USENIX Association

Proceedings of the 17th Large Installation Systems Administration Conference

San Diego, CA, USA
October 26–31, 2003

USENIX
THE ADVANCED COMPUTING SYSTEMS ASSOCIATION

Tossing Packets Over the Wall Using Transmit-Only Ethernet Cables

Jon Meek and Frank Colosimo – Wyeth

ABSTRACT

Solutions for transporting packets from an insecure DMZ into an organization's internal network are described. All of the solutions attempt to prevent the establishment of two-way traffic by physically cutting the transmit wires at the receiving device on the internal network. Because alternate paths to the packet sender could exist, a user-mode packet relay on the internal network is used to accept and re-transmit the packets to the appropriate destination.

Applications discussed include relaying of syslog and SNMP trap packets from DMZ systems to receiving hosts on a secure network, and monitoring traffic for IDS and diagnostic purposes using a system conveniently located on the secure internal network.

Introduction

An Internet service connection point often consists of multiple security zones. At a minimum, most organizations will have a screening router and a firewall forming a DMZ between the devices. In addition, service zones for Web, mail, DNS, and other servers may be present [Zwi2000]. In order to manage systems in these zones, it is desirable, but potentially risky, to allow the systems to syslog, send SNMP traps, and perform other communication to the "Inside Network." It might also be useful to watch traffic in the DMZs for application diagnostics and intrusion detection.

A common solution for the need to receive data for packet capture purposes, while blocking bi-directional IP communications, is to configure an Ethernet interface without an assigned IP address. While this may be effective, the risk that the interface might be accidentally reconfigured or be susceptible to a future exploit is too high when firewall bypass is the possible result. In order to receive IP traffic for other purposes, such as SNMP traps or syslog packets, the receiving interface needs to have an IP address assigned.

We demonstrate several methods for allowing certain classes of packets, primarily UDP or packet capture traffic, to be passed unidirectionally from a lower security DMZ to a higher security zone, such as the corporate network. These methods involve the use of a uni-directional Ethernet link and, in most applications, a packet relay script running in user-mode. The major solution described has several pieces that are Cisco switch-specific, but enterprise switches from other vendors should have similar capabilities.

The goal, illustrated schematically in Figure 1, is to provide a mechanism to receive syslog and SNMP trap packets, while not allowing any outgoing traffic associated with the incoming traffic. In other words, we want to break TCP and other bi-directional protocols. It is very important to realize, however, that simply snipping the transmit wires at the receiving system will usually not prevent the return traffic that we are trying to avoid because the system will always need another, fully capable, interface in order to be of any use on a network. If the regular interface can communicate with the source of the packets received on the "listen-only" interface, then the lack of transmit wires on that interface provides no protection.

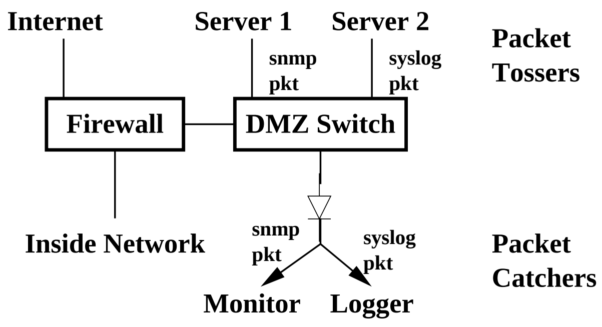

Figure 1: Packet tossing architecture.

Previous Hardware Methods

A solution for safely accommodating UDP applications and traffic monitoring across security zones includes the use of "Transmit-Only Ethernet Cables." It would be nice if such a cable could be made by simply cutting the transmit pair at the receiving system. Standard 10/100 Mbps Ethernet hardware, however, requires link heartbeats that ensure devices are connected on each end of the cable. Cutting the transmit pair at a host causes the corresponding hub/switch receive pair to lose the link signal, causing the port to be shutdown.

A search for such technology yields a number of possible solutions, most of which have significant problems. One solution places a capacitor across the transmit lines [Ng2001] which is supposed to mangle the data enough so that it is unintelligible on the other side, but not so much that the link integrity signal is lost. Other signal disruption techniques untwist the transmit pair or solder a paper clip antenna onto the transmit lines at the NIC card [Gra2000]. When security is the goal, this sort of solution is just not safe enough.

