Synchronization of VM probes for observing P2P traffic and application behavior using EtherIP

Ruo Ando, Youki Kadobayashi and Yoichi Shinoda

National Institute of Information and Communication Technology,
4-2-1 Nukui-Kitamachi, Koganei,
Tokyo 184-8795 Japan
ruo@nict.go.jp

Abstract. Recently security incident caused by P2P application has become serious threat. Particularly, it is difficult to trace P2P traffic and application behavior with the single node based current technologies. In this paper, we propose a synchronization method of VM probes for observing P2P traffic and application behavior using EtherIP. We apply EtherIP to connect the guest domains (virtualized domains) in physical machine on different locations to provide illusion that these are running on the same segment, local area network. Therefore, using Ether IP makes it possible to synchronize nodes by sending command to trace traffic and application behavior. Proposed system is implemented on VNET on Xen virtual machine monitor. VNET is the virtual network bridging based which enables the guest domains in physical machines on different locations connected and synchronized to each other. Prosed sytem is desinged towards observablity and traceability of P2P appliction and its networks.

Keywords: Synchronization of VM probes, P2P traffic, P2P application, EtherIP, VNET of XEN.

1 Introduction

1.1 Towards observability and traceability of P2P networks

Recently, security incidents caused by P2P application has become serious threat. The incidents are classified into two categories: information leak and P2P based malware. Once the confidential files are moved to upload folder and spread over P2P, with current technologies of single node based trace system, it is almost impossible to trace the leakage. Another problem of P2P networks are the large number of nodes connected to P2P networks. For example, Winny networks has still 300,000 - 400,000 nodes currently, which makes it impossible to observe network traffic to trace information leakage. Towards observability and traceablity of P2P networks, we would like to emphasize two points. First, application behavior is necessary to track. Second, distributed and massive probing is important. For traceability of P2P network, we need to gather information from muilti-layer (application behavior and network traffic) and distributed probing.

1.2 Connection graph of VM probes

We track the connection graph of Winny on VM probes. It is shown that connection graphs of VMs are completely different even if VMs has the same conditions (same OS image, synchronized, etc). This means that it might be possible to monitor P2P network if probes is on the physical locations. Because VMs on the same physical machines have different connection graphs. The difference of connection graphs means that distributed monitor need not always to be deployed in different physical machines. To some extent, synchronizing P2P application of each domain on virtualized local area network is effective even if researchers does not deploy distributed monitor in physical machines.

2 Probe synchronization on VMM

In this paper we propose a method to inspect of P2P query spread and response. To inspect the state of P2P network, the state of client machines need to be exactly same. In proposed system, we generate the same virtual machines and generate the same query at the same time. We call it P2P query synchronization on VMM in this paper. The interesting event we found is that the responses are different even when the conditions of client machines are all the same. To synchronize P2P traffic, we apply VNET of XEN virtual machine monitor.

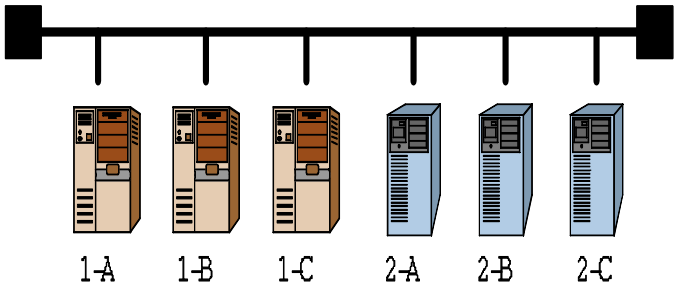

Fig. 1. Traffic synchronization on VMM. The guest domains with different segments can be virtually connected and synchronized.

Figure 2 shows proposed monitoring system using VMM. P2P application such as winny is running on VM. Each VM has birdge connection to ETH1 of control domain. Probes of application monitor and tcpdump are set on each VM. In proposed system, we can set condition of P2P application on VM, which makes it possible to measure effect of those conditions.

