Securing against Denial of Service attacks in remote energy management systems

Nikolay Rumenov Kakanakov and Grisha Valentinov Spasov

Abstract - **This paper presents a review of denial-of-service attacks and methods of defense in the field of SCADA and embedded applications. It investigates the vulnerabilities of embedded systems to the general security leaks, known from the general purpose and enterprise distributed systems. The similarities and differences of the attacks that desktop and embedded users experience are discussed. An embedded application for remote management of electric energy, based on ARM9 processor and Linux OS is used as an example to test the security leaks and vulnerabilities. The security tests are run using Nexus software, which is capable of determining general security risk as long as typical SCADA vulnerabilities.**

Keywords – **Embedded security, Remote measurement, DoS, DdoS, SCADA.**

I. Introduction

The continuous growth of cyber security threats and attacks including the increasing sophistication of malware is impacting the security of critical infrastructure, industrial control systems, and remote energy management systems. The reliable operation of modern infrastructures depends on computerized systems and SCADA systems. Since the emergence of Internet and World Wide Web technologies, these systems were integrated with business systems and became more exposed to cyber threats [1].

The importance of these Internet service applications makes their resilience to attacks and failures critical. However, studies show that the security and availability of Internet service applications are increasingly threatened by a variety of attacks.

Among these incidents, Denial-of-Service (DoS) attacks pose one of the most serious threats to Internet service applications [2] [3].

A key element of embedded systems is their power consumption. Some of the devices should work in remote places on battery power and without human control. In these cases the DoS can be used to reduce the battery life. Special care must be taken to reduce this possibility with defense mechanisms that do not consume much power. In many cases the aim of the attack is not to spoof the device but to turn its defense mechanisms on many times to consume power [4].

N. Kakanakov is with the Department of Computer Systems and Technologies, Faculty of Electronic and Automation, Technical University – Sofia, Plovdiv branch, 25 Tsanko Djustabanov Str., 4000 Plovdiv, Bulgaria, e-mail: kakanak@tu-plovdiv.bg,

G. Spasov is with the Department of Computer Systems and Technologies, Faculty of Electronic and Automation, Technical University – Sofia, Plovdiv branch, 25 Tsanko Djustabanov Str., 4000 Plovdiv, Bulgaria, e-mail: gvs@tu-plovdiv.bg

II. Denial-of-Service Attacks

A DoS attack on an Internet service application can be achieved by consuming critical resources (such as network bandwidth, server memory, disk space, or CPU time) on which the application or access to the application depends. Depletion of these resources can prevent the application from functioning, or disconnect the application from the Internet, and thus make the application unavailable to its users. A DoS attack occurs either at the infrastructure-level by attacking the resources directly (e.g. by flooding the applications sub-network with IP packets), or at the application-level by attacking through the application interface (e.g. by overloading the application with abusive workload) [5] [6].

In a typical DoS attack, an attacker first compromises a number of hosts (chosen from the millions of vulnerable hosts) in the Internet, and then instructs them to attack an application by sending either infrastructure-level or application-level attack traffic to it [5].

A. TCP SYN Flood

In this attack a weakness in the TCP three-way handshake is used. On receiving a SYN segment, replies with SYN/ACK segment and the server changes its state to SYN_RCVD waiting for the final ACK to move to ESTABLISHED state. Every server has limited space (list) for storing information for half open connections (SYN received, but not yet established) and attackers send many false SYN segments to flood the service – Figure 1 [6].

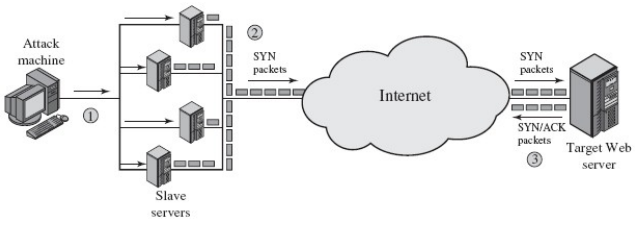

Figure 1. Distributed SYN Flood Attack

B. LAND (Local Area Network Denial)

In this type of attack host from the network sends a TSP SYN packet in which the source address and port are substituted with that of the receiver. The reason a LAND attack works is because it causes the machine to reply to itself continuously. Receiving of such packet on some systems can lead to temporarily unresponsiveness of the communication subsystem. In modern operating systems even in the embedded world this problem is solved.