We describe a number of techniques that physically break the transmit circuit on the monitoring system. These techniques include the use of UTP to AUI transceivers for 10BaseT [Gra2000], and for 100BaseT, specially constructed cables or junction boxes, and commercial Ethernet taps.

Hardware Implementations

Two hardware implementations of the "packet over the wall" technique will be reviewed. It is important to analyze each technique in the context of the complete network to understand the possible impact on security. These techniques should significantly reduce risks, but are not a guarantee.

AUI Interface

This solution is useful for packet capture from 10 Mbps Ethernet for diagnostic or intrusion detection purposes. It might also be useful for low-rate syslog or SNMP trap packet relay. The implementation, shown in Figure 2, requires parts that are not readily available as new products: an Ethernet card with an AUI connector (3COM 3C900-COMBO), an AUI cable, and a AUI/UTP 10BaseT transceiver. A 10 Mbps hub, a piece of equipment that might be getting more difficult to buy new, is also required.

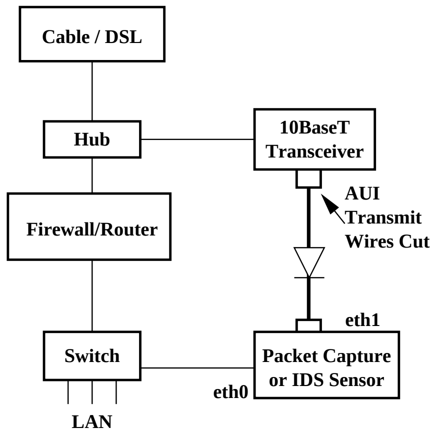

Figure 2: 10BaseT transceiver with transmit wires (pins 3 & 10) disconnected.

In this implementation, the AUI transmit wires (pins 3 and 10) are cut, or removed from the DB15 connector, at one end. There are no issues with the Ethernet link signal since it is supplied by the UTP transceiver.

Fast Ethernet Switch

This implementation, shown in Figure 3, will likely be the most widely used solution. The switch is located on the lower-security DMZ. The connection from the DMZ has a single pair, the transmit wires, running from the DMZ to the packet relay so that data can only be transmitted to the secure LAN.

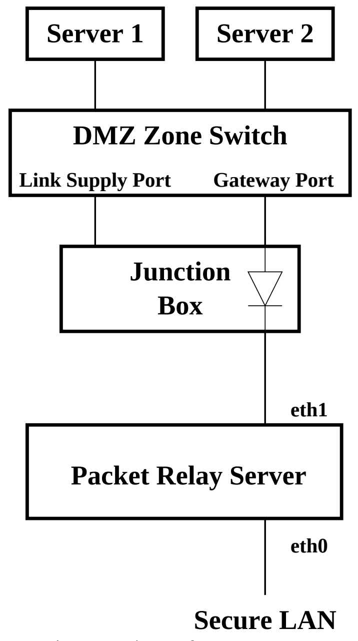

Figure 3: Tossing packets from DMZ to secure LAN via packet relay.

Our first technique for supplying the required link signal to the "gateway port" used a low cost hub connected to one of the RJ-45 connectors of a special three connector cable. We successfully used a Linksys FEHUB05W 100 Mb hub as the source of the link signal. However, numerous other low-end hubs and switches were tried without success. Most low end switches and hubs seem to require a link beat in order to supply one themselves. While this seems like a "Catch-22" situation, these devices do all work when used in the conventional way.

Since the requirement for a separate, nearly obsolete, hub is quite cumbersome, we had to find another solution. A significant amount of consultation with Cisco technical support resulted in no "magic solution" by way of an undocumented configuration command to force a switch port "up" without a link signal. Another problem with the first solution was the difficult-to-construct cable that stuffed two UTP cables into a single RJ-45 connector.

