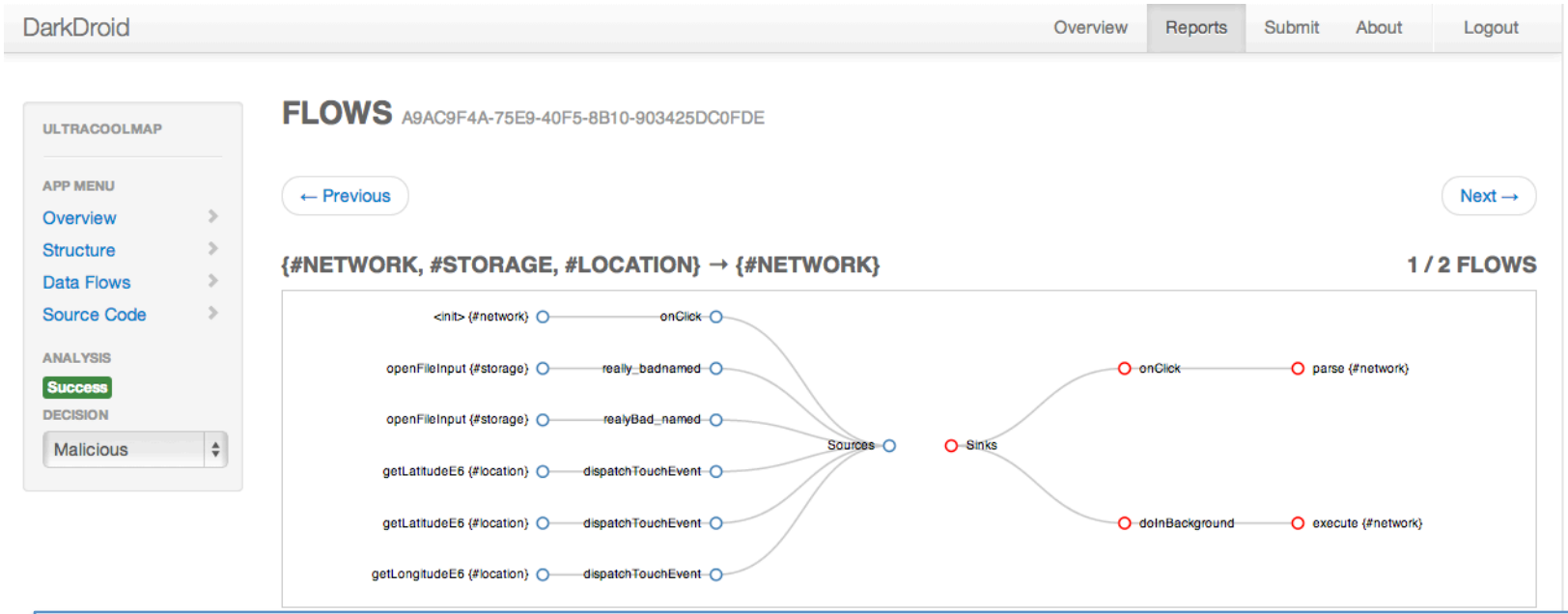
DATA SOURCES

Expression	.*
Call Site	com.ultracoolmap.UltraCoolMapActivity:dispatchTouchEvent(android.view.MotionEvent): boolean @ 352c
Call Target	int <=com.google.android.maps.GeoPoint:getLongitudeE6()
Classification	#location

DATA SINKS

Expression	.*
Call Site	com.ultracoolmap.UltraCoolMapActivity$ReallyBadName:doInBackground(java.net.URI[]): java.lang.Void @ 2de2
Call Target	<=org.apache.http.HttpResponse <=org.apache.http.client.HttpClient:execute(<=org.apache.http.client.methods.HttpUriRequest request)

Example APAC Tool (UCSB's DarkDroid April 2014)



Research that enabled tools like this one:

- Techniques to enable increasingly sound and precise mapping of control- and data-flows at scales sufficient to handle mobile apps,
- Techniques to overcome difficult language features like reflection and exceptions,
- Techniques to Improve value set analyses, particularly for strings, and
- Techniques to automate the summarization of the huge Android API.