C. Distributed denial of service (DDoS) attack

Distributed denial of service is very similar to the typical DoS. The difference is that the attacking packets are sent through distributed hosts in the Internet or victim's network that are somehow involved in the attack. These distributed hosts are unaware of the attack and are infected with some kind of malicious software (virus/worm/troyan). The attacker first infects some hosts that are useful for the attack and start some agent software on them that can be synchronized to simultaneously make the attack on the victim – figure 2 [5]. Then the attacker have time to hide the traces before the "zombie" agent start attacking.

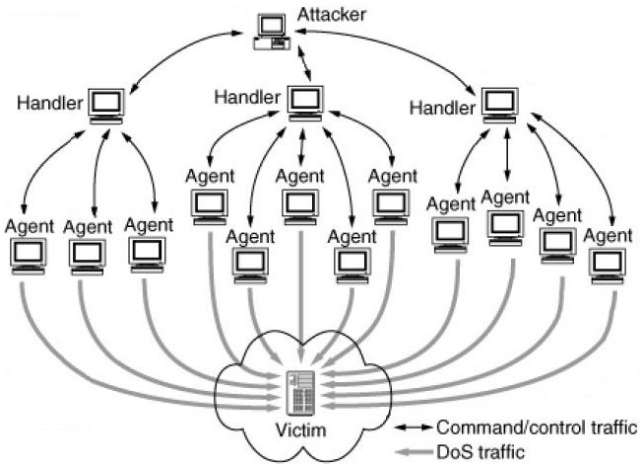

FIGURE 2. DISTRIBUTED DOS

D. Smurf attack (Distributed ICMP flood)

This type of attack is based on the weakness of the ICMP protocol and more accurate to broadcast ping. It uses intermediate network, called reflector, for the real attack. The attacker sends and ICMP ECHO request to the broadcast address of the reflector network, substituting the source address with that of the victim. Then all hosts in the network will reply to the victim simultaneously, causing a flood of the network interface (Figure 3 [6]). The problem with this attack is that it does not use the vulnerability of the victim but the vulnerability of other hosts in victim's network.

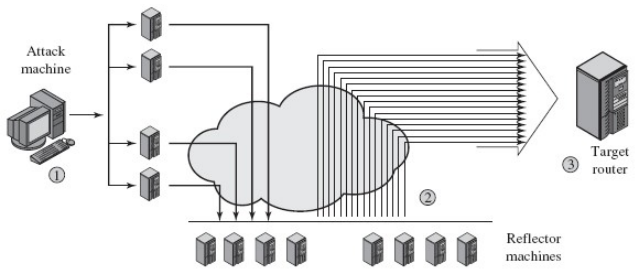

FIGURE 3. DISTRIBUTED ICMP ATTACK

A different implementation of this type of attack is the "Fraggle" attack. The difference is that in this case instead of broadcast ICMP ECHO, a broadcast UDP packet is sent on destination port 7 which is used for Echo service.

Many SCADA systems have custom hardware and operating systems. This makes them not so vulnerable to the common security holes but makes them open to the special SCADA attacks. These SCADA attacks search vulnerabilities in industrial controllers through industrial communication protocols. These types of attacks are very rare but the problem with them is that embedded security software is updated irregularly unlike general purpose software that is updated in days (and even in hours) for every newly found security leak.

III. DOS DEFENSE

The main goal of the DoS defense is its effectiveness. But the effectiveness is a complex term. It should be taken in mind the cost of the operation defensive policies and the cost of their installation. But it is very important that the defense policy should have very low rate of false-positives and not to stop the legal users from reaching the service. In the case of DDoS attack, it is not a good idea just to block packets from the "zombie" network because then a collateral damage will occur. The term collateral damage refers to the situation when third party suffers losses due to the attack [5].

There are three main aspects of fighting with the DoS attacks [5]. First, it is the prevention of the negative effects from the attack – the service continues to work during the attack. It often includes reserving additional resources by means of processing, memory buffers and bandwidth. Second aspect is the reaction to the attack. It often requires more engineering in the design phase but fewer resources in the implementation. Third aspect is the network protection. It is the best way for defense in the corporate network but it is difficult to put on stand-alone remote server.

