Detecting Rogue Switches through Active Verification

Kevin Kredit *GVSU CIS Department*
Grand Rapids, Michigan
Email: k.kredit.us@ieee.org

***Abstract*—Though switches bear great responsibility in LAN security, detecting rogue switches is difficult. This paper reviews the threat landscape around switches, overviews rogue switch prevention measures, and proposes a new double-pronged rogue switch prevention and detection mechanism. The first prong is TPM-enabled remote attestation of switches and setting configurations. The second prong is behavioral verification of switches using generated traffic. The tool leverages defined switch behavior and the presence of switch security features to detect misconfigured and potentially malicious switches. Together, the forms of verification support strong assurance as well as backwards hardware compatibility. Finally, a set of policies designed to create a trusted network backbone is described.**

***Index Terms*—network security, security management, local area networks**

I. Introduction

As trusted elements of basic networking infrastructure, Ethernet switches hold great responsibility. A compromised or maliciously programmed switch poses fearsome threats to network security. Due to its position between nodes on a LAN, a switch can greatly impact confidentiality, integrity, and availability of network services. Switches face threats to their inherent functionality as well as their features designed to secure higher level protocols.

As the threat to a LAN from a compromised switch is so significant, a number of prevention-oriented mitigations already exist. However, the impact of a compromised switch is great enough to warrant further research, particularly in how to detect rogue switches.

A. Detecting a Rogue Switch

How does one detect a disobedient network switch? One method is passive detection, usually by observing network traffic and parsing it for suspicious behavior. This approach lies solidly within the realm of intrusion detection systems (IDSs). Machine learning promises a new level of effectiveness for IDSs, though they are historically only partially effective and by their nature offer only probabilistic insight.

A second method is proactive verification. This approach could take two forms: self-verification and behavioral verification. Self-verification means using switch attestation to validate that it is running trusted firmware as well as booting into an approved configuration. Behavioral verification means running the switch through a suite of tests and verifying correct behavior.

B. This Paper's Contribution

This paper begins by summarizing the current state of switch security. It next describes self-verification, introduces a tool designed to enable behavioral verification, and builds upon the two to introduce a network management model that uses both techniques to establish a verified network backbone.

II. The Switch Security Threat Landscape

One could classify switch security threats many ways. In this paper, switch security is classified according to threats inherent in its network position and threats relating to higher level protocols.

For better or for worse, the switch's role in network security is not limited to switching packets correctly. As arbiter of LAN access, switches have been tasked with enforcing policies on higher level protocols that assume particular network behavior. This type of threat is analyzed separately.

A. Inherent Security Threats

The switch's core job is route packets on a LAN from one level 2 host to another. To perform its job correctly, it must route packets to the correct destination without modification or delay. This job can be subverted in three ways, corresponding to the security triad of confidentiality, integrity, and availability:

1) Traffic redirection (confidentiality)
2) Traffic modification (integrity)
3) Denial of service (availability)

Traffic redirection is to route packets to the wrong port, an additional port, or even internal storage. Redirection can be used for man-in-the-middle (MITM) or exfiltration attacks. Traffic modification is to rewrite data as it passes through the switch. This is an MITM attack. Denial of service is failing to deliver packets to their destination. This could be done selectively or completely. All three could be designed to evade IDSs. Redirection could be to internal storage, which would leave no network trace; modification could be selective, and require comparison of all inbound and outbound traffic to detect; and denial of service could be selective or strategically timed such when it is detected, it is too late.

While switch integrity is clearly violated in an arbitrarily programmed malicious switch, non-malicious switches can also be compromised by malicious hosts. Two ways this is done are Content-Addressable Memory (CAM) table overflow and MAC address spoofing.

1) CAM Overflow: CAM overflow occurs when a switch's limited MAC address to port mapping memory is exhausted, at which point the switch falls back to behaving like a hub; this is achieved by spoofing MAC addresses on an interface until the CAM table is full [1]. CAM overflow is a confidentiality (misrouted packets) and denial of service (poor network performance) attack.

2) MAC Address Spoofing: While MAC address spoofing is an element of the CAM overflow attack, a MAC address spoof attack is oriented not at overwhelming the switch, but at impersonating another host's valid MAC address so that the switch updates its CAM table entry and subsequently forwards future traffic intended for the valid host to the malicious host [1]. This is a confidentiality attack that enables MITM.

