Implementation of NAC at ORNL

Paige Stafford
Summer 2009
ESCC/Internet2 Joint Techs
Indianapolis, IN
July 19-24, 2009
Outline

• Background
  – ORNL’s network
  – NAC defined
  – Origins of ORNL’s NACmgr

• NACmgr implementation
  – Focus on Detection (Polling)

• Future direction and conclusion
ORNLS’s Network

- 4000 employees
- 3000 guests
- 2 class B’s
- Segmented
  - 10 Enclaves
  - 32 routers
  - +900 subnets
  - +600 switches
  - +20,000 registered devices
- Mix of Cisco, 3Com, Foundry
ORNLS Network, Cont.

- +98% clients DHCP enabled
  (mostly Windows)

- Wireless Network
  - WPA
  - DHCP access only
  - Available in all buildings
  - Visitor Network
NAC Implementation Choices

- Looked seriously at Cisco, Lockdown
  - Both Required supplicant
    - user implementation hurdle
  - Cisco
    - Too expensive (have to replace non-Cisco) (total ~$3M)
  - Lockdown
    - Better price
    - Accommodates current switch vendors
    - uncertainty: viability/service support
NAC Implementation Choices, Cont.

• Not ready for COTS quite yet…

• Looked at the nuts and bolts
  – “…we really could do this ourselves…”
  – Started with detection and enforcement

• So, what is NAC?
NAC Defined

Network Access Control (NAC) is a set of technologies and defined processes that aim to control access to the network, allowing only authorized and compliant devices to access and operate on a network.

Here are the elements:

1. Detection
2. Quarantine and Remediation
3. Enforcement
4. Post-Admission Protection
5. Authentication
6. Compliance
7. Authorization

From Ofir Arkin: Bypassing NAC v2.0
ORNL Already Had Most Elements

NACmgr is part of and ties together ORNL’s NAC system
NACmgr

Implementation
NACmgr Implementation – Detection

- Poll all switches/routers
- Every 300 s
- Using SNMP
- Information stored for Detection
  - MAC address (defines the host)
  - IP address(es) used by MAC address
  - Switch and Port
  - Vlan and Router
  - Date/time First and Last polled
NACmgr Implementation – Enforcement

• If DHCP-client
  – “Quarantine”
    • DHCP issues special IP configuration to host
    • requests Issued by ORNL’s compliance monitoring system
    • Changes network registration status
      – Triggers change in DHCP configuration
    • Sends notification to owner, sysmgr

• Else
  – “L2-block”
    • Disable access at the switch
    • MAC drop / Port Disable
NACmgr Implementation – Enforcement, Cont.

- Detects unregistered, non-DHCP clients
  - L2-blocks these
    - DROP mac on Cisco
    - Disable port on 3Com, Foundry

- Masquerading MAC address monitoring
  - Monitors ARP caches
    - for MAC addresses showing up in more than one LAN
NACmgr Implementation – Enforcement, Cont.

• Enforcement must be monitored
  – Since host can move
    • From DHCP enabled to static (and visa versa)
    • To a different port/switch/network
NACmgr – Technical Details

• Large Primary Server
  – Web interface (Apache)
  – NAC database (PostgreSQL)
  – Outpost Server

• Outpost Server (2)
  – Primary duty is polling
  – One or many virtual outposts
NACmgr Code Specifications

• Programming language
  – Researched benchmarks of execution time of hash algorithms of C, C++, Java
    • C++ came out on top
    • http://bruscy.multicon.pl/pages/przemek/java_not_really_faster_than_cpp.html
    • http://members.lycos.co.uk/wjgoh/JavavsC.html
    • http://www.kano.net/javabench/data

• Libraries
  – Net-SNMP: SNMP library for C
  – pqxx: PostgreSQL library for C++
  – Oracle (Network Registration)
  – RudeCGI: C++ CGI library (web interface)
  – pThreads: POSIX threaded library
NACmgr’s Network Model

- Network has three parts
  - L3 (router)
    - Arp Caches
    - One or many Vlans
  - Vlan
    - Ties L3 to L2
    - One to many subnets
  - L2 (switch)
    - Bridge Table
NACmgr Database Model

Corresponds to the Network Model
NACmgr Polling Optimization, Cont.

- Each L3-network is assigned to an Outpost
  - Load Distribution is optimized among outposts
NACmgr Polling Optimization, Cont.

- L3 distribution example

NACmgr web interface

<table>
<thead>
<tr>
<th>Outpost</th>
<th>status</th>
<th>Date/Time Last Polled</th>
<th>Duration (seconds)</th>
<th>L3 Networks Defined</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAC1</td>
<td>ACTIVE</td>
<td>2008-04-01 12:59:34</td>
<td>109</td>
<td>ICS-TEST-CORESW1, PHYSICS_LANS, SWGE1505, SWGE3025, SWGE6010, SWGECSB-2, SWGEJICS</td>
</tr>
<tr>
<td>NAC2</td>
<td>ACTIVE</td>
<td>2008-04-01 12:58:04</td>
<td>69</td>
<td>SWGE4500S, SWGE7603</td>
</tr>
<tr>
<td>NAC3</td>
<td>ACTIVE</td>
<td>2008-04-01 12:59:21</td>
<td>68</td>
<td>NTRCGWY, SWGE1060, SWGE2525, SWGE4500N, SWGCNMS, SWGESNS</td>
</tr>
<tr>
<td>NACmgr</td>
<td>ACTIVE</td>
<td>2008-04-01 12:59:16</td>
<td>61</td>
<td>MECFW1, SWGECSB-1, WIRELESS3750</td>
</tr>
</tbody>
</table>

NACmgr polling time snapshot taken at 13:00

The L3 Network model
NACmgr Polling Optimization, Cont.

• Processing time constraint on data
  – Depends on number of hosts
  – Network Latency is relatively insignificant
  – Outposts complete all polling within 50-90 s
    • Dependent on on time of day
    • e.g. 08:00 load higher than that at 20:00
NACmgr Limitations

• The host is already on the network
  – Before non-compliance is detected

• Switches must be set up correctly
  – Passwords, SNMP access, TTL, etc

• Wireless Network doesn’t poll Access Points
  – Polling and blocking is at the L3 only
Future Direction

• Room for Improvement of existing tasks
• Adding more tasks/functions
  – Adding SHUNs to mix
  – Looking to use vlan assignment at L2 port
• Centralizing ORNL’s NAC systems
• Looking to hire another programmer

[accepting applications now]
Conclusion

- NACmgr is part of the NAC system at ORNL
  - Accommodates current network
  - Managed Out-of-Band, no client supplicant
  - Simple deployment and operation
  - Cost Effective
  - Good solution
    - vs. No Solution
    - Effectively detects and enforces compliance
    - no NAC solution is 100%
Questions?