IT Services

University of Reading

Intrusion Detection and Network Monitoring …

… On No Budget

Chris Wakelin – University of Reading

April 15, 2011

University of Reading

Introduction

- Why Monitor
- Performance Issues
- Solutions
- Flow capture
- University of Reading Setup
- Results

University of Reading

Why Monitor?

- Socially-engineered Malware difficult to stop
  - Look for compromised PCs
- Look for visits to “bad places”
  - Phishing forms
- Verify reports of abuse
  - Copyright infringement
  - Audit trail of compromises
- Gain insight into the use of the network
  - Who’s using IPv6?

University of Reading

Performance

- Inspired by talks at NetWorkshop/JANET CSIRT
- Particularly talk by David Ford of Oxford
  - "Effective use of Snort on Large Networks"
    - http://www.oucs.ox.ac.uk/network/security/documents/20101020Snort.pdf
- Described their (expensive) hardware solution
  - Also described the importance of tuning the ruleset

Performance: The problem

University of Reading

- Networks are very fast
- We now have 2x10Gb/s links to JANET
  - Utilisation peaks at about 500Mb/s
    - 700Mb/s on Residential Network
  - Something like 80,000 packets per second
    - 120,000 packets per second on Residential Network
  - May grow substantially!

Performance: The problem

University of Reading

- Ran Snort on an (old) quad core machine
  - Dropped 10% of the packets
  - Only one of the CPUs was actually used ...
- CPU cores have pretty well ceased to increase in speed
  - But increase in number of cores instead
- Hardware solutions exist but are expensive

Performance: The problem

University of Reading

- Quite a debate over whether it is beneficial to use multiple cores
  - See e.g.
    - http://vrt-blog.snort.org/2010/06/single-threaded-data-processing.html
    - http://www.inliniac.net/blog/2010/07/22/on-suricata-performance.html
- Argument is that you may lose as much in caching loss and memory transfers between cores as you gain from more CPU cycles

Solutions: PF_RING

University of Reading

- www.ntop.org/PF_RING.html
- Open source packet capture handling system for Linux
  - Written by Luca Deri of nTop fame
- Kernel module
  - (optional) modified ethernet drivers
    - mostly Intel, others you may have to patch yourself
- Userland libraries
  - Own libpfring (and API)
  - Modified libpcap (plus tcpdump)
    - Recompile “legacy” apps to use it
  - Snort daq library

Solutions: PF_RING

University of Reading

- Optimised to reduce copying of packets in memory
- Supports clustering
  - Several threads or instances of the same application can each see a portion of the traffic
  - Cluster by network flow or round-robin
  - (Sort of) the opposite of network interface-bonding
- Can also do extra things like packet defragmentation
- Optionally pass on packets to non-PF_RING apps

Solutions: PF_RING

University of Reading

- Luca has spent years trying to boost performance
- Special card-specific versions can do more
  - Especially Intel 82599-based (igb, ixgbe)
    - cards can split traffic into queues split between CPU cores
    - PF_RING + TNAPI can create virtual ethernet interfaces
    - Haven't tried it (we have older cards)
    - Small charge towards development cost (may be free to academia if we ask nicely ☺)
- Emphasis on getting the most from "commodity" hardware

Solutions: Suricata

University of Reading

- New IDS kid on the block
- Product of OISF - Open Information Security Foundation (www.openinfosecfoundation.org)
  - Partly funded by the US Department of Homeland Security
- GPL-ed engine built from scratch
- Mostly compatible with Snort rules
- Extra features

Solutions: Suricata

University of Reading

- Designed to be multi-threaded
  - Threading overhead means it still doesn't scale according to the number of cores
- Automatic protocol detection
  - No need to specify HTTP ports
  - Optionally log all HTTP requests
- Native support for PF_RING
  - splitting up the traffic for Suricata's threads
- Experimental support for CUDA
  - Offload pattern-matching to high-end graphics cards