B. Higher Level Protocols

Many core networking protocols are designed assuming trust within the LAN. DHCP assumes only one server (or set of distributed but coordinated servers) exists per LAN. ARP assumes hosts will not lie about their identity. Due to misconfiguration and malicious behavior, such assumptions are frequently wrong. Because of the switch's role as gatekeeper between a host and the LAN, it is a convenient place to implement defenses against attacks on these higher level protocols.

This paper focuses not on how to configure the switch security features that defend DHCP and ARP, but on rogue switch detection and security implications. The threat considered here is not "rogue DHCP" or "ARP spoofing," but that of a compromised switch which uses misapplications of these security features to enable further attacks. For example, a compromised switch could block the valid DHCP server and enable a rogue DHCP server.

In order to understand the impacts of a rogue switch, therefore, one must understand these high level protocol security features. In addition to their importance to rogue switch impact, these features provide a possible means for rogue switch detection.

Two protocols that switches protect are DHCP and ARP. Both have clear impacts on confidentiality and availability. DHCP attacks impact integrity if the rogue DHCP server sends wrong router and default gateway addresses. ARP attacks impact integrity if they are used to perform MITM attacks.

1) DHCP Snooping: DHCP is susceptible to spoofing and starvation; DHCP snooping is a switch security feature that enables DHCP offer filtering and generation of a DHCP binding table used to validate incoming packets and limit the number of IP addresses on a port [2].

2) Dynamic ARP Inspection: ARP is used by hosts to tie IP addresses to MAC addresses. ARP is susceptible to spoofing and "poisoning"; dynamic ARP inspection (DAI) is a switch security feature that uses the DHCP snooping binding table to validate ARP messages before forwarding them [2].

C. Threat Summary

The switch security threats described in the previous sections are summarized here, organized according to the CIA triad:

1) Confidentiality
    a) misrouted packets
    b) internally stored packets
    c) fall back to hub (all packets broadcast)
    d) compromised DHCP defense (rogue DHCP)
    e) compromised ARP defense (MITM)
2) Integrity
    a) traffic modification
    b) incorrect MAC to port mapping, enabling MITM
    c) compromised DHCP defense (rogue DHCP)
    d) compromised ARP defense (MITM)
3) Availability
    a) denial of service
    b) fall back to hub (decreased performance)
    c) compromised DHCP defense (exhausted DHCP)
    d) compromised ARP defense (poisoned tables)

D. Threat Vectors

Given the high impacts of rogue switches, the pressing question is "how could a rogue switch be introduced to the network?" The possibilities are as follows:

1) A valid switch could be compromised via
    a) a host sending malicious traffic (e.g., MAC spoofing)
    b) stolen (or guessed) management credentials
    c) software vulnerability
    d) physical modification
2) A malicious switch could be inserted to the network at
    a) a critical network location
    b) a public facing port

III. Current Mitigations

A. Basics

No new security invention replaces the need for strong fundamentals. A few necessary first steps are presented before this paper's contribution is considered.

1) Password management: Changing switches' administrative credentials from the default is the first step in securing the local area network. Changing the credentials is more important than enforcing physical security because failing to do so enables effortless, automated remote compromise. Even if the organization does not intend to implement switches' available security features, changing the default credentials is necessary to prevent an attacker from configuring the switch to his interests. This mitigates threat vector 1(b), stolen management credentials.

2) Physical security: Preventing physical hardware access prevents bad actors from simply installing their hardware on your network. Particularly for backbone portions of the network, physical access must be controlled. This mitigates threat vectors 1(d), physical modification, and 2(a), switch insertion at a critical network location.

3) Patching and other fundamentals: Implementing a patching program, appropriate network segmentation, and intrusion prevention systems helps keep bad actors out of the network in the first place. To the extent switches and routers get software patches, they should be applied in a timely manner. This mitigates threat vectors 1(a), a host sending malicious traffic, and 1(c), a software vulnerability in the networking equipment.

4) Implementing available security features: After the previous steps have been taken, implementing the switches' available security features is the next logical step, and indeed is a prerequisite the more capable variations of the active verification approach. MAC limiting, DHCP snooping, DAI, and port security mitigate threat vectors 1(a), a host sending malicious traffic, and 2(b), switch insertion at public facing ports [1].