The final solution designates an unused port on the DMZ switch to supply the link signal. Simply connecting the transmit pair from the link-supply port to the receive pair of the gateway port does not, of course, work. By connecting the transmit and receive pairs on the link-supply port to each other, the link-supply port comes alive. The link signal is then supplied to the gateway port by tapping into the transmit-receive loop on the link-supply port as shown in Figure 4. This wiring feat is most easily accomplished using three CAT-5 RJ-45 "keystone jacks" commonly used in wall jack installation. Since the insulation displacement contacts (IDC) on the jack may not allow two conductors to be punched-down on the same pin, it is recommended that the jacks be wired with two pieces of twisted pair wire removed from a standard four pair UTP cable. The transmit and receive pins on the link supply jack and the receive pins on the gateway port are connected with one pair. The other pair connect the transmit pins of the "gateway port" to the "downstream port" that leads to the packet relay system.

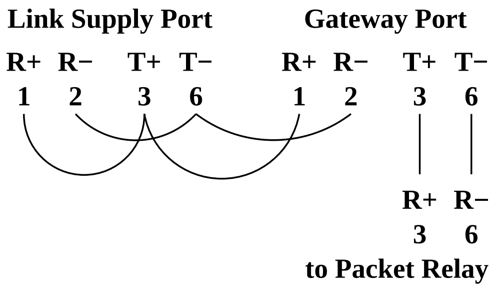

Figure 4: The junction box.

Finishing the "junction box" can be done using a standard plastic electrical wall box and a modular faceplate or a rack mount patch panel. The ports should, of course, be clearly labeled. Standard UTP cables are used to connect the junction box to the link-supply and gateway ports. Although a standard four pair UTP cable could be connected from the "junction box" to the packet relay system's Ethernet port, we recommend that a labeled single pair UTP cable be used in order to make it clear that this is a special circuit and to help prevent future mistakes since this wire will be crossing security zones.

In order to reduce the chance of problems due to the link-supply port hearing its own traffic, it might be a good idea to reduce or eliminate broadcast traffic on that port. It is also possible to use a low-cost 10/100 Mbps hub as the link-supply source. We successfully tested several brands of 10/100 hubs as the link supply source using the wiring scheme of Figure 4.

Switch and Sending System Configuration

The implementation of the method illustrated in Figure 3 requires special configuration of the DMZ switch and every system that will be "tossing packets over the wall." In this example, the gateway port on the DMZ switch is FastEthernet0/24, and the packet relay system's unnumbered interface has MAC address 00:04:5A:66:79:45.

The switch will be used as a pseudo-router, and it needs to know where to send packets destined for the "pseudo-host," the destination address 192.168.209.1 in this example. When configuring DMZ systems for remote syslog or SNMP traps, the "pseudo-host" IP address is used as the destination address.

On a Cisco Catalyst 2900 Series switch running IOS switch software, a MAC address can be assigned to a port using the mac-address-table command:

```
mac-address-table secure 0004.5A66.7945 \
        FastEthernet0/24 vlan 1
```

On Cisco Catalyst 5000 and 6000 series switches running Catalyst OS, a different command is used:

```
set cam permanent 00-04-5A-66-79-45 0/24 1
```

Our testing was done on 2924M XL and 6500 switches running IOS version 12.0(5)WC7 and Catalyst OS version NmpSW: 6.3(2) respectively.

Each system that will "toss packets" is configured to send traffic for the "pseudo-host" to 192.168.205.253, a "pseudo-gateway" on the same subnet as the hosts in the DMZ. Since there will be no way for the relay system to answer ARP requests, the arp table requires a manual entry.

On each Linux packet source system in the DMZ:

```
route add 192.168.209.1 gw 192.168.205.253
arp -s 192.168.205.253 00:04:5A:66:79:45
```

On Solaris systems:

```
route add -host 192.168.209.1 192.168.205.253 1
arp -s 192.168.205.253 00:04:5A:66:79:45
```

Using the configuration described above, a DMZ system with an IP packet to "toss" toward the high security zone will add an Ethernet header with the destination MAC address 00:04:5A:66:79:45. The switch will direct the packet to port 0/24 where it will be forwarded down the "transmit-only" cable to the "listen-only" interface on the relay host.

