Theoretical and Applied Informatics
ISSN 1896–5334
Vol.24 (2012), no. 1
pp. 95–106
DOI: 10.2478/v10179-012-0005-4

Applying of security mechanisms to middle and high layers of OSI/ISO network model

MARCIN KOŁODZIEJCZYK, MAREK R. OGIELA

AGH University of Science and Technology,
EAIiE department, Institute of Automatics
al. Mickiewicza 30, 30-059 Kraków, Poland
marcink@agh.edu.pl; *mogiela@agh.edu.pl*

Received 16 November 2009, Revised 31 January 2012, Accepted 16 February 2012.

Abstract: This article describes security mechanisms used by 3rd-7th layers in OSI/ISO network model. Many of commonly used protocols by these layers were designed with assumption that there are no intruders. Such assumption was true many years ago. The network situation has been changed for last few years and we should realize that some properties of existing protocols may be abused. Moreover, we should exchange some of them or create new versions. There are some methods and guidelines concerning secure programming, but there is also lack of guidelines about creating secure protocols. Authors see the necessity of such guideline and this article is an attempt at analysing existing solutions and selecting some universal and important patterns.

Keywords: AES, authentication, authorization, application, community string, dictionary attack, DDoS attack, DNS, DoS attack, hash function, HTTP, ICMP, IPv4, Ipv6, man in the middle attack, MD5, OSI/ISO model, password, port scanning, protocol, SHA-1, sniffing, SNMP, spoofing, SYN flooding, TCP, tunnelling, UDP

1. Introduction

The purpose of this article is to describe some security mechanisms and restrictions which may be used in third and higher layers of OSI/ISO network model. Before we start, we should realize that people started to care about network security much later than networks were involved. At the beginning there was no security, there were another needs and issues like too slow connections, too many faults on physical links, etc. Internet was not so popular and only small group of people had access to the network. Therefore, there was no reason to design secure protocols. Simple, not complicated protocols were the best. The same situation was with system architecture. Simpler was

better. For example PKI architecture or key exchange protocols are still not implemented in many systems. Many protocols, which are used today, were designed many years ago. There was no reason to think about intruders. Today, networks' attacks are a big problem, that there are no security mechanisms in many popular protocols. There are also many applications which were designed to work with these, unsecured protocols. These applications are not rewritten to secure versions, because of compatibility requirement or costs. Moreover, many times, users do not care about security, some useful features are much more important than data security. We will try to describe some important security issues and try to answer for the question: „is there any way to make network connections more secure?"

2. TCP attacks

TCP/IP is the most popular protocol in the internet. These protocols are used by most of network applications. TCP protocol is very popular because of very good fault control mechanisms. This protocol can adjust transmission's speed [3]. It is well tested mechanism, because TCP has been used for many years and is still very popular. However this protocol is not free of vulnerabilities. One of the most popular attacks is TCP hijacking.

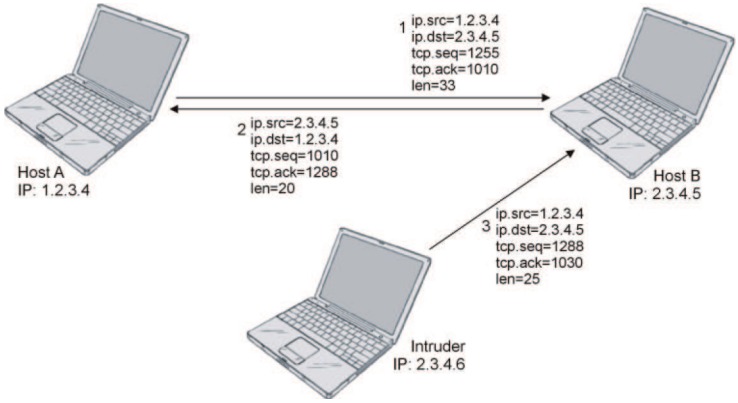

Fig. 1. TCP/IP hijacking diagram

TCP headers use sequence and acknowledgement numbers. The sequence number is an offset to first byte in current TCP packet, so it is increased in every next packet by total length of current packet. However, if SYN flag is set, sequence number is a number of first byte in the TCP session. Usually, initial sequence number is equal to 1. Acknowledgement number is used with ACK flag, it is an acknowledgment that packet with specific sequence number was received. It allows to a tricky attack, called TCP/IP hijacking. An example is shown in figure 1. First TCP/IP packet is sent from host A