Solutions:
Suricata Roadmap

University of Reading

- IP/DNS reputation
- File extraction/inspection
- Scoring thresholds
- Stateful pattern-matching/transaction-awareness
- …

University of Reading

Network Flow Capture

- Flows based on ‘5-Tuple’ criterion
  - Part of flow if source/destination address/port and protocol (TCP/UDP) all match within given timeout period
- Who talked to what, when
  - But not (much of) what they said
- Useful for
  - Auditing
    - Who visited reading-ac-uk.phishers.com
  - Measurement
    - How much IPv6 are we seeing from Eduroam?
    - Who’s using our bandwidth?

University of Reading

Network Flow Capture

- Network Switches
  - NetFlow/ Sflow etc.
  - Usually “sampled”, e.g. only 1/256 packets
- ARGUS 3.x (www.qosient.com/argus)
  - Open-source, mature flow-capture tool
  - Good command-line reporting tools
- nProbe (www.ntop.org/nProbe.html)
  - Open-source, small fee (again may be free if you ask nicely ☺)
  - Supports FastBit (column-orientated + bitmap index) databases
    - Very fast, but reporting tools rather rudimentary
  - Native PF_RING support (naturally!)

University of Reading Installation

University of Reading

- Have some old Dell servers
  - Two ex-VMWare servers with 8 CPU cores and 16GB of RAM
  - Two with 2-hyperthreaded CPU cores and 4GB of RAM
  - All have dual Intel 82571EB (e1000e) 1-Gb ethernet cards
  - Ubuntu 10.04
- Mirrored streams from core and border switches
- PF_RING + PF_RING-enabled e1000e driver
  - “transparent_mode=2” (only PF_RING-enabled apps see the traffic) and “enable_tx_capture=0” (only capture inbound)
    - Small increase in performance