Note that we could use an entire "pseudo-subnet," and the destination address could be used for further routing in the packet relay application. We have found, however, that source IP address and destination port number are sufficient for our applications.

Alternative Methods for Packet Capture

For the purposes of diagnostic packet capture, or intrusion detection, there are alternatives to the "transmit-only" Ethernet cable. For 100 Mbps/Full Duplex Ethernet the Net Optics 96135 Fast Ethernet Tap [Net2003] is one solution. This tap is placed in-line and the data are replicated for each direction on two separate 100 Mbps Ethernet ports. The outputs can be safely connected to unnumbered packet capture ports on a secure network. For most purposes it is probably beneficial to bond the two monitor ports into a single interface so that data for each direction are presented in a single stream. On Linux this is accomplished using the "bonding" module and the ifenslave configuration utility. When a busy 100 Mbps/Full Duplex Ethernet must be monitored, the tap method is superior to any method that uses a single channel (mirror port with or without "transmit-only" cable) because the aggregate traffic can easily exceed 100 Mbps causing packets to be lost.

Applications

Intrusion Detection Systems

Many IDSs consist of sensors that collect and analyze network traffic and a database server that collects and correlates events from the sensors. Our initial deployment of the Snort IDS [Sno2003] and ACID database [ACI2003] without the transmit-only Ethernet cable limited the placement of sensors. Our security policy strictly limits inbound communications from the less secure DMZs to the inside network, and this prevented sensors in those DMZs from being integrated into the complete Snort/ACID system. With the transmit-only Ethernet cable, a dual NIC Snort sensor can be placed directly on the inside network with the second Ethernet used to accept packets from the DMZ. TCP communication with the ACID database server is then safe and transparent with the sensor and database in the same security zone.

Packet Capture

Packet capture is often a critical part of application or infrastructure debugging. While is it desirable to have packet capture systems located in every security zone, it might not always be possible. The transmit-only Ethernet cable allows the packet capture system to be located on the inside network and use a second NIC to collect data from any security zone. With additional NICs, it is possible to collect packets from multiple security zones on a single system, keeping the data together and without the need for careful time synchronization between multiple packet capture systems.

Syslog/SNMP Traps

We generally prefer the robustness of TCP based applications for reporting alarm events, especially for network problems when UDP communication may be unreliable. UDP methods do have their place however, and allowing their appropriate use is beneficial. Previously, we monitored DMZ network hardware by syslogging to a monitor system located in each DMZ, and mirroring the syslog files to a system on the internal network using rsync over ssh. The files were then monitored with swatch [Swa2003] or mon [Mon2002] and syslog.monitor [Msl2002]. While this is a reasonable solution, it is somewhat complex, and the time-to-alarm depends on when in the polling interval an event occurred. The mirrored syslog file method does not serve organizations with enterprise network management system that require SNMP traps unless the internal monitor can generate a spoofed trap that appears to come from the original device.

When communicating from a less secure to a more secure zone the delivery of UDP syslog and SNMP traps can be done with the transmit-only Ethernet cable and packet relay.

With a transmit-only Ethernet cable, syslog and SNMP traps can be sent directly from insecure DMZs to the internal network with some routing and ARP configuration. The destination of SNMP traps can be an enterprise monitoring system, and the source address that of the originating system in the DMZ.

Packet Relay

As pointed out earlier, the existence of a return path using another interface on the receiving system will render the cutting of the transmit wires a useless exercise. To mitigate that risk, we developed a user-mode packet relay method. A packet capture program listens on the receiving interface which has no IP address assigned. A packet meeting certain criteria, usually a UDP packet carrying SNMP trap, syslog, or mon trap data, is read and then forwarded to a final destination. The packet relay is implemented as a simple Perl script, shown in Appendix 1, thanks to libpcap [Lib2003] and the RawIP [Kol2003] interface to libpcap. On an operating system that allows the permissions on a network interface to be altered, the packet relay script should run as a non-root user to provide another layer of insulation. The Linux kernel has a "capabilities" module that may provide appropriate access to a non-root user when tools to configure the module become widely available.

```
Interface eth2
Filter udp and (port 162 or port 514)
#          Source IP   Port  Destination IP
Redirect       *        514  10.1.1.19
Redirect 192.168.205.45 162  10.1.1.23
#Debug
```

Figure 5: Sample packet relay configuration file.