B. Encryption and Higher Level Protocols

When mitigating security risk, besides decreasing likelihood, decreasing the cost of successful attacks is another valid approach. Using encryption in higher level protocols such as IPsec, TLS, SSH, and SFTP, or even using level 2 encryption [3] nearly eliminates a rogue switch's capability to violate confidentiality or integrity.

C. IEEE 802.11i Wi-Fi Security

At first blush one may suggest implementing Wi-Fi 802.11i security on the wired Ethernet LAN. After all, there exists more research on the subject [4] [5]. However, Wi-Fi security measures are not well suited to wired LANs. The main reason is practical: the overhead is significant and the threat likelihood is much less on a LAN. Wireless networks can be passively observed by anyone within range, whereas Ethernet LAN attacks require either a remote exploit or physical access.

Additionally, wireless and wired networks are inherently structured differently: wireless networks are broadcast traffic routed through the access point, whereas wired networks support point to point communication and have switches that arbitrate access.

IV. Switch Self-authentication

The preferred way to detect and prevent rogue switches from joining a network is to place the verification burden on the switch.

A. Self-authentication with a TPM

Using a Trusted Platform Module would enable the network to verify that switches are operating trusted software stacks on trusted hardware. Such attestation methods for verifying client integrity are well understood [6]. In particular, when a TPM-enabled device boots, the TPM generates a certificate signed with its attestation identity key and a certificate from a trusted certification authority that was installed at time of manufacture; this certificate is sent to challengers during attestation; challengers verify the signatures and assign trust to the client [7].

The basic authentication flow would be as follows:

1) A unknown switch joins the network. It is connected to a managed switch that is already part of the verified network.
2) The existing switch challenges the new switch's identity.
3) The unknown switch either:
    a) Responds with a trusted attestation key. It joins the network as a verified element.
    b) Responds with an untrusted attestation key. It is treated according to the network's security policy.
    c) Is unequipped for the identity challenge. It is treated according to the network's security policy.

This mitigates threat vectors 2(a) and 2(b), switch insertion. For the case of verified switches becoming compromised after initial attestation, periodic re-attestation of their configuration is required. If a switch is found to have been compromised, it is treated according to the network's security policy. This mitigates the remaining threat vectors.

The "network security policies" relating to how to treat untrusted switches are discussed in section VI-B.

B. Self-authentication Without a TPM

Self-authentication can be achieved without a TPM if switches are pre-configured with a shared secret. While this allows self-authentication with current-generation hardware, it is vulnerable to spoofing if the secret is compromised and is less automated as it requires manual configuration.

V. The Behavioral Verification Tool

Self-authentication is the preferred means of switch validation. However, switches typically are not outfitted with TPMs. Even if that were to begin, organizations would rightly not be able to justify the cost of updating the millions of deployed switches worldwide for the sake of this feature. In order to build a verified network backbone, there needs to be a way to validate currently deployed switches.

This paper proposes a behavioral verification tool that establishes trust in switches by putting them through thorough tests. Such test-based trust is not as strong as attestation. All cases could not be tested; for example, a malicious switch might only filter packets from certain source MAC addresses. The tests could be evaded; for example, the device may detect that it is under test, and perform as expected until the test is over. These possibilities are of little concern for two reasons. First, both such cases require not just a maliciously *configured* switch, but a maliciously *programmed* switch. This is a less likely threat, and can be partially addressed through the switch security policies. Second, as with attestation, tests would be repeated periodically, making test detection non-trivial.

The behavior verification tool works by functionally testing corner cases as well as random cases of each switch configuration policy. If a switch is correctly configured and acts as expected, it passes the tests. If a switch is mis-configured or acts in any unexpected way, it fails. Passing switches join the network as a verified element. Failing switches are treated according to the network's security policy.

The proposed tool has three modes that offer increasing thoroughness at the cost of increasing configuration and labor.

A. Deployed Switch Server-only Mode

This mode of the tool only requires only a server. It is called "deployed switch" because the steps can be performed on switches that are already deployed and new switches require no special steps prior to deployment. The server has knowledge of the network topology and generates traffic in such a manner that tests the policies of the new switch. Server-only mode is limited in what traffic it can send. For example, if it sends traffic that violates security policies, it will be rejected on the way to the target switch. Even so, server-only mode still allows for automatic switch configuration: when a new switch joins the network, the server detects the make and model, logs in, and sends the appropriate configuration commands.

