SSID STRIPPING

The SSID Stripping Vulnerability: When You Don’t See What You Get

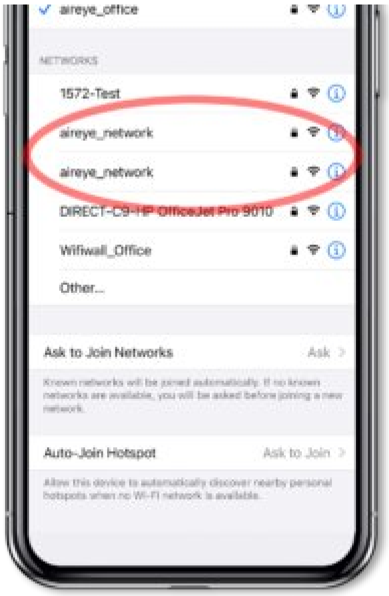
POSTED ON SEPTEMBER 13, 2021 BY AMICHAI SHULMAN

Introduction

AirEye’s research team in collaboration with the Computer Science faculty at the Technion – Israel Institute of Technology have found a vulnerability, dubbed SSID Stripping, which causes a network name – aka SSID – to appear differently in the device’s “List of Networks” than its actual network name.

The significance? Unsuspecting users may connect to an attacker-controlled network they did not intend to connect to.

The SSID Stripping vulnerability affects all major software platforms – Microsoft Windows, Apple iOS and macOS, Android and Ubuntu.



✓ aireye_office

NETWORKS

1572-Test

aireye_network

aireye_network

DIRECT-C9-HP OfficeJet Pro 9010

Wifiwall_Office

Other...

Ask to Join Networks Ask >

Known networks will be joined automatically. If no known networks are available, you will be asked before joining a new network.

Auto-Join Hotspot Ask to Join >

Allow this device to automatically discover nearby personal hotspots when no Wi-Fi network is available.

Figure 1: The first “aireye_network” is real while the second has the “%p%s%s%s%s%n” as a suffix that is not visible

Attack Scenarios

Attackers can leverage SSID Stripping to conduct the following attacks:

1. **Create a more effective rogue Access Point (AP), easily deceiving the user into connecting to a rogue network.** Since the attacker creates a rogue AP with a name that looks exactly like the known legitimate network name, users are more likely to fall prey to this attack.
   Operating system vendors have put in place controls to prevent users from connecting to rogue APs displaying the same network name as legitimate networks. These controls mainly rely on the fact that the device is configured to use the same security measures, such as a certificate, every time it connects to a network name it already has in its memory. Thus, a device cannot connect to a rogue AP with the same network name since the rogue AP does not require the same security measures.
   SSID Stripping bypasses these security controls since the device itself processes the network names as they actually are, not as they are displayed. Hence, the devices do not consider the rogue AP to have the same name as the legitimate network.

2. **Incorporate an attack within a network name without raising a user’s, or the IT team’s, suspicions.** One example of such an attack is the

recently disclosed format string vulnerability in the processing of SSID by iPhones. Usually such attacks would involve network names that are shady looking (e.g. “%p%s%s%s%s%n”). Hence, the chances of a user to actually attempt a connection to such a network are slim.
With SSID Stripping, it is possible to create a network name in a way that its display only shows a prefix that is similar to a legitimate network name (e.g. the corporate network name) while the actual network name includes the additional specially crafted information.

Are You Vulnerable?

To check if your organization is susceptible to the SSID Stripping vulnerability, AirEye created a free simple Windows-based tool called **Hide ‘n Seek**.

The tool publishes numerous network names using SSID Stripping techniques, based on the original SSID that the user provides. Users can then find out how these network names are displayed on the various devices in their organization to get a sense of how vulnerable their environment really is.

Download the free tool here.

Mitigation

We urge enterprises to constantly monitor their network airspace for network names that use SSID Stripping techniques and prevent any connection by corporate devices to such networks.

