Disarming offense to facilitate defense

Emilia Rosti
Dip. di Informatica e Comunicazione
Università` degli Studi di Milano
Background

• The motto that has characterized network security research so far can be phrased as follows:

“Networks are dangerous places, evil is outside (inside), protect your computing resources as much as you can“
Background

• People hired consultants for threats analysis
• People started building walls around their computers (firewalls)
• People started monitoring their networks and looking for suspicious behavior (IDS)
The common perspective

- Defense-centered computer security
  - security as a property of individual systems
  - security as the result of stronger in-depth defenses
    - firewalls, IDS, access control systems, etc.
Background

• In the last five years (too few) people have been involved in such a “rush for protection”, but we missed a crucial point
  “even if well protected your system can be subverted and become an attacker i.e., a threat for the entire community“
Background

- Some secondary effects of such an approach
  - a network with 100 Million potential attackers
  - difficulties in finding solutions to popular forms of attacks
    - DoS, Spoofing
  - difficulties in finding solutions to legal problems
    - liability
...and the future

• As end-to-end traffic encryption protocols spread (SSL, IPSEC, VPN), protection mechanisms will be less effective

• A dramatic increase of poorly protected computers such as videogame boxes and domestic appliances connected on the network

• An ever increasing quality of attack techniques
Our proposal

• It is our belief that a solution to the problems mentioned before can be obtained by reverting the
  • “protect yourself” approach
• into
  • “protect the others from yourself”
Our proposal

• Security as a global property
  • a network where only few hosts are a threat

• Security as a result of
  • individual harmlessness
  • in-depth defenses
Our proposal

• Individual harmlessness
  • innocuous COTS computers
    • prevent harmful activities originating from a host
  • “guarantee” inoffensiveness
    • protect from liability
Our proposal

• **DISARM COMPUTERS**
  
  “a disarmed computer is a host equipped with tools that turn off the host attacking capabilities and that force the host to be re-installed for it to be subverted“

• Two main ingredients
  
  • a “network reference monitor”
  
  • its kernelization
Our proposal

• Network “reference monitor”
  • monitors host activity
  • filters out “offensive” activity
  • blocks attacks at the source
    • characterization of attacks at the source
    • signatures for attacks that can be prevented at the source
• not a real reference monitor
  • cannot be proved correct and secure
Our proposal

• Kernelization
  • force system reboot and system re-install in order to remove the reference monitor
    • password protected install/remove
  • user transparent system component
    • prevent easy exploitation of hosts as unaware participants of attacks
    • protect intranets from inside
Our proposal

• Main effects
  • intercept encrypted traffic at source
  • guarantee trustworthyness of network hosts
    • can help to build white lists
  • can improve FW & IDS performance through white-lists
  • first defense against script kiddies
  • first response to some liability problems
Criticisms

• No silver bullet
  • can be bypassed (ASIC implementation?)
• Geographic deployment necessary for effectiveness
• Update & extensions
• Marketing impact of “restrained” computers
A disarming module

- Is it feasible?
- How much does it cost to implement our disarming module?
- What is the impact on network performance of the executing machine?

LET’S BUILD A PROTOTYPE
A network oriented monitor

- Filter against popular DoS attacks
  - Land
    - spoofed source address = dest. address
  - Smurf
    - source address spoofing & ICMP echo request with broadcast
  - SYN flood
    - uncompleted connections
  - Teardrop, Ping of Death
    - oversize packets
A network oriented monitor

- Filter against popular remote buffer overflows in
  - HTTP
  - FTP
  - Telnet
  - Sendmail
  - packet inspection looking for attack signature
A network oriented monitor

- **HOrstile Traffic Interceptor (HOT-I), now Angel**
  - Linux
  - static kernel module
  - packet filtering at IP level on outgoing traffic
    - rules and actions
First results

• IT WORKS !!!
• It requires a very good skill in network and system programming
Experiments

• Evaluate impact of HOT-I on network performance
  • latency
  • sustainable traffic generation rate
    • not a problem when the traffic is hostile
    • positive as denies the service to the attacker
  • mostly incoming traffic
    • servers are well protected
    • HOT-I is intended for “unattended” hosts
Experiments

• Experimental platform
  • Intel Pentium I, 120 MHz, 48 MB RAM
  • Debian GNU/Linux, 2.2.14 kernel
  • source code instrumentation
    • ip_queue_xmit, ip_build_xmit
    • execution time with and without the module
      with legitimate and hostile traffic
  • average over 1000 packets
    • std deviation negligible
Results

- From the performance point of view it does not seem too heavy
  - a slow-down of the outgoing packet flow of 7-8%
    - ACCEPTABLE!
What we still need

• Characterization of other attacks
• Extension of the set of rules to handle new attacks
• Minimize false positives
• Stronger module circumvention prevention
• Porting to other platforms
  • a new version exists for W2K
A host oriented monitor

- Filter against buffer overflows
  - `exec(/bin/sh), EUID = 0`
  - search for known “shell codes” (`/bin/sh` signature) before completing the `exec` syscall
  - raise alarm if OK, block execution otherwise
A host oriented monitor

- Filter against local DoS
  - fork/malloc bombing
    - wrappers on fork(), vfork(), clone(), brk()
Remarks

• Complement defense
  • intercept encrypted traffic at source
  • partly simplify fw&IDS’s work
    • white lists
  • guarantee trustworthyness of network hosts
  • first defense against script kiddies
Related work

• Pattern Analysis [Sekar 99]
  • a specification language based on regular expressions for events allows to characterize a program based on the normal/abnormal sequence of system calls it makes, taking into consideration also their arguments
  • at run time, an interception mechanism captures each system call and efficiently (5% overhead) matches the current pattern against the one defined, possibly taking actions against the process if necessary
Related work

• Generic Software Wrapper [Fraser et al. 99]
  • a wrapper definition language allows to define
generic wrappers for all possible system calls and
a wrapper support subsystem is implemented as a
loadable kernel module
  • a wrapper is activated when the activation criteria
defined by the user for that wrapper are verified
  • GSW is meant for systems with a sensible
administrator who would be in charge of installing
it and defining the wrappers needed.
Related work

• Neural network based code discovery [Cunningham98]
  • for remote exploits only
  • the source code, be it C or shell code, is examined by a NN and the presence of typical attack code features are identified
Related work

- STAT [Vigna et al. 00]
  - based on the definition of attack scenarios that abstract from the system specific details of attack signatures
    - opposite to HOT-I/Angel
    - like HOT-I/Angel, misuse detection
  - no trouble for signatures DB update
Related work

• Personal Firewall (IPchains, etc.)
  • user configured vs system defined as part of the OS
  • user awareness vs total transparency
References

D. Bruschi, E. Rosti,
“Disarming offense to facilitate defense,”

D. Bruschi, L. Cavallaro, E. Rosti,

D. Bruschi, E. Rosti,
“Angel: a tool to disarm computer systems,”