Performance Evaluation of On-Demand IP Address Assignment for Layer-2 Devices

R.Gad, D.Baulig, M.Kappes and R.Mueller-Bady

University of Applied Sciences Frankfurt am Main, Frankfurt am Main, Germany
e-mail: {rgad|dbaulig|kappes|mueller-bady}@fb2.fh-frankfurt.de

Abstract

While data link layer devices require no IP address for their operation, they often are run with an IP address assigned for configuration or monitoring purposes rendering the device potentially susceptible to attacks over the network. In this paper, we analyze the performance aspects of a prototypical implementation for assigning an IP address to such a device on demand analogous to port knocking on firewalls, allowing a safer IP-less operation when IP connectivity is not needed while retaining the possibility to connect to the device over IP at any time. Our results indicate that our technique can be employed with virtually no performance penalty.

Keywords

Network Security, Performance, Data Link Layer, Network Layer

1. Introduction

Data link layer devices such as bridges, switches, or bridging firewalls (Eggendorfer & Weber 2007), by definition, operate in a manner that is transparent to higher level protocols like IP, TCP, or UDP. Thus, such devices are fully capable of fulfilling their tasks without the need to run protocols above and beyond the data link layer. Consequently, there is no need to assign an IP address to these devices during regular operation.

However, many modern data link layer devices feature additional functionality that needs to be configured, maintained, and monitored. These tasks are typically performed remotely over the network using application-layer protocols such as SNMP, SSH, or HTTP(S) that are based on UDP or TCP respectively and, in turn, on IP. Thus, such managed data link layer devices are often configured with IP addresses and have services running that listen on publicly accessible ports in order to enable connections for management tasks.

A device with a permanently assigned IP address can be detected much easier by an attacker; furthermore, the device becomes susceptible to multiple attack vectors which can be used in order to, e.g., conduct denial of service attacks or gain access to services or sensible data. This problem is well known and has lead to approaches such as the creation of dedicated management networks disjoint from other "business" networks for such purposes. While this concept may mitigate some security threats, management networks are difficult to set up and maintain as they need to be completely separated from the business networks. The isolation between the networks can easily fail, e.g., due to configuration errors. In practice, to guarantee the separation of networks within a complex network setup, e.g., with large numbers of VLANs or even virtualized environments is an intricate and costly task requiring specialized and highly qualified personnel.

In small to medium sized business (SMB) environments, however, IT-Security is constrained by the human and financial resources at hand (Kappes & Happel 2009). Sophisticated security measures like dedicated management networks often do not exist in these environments at all or are not maintained properly leading to numerous security threats. The approach presented and analyzed here was developed in the context of a project targeted at SMB environments and aims at providing a pragmatic and cost-efficient solution to significantly mitigate risks while keeping deployment and management simple. The goal of our method is to provide an additional layer of security by operating data link layer devices with an IP address only when needed, if other solutions are not viable.

In many cases, data link layer devices such as managed switches do not need to be permanently accessible via services that operate on top of IP. Most use cases such as configuring a device or downloading log data only require access to such devices for very short time spans. Thus, IP addresses and services can be disabled for most of the time without affecting the regular operation of these devices. By disabling IP addresses and services that operate on top of IP a data link layer device remains hidden for the most part and the number of possible attacks against the device, its services, and operating system can be significantly reduced. While it is beneficial if IP addresses and the according services were only active for the duration of these maintenance periods, a method for securely triggering the assignment of an IP address when needed is required.

For services on the transport layer there are established techniques like port knocking (Krzywinski 2003) or more modern variants (deGraaf et al. 2005) (Al-Bahadili & Hadi 2010) that make it possible to enable access to services on demand, but hide or close the service ports during normal operation. The current implementations of these approaches have the limitation that network devices need to be accessible via IP and hence need to have configured IP addresses or provide out-of-band channels (Liew et al. 2010) for port knocking and similar approaches to work.

