Surveillance of / with Small-scale systems

Chamseddine Talhi
École de technologie supérieure (ÉTS),
Montreal

Department of Software Engineering and Information Technology
Agenda

- Project Presentation
- Why surveillance of/with small-scale systems?
- Surveillance of Small-scale Systems
- Surveillance with Small-scale Systems
- Project Summary
- Feedback?
Agenda

- Project Presentation
- Why surveillance of/with small-scale systems?
- Surveillance of Small-scale Systems
- Surveillance with Small-scale Systems
- Project Summary
- Feedback?
Project Presentation

• New research thread to Advanced Host-Level
  o 1 year project! 2013/2014

• Team:
  o 1 professor
  o 2 Master students
  o 1 part time research professional
  o Part-time graduate/undergraduate students

• Objectives:
  o Surveillance of small-scale systems
  o Use of massively parallel small-scale systems for the surveillance of other systems
Agenda

• Project Presentation
• Why surveillance of/with small-scale systems?
• Surveillance of Small-scale Systems
• Surveillance with Small-scale Systems
• Project Summary
• Feedback?
Small-scale systems, why?

From Mobile Phones to general-purpose small devices

• « Cabir » 2004: first mobile phone malware
• « CommWarrior » & « Doomboot » 2005:
• And …

2 years of mobile malware evolution <=> 20 years of Computer malware evolution!!!

More than 1 000 000 variants of malware targeting Android
Small-scale systems, why?

Mobile malwares – Evolution
Small-scale systems, why?

Mobile malwares – 2013 Statistics

Small-scale systems, why?

Small-scale systems are not limited to Smartphones!
- Linux/Android based devices.
- Shodan: Computer Search Engine
Small-scale systems, why?

Shodan: Computer Search Engine
Privacy? Security?
## Small-scale systems, why?

- **Malwares in Embedded Systems: next (r)evolution!**

<table>
<thead>
<tr>
<th>Year</th>
<th>Malware /attack</th>
<th>Target</th>
<th>Threats</th>
</tr>
</thead>
<tbody>
<tr>
<td>2009</td>
<td>psyb0t</td>
<td>Linux-based routers and DSL modems</td>
<td>DDoS</td>
</tr>
<tr>
<td>2010</td>
<td><strong>Chuck Norris Botnet</strong></td>
<td><strong>Linux-based routers, DLS modems</strong></td>
<td>DDoS +DNS Spoofing</td>
</tr>
<tr>
<td></td>
<td>Stuxnet</td>
<td>industrial control systems (ICS)</td>
<td>alter PLCs for supported facilities</td>
</tr>
<tr>
<td>2012</td>
<td>DNSChanger</td>
<td>computers and routers</td>
<td>DNS spoofing/poisoning</td>
</tr>
<tr>
<td>2013</td>
<td>JUL: GPS attack</td>
<td>GPS based systems</td>
<td><strong>total control of system</strong></td>
</tr>
<tr>
<td></td>
<td>Sept: Linux/Flasher</td>
<td>wireless routers</td>
<td>login credentials captured and transferred to remote web servers.</td>
</tr>
<tr>
<td></td>
<td>Nov 26 : Linux.Darlloz</td>
<td>Linux-based computers, industrial control servers, routers, cameras, set-top boxes.</td>
<td>generates IP @ randomly, accesses a specific path on the machine with well-known ID and passwords, and sends HTTP POST requests.</td>
</tr>
</tbody>
</table>
Small-scale systems, why?

- Stuxnet Malware (2010)!
Small-scale systems, why?

- Resource Limitations

  Low power CPUs
  - Lightweight processing
  - Limited multitasking

  Battery life

  Memory limited to Megabytes

  [Image of battery status and memory usage]

  - Total Memory: 196,608 B
  - Free Memory: 31,256 B (15%)
  - Used Memory: 165,352 B (85%)
• Project Presentation
• Why surveillance of/with small-scale systems?
• Surveillance of Small-scale Systems
• Surveillance with Small-scale Systems
• Project Summary
• Feedback?
Signature vs. Anomaly Detection - Challenges

Signature-based detection
• *Best Multi pattern matching algorithms?*
• *Optimization:* data structures and algorithms, compression, parallel programming, etc.
• Need for Cloud/Server: signatures Database storage, Remote scan.