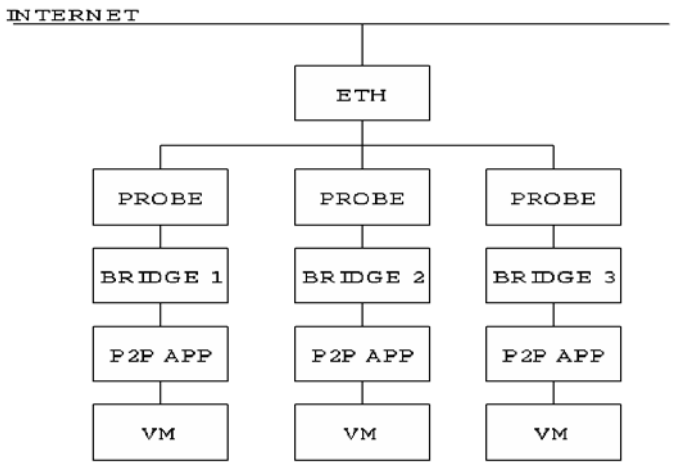

Fig. 2. P2P monitoring on VM.

3 Virtualized L2 datalink

3.1 VNET of virtual machine monitor

VPN (Virtual Private Network) is widely used because physical machines in different location can be connected by VPN. VNET is a kind of VPN utility (providing similar feature) of XEN virtual machine monitor as shown in Figure 3. By Vnet, the guest domain in different network segments are connected and virtual network segment is emulated. It provides a network illusion that the guest domains in different domains are in a same local private virtual networks. VNET is the bridging virtual network based. VNET tunnels the virtual Ethernet traffic between domains.

3.2 EtherIP:Tunneling Ethernet Frames in IP Datagrams

The EtherIP protocol is used to tunnel Ethernet and CSMA/CD MAC frames. Vent applies EtherIP to provide illusion that the guest domains in different networks can be connected in the same segment. Ether IP can be implemented in as an endpoint to enable tunneling for in a bridge-like station to enable tunneling for multiple domains linked to virtual local area network segment. Figure 4 shows header of EtherIP (RFC3378).

4 Probing, decoding and aggregating techniques

4.1 P2P malware capture

Recently P2P malware has become a serious threat which causes information leak and botnet attack. In P2P network, observing network topology and traffic is sometimes very difficult. With the difficulty of observation of P2P traffic,

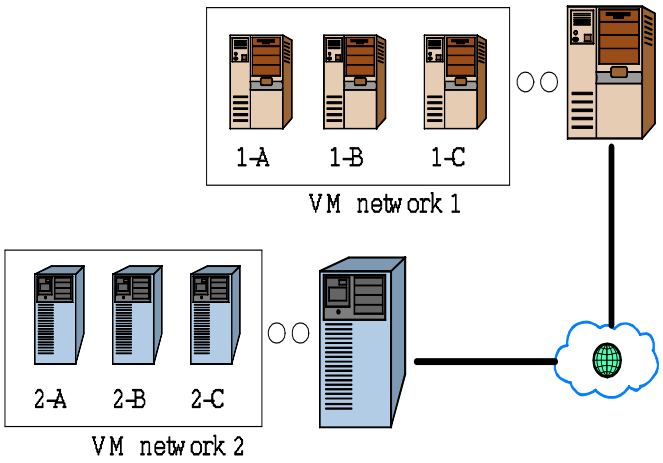

Fig. 3. Virtualized L2 datalink. The guest domains of physical machines in different segments are (virtually) connected using EtherIP.

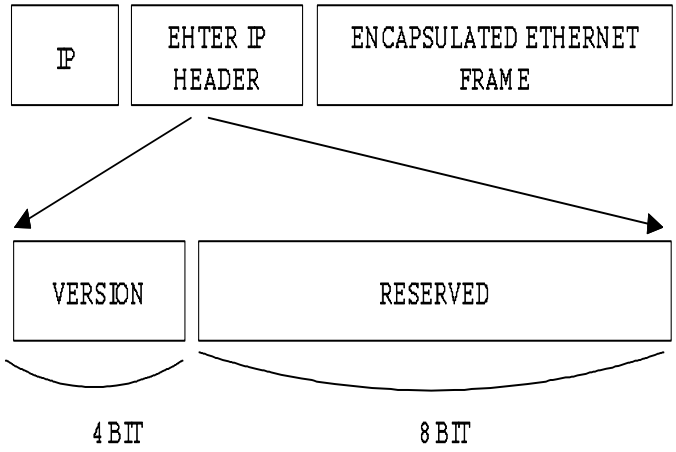

Fig. 4. Sending query using ETHERIP reserved header.