1) Advantages:

- Low configuration overhead
- Can automatically configure new switches

2) Disadvantages:

- Very limited live-traffic behavioral verification

B. Deployed Switch Client-server Mode

This mode of the tool adds the concept of clients that assist the server in generating and detecting policy-testing traffic. Suitable hosts for the client application include endpoints and managed switches themselves. The server coordinates the clients to generate as thorough a set of test traffic as possible. Because of the additional nodes available to the tool, possibly including devices attached directly to the switch, the properties it can test in this mode are more thorough. The server still logs in to perform administrative functions on managed switches.

1) Advantages:

- Can automatically configure new switches
- More capable live-traffic behavioral verification

2) Disadvantages:

- High configuration overhead

C. Test-bench Mode

This mode of the tool requires only a server, but requires that the switch be taken out of deployment or tested before deployment. By sequentially testing each port of the switch with arbitrary traffic, the server is able to fully verify all the behavior expected of the switch. It begins by logging in to perform administrative functions on managed switches, then runs through the test suite. Once the switch has been tested, its MAC address and test result are saved in the server's database. When it connects to the network in deployment, the server recognizes it and assigns trust to it.

1) Advantages:

- Can automatically configure new switches
- Fully capable live-traffic behavioral verification

2) Disadvantages:

- Requires manual labor
- Cannot be performed on deployed switches

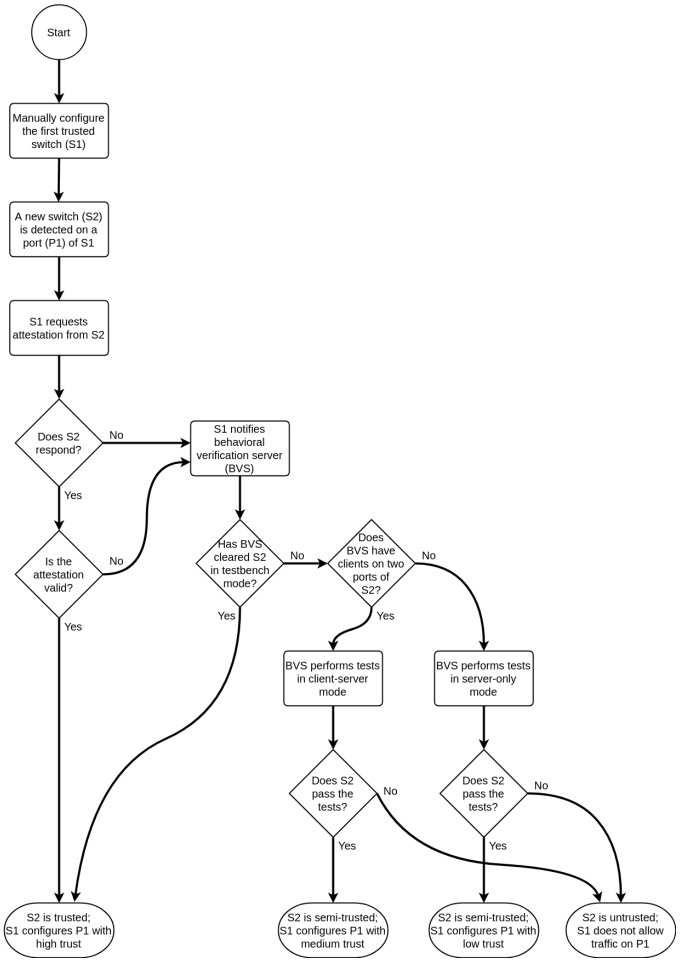
Start
Manually configure the first trusted switch (S1)
A new switch (S2) is detected on a port (P1) of S1
S1 requests attestation from S2
Does S2 respond?
No
S1 notifies behavioral verification server (BVS)
Yes
Is the attestation valid?
No
Has BVS cleared S2 in testbench mode?
No
Does BVS have clients on two ports of S2?
No
Yes
Yes
Yes
BVS performs tests in client-server mode
BVS performs tests in server-only mode
Does S2 pass the tests?
No
Yes
Does S2 pass the tests?
No
Yes
S2 is trusted; S1 configures P1 with high trust
S2 is semi-trusted; S1 configures P1 with medium trust
S2 is semi-trusted; S1 configures P1 with low trust
S2 is untrusted; S1 does not allow traffic on P1

Fig. 1. The process of dynamically growing the trusted network backbone.

