Honeypots
Building Honeypots

Commercial honeypots-emulating services
- Specter, Honeyed, Deception Toolkit.

Setting up of dedicated firewall (data control device)

Data collecting devices
- Firewall logs
- System logs
- Packet sniffer
- IDS logs
Stand alone Honeypots

- Easy to set up and no limit on any operating system installation
- Disadvantages
  - Sub-optimal utilisation of computational resources
  - Reinstallation of polluted system is difficult
  - Difficulty in Monitoring of such systems in a safe way
Virtual honeypots

- Virtual machines Allows different os to run at the same time on same machine
- Honeypots are guests on top of another OS
- We can implement guest OS on host OS in 2 ways
  - Rawdisc-actual disc partition
  - Virtual disc-file on host file system

contd..
- Advantages
  - Can peek into guest operating system at anytime.
  - Reinstallation of contaminated guest is also easy
  - And it is cheaper way
- Disadvantages
  - detecting the honeypot is easy.
Building honeypot with UML

- UML allows you to run multiple instances of Linux on the same system at the same time.
- The UML kernel receives system calls from its applications and sends/requests them to the Host kernel.
- UML has many capabilities, among them:
  - It can log all the keystrokes even if the attacker uses encryption.
  - It reduces the chance of revealing its identity as honeypot.
  - Makes UML kernel data secure from tampering by its processes.
Firewall rules
variables

Scale = “day”
Tcprate=“15”
Udprate = “20”
Icmprate= “50”
Otherrate=“10”

$laniface-internal lan interface to firewall
$ethiface-ethernet interface to outside from firewall
- Iptables -F
- Iptables -N tcpchain
- Iptables -N udpchain
- iptables -N icmpchain
- Iptables -N otherchain
Inbound traffic

- For broadcasting and netBIOS information
  - Iptables –A FORWARD –s honeypot –d 255.255.255.255 –j ACCEPT
Inbound TCP

- Iptables 
  
  `\n  
  -A FORWARD \-d honeypot \-p tcp \-m state \--state NEW \-j LOG \-log-prefix "tcpinbound"
  
  -Iptables \-A FORWARD \-d honeypot \-p tcp \-m state \--state NEW \-j ACCEPT
  
  >>> inplace of tcp use udp, icmp for respective data.

  >>> for established connections

  - Iptables 
  
  `\n  
  -A FORWARD \-d honeypot \-j ACCEPT

  contd...`
Outbound traffic

- DHCP requests
  - Iptables – FORWARD -s honeypot -p udp -sport 68 -d 255.255.255.255 -dport 67 -j ACCEPT
- DNS requests
  - Iptables –A FORWARD -p udp -s host -d server -dport 53 -j LOG --log-prefix “DNS”
  - Iptables –A FORWARD -p udp -s host -d server -dport 53 -j ACCEPT
- honeypots talking to each other
  - Iptables –A FORWARD -i $laniface -o $laniface -j LOG --log-prefix “ honeypot to honeypot”
  - Iptables –A FORWARD -i $laniface -o $laniface -j ACCEPT
*Counting and limiting the outbound traffic*

- Iptables -A FORWARD -p tcp -m state --state NEW -m limit --limit $tcprate/$scale --limit -burst $tcprate -s honeypot -j tcpchain
- Iptables _a FORWARD -p tcp -m state --state NEW -m limit --limit 1/$scale --limit-burst 1 -s honeypot -j LOG --log-prefix “drop after $tcprate attempts”
- Iptables - A FORWARD -p tcp -s honeypot -m state - -state NEW -s $host -j DROP

- For related information of a connection
- Iptables - A FORWARD -p tcp -m state --state RELATED -s $host -j tcpchain

- Same rules goes for UDP and icmp otherdata also
to allow all the packets from the established connection to outside

- Iptables -A FORWARD -s honeypot -m state --state RELATED ESTABLISHED -j ACCEPT

- TCPchain
  - Iptables -A tcpchain -j ACCEPT

- UDP chain
  - Iptables -A udpchain -j ACCEPT

- ICMP chain
  - Iptables -A icmpchain -j ACCEPT

- other chain
  - Iptables -A otherchain -j ACCEPT
- Iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT

- Firewall talking to itself
- Iptables -A INPUT -i lo -j ACCEPT
- Iptables -A OUTPUT -o lo -j ACCEPT
Default policies

- Iptables –P INPUT DROP
- Iptables –p OUTPUT ACCEPT
- Iptables –P FORWARD DROP