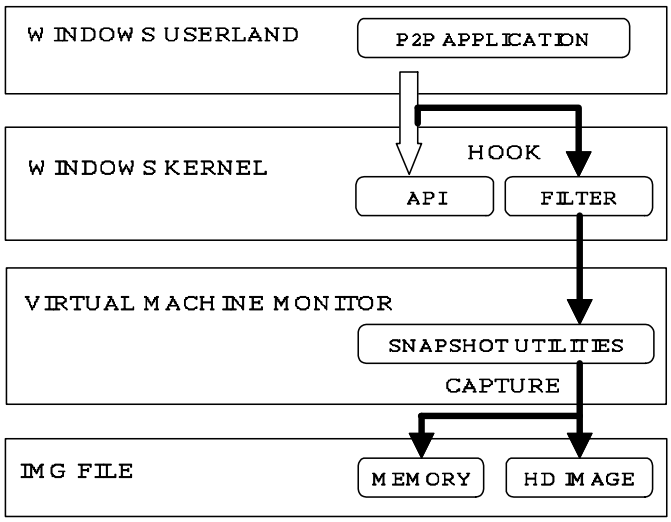

Fig. 5. Capturing P2P malware. Illegal resource access of P2P malware is detected by API hooking. Then injected routine invokes snapshot utilities of VMM to capture P2P malware.

alternative detection and prevention methodology is required for detecting P2P malware. Proposed system has been implemented on full-virtualized Microsoft Windows (TM). As shown in Figure 5, on guest Windows(TM) OS, we apply three kinds of interruptive debugging techniques to detect the incidents and invoke capture (snapshot) functions of VMM. On VMM side, we modify debug register handler to execute snapshot routine of VMM. Receiving notification, proposed system can capture and prevent P2P malware when the event such as directory access and packets send / recv are occurred in guest Windows(TM) OS. Proposed system is implemented XEN virtual machine monitor and KVM (Kernel Virtual Machine). Our system is used to capture P2P malware for analysis on the test bed of our group.

Figure 6 shows an implementation of proposed system in KVM (Kernel Virtual Machine). KVM makes Linux as hypervisor. In implementation of KVM, a simple user defined signal is applied for the asynchronous notification. When the incident is detected by guest OS, the value of special registers is changed (vector [1]). When the system control is moved to VM root operation, the change is caught by register handler. Then, user defined signal is sent to QEMU modules of KVM by control application or directly from kernel (vector [3][4][5]). Finally, signal handler invokes memory snapshot facilities using QEMU I/O module.

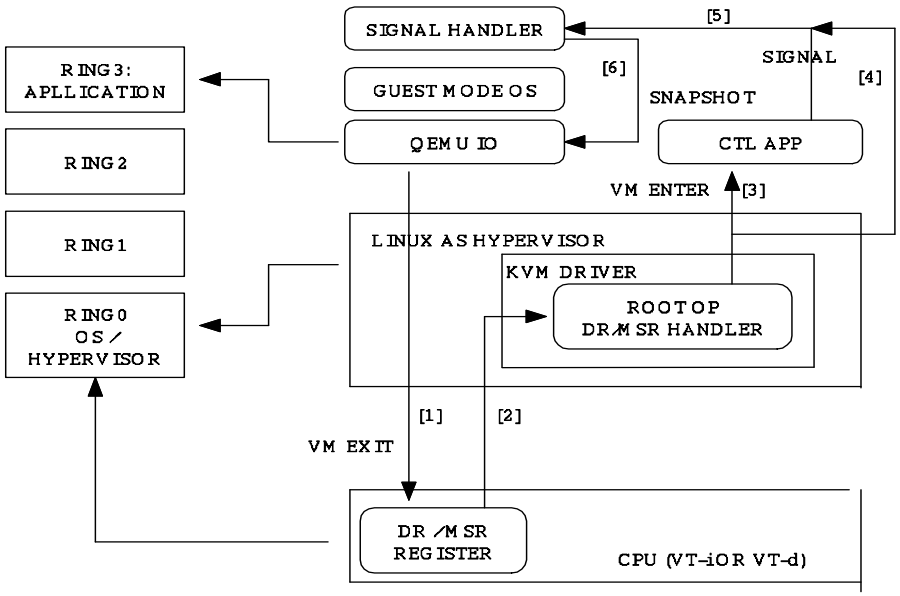

Fig. 6. Proposed system implemented on KVM. When a incident is detected, guest OS changes debug register. The change is caught in KVM module. Then, signal is generated and sent guest OS to take snapshot.