In order to provide an additional layer of security and to significantly reduce the number of possible attacks in a network we implemented a solution for operating data link layer network devices without configured IP addresses during regular operation and a way for securely activating and configuring IP addresses and services on demand for these devices that only allows authorized persons to activate and configure the higher layer functionalities. The authentication as sketched in our approach is performed via a mechanism that ensures only authorized people are permitted and that is resilient against attacks. This way data link layer devices and administrative services can be effectively hidden from an attacker during normal operation but are still accessible for administrators.

The remainder of this paper is organized as follows: in the next section, we outline our general approach for IP-less operation and a secure and authenticated on-demand IP address and service configuration. Then, we give an overview of the prototypical implementation we used for evaluation. In the main part, we present the procedures and results of performance and load tests that were conducted using our prototypical implementation in order to assess the reliability and robustness of our approach. Finally, we conclude our work and give an outlook on possible further research topics.

2. IP-less Operation and Secure On-demand IP Address Assignment

In the following, we propose a mechanism for the network layer that works somewhat analogously to port knocking on the transport layer. The requirements for our approach can be summarized as follows:

1. The target device does not possess any IP addresses beforehand.
2. An IP address can be assigned to the target device on demand.
3. IP address assignment must be possible from an arbitrary computer within the same subnet.
4. Only authorized entities can trigger IP address assignment.
5. The solution is resilient against different types of attacks such as, e.g., replay or denial of service attacks.
6. Device performance is not significantly impacted by the technique.
7. After IP address assignment, the device behaves like any IP device, i.e., any IP-based service may be used.

In our scenario the device in question is a bridge/switch that forwards packets (please note that in the context of this paper, we do not distinguish between packets and frames) between the connected Ethernet segments (see Figure 1) that are each connected to one of its network interfaces.

Our proposed approach is not cognate to existing device discovery mechanisms as we propose a method for concealing the presence of a device instead of publicly announcing it. Our approach presumes that the legitimate administrator is aware of the presence of the device. The aim of our approach is to hide a device in a network that can only be activated (assigned with an IP address) by its legal operator.

In order to meet the specification above, the device analyzes packets arriving on its interfaces and checks for specific cryptographically protected "Wake-Up" packets as depicted in Figure 1. Via these "Wake-Up" packets the IP address assignment is triggered and information about IP addresses or possibly other management data is securely sent to the device. While specific details are beyond the scope of this paper, we would like to point out that a multitude of possible management and configuration options further improving the security of the device exist including, but not limited to, restricting communications to particular IP addresses, assigning a specific IP address and enabling specific services on the device. For authentication and replay protection our prototype uses pre-shared one-time tokens. Other, standard cryptographic one-way designs can be easily employed as well.

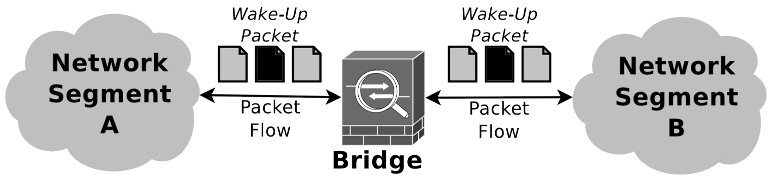

Figure 1: Transparent layer 2 device looking for "Wake-Up" packets

Another problem to tackle is how "Wake-Up" packets are actually sent to the device. Since the device does not possess an IP address it will not respond to any TCP/IP-based protocols including ARP and hence cannot be addressed on the network layer. Moreover, the MAC address of the device cannot be detected unless further protocols on the data link layer such as the IEEE 802.1D Spanning Tree Protocol (IEEE 2004) are active; thus, it cannot be addressed on the data link layer as well. However, as a layer 2 connectivity device, it is an integral part of the network structure and can processes all packets that reach one of its interfaces no matter if these will then be forwarded or not as shown in Figure 1.The task, however, is to ensure that “Wake-Up” packets reach the device independent of its position in the network structure. Our experiments showed that this can be most easily ensured for broadcast packets. The suitability of multicast packets depends on the installed network equipment; in our experiments we encountered situations in which multicast packets were not forwarded by certain layer-2 equipment like some switches. Finally, unicast packets are only suitable for being used as “Wake-Up” packets if it can be guaranteed that our “IP-less” device is located on the path in the network between the source and destination as otherwise these packets won't reach our device at all. More sophisticated approaches could utilize steganography for concealing authentication messages, e.g., in DHCP or other commonly used protocols which rely on broadcasts. However, such approaches are still subject to further research and are not covered in this paper.