AirEye prevents corporate devices from connecting to non-corporate networks, including rogue APs. Additionally, AirEye is the only solution that detects an attack within an SSID and alerts which corporate devices have been affected.

A Deep-Dive Into SSID Stripping

What’s in a Name?

In the Wi-Fi world, networks are identified by an element that we humans refer to as “network name” and is formally called ESSID or

SSID. We tend to think of the SSID as “network name” because this is the identifier that we see on our display when we choose to connect to a specific network. That network could be provided by multiple different devices, aka Access Points (APs), even across different geographies. As long as all the APs publish (and respond to) the same name – and share the same security parameters – our wireless enabled devices will connect to any of them upon request (or even automatically).

Some devices may deny connection to an AP if it publishes a name that was already used for connecting the device, with different security attributes – either type of security or credentials. This behavior protects the user from connecting to rogue APs. Additionally, the SSID published by any AP in the proximity of a wireless client is processed by that client – regardless of whether there is any trust between the client device and the AP. Hence an attacker may attempt to include malicious payload within the SSID in an attempt to exploit a vulnerable client implementation. An example of such vulnerability is CVE-2021-30800, affecting Apple devices. Since a network name is displayed to a user before attempting connection, it may be difficult for an attacker to entice a user into connecting to an unrecognized network with a particularly weird name (e.g. one that includes “%p%s%s%n”).

Research and Findings

With the importance of SSID in mind, we set out to test whether it is possible to create network names that are displayed incorrectly by devices, therefore tricking a user into connecting to a rogue network. To do so, we joined forces with students from the Computer Science faculty of the Technion – Israeli Institute of Technology.

Our first, and most important, observation was that while we think of the SSID as a string of characters, it is interpreted and handled by the communication layer as a stream of binary data. Thus, different devices make different choices when translating this stream of data into human readable characters for display.

We introduced different special characters — in particular “non-printable” characters — into SSIDs. We published these SSIDs, observed the network name as it was displayed by different types of devices, and recorded the network traffic when actually attempting a connection to the specific network. We tested our specially crafted network names with the following types of devices: iPhone, Apple Mac, Microsoft Windows computer, Android phone and Ubuntu Linux computer. We were able to generate three types of display errors:

- Type 1: A display of only a prefix of the real network names
- Type 2: Omissions of some characters from display
- Type 3: Some characters are pushed outside of the visible portion of the display

Errors of type 1 and 3 can assist in exploiting vulnerabilities related to the processing of the network name itself. We observed that when Type 2 errors exist, the inserted special characters also affect the sort order of network names in the display. Hence it is possible to make sure that the specially crafted network name is inserted in a specific location in the list of available networks. Any of the error types makes it easy for an attacker to trick users into connecting to rogue networks.

Type 1 errors – display a prefix

We found out that when a NULL byte is inserted as part of a network name, Apple devices (including iPhone and Mac) only display the part of the name before the NULL byte, while the communication layer handles the entire SSID. For example, a network name of the form "aireye_networkrogue" (where " denotes the NULL byte) is displayed to the user as "aireye_network".

On Microsoft Windows machines a similar behavior is observed when we used multiple New Line characters. For example, a network name of the form "aireye_networknnrogue" is displayed to the user as "aireye_network" with a single blank line before the next network name (combined with Type 2 errors that blank line would go unnoticed as we place our network last).

Type 2 errors – character omission

This is by far the most common type of error. We found out that many special characters are simply omitted from the actual display (especially those considered “non-printable” characters).

For example, the NULL byte when introduced into a network name is not part of the display on Android phones. A network name of the form “aireye_network” would be displayed exactly the same as “aireye_network”.

The same holds true for Ubuntu machines when handling a NULL byte.

Other “non-printable” characters have similar effect on iPhone and Mac devices. For example, the network name “aireye_x1cnetwork” (with x1c representing a byte with the value 0x1C hex), is displayed exactly the same as “aireye_network”.

