International Journal of Civil Engineering and Technology (IJCIET)
Volume 8, Issue 11, November 2017, pp. 35–41, Article ID: IJCIET_08_11_004
Available online at http://http://www.iaeme.com/ijciet/issues.asp?JType=IJCIET&VType=8&IType=11
ISSN Print: 0976-6308 and ISSN Online: 0976-6316

HOCS: HOST OSCOMMUNICATION SERVICE LAYER

Jayakumar Sadhasivam
School of Information Technology and Engineering, VIT University, Vellore, India

Senthil Jayavel
Computer Science and Engineering, Kumaraguru College of Technology, Coimbatore, India

Balajee Jeyakumar
School of Information Technology and Engineering, VIT University, Vellore, India

Shoaib Merchant
School of Computing Sciences and Engineering, VIT University, Vellore, Tamil Nadu, India

ABSTRACT

In this paper, we introduce the concept of an additional layer between the Session and Transport layer in the OSI Model, acting as a mediator for all the network communication performed by the application protocols. This network service layer will have to be incorporated in the OS level replacing the default OS' network libraries. Further in the paper we have discussed the benefits offered by implementing this layer in terms of network speed, security, reliability, and application portability. Major security threats like Port Scanning, OS Finger Printing, and TCP Session Hijacking are handled by the layer. In the current Operating Systems design the OS has not much control over the activities performed by the Application Protocol, the OS simply provides an API to create sockets and communicate using them to a remote host via the standard protocols. The HOCS Library will enable the OS to moderate and control the application protocols, this will specifically help in detecting and blocking a Worm attack in the Host system, which is undetectable in the transport layer. Apart from these benefits, the HOCS library will be uniform for different Operating Systems providing transparency and uniformity in the application protocols. Applications developed using the HOCS library will be able to perform the same networking functions on different operating systems. An implementation of HOCS for a server computer is also discussed apart from the conventional client host implementation.

Key words: OSI Model, Network Model, HOCS, Operating System.

Cite this Article: Jayakumar Sadhasivam, Senthil Jayavel, Balajee Jeyakumar and Shoaib Merchant, HOCS: Host Oscommunication Service Layer. *International Journal of Civil Engineering and Technology*, 8(11), 2017, pp. 35–41.
http://www.iaeme.com/IJCIET/issues.asp?JType=IJCIET&VType=8&IType=11

1. INTRODUCTION

In the conventional socket based application programming, the programmer creates a socket and specifies a socket address, port number and transport protocol (TCP/UDP) which is bound to that socket connection using the Operating System's network libraries. Once this connection is established the Operating System loses control over this connection and cannot monitor the messages incoming-outgoing from this port. The OS involvement in this is simply to open the corresponding port on the firewall to enable communication. Therefore, in such a case all methodologies for host protection over a network have to be implemented on the application layer protocol being used. The HOCS overcomes this major issue by providing its own libraries capable of securing the host system in case of a network attack and monitor the messages in-out of the port.

Apart from addressing this problem, HOCS is capable of achieving uniformity in network programming; application programs built for an OS over HOCS will support any OS which has HOCS implemented. Just to be clear, the HOCS will not be modifying the TCP Headers it will simply modify the Message format of the Application layer.

2. SYSTEM DESIGN AND IMPLEMENTATION

2.1. HOCS Design Overview

The HOCS layer lies between the session and the transport layer. In the OS level it exists directly above the native Networking APIs. Any application level service which wishes to communicate in the network interacts with this layer using a common set of functions exposed by the HOCS Network API.