3. Prototypical Implementation

We implemented a prototype for our method on an embedded, x86 Linux-based device configured as switch. As network layer protocol we used IPv4. In the following, we present some details of our prototypical implementation. We also outline possible alternatives where indicated. For other types of devices, additional implementation options may exist.

In order to check for "Wake-Up" packets while no IP address is assigned, we employ a sniffer to capture packets. For packet capturing we use jNetPcap (Sly Technologies, Inc. 2012), which builds on top of the libpcap (Tcpdump/Libpcap 2012) library. For analysis, the captured packets are copied into user-space and processed.

Promiscuously analyzing packets in user-space may result in performance issues such as high CPU load, skipped packets, or other negative side-effects, e.g., when the device is exposed to a high network load. Hence, we do not capture the whole network traffic but selectively capture only packets that are likely to be "Wake-Up" packets using kernel level filtering. This way, the number of packets getting passed to user-space is significantly reduced, keeping processing time low and allowing a high detection rate. This is particularly important as otherwise attack vectors for denial-of-service attacks might open up. In the performance evaluation in the next section we will analyze the effects of such flooding attacks in detail.

In our prototypical implementation, "Wake-Up" packets use IP broadcast addresses in conjunction with UDP and a specified port. We use this combination of properties as filter criterion at kernel-level to preselect potential "Wake-Up" packets within the regular network traffic. Clearly, for efficient filtering it is paramount to choose a combination of filtering properties such that only very few of the common network traffic packets match these criteria.

Using broadcasts ensures that the packets are sent to all endpoints on the data link layer level network segment including our device. Using single- or multicast would require us to make assumptions about the network topology and capabilities, which we neither can nor want to make. Since the focus here is on analyzing the performance characteristics of our method, which is not affected by the distribution method, we opted for the straight forward broadcast approach. However, broadcasts can be detected by any other host in the broadcast domain; thus, we are currently developing alternatives for further concealing the presence of our hidden device.

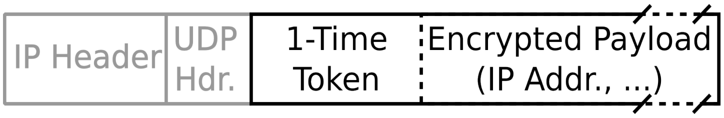

Figure 2: "Wake-Up" Packet Format

In our implementation, the device cannot respond to any wake-up calls until it has acquired an IP address. This necessitates the use of a replay-safe, one-way authorization method such as Single Packet Authorization (SPA) (Rash 2006). SPA allows cryptographically secured authentication by a single, one-way communication channel. In our prototype we use random one-time tokens for authentication as depicted in Figure 2. These randomly created one-time tokens are essentially shared secrets that are stored on both the administration computer and the hidden device. After each successful “Wake-Up” operation once an IP address had been assigned new, randomly generated one-time tokens are exchanged between the formerly hidden device and the administration system via an encrypted connection. Other cryptographic methods like S/KEY are also possible (Haller 1995) (Worth 2004) and the respective data would be placed at the beginning of the “Wake-Up” payload at the same position as the one-time token. Generally, every form of secured authentication is possible that enables authentication via a single, one-way message from the administration system to the hidden device. In our scenario the hidden device will be shipped with pre-generated one-time tokens that get replaced on the first successful “Wake-Up” using a sort of “bootstrap”. From there on only the legit administrator will be able to trigger the on demand IP address assignment as he will be the only one in possession of the required one-time keys.

4. Performance Evaluation

We measured the performance characteristics of our prototype with respect to the following dimensions critical for the practical usability of the mechanism: performance during regular operation, reliability of capturing and identifying "Wake-Up" packets, and performance during flooding-attacks such as brute force or denial of service (DoS) attacks.