In the filed of embedded or industrial networking, the prevention mechanisms are too expensive to use. Putting a one more chip of memory to a device produced in millions, costs millions. The defense of embedded/industrial systems can be separated in two main groups – network area protection and personal device protection. As long as industrial networks often are built on multilayer architecture, the defense on the different layers can be applied. On the top layers the defense is straightforward as in non-embedded applications. The defense on the controller level and in controller networks needs different approaches. In some cases it is enough to restrict the types of traffic that can be exchanged between the layers and thus reducing the risk of popular DoS attacks. Second step is applying MAC/IP level filters on the controllers that allow only specified host to communicate with. In such cases the Resurrecting Duckling mechanism [4] can be used. It is very useful in wireless networks to restrict the association of unknown hosts in the network. If the controller network is built using smart Ethernet switches they can be used to separate the traffic and isolate the important communications between hosts from the "junk" traffic.

The main challenge in the embedded/industrial world is the defense of remote stand-alone devices that should provide network services. In such case the power consumption, CPU and memory availability, and network bandwidth are expensive and important features. Some vendors provide API's for packet filtering, or even provide built-in defense to most common leaks.

IV. Implementation of D/DoS Prevention in Linux kernel

As an example application, an embedded system for remote measurement of electric power and energy based on Web services and EP9302 embedded platform is proposed (Figure 4). It runs kernel 2.6.24.7, specially built for ARM9 (armv4tl) with Emdebian/arm packages. The presented example application gets data form power management sensor via its serial port and uses UDP socket server and HTTP server for exchanging measured data with remote database server [7].

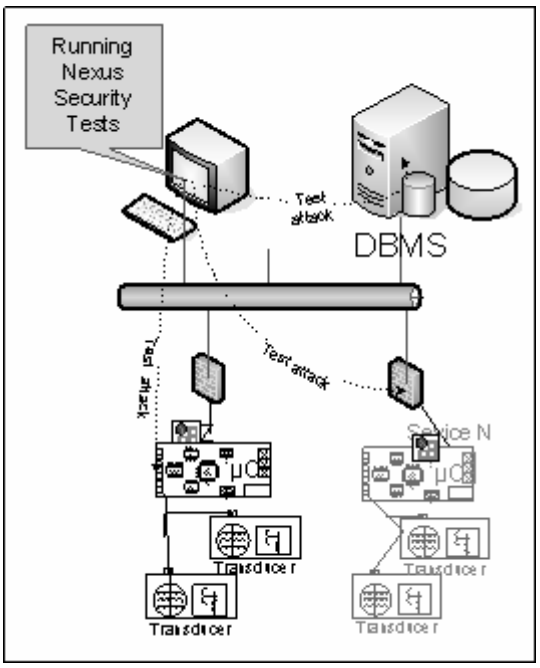

Figure 4. Example System and Test Set-Up

As long as the system uses TCP/IP networking, it can be a subject to DoS attacks. As long as the kernel is very new, it has built-in security. It is not vulnerable to the most common attacks like Teardrop, LAND, Smurf, TCP SYN flood. For the latter, Linux comes with an effective defense called SYN cookies. Instead of allocating space in the connection queue after receiving the first packet, the Linux kernel just sends a cookie in the SYN+ACK packet and allocates space for the connection only after receiving the ACK packet. SYN cookies can be activated on a Linux machine by adding '1' to */proc/sys/net/ipv4/tcp_syncookies*. For the Smurf attack, kernels newer than 2.2 provide an implementation of the Spoof protection described in RFC1812 that will suit most simple network configurations. The LAND and Teardrop attacks are not subject after kernel versions 2.0.32 and 2.1.63. Another step in securing the board is to stop all unnecessary network services in *inet* daemon. As long as it is an embedded systems, it will not need X windows, Mail Transfer Agent, Remote Procedures (except if using NFS), and other user software. So they must not be installed. To make the security of the embedded Linux complete, it is a good idea to add some restriction rules in the IPTABLES tool. Such rules can deny all traffic except the required one (e.g. HTTP, SSH, UDP on specific ports, etc.) and block all incoming echo requests (ICMP or TCP/UDP) except from local network or specific hosts and echo replies to a broadcast ping. This may protect the embedded system not only from attacks but it can assure it cannot be used as "zombie" tool for attacking the local area network. Implementing these steps will improve the security but cannot stop new attacks or overloading network bandwidth. More can be done to secure the whole network with firewall, intrusion detection/prevention system, and secure communication channels.