Anomaly-based detection
• *Machine Learning algorithms:* accuracy (eg. false positives), overhead (eg., memory and power, etc.)
• *Need for remote Cloud/Server:* traces storage and exchange
Signature Detection: Multi Pattern Matching (1)

Empirical study

- Required memory budget: varying numbers of signatures.
- Dataset: Android malwares signatures (MD5 hash).
- Memory budget compared with available memory on a Samsung Galaxy S Vibrant phone.
Evolution of Smartphone Memory

<table>
<thead>
<tr>
<th>year</th>
<th>phone model</th>
<th>memory size (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2002</td>
<td>Blackberry 5810</td>
<td>1</td>
</tr>
<tr>
<td>2003</td>
<td>BlackBerry 7210</td>
<td>6</td>
</tr>
<tr>
<td>2004</td>
<td>Nokia 6630</td>
<td>10</td>
</tr>
<tr>
<td>2005</td>
<td>HTC Universal</td>
<td>64</td>
</tr>
<tr>
<td>2006</td>
<td>HTC TyTN 100</td>
<td>64</td>
</tr>
<tr>
<td>2007</td>
<td>Iphone</td>
<td>128</td>
</tr>
<tr>
<td>2008</td>
<td>HTC dream</td>
<td>192</td>
</tr>
<tr>
<td>2009</td>
<td>HTC Magic</td>
<td>288</td>
</tr>
<tr>
<td>2010</td>
<td>Samsung Galaxy S</td>
<td>512</td>
</tr>
<tr>
<td>2011</td>
<td>Samsung Galaxy S2</td>
<td>1024</td>
</tr>
</tbody>
</table>
Empirical study

512 MB should be enough ... BUT

- Memory reserved by hardware = 32 MB
- Android fixed components = 80 MB
- Launcher = 30 MB
- Live wallpaper = 20 MB
- 5 widgets = 20 MB
- Android System = 208 MB

- Available memory = 121 MB!
Empirical study

512 MB should enough ... BUT
From 121 MB

- Video media player consumes 10.34MB
- Internet browser consumes 31.07MB

Finally: Available memory is 88MB!!!
Small-scale Sys Surveillance

Signature Detection: Multi Pattern Matching (5)

Empirical study
Signature Detection : Lessons learned

• Fast evolution of signatures database: memory of small-scale systems will never be enough!!!
• A subclass of most important signatures should be maintained
• subclasses of malwares => sub-databases
• Optimize, optimize, …., and optimize
Small-scale Sys Surveillance

Anomaly Detection (1)

Analysis of sys call n-grams

- Look-ahead pairs
- n-gram Trees
Small-scale Sys Surveillance

Anomaly Detection: sys call n-gram Analysis (2)

Lookahead pairs

n-gram Trees

<table>
<thead>
<tr>
<th>Appel système</th>
<th>1 appel après</th>
<th>2 appels après</th>
</tr>
</thead>
<tbody>
<tr>
<td>Appel 1</td>
<td>open</td>
<td>open, read</td>
</tr>
<tr>
<td></td>
<td></td>
<td>read, gettime, close</td>
</tr>
<tr>
<td>Appel 2</td>
<td>read</td>
<td>gettime, close</td>
</tr>
<tr>
<td></td>
<td></td>
<td>open</td>
</tr>
<tr>
<td>Appel 3</td>
<td>gettime</td>
<td>open</td>
</tr>
<tr>
<td></td>
<td></td>
<td>read</td>
</tr>
</tbody>
</table>
Possible optimization: Sorted n-gram Tree

Most frequent n-grams of Angrybirds game
- N-grams sorted according to their frequency inside the normal model
- => Improve analysis time
Small-scale Sys Surveillance

Anomaly Detection: sys call n-gram Analysis (4)

Experimental results – injecting 3 function calls in an open source application

Lookahead anomaly rate

n-gram Tree anomaly rate

3 new function calls injected in the application: traces 43-48
Anomaly Detection: sys call n-gram Analysis (5)

Experimental results – Lookahead model
Angrybird maliciously updated by Droid-KungFu malware

Maliciously updated Angrybird

=> Windows >= 5 are good candidates for anomaly detection
Small-scale Sys Surveillance