to host B. Then, host B sends an acknowledgement, that that packet with specific SEQ and ACK numbers was received[1]. If an intruder listens and receives whole packets, next sequence and acknowledgement numbers can be counted in easy way. The attacker can try to send a prepared TCP/IP packet to a victim with previously counted sequence and acknowledgement numbers. Figure 1 illustrates such situation. There are at least three consequences of sending such, prepared packet. First of all, packet will be accepted and processed by a receiver, by a victim. Second consequence is that the real sender will send TCP/IP packets with wrong numbers, because he will not increment sequence number. Third consequence is that ACK packets, sent by a victim will be ignored because of wrong acknowledgement numbers.

What can be done against such attacks? First, well known method is to used initial sequence numbers (ISN) which are different than 1 [7]. The TCP protocol description recommends that the value of this 32-bit counter should be increased by one every four microseconds. Second, better method is to use random generated initial sequence numbers recommended by DISA in [7]. It makes ISN prediction more difficult. The second fact is that real sender can detect TCP/IP hijacking. Authors are considering about other TCP attacks against the intruder, for example RST attack which will be described in next paragraph.

A very simple form of injecting TCP/IP packets is RST injecting. If the source is spoofed and acknowledgment number is correct, the receiver will accept injected RST packet and reset the connection. It is very hard to prevent against RST attacks. The only way it to prevent and block spoofed packets. Moreover, such prevention, blocks also TCP/IP hijacking attacks.

TCP protocol requires some resources for creating and for existing connections. New connection is created by three-way handshake process described in [3]. This is required by fault management in TCP. This fact can be utilized by attackers. Every TCP packet with SYN flag reserves some resources for new connection. If a victim receives such packet, sends back SYN/ACK packet and waits for third – ACK packet. If ACK packet is not sent, resources are still allocated for new connection. Many TCP packets with SYN flag can cause very effective “Denial of Service”. Such attack is known as SYN flooding attack.

3. Internet Control Management Protocol tricks

ICMP is one of the core internet protocols. This protocol is used in many situations, for example diagnosing a connection or determining route of packets from source to destination. It is very helpful in many fault situations. There is another side of this

[1]TCP.ACK number in ACK packet is equal to sequence number plus data length. TCP.SEQ number in ACK packet is equal to TCP.ACK number in first packet.

protocol. ICMP can be abused in many ways. The most popular way of attack is “Denial of Service”. Ping requests may be sent one by one. Such attack is called ping flooding. The goal of ping flooding is to use whole victim’s bandwidth for ICM packets. In that way, other services will be unavailable. Moreover, maximum length of ICMP data is $2^{16} = 65\,536$ bytes. Huge number of such, large echo request packets[2] can cause strange behaviour of many network devices. Several operating systems crashed if there were sent long ICMP echo requests. Such attack is known as “Ping of Death” attack.

Denial of Service by ping flooding can be detected in an easy way by border routers, firewalls or Intrusion Detection System. ICMP echo requests can be blocked on these devices in such situations. There is another, more interesting attack called Distributed Denial of Service. Source address can be spoofed. It is much more difficult to detect such attacks, because source addresses are different in every packet. There are no repeated, identical echo requests. The second, more efficient way is to infect as many hosts as possible all over the internet and then simultaneously sent ping requests from these, thousands of hosts in the same time. Such attacks are very hard to detect and are very effective.