In this case the vulnerabilities of embedded Linux system appear only on the application layer using specific leaks and bugs in the applications. An example of such leak is vulnerability of Apache web server to opening multiple connections and keeping them open for an infinite period. This could be done with sending a good HTTP header lines without ending two new lines and constitute to send some pointless data with good syntax continuously just to keep connection open. There is no good protection against this issue as the community offers to increase maximum number of connections which is not applicable for embedded applications. Some apache developers [8] offer using of blacklist and *.htaccess* files to block all known bad IP addresses and domains. It reduces the potential attacks but do not protect the systems as the IP addresses of attacker are constantly changing. Even more the system must still provide services to all possible users. Better protection is using SSL connection and client certificates.

The example embedded system is tested for most of known vulnerabilities after applying the commented securing steps. The test is made with popular Nexus tool [9] in local area network environment to cover all issues (Figure 4). It tests for vulnerabilities in TCP/IP stack and some popular applications as *ssh*, *mail*, *web*, *telnet*, *ftp*, *time* servers as long as some specific SCADA leaks. The results show that the system is durable to all tested attacks.

V. Conclusions and Future Work

The presented paper shows that embedded Linux applications are more durable to known security leaks in SCADA systems. Moreover, using Linux on embedded devices provides a platform for easy integration of security policies of service and automation level of SCADA systems. The drawback of the embedded Linux is that it can be prone to popular Internet exploits and must be updated regularly. The main conclusion of the security tests made is that embedded Linux applications can be secured from popular DoS attacks without much effort with just few common steps.

Further experiments should be made to evaluate some of defense mechanisms against CPU and memory use and especially power consumption. Some additional tests should be carried out to investigate the best place for applying firewall rules – network entry points, each device, or both.

VI. Acknowledgments

The presented work is supported by Technical University of Sofia, project "102ни200-3/2010", entitled "Investigation of technologies for development of Web-based systems for measurement and control of electric power systems".

References

[1] Mariana Hentea, "Improving Security for SCADA Control Systems", Interdisciplinary Journal of Information, Knowledge, and Management Volume 3, 2008, pp. 73- 86.

[2]Chee-Wooi Ten, Govindarasu, M., Chen-Ching Liu, "Cybersecurity for electric power control and automation systems", IEEE International Conference on Systems, Man and Cybernetics, ISIC 2007, 7-10 Oct. 2007, pp. 29 – 34, ISBN: 978-1-4244-0991-4.

[3] Ghosh and E. Turrini, "Cybercrimes: A Multidisciplinary Analysis", pp. 27-44, Springer-Verlag 2010, ISBN 978-3-642-13546-0.

[4] Frank Stajano, Ross Anderson, "The Resurrecting Duckling: Security Issues for Ubiquitous Computing (Supplement to Computer Magazine)," Computer, pp. 22-26, April, 2002.

[5] Jelena Mirkovic, Sven Dietrich, David Dittrich, Peter Reiher, "Internet Denial of Service: Attack and Defense Mechanisms", Prentice Hall 2004, ISBN: 0-13-147573-8.

[6] William Stallings, "NETWORK SECURITY ESSENTIALS: APPLICATIONS AND STANDARDS", Pearson Education 2011, ISBN 10: 0-13-610805-9.

[7] N. Kakanakov and G. Spasov, "Web enabled system for remote energy management, " in Journal of Electronics, vol. 4, no.2, pp. 95- 98, 2010, ISSN 1313-1842.

[8] Apache Security Tips. Online: httpd.apache.org/docs/2.0/misc/security_tips.html, [10.06.2011].

[9] Russ Rogers, "Nessus Network Auditing, 2nd Edition", Syngress Publishing, 2008.