Anomaly Detection : sys call n-gram Analysis (6)

Experimental results – n-grams Model
Angrybird maliciously updated by Droid-KungFu malware

Maliciously updated Angrybird    “Safe” update of Angrybird

=> Windows >= 5 are good candidates for anomaly detection
Small-scale Sys Surveillance

Anomaly Detection: sys call n-gram Analysis (7)

- CPU Overhead

![Graph showing CPU Overhead vs N-gram size]

- Lookahead
- N-gram Tree
- Sorted n-gram Tree

CPU Overhead (%)

N-gram size
Small-scale Sys Surveillance

Anomaly Detection: sys call n-gram Analysis (8)

- Memory Overhead

![Graph showing Memory Overhead vs N-gram size]

- N-gram size

- N-gram Tree

- Lookahead
Signature vs. Anomaly Detection - Challenges

Signature-based detection
• **Best Multi pattern matching algorithms?**
• **Optimization**: data structures and algorithms, compression, parallel programming, etc.
• **Pragmatic approach**: periodicity, prioritized / specialized signatures, devices collaboration, alert-based, etc.
• Need for Cloud/Server: signatures Database storage, Remote scan.

Anomaly-based detection
• **Machine Learning algorithms**: accuracy (eg. false positives), overhead (eg., memory and power, etc.
• **Adaptive approach**: resource usage of the device, different speeds of the same algorithm, different algorithms, etc.
• **Need for remote Cloud/Server**: traces storage and exchange
Small-scale Sys Surveillance

Evaluation Boards

- PandaBoard,
- BeagleBoards
- Arndale Board,
- OMAP5432
Evaluation Boards : Use cases

BeagleBone Black:
- Spectrum Analyzer http://www.youtube.com/watch?v=6YhrKMBrJ2g
- Motor Controller http://www.youtube.com/watch?v=34xJIR-mD4A
- Game console http://www.youtube.com/watch?v=U4P_s-7dDRQ
- Web server http://www.youtube.com/watch?v=CDhyVdpXuqQ

Beagleboard-XM:
- Robot Controller http://www.youtube.com/watch?v=FZKtQLj8NLE
- Motor controller http://www.youtube.com/watch?v=bahmjwWKWIo
- Domotic Control System http://www.youtube.com/watch?v=eIAWYCFv0Rw

Pandaboard ES:
- Robot http://www.youtube.com/watch?v=ZWbZBBs9WSs
# Small-scale Sys Surveillance