Multiple Approaches Attacking the Same Problem

1. Automating the learning of new APIs will speed the development of tools on new platforms.

2. String value analysis is key to understanding communication via Android intents.

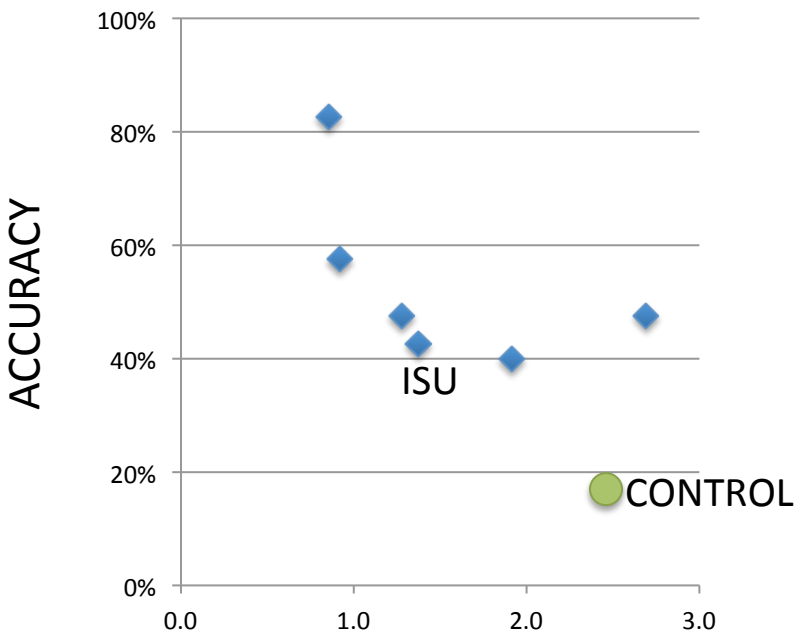
3. These are the key hard scalability problems with these two techniques.

4. Scalable precise points-to-analysis will make many higher-level analyses more scalable and precise. No clear best approach yet.

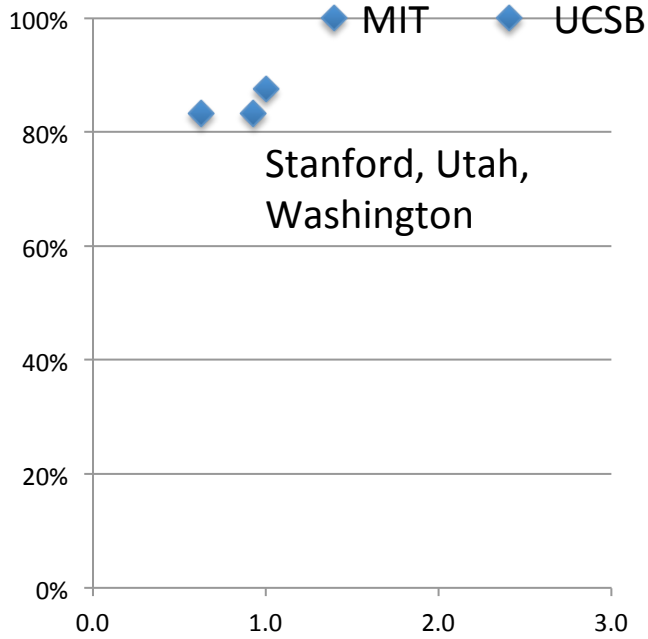
Research Challenges	ISU	UCSB	MIT	STANFORD	BAE	UTAH	UW
AUTOMATION FOR API COVERAGE		X		X			
DOES NOT NEED SOURCE		X	X		X	X	
EXTENSIBLE BY OPERATORS	X						
STRING VALUE ANALYSIS	X	X	X				
SEEKS HIDDEN TRIGGERS		X			X		
IMPROVED TYPE INFERENCE							X
REDUCED ABSTRACT STATE EXPLOSION						X	
IMPROVED CONCOLIC DIRECTION					X		
IMPROVED POINTS-TO ANALYSIS			X	X			

APAC Engagement Results

ENGAGEMENT 3
March-April 2014 (2 years in)



ENGAGEMENT 5
March-April 2015 (3 years in)