First, the network performance of the device itself during regular operation with our proposed technique enabled should be similar to its performance without the secure IP on-demand assignment mechanism. This means in terms of measurable characteristics that, e.g., the throughput should not decrease and round-trip times should remain similar. Factors that could negatively affect these performance characteristics may be, amongst others, the system load caused by running the authentication software, the packet capturing itself, or the kernel-level traffic filtering. We assessed this aspect by performing benchmarks on our test hardware with and without running our prototypical software implementation and comparing the results. We repeatedly performed throughput measurements via UDP and TCP using different IP packet sizes. Ten measurements had been made for each packet size and protocol. Furthermore, we also measured the round trip time via UDP. The test setup was made up of two computers that were directly connected to one of the bridge ports each. The benchmarks were run from these computers in both directions across the bridge.

Our results show that the device performed equally in both cases; the network performance was not affected significantly by running our prototypical implementation (compare Table 1 and Figure 3). In Figure 3 the average UDP throughput for different IP packet sizes during regular operation and while running our prototypical implementation is shown. Some of the depicted values show a slightly higher throughput while running our prototype. These differences are due to common variation in throughput caused by the used embedded x86 Linux platform. For an IP packet size of 1500 byte the average throughput reached about 96.5% of the theoretically achievable throughput in both cases.

	Regular	**With Method**
Mean [ms]	0.3679	0.3705
Standard Deviation	0.02538	0.02581

Table 1: Results of UDP RTT Measurements

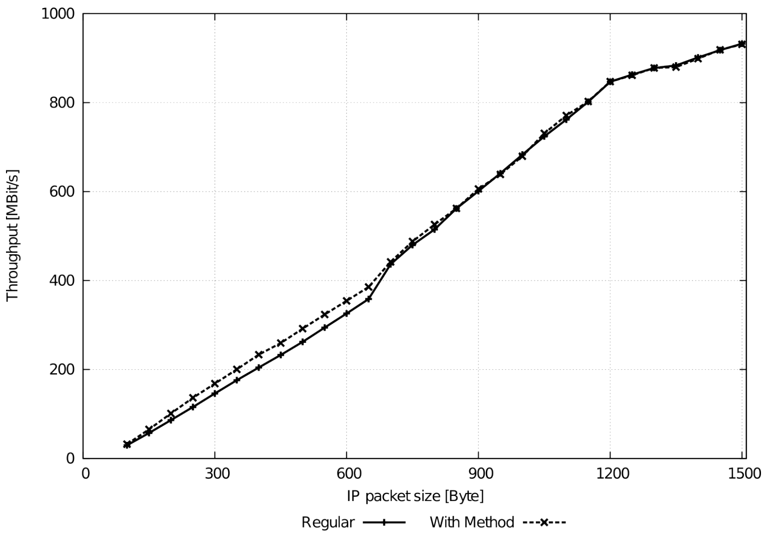

Figure 3: UDP Throughput Comparison

Second, we study the reliability of capturing and identifying "Wake-Up" packets. If individual packets are dropped or not analyzed, e.g., due to high load, the device might miss "Wake-Up" packets rendering the IP address assignment mechanism ineffective. In the worst case this could result in situations in which the device cannot be accessed at all. The reliability with which "Wake-Up" packets are captured and processed, depends on two different circumstances: first, the packets must actually reach the bridge and may not be discarded beforehand, for example due to high load in the attached network segments. Second, the packets must be successfully captured and processed independently from the device's network load; i.e., even when the maximum throughput is transferred across the device the packets must be received and processed correctly for the authentication mechanism to work reliably.

The first effect solely depends on the details of the network segments attached to our bridge and is independent from our IP-less operation and IP address assignment mechanism as discussed in this paper. Hence, we only assured that this aspect did not affect our results when examining the reliability of the receive and processing mechanism under full link load.

We tested the reliability with which "Wake-Up" packets are received by flooding the bridge with UDP traffic at full link speed. Within this traffic we sporadicly transmitted single "Wake-Up" packets. We performed this test repeatedly with and without flooding the bridge with UDP traffic. In both cases the detection rate of "Wake-Up" packets was 100%.