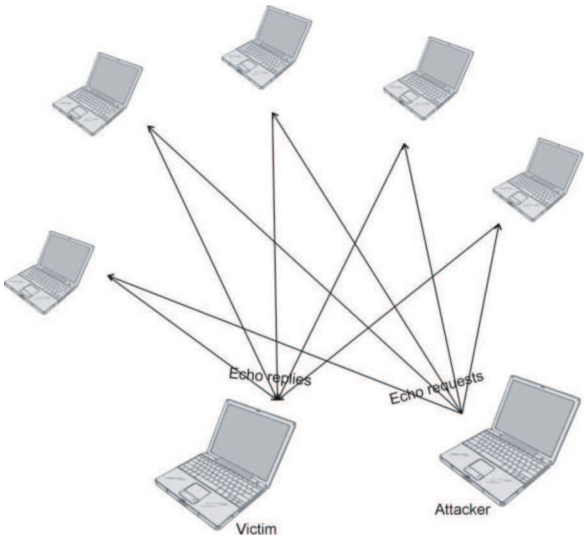

Fig. 2. Amplification ICMP attack

The other, very clever ping attack is an amplification attack. Every echo request causes echo reply, sent back to the receiver. There is no authentication in ICMP, so source IP address can be exchanged in echo requests. If the attacker sends ping requests with victim’s IP to many hosts in the network, all these hosts will send back replies to the victim. In that way, huge percentage of victim’s bandwidth will be used by ICMP re-

[2]ICMP echo request packet is sent as a result from ping command.

sponse packets. This is another example of tricky DoS attack. More detailed description of mentioned attacks is in [1].

4. Services discovering by port scanning

Port scanning is a well-known method for discovering network services. It is used by many network security scanners, for example Nessus or Foundstone. Every network service listens on specific, previously defined or dynamic port. These ports accepts new connections, rest of ports should be closed. The basic variety of port scanning is to send TCP packets with SYN flag. Every open port should response with SYN and ACK flags. Such algorithm may cause Denial of Service, because of half open connections. After received SYN/ACK packet, attacker should send ACK packet to finish three way handshake and then close the connection or just send the RST packet to tear down the connection immediately, as described in [1]. Port scanning by SYN flooding can be easily detected and blocked, because of large number of ports – 65536. Therefore, hackers involved next variety of port scanning, called spoofing decoys. This method spoofs connections from various decoy IP addresses between each real port scanning connections.

RFC 793 defines some specific behaviour of closed TCP ports [14]. According to this document[3], RST packet should be sent by closed ports as a response for FIN, X-mas (with Urgent, Push and Fin flags) or null packets (all flags are set to zero). If a port is listening, RST packet will be not sent. This vulnerability may be abused to detect open ports. If the attacker knows open port numbers, he can guess what kind of services are running. Standard port numbers are described by IANA organization in [13]. Vulnerabilities in standard, well known services are described in the internet and may be utilized by potential intruders.

Last method of port scanning – which is described in this article – is idle scanning. The attacker needs to find an idle host that is not sending or receiving any other network traffic and has a typical TCP/IP implementation. IP identification number is usually incremented by a constant value in every packet. The ID difference between two packets can be checked by sending few TCP/IP packets and receiving answers. The next step is to get know current ID in IP packet. It can be done by for example SYN/ACK packet and received RST packet. Afterwards, the attacker may send SYN packet with spoofed source IP address which should be equal to idle host's IP [11]. If a target, a victim has an open port, SYN/ACK packet will be sent to idle host and idle host will response to a victim using RST packet with incremented ID. In case of closed port, nothing will be sent to an idle host and ID will not be incremented. Then, the attacker may check second time the current ID of idle host and check if RST was sent between these two tests.

[3] Some implementations of TCP are not 100% compatible with the standard and do not send RST packets.

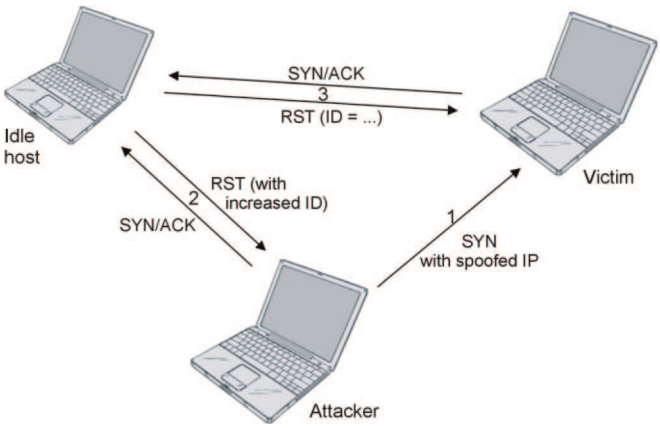

Fig. 3. Idle port scanning – use case with open port

To sum up, some easy SYN, FIN, X-mas or null packet attacks can be easy detected by border devices like firewalls or IDSes. If such attack is detected, IP address of an attacker can be blocked. The next question is what about other methods of protection against port scanning. The best way is to cheat an attacker. For example, some modifications in TCP/IP stack implementation may be introduced. Every port can send SYN/ACK as an answer for SYN packet, independently of port state. In this way, an attacker will always receive SYN/ACK responses. Moreover RST responses in some, above described situations may be blocked. Such methods cause some incompatibilities with a standard but increases security level and are known as proactive defence.