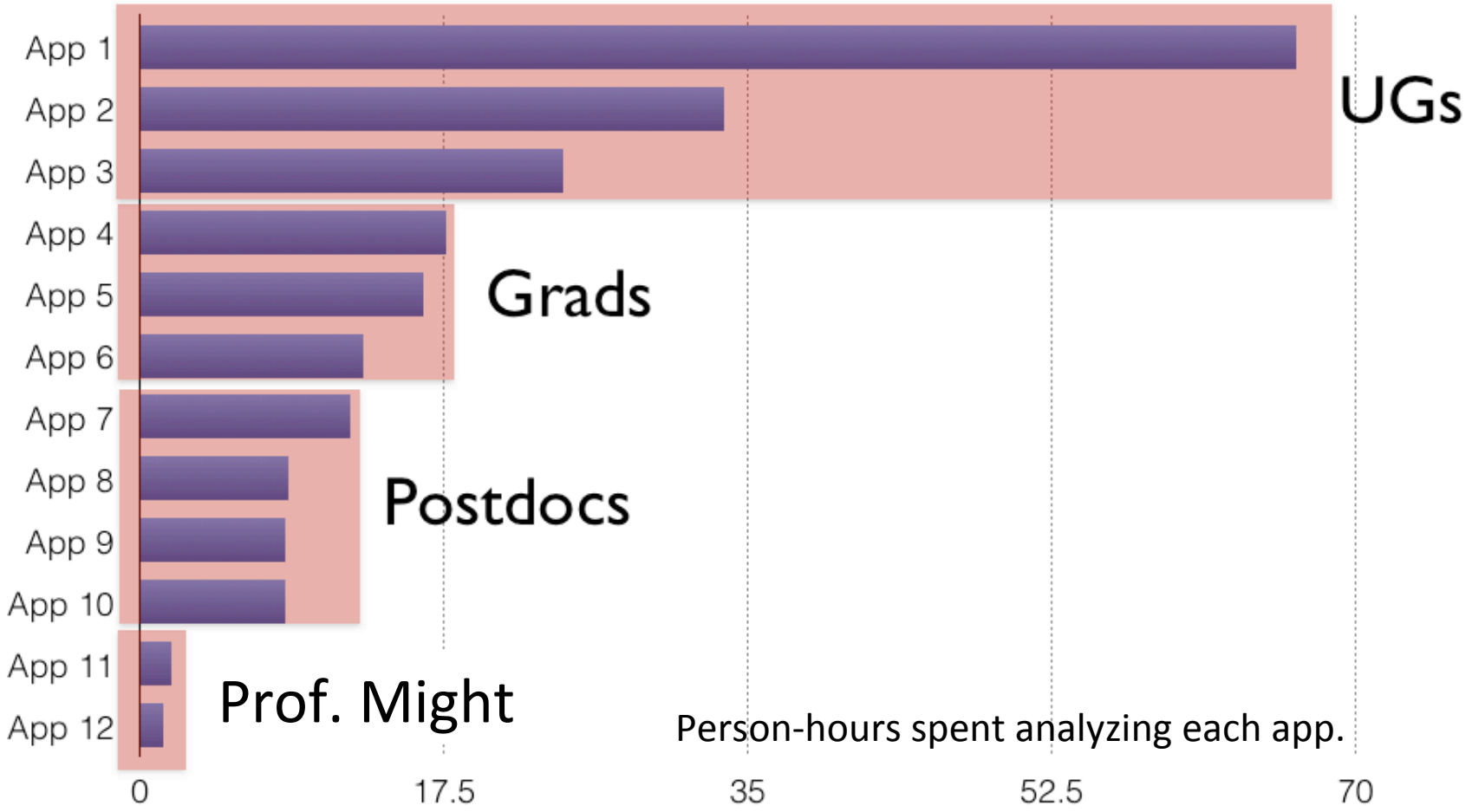


APPS ANALYZED IN 8 PERSON-HOURS

APAC after three years: using their APAC tools, analysts can vet about 1 app per day at >= 80% accuracy – far more accurately than the Control team using today's tools.

Raytheon BBN is APAC's Adversarial Challenge Team. Five Directions is the Experimentation Lead.

APAC Engagements Don't Account for Analyst Skill



Detail of U. Utah's analysis times for Engagement 5A March 2015.

More experienced analysts are much faster.