4.2 TCPDUMP on virtualized NIC

Vnet is the bridging virtual network based. We use TCPDUMP on virtualized bridge interface on host OS (controller domain) to observe traffic of the guest domains like:

```
tcpdump -i vbridge1 -n -s 65535 -w dump.file
```

In the following section, we discuss numerical output.

4.3 Traffic decoder

We implemented a prototype for inspecting Windows (TM) P2P software Winny. Intercepted functions are send, recv, connect, bind etc. In Windows(TM) XP SP2, P2P application such as winny applied WS2_32.dll. Function to replace is implemented as follows:

```
PROC pfnOrig = GetProcAddress(GetModuleHandleA("ws2_32.dll"),
"accept");
int res = ((PFN_ACCEPT) pfnOrig)(s, addr, addrlen);

sockaddr_in* addrin = (sockaddr_in*)addr;
```

```
PROC pfn_inet_ntoa = GetProcAddress(GetModuleHandleA
("ws2_32.dll"), "inet_ntoa");
PROC pfn_ntohs = GetProcAddress(GetModuleHandleA("ws2_32.dll"),
"ntohs");

char* chaddr = ((PFN_INET_NTOA) pfn_inet_ntoa)(addrin->sin_addr);
if (chaddr) {
port = ((PFN_NTOHS)pfn_ntohs)(addrin->sin_port);
sprintf(chbuf, "%s (%d)", chaddr, port);
}
```

By using DLL injection, we have decoded encrypted packet of Winny with RC4. Also, malicious packets can be detected by checking packet header. On our system we can capture P2P malware using snapshot if the system detect malicious packets. By using this, we can decode the downloaded / uploaded file and stop those by checking the keywords indicating illegal contents.

4.4 Directory access detection

P2P malware mainly aims at information and file leak which causes illegal directory access. For example, P2P malware moves the files of host computer to upload folder. Or file in user document directory is changed. To detect these actions, we apply modification of createFile of writeFile API.

```
BOOL WINAPI hook_WriteFile(
IN HANDLE hFile,
IN LPCVOID lpBuffer,
IN DWORD nNumberOfBytesToWrite,
OUT LPDWORD lpNumberOfBytesWritten,
IN LPOVERLAPPED lpOverlapped
) {

/* check routine */

PROC pfnOrig = GetProcAddress(GetModuleHandleA("kernel32.dll"),
"WriteFile");
BOOL res = ((PFN_WRITE_FILE_HOGE)pfnOrig)(hFile, lpBuffer,
nNumberOfBytesToWrite,
lpNumberOfBytesWritten,
lpOverlapped);
return res;
}
```

These modification enables us to check which file is changed (written) for malicious file uploading. If the file in the directory we are inspecting is changed, we can capture the malware using snapshot utility of VMM. For example, P2P

malware do illegal access C:document and settings directory which proposed system can prevent. Also, we can prevent exploited uploading causing information leak on proposed system with more fine grained filter compared with generaic AV scanner.

5 Implementation and experiment

5.1 Network setting

As we showed in Figure 3, proposed system is implemented on XEN VNET. Host machine of VM network 1 and 2 is in different segment while VMs 1A - 1C and 2A - 2C is in same segment 192.168.1.*. In VNET, if host machines of VM network 1 and 2 is connected by vn peer-add, VMs 1A - 1C and 2A - 2c can be connected as if these are in the same segment. Then we can send the control packet to 1A - 2C to generate (synchronized) query traffic at the same time.