5. Evolution of SNMP protocol

Simple Network Management Protocol is a good example of security evolution in network protocols. The basic idea is that there are two types of network devices[4] - agents and managers. Manager can sent some simple packets to agents like set or get requests and agents do some tasks. Agents can send only two types of messages: responses and traps [3]. First version of this protocol has all possible security vulnerabilities. SNMP does not specify protocol on transport layer. Both, UDP and TCP may be used with SNMP, so spoofing may be used in a very easy way. The only security mechanisms in SNMP v.1 is so called “community string”. This field is constant in every SNMP message and can be sniffed and reused later. Moreover SNMP community string was on the top ten most critical internet security threads in 2000 year. In many implementation, community string is set to default, is not changed. SNMP v.2 adds few types of com-

[4]The same device may work as an agent and master in the same time. SNMP v.3 does not separate agent and master roles.

mands and errors, but the security level is the same as in version 1. The most interesting is version 3 of SNMP. One of goals of this version is to add new security mechanisms and to prevent some basic network attacks. All security features added in SNMP v.3 are optional. SNMP v.3 may be used with three different security levels: no authentication, authentication and no privacy, authentication and privacy. Privacy may be implemented by using symmetric algorithms like DES or AES and authentication is provided by one-way hash functions like MD5 or SHA-1, well described in [4] and [5]. We should give attention that it is not enough, because such attacks like brute force or dictionary attacks may be still used [2]. However evolutions of this protocol shows how important are and still become security issues.

6. Domain Name Servers

Domain Name Servers are one of the most popular servers all over the internet. People prefer words' names than IP numbers. DNS servers translate IP addresses to domain names, so at the beginning almost every connection uses this service [3]. DNS queries and responses use UDP protocol. DoS attack with UDP is hard to perform, because this protocol does not require any resources, but there is no problem with sniffing on UDP layer [11]. DNS queries and responses have to be connected somehow and this is the reason on Transaction ID field in DNS packets. This field causes spoofing attack more difficult, but such attack is still possible. Both, DNS servers and hosts using DNS have DNS cache. There is an assumption that domain names are not changed very often, so DNS entries may be cached. After some time such entries are deleted and the host (or DNS server) has to send a query to the specific DNS server. It is possible to send queries between two servers, because there is a huge amount of DNS entries and this is too big number for single server. The attacker may send responses with wrong IP addresses, but the question is how to get the Transaction ID number? The attacker may create his own DNS server, for example: *attacker.domain.com*. The next step is to send a query about *abc.attacker.domain.com* to the *domain.com* DNS. The *domain.com* DNS sends back query about abc's IP and the attacker receives current Transaction ID [1]. Many DNS servers increases this ID by one in every packet, so the attacker may send many packets with ascending IDs, beginning from received ID plus 1.

There is a DNSSec project, but this protocol is still tested and not very popular. This protocol uses cryptography and provides such security mechanisms like authentication, transactions and key distribution. Authors put hopes on this protocol, because existing DNS protocol is one of the biggest internet vulnerabilities. DNS's vulnerabilities may be used with one of the most dangerous attacks - "man in the middle" attack [5]. This attack allows putting an intruder's host inside the connection between two victims. There are some tools which may generate very similar key signatures to original and then exchange

them. Users usually remember first and last few characters of signature, so similar, exchanged keys may be recognized as correct keys. DNS hacking is a first step of such attack and this step is very easy to perform.

7. Tunnelling issues

HTTP protocol (Hypertext Transfer Protocol) is also very popular in the internet. Many companies allows for outside HTTP traffic. Other services are blocked very often, but web pages are usually accessible [6]. Therefore much malicious code uses HTTP tunnels for its connections. This is the way which may be used for stealing confidential data from companies. This method is well known by administrators and HTTP traffic is often monitored and checked. Such monitoring is not a simple task, because of huge number of possibilities and potential HTTP issues. As we wrote HTTP tunnelling is usually checked but it is not the only allowed protocol in most of situations.