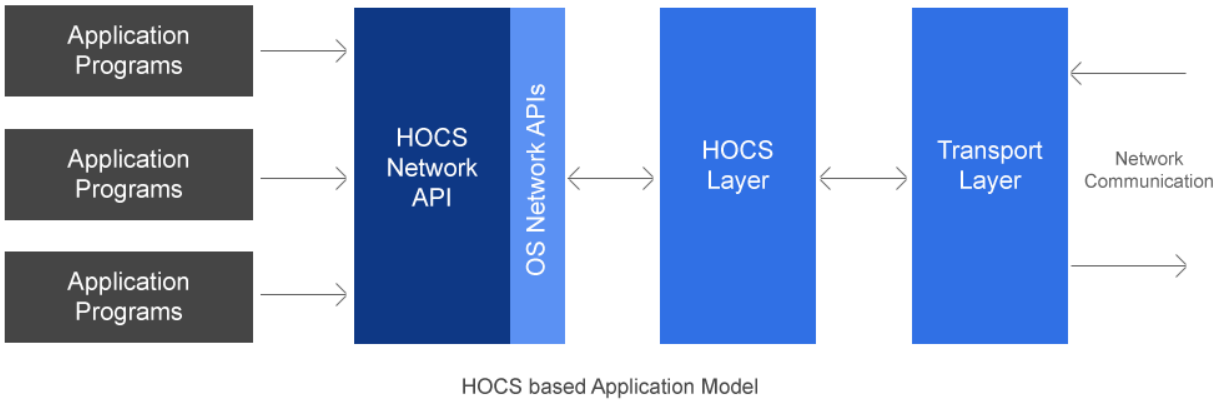


Figure 1 HOCS Based Application Model

The HOCS layer contains the following modules which perform different functions for various threats the host system in the network can face.

2.1.1. Port Mapping

Ports created by the application are not directly implemented on the host, instead a proxy port is used maintaining transparency. Also, states are maintained for each connection on every port.

2.1.2. Application Monitor

The application monitor moderates all the messages sent from the applications and verifies incoming and outgoing connection. This monitor includes a Stateful Firewall mechanism for defining rules over the ports and maintaining states. This is used for preventing Worm and

Trojan attacks by monitoring the applications and the traffic generated by them. This monitor also maintains a list of all the blacklisted attackers along with their host details.

2.1.3. HOCS Control System

The HOCS exercises control over the Host's network activity by setting the system in fixed network modes – Strict, Passive & Lockdown based on the information it receives from the Application Monitor. The user is also given the ability to switch between Strict & Passive via the Operating System.