The Perl script uses a simple configuration file, shown in Figure 5, that allows the listen interface to be selected, a custom libpcap filter to be applied to the interface, and entries to be set in a forwarding table. The default filter passes all UDP packets to the script but it is best to allow only packets that should be forwarded to another system, such as SNMP traps and syslog data. The script can forward all packets based on destination port number alone, or a combination of port number and source IP address. Using the source address as an additional qualifying parameter provides the capability to deliver packets to different destinations based on the owner of the system, if is a production or development server, etc.

We have found that the packet relay rate using a 1 GHz Pentium III is more than 800 packets per second for syslog packets with a 150 byte payload. If the Perl script is not fast enough for a particular application, a re-implementation in C would presumably increase performance.

If the user-mode packet relay script cannot be implemented, it is highly recommended that the receiving system be placed on an isolated network that has no path back to the source of the packets. That might be difficult to implement since most enterprise monitoring systems need to be connected to the main network to allow remote communications, send alarms, etc. In addition, monitoring systems benefit from being able to poll systems, including those that are sending packets "over the wall." If a monitored system fails catastrophically, or if there is a network failure, there may be no chance for a UDP packet to arrive at the monitor system. Thus, a polling monitor must be part of a complete monitoring solution for any system or application.

Conclusion

We have discussed variations on a method to send UDP packets from a less secure DMZ into a more secure network. In addition to using unidirectional Ethernet, we implemented a user-mode packet relay to insulate systems on the inside network from attack. As with all systems involving security, the implementation of this technique should be carefully scrutinized and monitored to reduce risks, current or future. For example, we have not even considered any methods to stop denial of service attacks that might be launched over the unidirectional Ethernet circuit. So far, the main use of the technique has been to send syslog and SNMP data to monitoring systems that are conveniently located on the corporate network.

Acknowledgments

The authors would like to acknowledge Jim Trocki, Nick Fiekowsky, and Rich Hollenbach for valuable discussions on the implementation of these techniques and Æleen Frisch for her review and editing of the final paper.

Jon Meek is a Senior Consultant in the Border Network Services Group at Wyeth. He received BS and MS Degrees in Physics, and a Ph.D. in Chemical Physics all from Indiana University and has worked in Nuclear and Chemical Physics, Analytical Chemistry, and Information Technology. His research interests include network performance measurement, security, and data integrity. He can be reached at meekj@wyeth.com or meekj@ieee.org .

Frank Colosimo is an Enterprise Consultant in the Border Network Services Group at Wyeth. He received a BS Degree in Chemical Engineering from the University of Michigan and has worked with networked systems of various types for Eastman Kodak, Siemens, Unisys, and as an independent consultant. In recent years he has specialized in Cisco IP routing and switching technology and network security. He is a Cisco Certified Internetworking Expert (CCIE) and Cisco Certified Security Professional. He can be reached at <colosif@wyeth.com>

References

[ACI2003] ACID, http://www.andrew.cmu.edu/˜rdanyliw/snort/snortacid.html .

[Gra2000] Sniffing FAQ, http://www.robertgraham.com/pubs/sniffing-faq.html .

[Kol2003] Net::RawIP Perl module, Sergey Kolychev.

[Lib2003] libpcap, maintained by "The Tcpdump Group," http//www.tcpdump.org .

[Mon2002] mon Monitoring framework, http://kernel.org/software/mon/ .

[Msl2002] syslog.monitor for mon, http://wanpcap.sourceforge.net/mon/ .

[Net2003] Net Optics, Inc., Sunnyvale, CA, http://www.netoptics.com/ .

[Ng2001] *How to make a sniffing (receive only) UTP cable*, Sam Ng, http://www.geocities.com/samngms/ sniffing_cable/ , 2001.

[Sno2003] http://www.snort.org/ .

[Swa2003] http://swatch.sourceforge.net/ .

[Spu2000] *Ethernet: The Definitive Guide*, Charles E. Spurgeon, O'Reilly, February, 2000.