Outside DNS traffic is allowed too, in most of companies. Many administrators do not realize that DNS may also be abused for tunnelling purpose [7]. The idea is that the intruder should have his own domain for example: *thief.domain.com*. If somebody (or malicious software) wants to send some information, it may be done by converting this information to Base32 standard and then send a DNS request with previously encoded bytes, for example: *confidential.and.stolen.information.thief.domain.com*. Responses from DNS are also allowed, so pseudo-DNS server may send control information back. Such traffic may be discovered because of big number of DNS queries from one, specific host. However, there is no problem with spoofing, because of UDP usage. Source addresses may be exchanged to every address from the network.

As we see, there is another problem with protocols: abusing of them. It is very hard to create a protocol which cannot be abused for some illegal purpose. Moreover, in many situations, it is very hard to detect such illegal usage. Authors are considering what restrictions may be used? What mechanisms can block abusing of network protocols? This question is very difficult. Perhaps, the research should focus on the entropy, but it is only suggestion and the question is still open. There is no universal rule now. People who involved DNS tunnelling are very excited of DNSSec. The basic goal of DNSSec was to build more secure networks, but there is the other side of the coins. DNSSec is a protocol which is much more secure than typical DNS, but detection of DNSSec tunnelling may be much more difficult.

8. Perspectives

This article describes only the most popular network interfaces and protocols used in high and middle layers of OSI/ISO model. The conclusion is that we are still using

insecure protocols. Vulnerabilities in these protocols may be utilized by many potential attackers. As we saw in last chapter, attackers can also take advantage of changing aim of well-known protocols. The last conclusion should be taken into consideration by all administrators and developers creating network protections. Many possible attacks can be eliminated by simple methods, for example by using random numbers which are hard to predict, to use IPv6 with autoconfiguration mechanism as spoofing prevention method, etc. Table 1 contains above described network vulnerabilities and some good advices how to protect against attacks which utilizes described bugs.

Vulnerability	**Protection method**
TCP/IP hijacking	• using random initial sequence numbers • resetting of hijacked TCP connection
SYN flooding	• do not allocate additional resources and send back error packet (method not implemented, it is only authors' idea) • detect, block and report unwanted SYN packets
ICMP DoS attack	• detect, block and report unwanted ICMP requests and all traffic from attacker's IP
Amplification ICMP attack	• using Ipv6 with auto-configuration mechanism.
SYN port scanning	• detect and block such traffic on firewall • send back SYN/ACK packets also for closed ports
FIN port scanning	• do not send RST packets.
Idle port scanning	• generating random IP identification numbers
SNMP v.1 and v.2 attacks	• using of SNMP v.3 only in authentication and privacy mode
DNS poisoning	• using random Transaction IDs • using DNSSec
Stealing data through HTTP or DNS tunneling	• using Intrusion Detection Systems or Intrusion Prevention Systems to detect and analyze contents of every packet
dictionary attack	• using of long and strong passwords and keys
man in the middle attack	• protection of DNS servers • using PKI and certificates

Tab 1. Vulnerabilities and protection methods

Authors are working on some guidelines which can be used for existing and new protocols [12]. Knowledge and security experiences in existing protocols may be im-

portant and useful. First, we should analyse existing protocols, protection and security mechanisms, existing bugs, faults and vulnerabilities. The second step is to gather all this things together and then to start formalization process. There is big number of some guidelines and tutorials how to create secure programs, but it does not guarantee that software is or will be secure. Authors strongly believe that wrong, unsecured protocols can cause faults and security issues in software. Sometimes protocols force damageable solution in created software. For example if protocol requires clear text password for authentication, the application cannot be secure. We have described popular and known protocols and we see the necessity of protocol protection against network security issues.