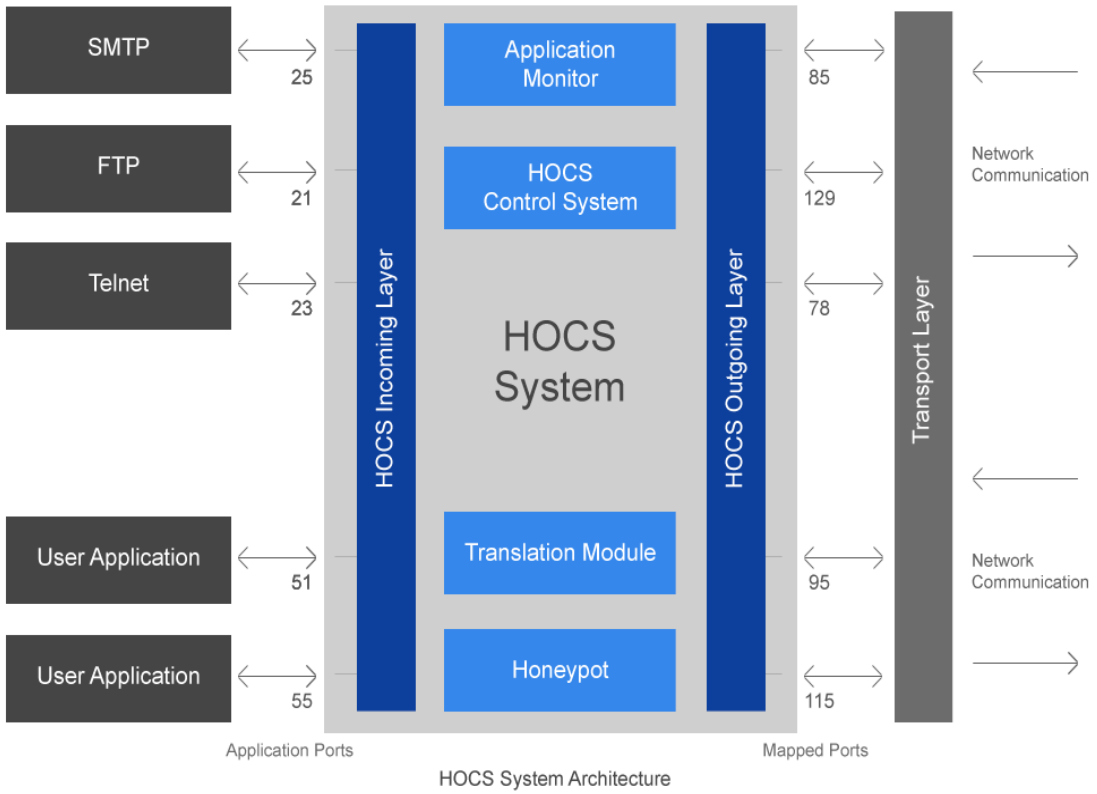
2.1.4. Translation Module

The applications which still have not adapted the HOSC layer can send messages but these messages reach the HOCS Control which translates these messages to the HOSC Message format, removing any OS specific information and updating the port details.

2.1.5. Honeypot Service

A Trap mechanism for attackers, wherein access is given to fake data in order to get the attacker host information which is then used to blacklist the attacker.
This service runs on all unmapped ports, and acknowledges any incoming connection.

2.2. Schematic Diagram



HOCS System Architecture

2.3. Network Operation Modes

The HOCS controls the network traffic from the host through three different network operation modes.

Passive – In the passive mode of operation, the HOCS allows excessive traffic generated by the application protocol but generates a warning for the user to inform that it detects suspicious activity. Moreover, Remote Access commands are also executed without the

HOCS intervening. This low security mode can be switched to, by the user if the connections on the networks are pre-secured and trusted.

Strict– The HOCS does not allow excessive traffic from an application and blocks any suspicious communication from the applications until they are added as an exception by the user in case of trusted connections. Also no remote access commands are issues without adding the respective applications as exceptions. The HOCS will operate in this mode by default unless switched to Passive mode by the user. The strict mode also blacklists any host which tries to communicate to the host incorrectly considering as an attacker.

Lockdown– This network operational mode will be activated automatically, when the HOCS detects a malicious code in the form of a worm or an application which is causing network traffic in the system. In this mode, all the outgoing & incoming connections from the host will be blocked; isolating the host from the network till the system is uninfected from the malicious code. The outgoing traffic burst will be monitored and the user will be notified of the application causing it.

2.4. System Workflow

2.4.1. Outgoing Connection

Let's say an SMTP application wishes to communicate over the network. This layer uses the APIs provided by our layer. This request is then sent to the HOCS layer which maps the "well known" port to a random port and sends out the message to the Translation module which adds the required port info and sends it to the Transport layer which treats it as a regular message. The incoming messages of this connection are taken in through the mapped port only; this hides the actual port thus hiding the application details. The Application Monitor will keep a check over this connection and will sense if an irregular amount of traffic is being generated on it, which can be the case if the application is infected with a worm. In that case, the Monitor will send this information to the HOCS Control which will switch the current network mode to Lockdown, till the user resolves this issue.

2.4.2. Incoming Connection

For an incoming connection, the initiator will send a request to the "well known" port corresponding to the application he requires and if the headers match the protocol and the port number the HOCS allows communication to the application on that port. If not, then the HOCS forwards the connection to the 'Honeypot' dummy service which acknowledges the connection and in turn gives access to some dummy data and tries getting the attacker details and once it obtains these details, they are sent to the Application Monitor to add in the blacklisted addresses list.