5.2 Numerical result

In this section we discuss the numerical result of proposed system. Three series on Figure 7, 8 and 9 shows traffic on synchronized P2P application Winny with bandwidth 200 Kbytes, 120 Kbytes and 50 Kbytes. As the observation time is passed, the difference is occurred mostly in the throughput (I/O) packets. As we discussed in Figure 2, the node with high bandwidth go upper in the P2P network, which causes increasing the difference between 200 Kbytes and 120 (or 50) Kbytes. On the other hand, about SYN packets, the difference keeps constant (not increasing) between three lines partly because we can generate the same query traffic. The packet with 11 length is particular to the application Winny. 11-length packet is the initializing packet for Winny's operation. About these packets, with 120 Kbytes optimal for our ASDL (MB), Winny with 120 Kbytes seems to work reasonably.

5.3 Analysis

Result shown in Figure 8 is expected result because we have run P2P application with 200, 100 and 50 Kbytes which is under the limit of ADSL bandwidth. Let F(x) be the throuthput (I/O packet) of each node.

$$\frac{dF(200)}{dt} = a * \frac{dF(120)}{dt} = b * \frac{dF(50)}{dt}$$

Regardless of the throughput of each node, the speed of chaning connection graph on P2P network determines how many neighbors connected to the node. Let G(x) be the connection / disconnection speed of P2P network.

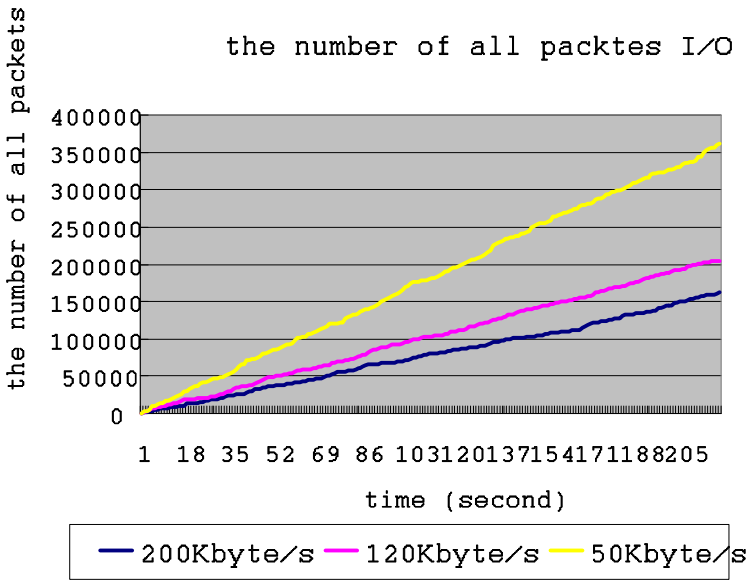

Fig. 7. The number of all packets I/O with 200 Kbytes, 120 Kbytes and 50 Kbytes.

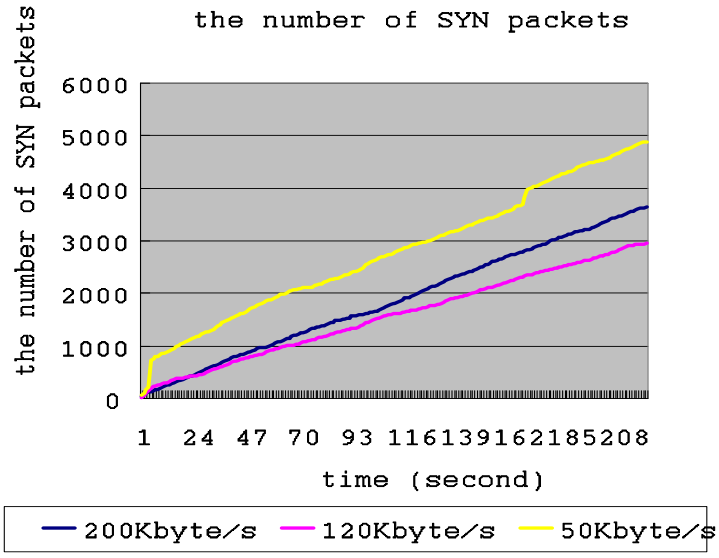

Fig. 8. The number of SYN packets with bandwidth 200 Kbytes, 120 Kbytes and 50 Kbytes.