## OMAP SOC

<table>
<thead>
<tr>
<th>Feature</th>
<th>BeagleBone</th>
<th>Overo® FE COM (Gumstix)</th>
<th>Gumstix (DuoVero) Zephyr COM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Manuf.</strong></td>
<td>BeagleBoard.org</td>
<td>Gumstix Inc</td>
<td>Gumstix Inc</td>
</tr>
<tr>
<td><strong>CPU</strong></td>
<td>AM335x, 720MHz ARM Cortex-A8</td>
<td>OMAP 3530, 600 MHz ARM Cortex-A8</td>
<td>OMAP4430, Dual-Core : 1 GHz, Cortex-A9</td>
</tr>
<tr>
<td><strong>GPU</strong></td>
<td>NEON (SIMD) 2D/3D graphics</td>
<td>OpenGL POWERVR SGX for 2D and 3D graphics acceleration</td>
<td>PowerVR SGX540 ™</td>
</tr>
<tr>
<td><strong>Memory</strong></td>
<td>256 MiB DDR2 4GB microSD, Cloud9 IDE on Node.JS</td>
<td>512 MB RAM 512 MB NAND microSD slot</td>
<td>RAM : 1GB microSD slot</td>
</tr>
<tr>
<td><strong>Features</strong></td>
<td>USB client and Host, Ethernet, 2x 46 pin headers, Power consumption 2w</td>
<td>Bluetooth and 802.11b/g, Performance up to 1,400 Dhrystone MIPS, Powered via expansion board (Overo series or custom) connected to dual 70-pin connector</td>
<td>Ethernet (10/100 Mbps) Wifi, Bluetooth, USB OTG Power: SmartReflex technologies</td>
</tr>
<tr>
<td><strong>OS</strong></td>
<td>Android, Linux</td>
<td>Linux distribution pre-installed. Android</td>
<td>Linux, Android</td>
</tr>
<tr>
<td><strong>Size</strong></td>
<td>76.2 ×76.2 ×16mm</td>
<td>58mm x 17mm x 4.2mm</td>
<td>58mm x 17mm x 4.2mm</td>
</tr>
<tr>
<td>SOC</td>
<td>Nautiz X1</td>
<td>Sabre-Tooth</td>
<td>SCORPIN H2</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------</td>
<td>-----------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>CPU</td>
<td>OMAP (TI)</td>
<td>MediaTek</td>
<td>Qualcomm</td>
</tr>
<tr>
<td></td>
<td>OMAP 4430, dual core, (1 GHz)</td>
<td>MT6515, dual-core (1 GHz)</td>
<td>Snapdragon S3, dual core(1.5GHz)</td>
</tr>
<tr>
<td>Memory</td>
<td>RAM : 512 MB, flash: 4 GB, MicroSD card slot</td>
<td>RAM : 512 MB</td>
<td>RAM : 1MB, Flash : 16 GB, expandable to 32GB micro SD</td>
</tr>
<tr>
<td>Connectivty</td>
<td>GSM, CDMA, GPS, Bluetooth, 802.11 b/g/n WiFi</td>
<td>Wi-Fi: 802.11 b/g/n, 2G: GSM, Bluetooth</td>
<td>3g/4G compatible, Wi-Fi 802.11 and Bluetooth, GPS</td>
</tr>
<tr>
<td>Connectors</td>
<td>E-compass and G-Sensor, Extended battery, Vehicle cradle, 5-megapixel camera, LED flash</td>
<td>2x GSM, Micro SD Card Slot, Micro USB, Gravity and Linear Acceleration Sensor</td>
<td>tactical data radios, extended battery life</td>
</tr>
<tr>
<td>features</td>
<td>survive humidity, vibration, drops/extreme temperatures. waterproof and impervious to dust and sand. runs Android 4.0</td>
<td>Water Resistant, Shockproof, Dustproof, Battery Standby: 72 Hours, dimensions: 136x75x18mm , weight: 144g</td>
<td>run/charge simultaneously via USB port, batteries, or vehicle power. vibration, shock, drop, humidity Runs Android 4.0</td>
</tr>
</tbody>
</table>
• Project Presentation
• Why surveillance of/with small-scale systems?
• Surveillance of Small-scale Systems
• Surveillance with Small-scale Systems
• Project Summary
• Feedback?
- Massively parallel small-scale embedded systems
- Opportunities for better performance of surveillance techniques
Massively parallel small-scale embedded systems

GPUs: Expanding field for massively parallel computing

- A Graphics Processing Unit: A co-processor that takes on graphical calculations and transformations so that the main CPU does not have to be burdened by them
- GPUs are the most used platforms for massively parallel programming systems.

- Accelerates data-parallel computation while reducing system work load
- Reduces platform energy consumption while increasing system throughput
- Enhances your system's value by consolidating functionality while reducing programmer effort
Massively parallel small-scale embedded systems

Evolution of Embedded GPUs

- PowerVR 5XT
- Mali T604
- Adreno 320
- PowerVR 6
- Adreno 330
- Mali T628
- Tegra 5

40% more GFLOPS/quarter

Estimated at sustained peak performance. Likely to be much less in practice.
Mobile Compute driving Imaging use cases

- Requires *significant* computing over *large* data sets
Massively parallel small-scale embedded systems

Parallella: Super computing for everyone

- Project goal: to democratize access to parallel computing through providing an affordable open hardware platform and open source tools
- The Parallella platform is an open source, energy efficient, high performance, credit-card sized computer based on the Epiphany multicore chips developed by Adapteva.
Opportunities for better performance of surveillance

- Accelerating/optimizing surveillance Using Multithreaded Algorithms
Small-scale Sys 4 Surveillance

Opportunities for better performance of surveillance

Parallel processing of input stream with *Boundary detection problem!*

Parallel processing of input stream with *Overlapped segments*

Accelerating String Matching Using Multi-Threaded Algorithm on GPU  GLOBECOM 2010
Opportunities for better performance of surveillance

AC automaton without failure transitions

Accelerating String Matching Using Multi-Threaded Algorithm on GPU  GLOBECOM 2010
Opportunities for better performance of surveillance

Promising improvements !!!