3. MAJOR FEATURES

3.1. Port Mapping

According to the design, each application protocol will operate in their standard ports for incoming connections, for e.g. the SMTP will function on 25, and will accept all the connections on this port. Whereas, for the connections initiated by HOCS based client the port will be mapped to a random port to maintain transparency for network application. The receiving host will communicate only via the mapped port of the host.

Through this the attacker sniffing the packets on the network, will not be able to access the details about the application running on the host.

3.2. Honeypot Implementation

All the ports which have not been currently mapped to any application will be by default set as open and a dummy service in the HOCS layer will be listening on these ports, therefore this service will try to initiate a connection with the attacking host and will give access to a dummy file system in order to entice the attacker to get further details regarding the attacker like IP and location for monitoring purposes.

3.3. OS Control over Network Security Modes

The HOCS will provide the OS a facility of switching over to different network security modes for application communication over the network. For e.g. If the HOCS detects a malicious worm in the system which is sending multiple scans to different machines, the HOCS will inform the OS and switch to a Lockdown mode till the appropriate action is taken to remove the worm, this will result in network protection.

3.4. HIDS Replacement

The HOCS based system will inherently perform all the tasks of a Host based Intrusion Detection System by monitoring all the network applications as all the packets sent by each of them have to be sent through HOCS. Therefore, the additional load of having a HIDS System in the Host is not required.

3.5. Implicit Firewall Implementation

The HOCS itself acts as an advanced application layer firewall by monitoring all the host ports and providing additional facilities like blacklisting detected attackers, honey pot based mechanism for all ports, and malicious scans detection apart from performing all the tasks that a generic Stateful Firewall will perform.

The default system firewall will be overridden by the HOCS System.

3.6. Code Portability

All Operating Systems have their own proprietary libraries which provide a Networking API to users; therefore, network based applications developed for a particular OS cannot be ported to other systems. OS with HOCS implementation will share a same set of APIs to its users, built over their proprietary libraries.

3.7. Backwards Compatibility

For application protocols not using the HOCS API for their network communication functionalities, their messages will be intercepted by the HOCS based Firewall and will be sent to a translating service which will translate the message in the HOCS format and send it again.

4. SECURITY THREATS

We have highlighted some major network threats and their prevention using HOCS based system.

4.1. OS Fingerprinting

Different Operating systems set the default configuration attributes for a network connection differently. The combination of these default parameters can then be used by hackers to infer the operating system being used. This is OS Fingerprinting.

The HOCS system will set the configuration attributes like Initial packet size, Initial TTL, Window size, Max segment size, Window scaling value, "don't fragment" flag, "sackOK" flag,"nop" flag by itself. Therefore, the configuration attributes are OS independent and hence the hacker won't be able to infer anything about the OS.

4.2. Port Scanning

Attackers use port scanning techniques to search for open ports on which exploitable applications are operating on a host machine. This enables them to exploit these applications and gain access to the remote machine stack. This is called port scanning.
In the HOCS system, all ports are open by default and are mapped to a dummy service which is used as a honeypot hence rendering these techniques useless.

4.3. Worms Attack Detection and Safeguarding

Worm is a malicious code which spreads automatically without user interaction. They mainly spread through mails, scanning and infecting the contacts of the infected host.
Our HOCS layer monitors the applications and their traffic and looks out for applications sending/receiving huge amount of data in a short period of time and alerts the user depending on the HOCS mode. The performance of this system can also be increased by implementing an unsupervised machine learning approach.

4.4. Remote Access Trojans Attacks

A Remote Access Trojan (RAT) is a malware program that includes a back door for administrative control over the target computer. RATs are usually downloaded invisibly with user-requested program such as a game or sent as an email attachment. The attacker can then remotely execute system commands on the infected host.

The HOCS layer will recognize and identify patterns where-in a system command is executed just after receiving a packet. In passive mode, the user will be alerted of such behavior and in the strict mode -the HOCS system will block all such connections.