Direct Impacts of the APAC Program

Impact on Academic Research:

- Over 50 papers published.
- New areas of research opened, most significantly Automated API Modeling

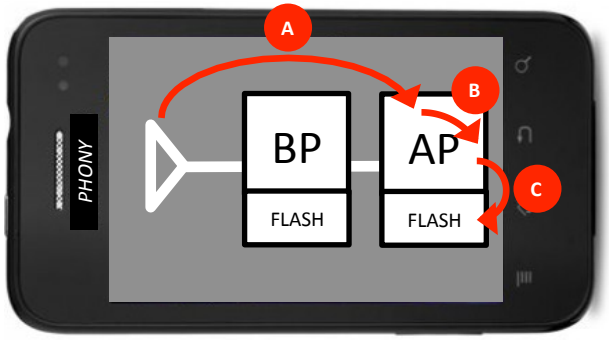
Impact on industry:

- From U. Washington's Michael Ernst: Oracle incorporated Washington's APAC code to support type annotations developed by APAC project into Java 8, and is improving this support to reflect experience from Washington's ongoing work in Java 9.
- From U. Utah's Matt Might: Shuying Liang took her APAC work on pushdown analysis, abstract garbage collection, and entry point saturation with her to HP Fortify and implemented it in their security auditing tool.

Impact on DoD: Upcoming pilot experiments with:

- US Army's Communications-Electronics Research, Development and Engineering Center (CERDEC)
- US Navy's Space and Naval Warfare Command Systems Center SPAWAR Pacific (SSC Pacific).

Hidden Malice and Accidental Bugs: Equally Dangerous



A. Get execution on the device.
B. Escalate privilege without proper authorization.
C. Modify software in Flash RAM.

A + B + C = our adversary can remotely reprogram the device.


```
-parseFloat("NAN(ffffe00572c60)")
```

CVE-2010-1807 Packetstormsecurity.org

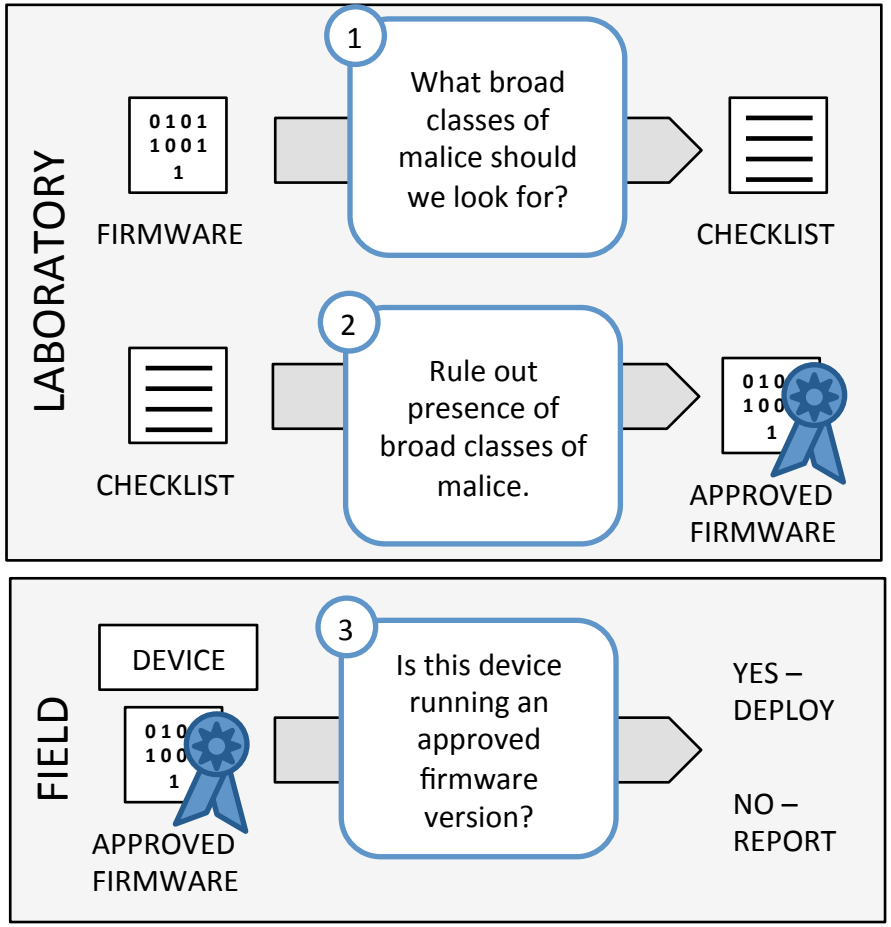

```
$ sync_agent ztex1609523
# id
uid=0(root) gid=0(root)
```

CVE-2012-2949 Pastebin.com

DoD needs an effective and efficient way of gaining measurable confidence that the COTS IT equipment it procures does not contain hidden malice.