Finally, the impact of flooding the device with valid or invalid "Wake-Up" packets needs to be evaluated. Such a situation could occur either in the case of brute force attacks when an attacker repeatedly sends "Wake-Up" packets in order to guess the correct combination or when an attacker attempts to perform a denial-of-service attack against the device in order to impair the network functionality or the ability for others like the legit network administrators to authenticate with the device.

Technically there is no difference if an attacker is performing a denial of service attack or tries to break the authentication mechanism using brute force. In both cases an attacker will flood the device with "Wake-Up" packets which may affect the device negatively, e.g., by exhausting its resources. Such resources are: network bandwidth, CPU usage, or memory usage. In this context UDP is problematic as it can be used generally for denial-of-service attacks on networks by simply flooding datagrams. This is a known issue and solutions as well as their limitations are well researched (Cabrera, et al. 2001) (Mirkovic & Reiher 2004). Since this is a general issue that is not specifically connected to our proposed IP-less operation we do not assess this here but rather analyse the effects of flooding the prototype with invalid "Wake-Up" packets on the CPU and memory performance.We assessed the impact of flooding authentication tokens by comparing the effects of putting our test device under heavy UDP load and by flooding it with an authentication message as could be done in an attempted denial of service attack. The test setup was identical to the one used for the throughput and round trip time performance benchmarks above. Flooding UDP traffic and sending authentication messages was done from the same computer. During flooding with UDP traffic we could not observe any significant CPU and memory utilization. While flooding with "Wake-Up" packets, however, we noticed a change in the CPU usage: as shown in Figure 4 nearly 30% of the CPU time was used for executing kernel level code and another roughly 30% of the CPU time was used for executing user level code. However, we see very high potential for increasing the performance of our prototype. Primarily the explicitness of kernel-level filters could be improved. There are also more approaches to improve packet-filtering and packet-processing in general (Fusco et al. 2010)(Leogrande et al. 2011), which could be applied to further increase the performance.

Furthermore, the prototypical software implementation was primarily designed as a proof of concept and the critical code paths have not yet been optimized. Finally, there might be a chance to increase the performance even further by implementing the critical code paths in lower level languages like C or assembly.

5. Conclusion

In this paper, we evaluated the performance and practicability of a mechanism for IP-less operation and on demand IP address assignment for managed data link layer devices. Analogous to port knocking on firewalls, the IP-less operation and on demand IP address assignment allows a safer operation when IP connectivity is not needed while retaining the possibility to connect to the device over IP at any time.

We implemented a prototype on an embedded x86-based Linux system and assessed its performance during regular operation, the reliability of capturing and processing packets for triggering IP address assignment, and the impact of flooding attacks like brute force or denial of service attacks.

Our results show that the network performance in regular operation is not significantly affected. Moreover, the reliability of capturing and analyzing "Wake-

Up" packets was 100% even under high network load and flooding with other traffic. There was a noticeable increase in CPU-load during a DoS scenario, but bandwidth limitations will be reached long before serious CPU or memory issues arise. During regular operation, even when facing high traffic, there was no overly memory or CPU usage. Further optimizations of our implementation are possible and will be conducted in the future. In summary, our results show that our method for IP-less operation and on demand IP address assignment can be practically used.