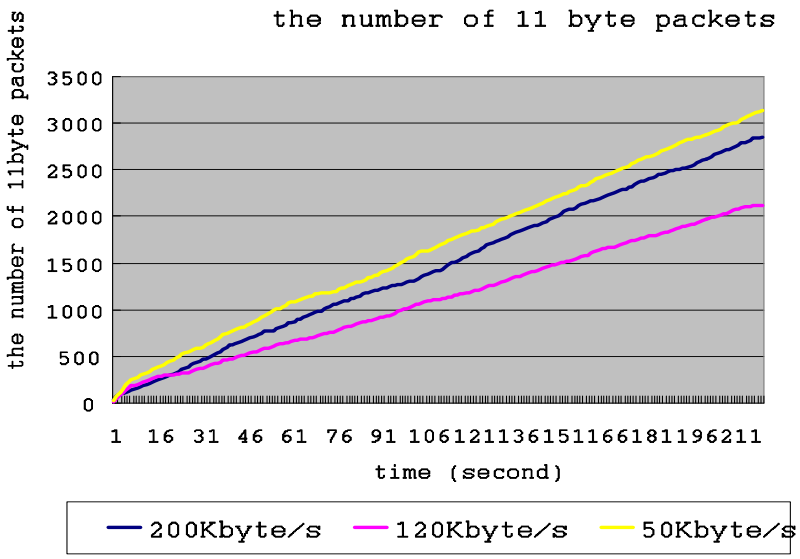

Fig. 9. The number of 11 byte initializing packet with bandwidth 200 Kbytes, 120 Kbytes and 50 Kbytes.

$$\lim_{t\to\infty} \frac{dG(200)}{dt} = c * \lim_{t\to\infty} \frac{dG(120)}{dt} = d * \lim_{t\to\infty} \frac{dG(50)}{dt}$$

11 byte packet is initializing vector for Winny connection before shake hand. Let H(x) be the number of 11 byte packets (initizalizing connection is succeeded). It is showed that $e \leq c$ and $f \leq d$.

$$\lim_{t\to\infty} \frac{dH(200)}{dt} = e * \lim_{t\to\infty} \frac{dH(120)}{dt} = e * \lim_{t\to\infty} \frac{dH(50)}{dt}$$

Although it's not for sure, the number of connection possible for each probe much depends on the speed of P2P network even if we deploy the machine with high bandwidth. The number of probes is more important for observe and trace P2P network.

6 Conclusions

Recently security incident caused by P2P application has become serious threat. Particularly, it is difficult to trace P2P traffic and application behavior with the single node based current technologies. In this paper, we haved proposed synchronization method of VM probes for observing P2P traffic and application

behavior using EtherIP. We have applied EtherIP to connect the guest domains (virtualized domains) in physical machine on different locations to provide illusion that these are running on the same segment, local area network. Therefore, using Ether IP makes it possible to synchronize nodes for tracing, observing networks and sending command. It is shown that connection graphs are different even probes is deployed on the same physical machine. This means that the synchronization is effective for tracing on virtualized domains, not always necessary on physical locations. Proposed system is implemented on VNET on Xen virtual machine monitor. VNET is the virtual network bridging based which enables the guest domains in physical machines on different locations connected and synchronized to each other. Proposed system has been designed towards observability and traceablity of P2P aplication and its networks.

Acknowledgement

We are indebted to JGN2plus project team of National Institute of Information and Communication Technology. This paper grew out of the ongoing project of “ Experiment of long-distance VM live migration and high-speed snapshot transfer ” with project Number JGN2P-A20.

References

1. ”A Distributed Decentralised Information Storage and Retrieval System”, an Clarke, Division of Informatics University of Edinburgh Dissertation, 1999 http://freenet.sourceforge.net/freenet.pdf
2. Javed I. Khan and Adam Wierzbicki, Foundation of Peer-to-Peer Computing, Special Issue, Elsevier Journal of Computer Communication, Volume 31, Issue 2, February 2008
3. Stephanos Androutsellis-Theotokis and Diomidis Spinellis. A survey of peer-to-peer content distribution technologies. ACM Computing Surveys, 36(4):335 • 71, December 2004.
4. XEN virtual machine monitor, http://www.cl.cam.ac.uk/Research/
5. Programming Applications for Microsoft Windows Forth Edition, Jeffrey Ritcher, Microsoft Press, 1999
6. RFC 3378 - EtherIP: Tunneling Ethernet Frames in IP Datagrams http://www.faqs.org/rfcs/rfc3378.html