VET: Broadening the Scope to Include New Problems

VET Workflow



Major Technical Challenges

"I have thought of all the possible malicious scenarios" is a difficult phrase to utter with confidence.

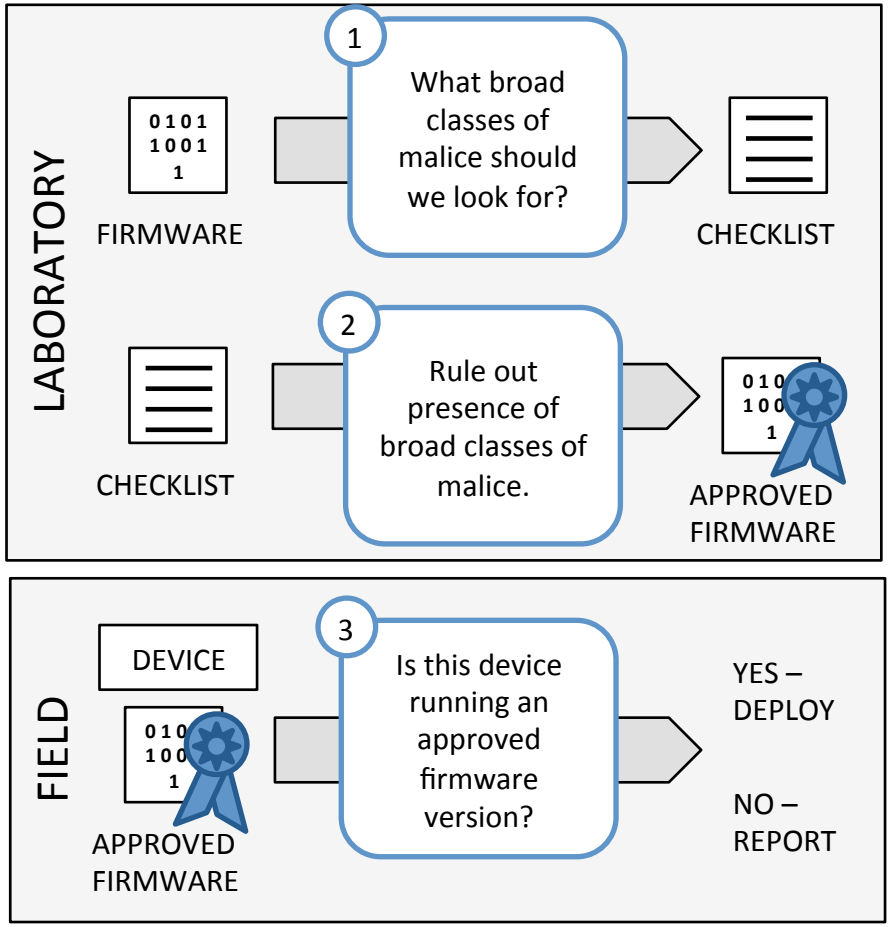
Demonstrating absence requires static program analysis – a fundamentally hard problem in computer science.

The adversary gets first-move advantage and can rig the device to lie about its health.

VET has a separate Technical Area for each step 1, 2, and 3 (R&D Teams).