4.5. DNS Poisoning Attacks

In DNS poisoning, the attacker introduces data into a DNS name server's database, thereby resulting in rerouting a request for a web page and causing the name server to return an incorrect IP address ultimately diverting traffic to another computer.
The HOCS system blocks any kind of direct modifications to a DNS name server's database. The HOCS system provides an internal mechanism to cater to such requests.

5. PREVENTION OF MODERN HTTP BASED DENIAL-OF-SERVICE ATTACKS

These attacks are usually performed on a server based system, a server with HOCS implementation will have the additional protection against DOS attacks.

5.1. Slowloris

In this method, the attacker tries to keep many connections to the target web server open and hold them open as long as possible by sending a partial request. It updates HTTP headers periodically, adding to the header but never completes the request. Affected servers keep these connections open, filling their maximum concurrent connection pool, eventually denying connection attempts from other clients.

The HOCS system scans such slow requests and checks the IP address of the requests. If the server becomes busy, it blocks such requests if multiple requests of this type map to the same IP address. Such requests are responded to when the load clears and sockets are made available. This will result in slow delivery of messages to the attackers and the legitimate users will remain unaffected and communicate at normal bandwidth.

5.2. Slowpost

In this method, the attacker sends POST headers with a legitimate "content-length" field that lets the Web server know how much data is arriving. Once the headers are sent, the POST message body is transmitted at a slow speed to gridlock the connection and use server resources.

The HOCS system will use a time-logging mechanism and save the average time between the POST messages and in this time, save the context and create a new connection for this socket to serve GET requests (GET requests will end before the time). Using a multi-threaded system if the POST request arrives before the expected time, it will be given a higher priority and the GET Request will be sent to a waiting queue till the POST request is served.

REFERENCES

[1] H. Yang, P. Li, Q. Zhu, and L. Xu, “The Application Layer Protocol Identification Method Based on Semisupervised Learning,” in 2011 12th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, 2011, pp. 115–120.

[2] J. Hwang, T. Xie, F. Chen, and A. X. Liu, “Systematic Structural Testing of Firewall Policies,” IEEE Trans. Netw. Serv. Manag., vol. 9, no. 1, pp. 1–11, Mar. 2012.

[3] M. N. Kondalwar and P. C. J. Shelke, “Remote Administrative Trojan / Tool (RAT),” vol. 3, no. 3, pp. 482–487, 2014.

[4] K. Salah, K. Elbadawi, and R. Boutaba, “Performance Modeling and Analysis of Network Firewalls,” IEEE Trans. Netw. Serv. Manag., vol. 9, no. 1, pp. 12–21, Mar. 2012.

[5] M. Y. Arafat and M. M. Alam, “A Practical Approach and Mitigation Techniques on Application Layer DDoS Attack in Web Server,” vol. 131, no. 1, pp. 13–20.

[6] S. Jayakumar, “NETWORK SECURITY – MAC ADDRESS BLOCK,” Int. Conf. Netw. Commun. Comput., pp. 419–422, 2011.

[7] S. Jayakumar, “Automatic Campus Network Management using GPS,” Int. J. Comput. Sci. Issues, vol. 9, no. 3, pp. 468–472, 2012.

[8] J. Sadhasivam, asha J. M, and N. S, “Intelligent Interior Mapping using Wall Following Behaviour,” Int. J. Trend Res. Dev., vol. 3, no. 6, p. 112, 2016.

[9] Benu Singh, Sunita Bansal and Puneet Mishra , Artificial Neural Network Modeling and Optimization In Honing Process , International Journal of Computer Engineering and Technology, 7(3), 2016, pp. 67 – 77 .

[10] V.Ranganayaki and S.N. Deepa, Optimal Neural Network Models For Wind Speed Prediction. International Journal of Electrical Engineering & Technology , 6 (7), 2015, pp. 01-12