References

1. J. Erickson-Hacking: *The Art of Exploitation – No Starch Press*, San Francisco 2003.
2. M. Howard, D. LeBlanc, J. Viega: *The 19 Deadly Sins of Software Security – McGraw-Hill/Osborne*, California 2005.
3. S. Andrew: *Tanenbaum – Computer Networks (4th Edition) – Prentice Hall*, New Jersey 2003.
4. A. J. Menezes, P. C. van Oorschot, S. A. Vanstone: *Handbook of Applied Cryptography* – CRC Press 1996 (online version).
5. M. R.Ogiela: *Security of computer systems* – Wydawnictwa AGH, Kraków 2002 (in Polish).
6. The Top 10 Most Critical Internet Security Threats – (2000-2001 Archive) - http://www.sans.org/top20/2000/
7. Security Technical Guidelines from DISA – http://iase.disa.mil/stigs/stig/index.html
8. M. R. Ogiela, U. Ogiela: *Linguistic Cryptographic Threshold Schemes*, IJFGCN – International Journal of Future Generation Communication and Networking, Vol. 2, No. 1, March 2009, pp. 33-40.
9. M. R. Ogiela, U. Ogiela: *Linguistic Extension for Secret Sharing (m, n)-threshold Schemes*, 2008 International Conference on Security Technology, December 13 ~15, 2008, Hainan Island, Sanya, China, pp. 125 – 128, ISBN: 978-0-7695-3486-2, DOI: 10.1109/SecTech.2008.15
10. M. Kołodziejczyk: *Rainbow tables as brute-force algorithm optimization* – Elektrotechnika i Elektronika, Kraków 2009 (in Polish, in press).
11. M. Kołodziejczyk: *Applying of security mechanisms to low layer of OSI/ISO network model* – Automatyka, wyd. AGH, Kraków 2010, vol. 14, pages 23-30.
12. M. Kołodziejczyk: *Nieinwazyjne metody audytu i poprawy bezpieczeństwa systemów komputerowych* – Studia i Materiały Informatyki Stosowanej, Wyd. Uniwersytetu K. Wielkiego, Bydgoszcz 2010, vol. 2, pages 23-31.

13. IANA port numbers – http://www.iana.org/assignments/port-numbers.
14. RFC 793 – Transmission Control Protocol Specification – http://tools.ietf.org/html/rfc793

Mechanizmy bezpieczeństwa w zastosowaniu do środkowych i górnych warstw sieciowego modelu OSI/ISO

Streszczenie

Artykuł opisuje popularne mechanizmy bezpieczeństwa stosowane w istniejących protokołach sieciowych środkowych i górnych warstw modelu OSI/ISO (od trzeciej warstwy włącznie). Wiele spośród tych protokołów zostało zaprojektowanych bardzo dawno temu i chociaż do dzisiaj są wykorzystywane i spisują się bardzo dobrze to jednak umożliwiają pewne nadużycia swojej funkcjonalności. Wiele z opisanych tutaj protokołów nie bierze pod uwagę ataków sieciowych. Autorzy postanowili wyselekcjonować pewną grupę protokołów, które mogą posłużyć do analizy zagrożeń sieciowych. Dzisiaj, mając pewne niebezpieczne rozwiązania jesteśmy bogatsi o pewne praktyczne doświadczenia z tym związane. Pozwala to analizować zagrożenie, klasyfikować je i wreszcie skutecznie im przeciwdziałać w nowych protokołach. Często są to rozwiązania lokalne, tworzone przez pewne grupy studentów lub konkretne firmy. Powstają również nowe, globalne rozwiązania. W pierwszym przypadku, nowy protokół może stanowić realne zagrożenie ze strony intruza. W drugim, należałoby poddać przynajmniej częściowej analizie formalnej nowe rozwiązanie. Istnieje wiele metod skupiających się na tworzonym oprogramowaniu, jednak często protokoły, jako rozwiązania autorskie są pomijane w analizie. Artykuł jest również próba wstępnej klasyfikacji zagrożeń i stworzenia pewnych uniwersalnych rad dla twórców nowych rozwiązań. W pracy zarysowuje się potrzeba zmian pewnych istniejących rozwiązań, których słabości są opisane w artykule. Autorzy są przekonani, że niebezpieczny protokół nie może być użyty w bezpiecznym programie, bo jak np. można stworzyć bezpieczny program, jeśli protokół nie posiada odpowiedniego mechanizmu uwierzytelniania? Ten i wiele innych aspektów bezpieczeństwa zostało w pracy poruszonych i omówionych na przykładzie istniejących rozwiązań.