VET: Broadening the Scope to Include New Problems

VET Workflow



Performers

Charles River Analytics
Raytheon BBN

CMU
MacAulay-Brown

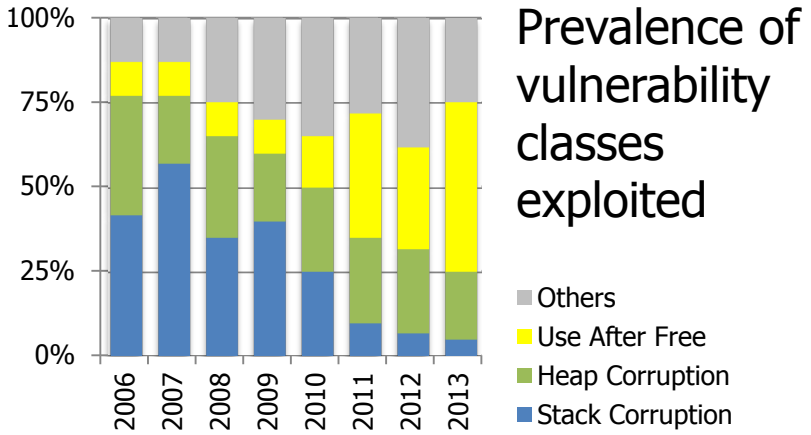
BAE
Grammatech
UCSB

USC/ISI

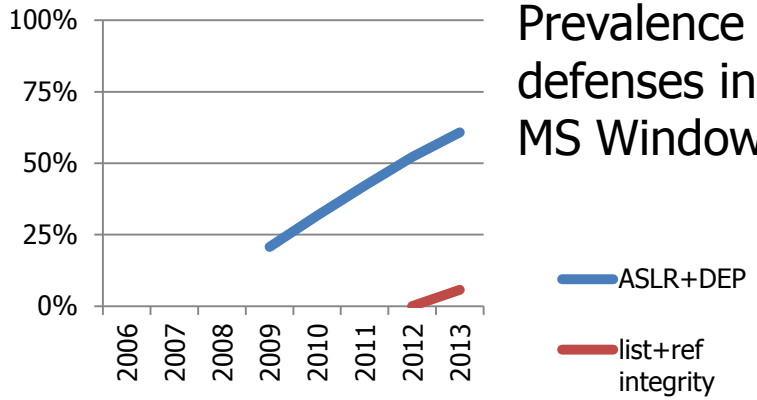
Shoshitaishvili et. al., "Firmalice: Automatic Detection of Authentication Bypass Vulnerabilities in Binary Firmware," NDSS 2015

Cyberpoint, Skaion are Adversarial Challenge Teams. Apogee is Experimentation Lead.

STAC: Facing the Next Generation of Vulnerabilities



Prevalence of vulnerability classes exploited



Prevalence of defenses in MS Windows

SOURCES: Microsoft 2013 Software Vulnerability Exploitation Trends and NETMARKETSHARE.COM

ALGORITHMIC COMPLEXITY ATTACKS

Small worst-case input causes a crippling space or time usage.

2011

Huge portions of the Web vulnerable to hashing denial-of-service attack

A flaw common to most popular Web programming languages can be used to launch …

by **Jon Brodkin** - Dec 28 2011, 2:25pm EST ARS TECHNICA

SIDE CHANNEL ATTACKS

Adversary deduces secrets by observing minute differences in space or time used.

2013

Gone in 30 seconds: New attack plucks secrets from HTTPS-protected pages

Exploit called BREACH bypasses the SSL crypto scheme protecting millions of sites.

by **Dan Goodin** - Aug 1 2013, 11:30am EDT ARS TECHNICA

Past: flawed implementations of algorithms. Future: flawed algorithms.

STAC: Facing the Next Generation of Vulnerabilities

R&D Teams:

- Draper Labs
- Grammatech
- Iowa State U
- Northeastern U
- U Maryland College Park
- U Utah
- Vanderbilt

Control Team:

- Invincea Labs

Adversarial Challenge Teams:

- Cyberpoint, Raytheon BBN

Experimentation Lead

- Apogee Research

ALGORITHMIC COMPLEXITY ATTACKS

Small worst-case input causes a crippling space or time usage.

2011

Huge portions of the Web vulnerable to hashing denial-of-service attack

A flaw common to most popular Web programming languages can be used to launch …

by **Jon Brodkin** - Dec 28 2011, 2:25pm EST ARS TECHNICA

SIDE CHANNEL ATTACKS

Adversary deduces secrets by observing minute differences in space or time used.

2013

Gone in 30 seconds: New attack plucks secrets from HTTPS-protected pages

Exploit called BREACH bypasses the SSL crypto scheme protecting millions of sites.

by **Dan Goodin** - Aug 1 2013, 11:30am EDT ARS TECHNICA

Past: flawed implementations of algorithms. Future: flawed algorithms.

www.darpa.mil