**TABLE 1: THROUGHPUT COMPARISON OF THREE APPROACHES**

<table>
<thead>
<tr>
<th>Input streams</th>
<th>CPU_AC Throughput (KBps)</th>
<th>Direct_AC Throughput (KBps)</th>
<th>PFAC Throughput (KBps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal Case</td>
<td>997</td>
<td>6,428</td>
<td>3,963,966</td>
</tr>
<tr>
<td>Virus Case</td>
<td>657</td>
<td>4,691</td>
<td>3,656,217</td>
</tr>
<tr>
<td>Ratio</td>
<td>1</td>
<td>~6.4</td>
<td>~4000</td>
</tr>
</tbody>
</table>

**TABLE 2: MEMORY COMPARISON**

<table>
<thead>
<tr>
<th>Conventional AC</th>
<th>PFAC</th>
<th>Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>states, transitions, memory (KB)</td>
<td>states, transitions, memory (KB), Reduction</td>
<td></td>
</tr>
<tr>
<td>Snort rule*</td>
<td>8,285, 16,568, 143</td>
<td>8,285, 8,284, 114, 21%</td>
</tr>
<tr>
<td>Ratio</td>
<td>1, 1, 1</td>
<td>1, 0.5, 0.79</td>
</tr>
</tbody>
</table>

* The Snort rules contain 994 patterns and total 22,776 characters.

Accelerating String Matching Using Multi-Threaded Algorithm on GPU  GLOBECOM 2010
## Opportunities for better performance of surveillance

<table>
<thead>
<tr>
<th>Work</th>
<th>Mechanism</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Malware detection</strong></td>
<td></td>
</tr>
<tr>
<td>GrAVity: A Massively Parallel Antivirus Engine</td>
<td>Applying a signature filter on GPU</td>
</tr>
<tr>
<td>A Taxonomy and Comparative Evaluation of Algorithms for Parallel Anomaly Detection</td>
<td>Combining different classes of anomaly detection algorithms and address the question of which combination of existing anomaly detection algorithms achieves the best detection accuracy.</td>
</tr>
<tr>
<td>An Efficient Parallel Anomaly Detection Algorithm Based on Hierarchical Clustering</td>
<td>Parallel processing of training and predicting phase Both phases have the same excellent detection performance with serial processing, and it also has better real time performance than serial processing</td>
</tr>
<tr>
<td><strong>Pattern Matching</strong></td>
<td></td>
</tr>
<tr>
<td>Accelerating String Matching Using Multi-threaded Algorithm on GPU</td>
<td>Proposing a novel algorithm that reduces the complexity of Aho-Corasick Algorithm The new algorithm on GPUs achieves up to 4,000 times speedup compared to the AC algorithm on CPU</td>
</tr>
<tr>
<td>A gpu-based multiple-pattern matching algorithm for network intrusion detection systems</td>
<td>A GPU-based pattern matching algorithm for NIDS has been proposed in this work. The proposed pattern matching algorithm is based on the concept of WM algorithm. The performance of the proposed approach is around twice of that of the MWM algorithm employed in Snort and can be applied on host-based antivirus systems.</td>
</tr>
<tr>
<td>Bit-Parallel Multiple Pattern Matching</td>
<td>Extension of the bit-parallel Wu-Manber algorithm to combine several searches for a pattern into a collection of fixed-length words. Presenting an OpenCL parallelization of a redundant index on massively parallel multicore processors, within a framework of searching for similarities with seed-based heuristics. Some speedups obtained with gpu are more than 60× on cpu.</td>
</tr>
</tbody>
</table>
Agenda

• Project Presentation
• Why surveillance of/with small-scale systems?
• Surveillance of Small-scale Systems
• Surveillance with Small-scale Systems
• Project Summary
• Feedback?
Ongoing Activities

• Signature based detection:
  o Experimenting existing tools:
    ▪ Antimalware for Smartphone
    ▪ Antimalware for embedded systems
  o Optimized pattern matching algorithms

• Anomaly-based detection:
  o Features selection
  o Lightweight and optimized algorithms
  o Adaptive algorithms
  o Experimenting and adapting algorithms developed by collaborators: Concordia University
Time for Feedback