D. Generating Test Traffic

Test traffic is generated from the server and clients in order to test the target switch's behavior. In order to test DHCP Snooping, for example, a client will send generated DHCP offers out of port 67. To test DAI, a client will send generated ARP messages. On a Unix-like system, this can be crudely achieved as simply as `cat message.file | nc -nu -p source_port dest_addr dest_port`. More direct approaches can be used. Depending on the host operating system, the client may need root access to send traffic on ports below 1024.

VI. Building a Verified Network Backbone

A. Switch Addition

By establishing trust in switches using self-authentication and behavioral verification, network administrators can establish a core or "backbone" of trusted switches. This backbone grows dynamically in accordance with the flow chart shown in Figure 1.

B. Security Policies

Connections between trusted switches can be configured permissively, while connections between trusted and untrusted switches can be configured with variable amounts of strictness depending on the level of trust assigned to the switch. Example policies include the following:

- Strict: until a new switch is successfully verified (attested or tested),
  - deny all outbound non-attestation or non-test related traffic on the port
  - deny all inbound non-attestation, non-test, and non-configuration related traffic on the port
- Limited: until a new switch has successfully verified (attested or tested), or if a switch is unable to verify,
  - treat the port as untrusted
  - set a limit on the number of hosts allowed to be tied to the port
- Permissive: regardless of verification, treat the port like normal

Policies can vary depending on attestation or test method. For example, remotely-attesting switches could be considered fully trusted, thoroughly tested switches could be mostly trusted, partially tested switches could be less trusted, and untested switches could be untrusted. The level of trust in a switch translates into port security features applied to that switch's connection to the rest of the network, an example being the number of hosts allowed to connect through it.

VII. Conclusion

Rogue and compromised switches represent grave threats to LAN security. Though relatively low in likelihood, preventing and detecting rogue switches is a valuable step to take. After taking the necessary basic steps in securing the LAN and using encrypted communications when possible, active switch verification allows the creation of verified LAN backbones. TPM-based remote attestation enables strong verification, while behavior-based verification enables backwards compatibility with currently deployed equipment. Both measures mitigate the threat of rogue switches through active prevention and detection.

Acknowledgment

Kevin Kredit thanks Professor Vijay Bhuse and his CIS 616 "Data Security and Privacy" Winter 2019 classmates for their constructive input over the course of the semester.

References

[1] "Layer 2 Security Features on Cisco Catalyst Layer 3 Fixed Configuration Switches Configuration Example", Cisco, 2007. [Online]. Available: https://www.cisco.com/c/en/us/support/docs/switches/catalyst-3750-series-switches/72846-layer2-secftrs-catl3fixed.html. [Accessed: 05-Apr- 2019].

[2] Y. Bhaiji, "Security Features on Switches > Securing Layer 2", Ciscopress.com, 2008. [Online]. Available: http://www.ciscopress.com/articles/article.asp?p=1181682. [Accessed: 05- Apr- 2019].

[3] "Senetas CN Series Layer 2 Encryption solutions", Senetas. [Online]. Available: https://www.senetas.com/products/cn-encryptors/. [Accessed: 05- Apr- 2019].

[4] R. Beyah, S. Kangude, G. Yu, B. Strickland and J. Copeland, "Rogue access point detection using temporal traffic characteristics," *IEEE Global Telecommunications Conference, 2004. GLOBECOM '04.*, Dallas, TX, 2004, pp. 2271-2275 Vol.4. doi: 10.1109/GLOCOM.2004.1378413

[5] L. Watkins, R. Beyah and C. Corbett, "A passive approach to rogue access point detection," *IEEE GLOBECOM 2007 - IEEE Global Telecommunications Conference*, Washington, DC, 2007, pp. 355-360. doi: 10.1109/GLOCOM.2007.73

[6] R. Sailer, T. Jaeger, X. Zhang, and L. van Doorn, "Attestation-based policy enforcement for remote access," *The 11th ACM conference on Computer and communications security (CCS '04)*, ACM, New York, NY, USA, 2004, pp. 308-317. doi: 10.1145/1030083.1030125

[7] T. Garfinkel, M. Rosenblum and D. Boneh, "Flexible OS Support and Applications for Trusted Computing", in *HotOS*, 2003, pp. 145-150.