[Zwi2000] Zwicky, Elizabeth D., Simon Cooper, D. Brent Chapman, *Building Internet Firewalls, 2nd Edition*, O'Reilly, June, 2000.

Appendix 1 – Packet Relay Script

```
#!/usr/local/bin/perl

# Raw Packet Relay

# Accept a packet, usually from a "listen-only" interface
# Modify the destination address and re-transmit it

# Jon Meek  07-Jun-2003

# usage: sudo ./rawrelay relay.cfg &

use Net::RawIP;
use Socket;

# Set defaults

$Debug = 0;
$SnapLength = 1514;

# Default filter and interface
$Filter = 'udp';     # Only UDP packets will be worth processing
$Interface = 'eth1'; # An interface with no address assigned

$Timeout = 500;

$NumPackets = -1; # Run forever
$CheckSourceAddress = 0;

print "Starting $0\n" if $Debug;

# Read configuration data from configuration file specified on command line

$ConfigFile = shift @ARGV;
open(F, $ConfigFile) || die "Can't open configuration file: $ConfigFile";
while ($in = <F>) {
  chomp $in;
  next if ($in =~ /^\#/); # Skip commented lines

  if ($in =~ /^Debug/i) { # Debug mode displays information for each packet
    $Debug = 1;
  }

  if ($in =~ /^Interface /i) { # Override the eth1 default
    ($tag, $Interface) = split(' ', $in);
  }

  if ($in =~ /^Filter /i) {    # Set a custom filter
    ($tag, $Filter) = split(' ', $in, 2);
  }

  if ($in =~ /^Redirect /i) {  # Configure forwarding table
    ($tag, $src_addr, $dst_port, $dst_addr) = split(' ', $in);
    if ($src_addr eq '*') {              # Wildcard address
      $Redirect{$dst_port} = $dst_addr; # Forward by port number only
    } else {
      $CheckSourceAddress = 1;           # Set flag to check source address
      $Redirect{"$src_addr-$dst_port"} = $dst_addr; # Forward by port and address
      print "$src_addr-$dst_port --> $dst_addr\n" if $Debug;
    }
  }
}

# Initialize

$a = new Net::RawIP({udp=>{}});
my $pcap = $a->pcapinit($Interface, $Filter, $SnapLength, $Timeout);

# Capture and process packets

loop $pcap, $NumPackets, \&ModifyAndResend, \@a;

sub ModifyAndResend {
  $a->bset(substr($_[2],14));

  ($ipsrc, $ipdst, $source, $dest) =
    $a->get( {ip=>[qw(saddr daddr)], udp=>[qw(source dest)]} );
  if ($Debug) {
    print inet_ntoa(pack("N",$ipsrc))," [$source] -> ",
      inet_ntoa(pack("N",$ipdst))." [$dest]\n" ;
  }

  # Change the destination address

  $DestinationIP = 0;
  if ($CheckSourceAddress) { # Need to check the source for redirect destination

    $src_addr = inet_ntoa(pack("N",$ipsrc));
    if (exists $Redirect{"$src_addr-$dest"}) {
      $DestinationIP = $Redirect{"$src_addr-$dest"};
      print "redirecting $src_addr-$dest to $DestinationIP\n" if $Debug;
      $a->set({ ip => { daddr => $DestinationIP } })
    } else {
      $DestinationIP = $Redirect{$dest};
    }

  } else {
    $DestinationIP = $Redirect{$dest};
  }

  if ($DestinationIP) { # We have a place to send the packet
    $a->set({ ip => { daddr => $DestinationIP } });
    $a->set({ udp => { check => 0 } }); # Forces checksum recompute

    $a->send;     # Re-transmit the packet

    if ($Debug) { # Verify new packet content in Debug mode
      ($ipsrc, $ipdst, $source, $dest) =
        $a->get( {ip=>[qw(saddr daddr)], udp=>[qw(source dest)]} );

      print inet_ntoa(pack("N",$ipsrc))," [$source] -> ",
        inet_ntoa(pack("N",$ipdst))." [$dest]\n\n";
    }
  } else {
    print "No destination for packet\n" if $Debug;
  }
}
```