University of Reading Installation

University of Reading

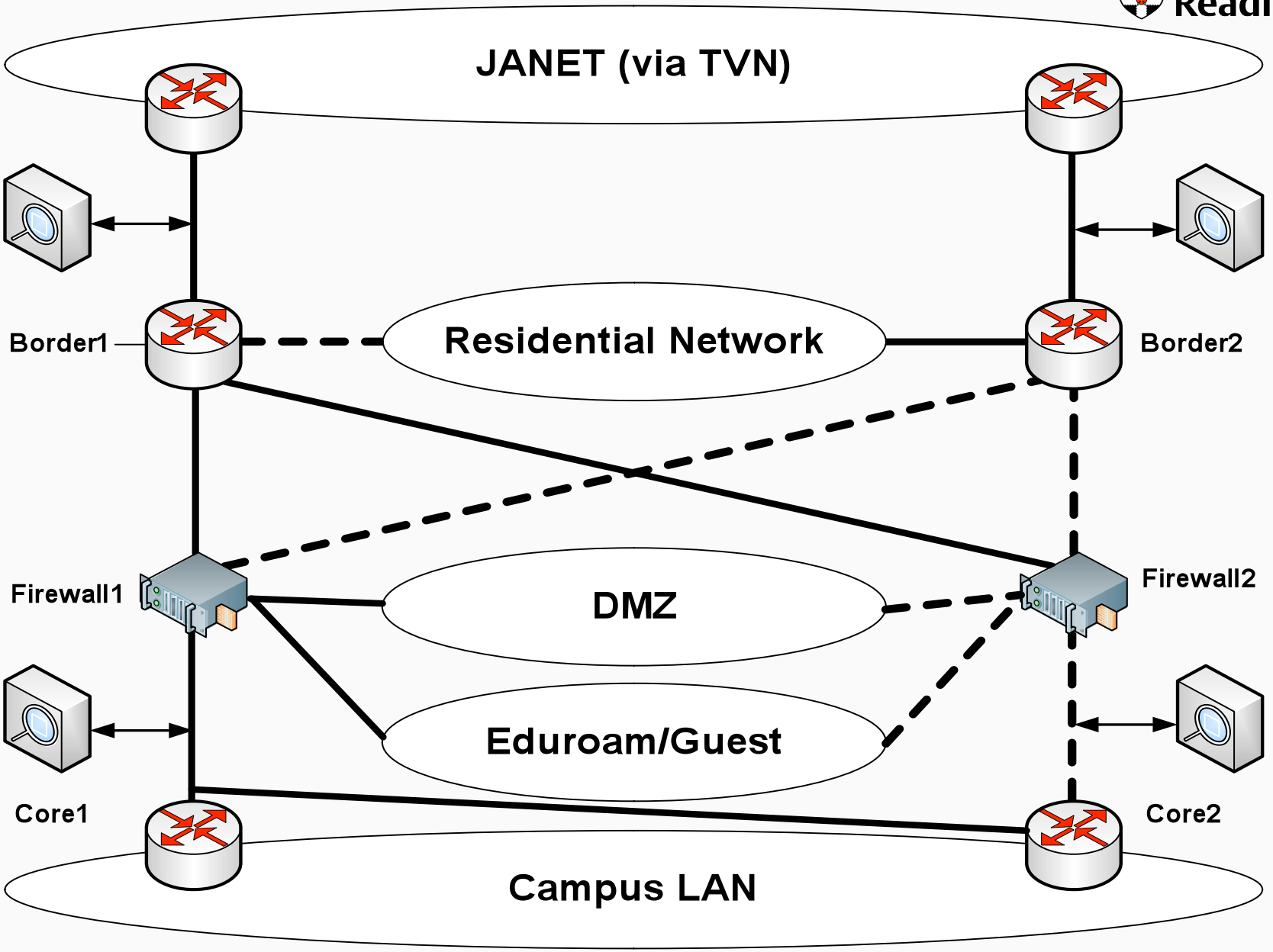
- Running ARGUS on all four
  - Compiled with PF_RING-enabled libpcap
  - Keeping 2-7 days worth of logs online
  - Log first 128 bytes of each flow as well
    - Enough to see most of a HTTP request
- Duplicated traffic for campus to/from the internet
  - the important stuff ☺
  - TODO – aggregate it using “radium”

University of Reading Installation

University of Reading

- Running Suricata on the two 8-core machines
- Using rules
  - Emerging Threats (http://emergingthreats.net)
    - ET Trojans/Current_Events (Malware/Viruses/Botcc)
  - some custom anti-phishing rules
  - ruleset built from APER phishing_links list
  - Important to tune the ruleset (or you'll be swamped!)

University of Reading



University of Reading Installation

University of Reading

Where	Monitoring (usually)	Tools
Border1	All traffic between internet and campus (including DMZ and Eduroam)	Suricata (IDS) ARGUS (flows) nTop (flows collected from Firewall1 monitor)
Border2	All traffic between internet and ResNet	Suricata (IDS) ARGUS (flows)
Firewall 1	Traffic between departments, internet, DMZ and Eduroam	ARGUS(flows) nProbe (flows sent to nTop on Border2 monitor)
Firewall 2	Nothing much (except failover)	ARGUS(flows)

University of Reading

Results

- Most interested in botnet infections of staff PCs
  - Then perhaps students on Residential Network + Wireless
  - Particularly Zeus
    - Spurt of infections in early December (6 PCs in a week)
      - ARGUS helped
      - Infection was via scam DHL e-mails with zipped EXEs
- Get lots of “Fun Web products” type spyware/adware
  - Cold probably remove those with Sophos policy
- Compromised Departmental Web Server
  - ARGUS logs told us how (insecure phpMyAdmin installation)

University of Reading

ToDo

- Consolidation of ARGUS logs
- Monitor our remote campus (Greenlands) too
- Streamline getting infected PCs cleaned up
- Anomaly detection in flows

University of Reading

Questions?