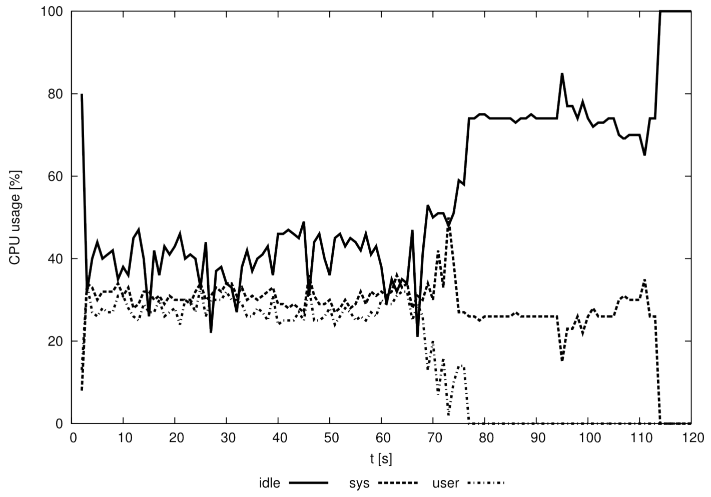

Figure 4: CPU Utilization while Flooding "Wake-Up" Packets

In the future, we are planning to research improved authentication mechanisms and protocols as well as means for further increasing the performance. Options for more sophisticated protocols include steganography techniques for hiding authentication data in other protocols like DHCP making it harder for an attacker to detect the presence of a concealed device, even during authentication. Furthermore, we will research options for applying our method to virtualized environments. While the network characteristics of virtualized environments may differ from physical networks (Gad et al. 2011a) (Gad et al. 2011b) we expect, based on the experience from our prior work on virtualization, that our results gained in this paper can be applied to virtualized network equipment as well. Moreover, we will also look into the possibility of having a separate, randomly chosen "hidden network" making it harder to find the concealed device even after it acquired an IP address. Finally, this technology could also be deployed by an attacker to secretly tap into a victims network. Hence, we are also going to research methods for detecting the presence of such concealed devices.

6. References

Al-Bahadili, H. & Hadi, A. H. (2010). Network security using hybird port knocking, International Journal of Computer Science and Network Security 10(8).

Cabrera, J. B. D., Lewis, L., Qin, X., Lee, W., Prasanth, R. K., Ravichandran,B. & Mehra, R. K. (2001). Proactive detection of distributed denial of service attacks using mib traffic variables-a feasibility study, Proceedings of the International Symposium on Integrated Network Management, pp. 609–622.

deGraaf, R., Aycock, J. & M. Jacobson, J. (2005). Improved port knocking with strong authentication, Proceedings of the 21st Annual Computer Security Applications Conference.

Eggendorfer, T. & Weber, D. (2007). Running a port forwarding firewall system on a bridge, Proceedings of the Fourth IASTED International Conference on Communication, Network and Information Security, pp. 122–126.

Fusco, F., Deri, L. & Gasparakis, J. (2010). Towards monitoring programmability in future internet: challenges and solutions, Proceedings of the 21st Tyrrhenian Workshop on Digital Communications: Trustworthy Internet (ITWDC).

Gad, R., Kappes, M., Mueller-Bady, R. & Ritter, I. (2011a). Is your virtualized network really what you think it is?, Fourth International Conference on Internet Technologies & Applications, Wrexham, UK, pp. 128–135.

Gad, R., Kappes, M., Mueller-Bady, R. & Ritter, I. (2011b). Network performance in virtualized environments, 17th IEEE International Conference on Networks, Singapore, pp. 275–280.

Haller, N. (1995). The s/key one-time password system, IETF RFC 1760.

IEEE (2004). Standard for Local and Metropolitan Area Networks – Media access control (MAC) Bridges – IEEE Standard 802.1D

Kappes, M. & Happel, F. (2009). Tackling network security in small to medium businesses, Proceedings of the 3rd International Conference on Internet Technologies and Applications, Wrexham, UK.

Krzywinski, M. (2003). Port knocking from the inside out, SysAdmin Magazine.

Leogrande, M., Ciminiera, L. & Risso, F. (2011). Modeling filtering predicates composition with finite state automata, Proceedings of the 19th International Conference on Software, Telecommunications and Computer Networks.

Liew, J., Lee, S., Ong, I., Lee, H. & Lim, H. (2010). One-time knocking framework using spa and ipsec, International Conference on Education Technology and Computer.

Mirkovic, J. & Reiher, P. (2004). A taxonomy of ddos attack and ddos defense mechnisms, ACM SIGCOMM Computer Communication Review 34(2): 39–53.

Rash, M. (2006). Single packet authorization with fwknop, ;login: USENIX Magazine 31(1).

Sly Technologies, Inc. (2012). jNetPcap, last accessed 02/10/2012. URL: http://jnetpcap.com

Tcpdump/Libpcap (2012). libpcap, last accessed 02/10/2012. URL: http://www.tcpdump.org

Worth, D. (2004). Ck: Cryptographic one-time knocking, Talk slides, Black Hat USA.