Type 3 errors – display overflow

These errors rely on enough characters being pushed out of the visible portion of the display. Luckily for attackers, mobile displays are not big. The trick is to introduce enough white space characters into an SSID that would push the suffix out of the display. Since the SSID is limited to 32 bytes (not characters) we need to use “efficient” characters such as New Line, Tab or Form Feed (a reminiscent of stone age printing technology).

For example, an SSID of the form “aireye_networknnnnnnnnnnnnrogue” (where ‘n’ denotes the New Line character) may be displayed by an iPhone as “aireye_network” since the word “rogue” is pushed out of the display. Together with type 2 errors this can be used to efficiently hide the suffix of a rogue network name.

Disclosure Process

We informed Apple, Microsoft, Android project and Ubuntu project of our results on July 11th, 2021.

Microsoft, Ubuntu and Android acknowledged the issue but regarded it as having minor security implications and hence did not commit to fixing it.

Apple acknowledged the issue but failed to provide any update on their fixing plans to this date.

Conclusions

Our findings point out an actual security issue with Wi-Fi communications implementation. The use of discrepancies between the actual data and the displayed data has been part of a hacker's toolbox in other domains as well, such as phishing, and is still a very effective technique.

We also think that our findings point to a bigger issue – that of the state of Wi-Fi deployment security. It seems that for years, researchers (and developers) focused their attention on the implementation of secure communications within the Wi-Fi standard and paid less attention to the inherently unprotected stages of the communication. The fact that we were able to find a security glitch in a mechanism that is so basic to establishing Wi-Fi communication (display of a network name) is just the tip of the iceberg with respect to other parts of these complex and extensive protocols.

Enterprises must realize that there is more to Wi-Fi security than setting the correct authentication method. Wireless capable devices are exposed to many threats that are related to the open nature of the medium – everyone can send frames into the air and every device with wireless capabilities is constantly processing such frames. Attackers can exploit the Wi-Fi medium in order to bypass existing network security controls and gain access to enterprise networks through vulnerable wireless devices. It is time for corporations to consider solutions for monitoring, controlling and protecting the network airspace around them.

The Hide 'n Seek tool assesses how vulnerable your company is to SSID Stripping. Download Now.

Detailed Findings

Platform specific details

iOS

Versions tested: 14.5.1, 14.6

Char	Type	Example	Comments
Null	1/2	MaorLiel%p%s%s%s%s%n is displayed as MaorLiel aireye_office is displayed as aireye_office	We were not able to establish a clear pattern of when it becomes a Type 1 or Type 2 error
Tab	3	aireye_officet%p%s%s%s%s%n is displayed as aireye_office<tab size space>%p%s%s%s%s%n	The display wraps line. By using enough tab characters, the strange suffix is pushed down and outside of the visible display
New Line	3	aireye_officen%p%s%s%s%s%n is displayed as aireye_office %p%s%s%s%s%n	By using enough new line characters, the strange suffix is pushed down and outside of the visible display
Form Feed	3	aireye_officef%p%s%s%s%s%n is displayed as aireye_office %p%s%s%s%s%n	By using enough new line characters, the strange suffix is pushed down and outside of the visible display
Non-printable	2	aireyex1c_office is displayed as aireye_office	We tested this with multiple non-printable characters

macOS

Version tested: 10.15.7

Char	Type	Example	Comments
Null	1/2	MaorLiel%p%s%s%s%s%n is displayed as MaorLiel aireye_office is displayed as aireye_office	We were not able to establish a clear pattern of when it becomes a Type 1 or Type 2 error
Tab	3	aireye_officetttttt%p%s%s%s%s%n is displayed as aireye_office...	This is different than iOS behavior
New Line	3	aireye_officen%p%s%s%s%s%n is displayed as aireye_office	By using enough new line characters, the strange suffix is pushed down and

		%p%s%s%s%s%n	outside of the visible display
Form Feed	3	aireye_officef%p%s%s%s%s%n is displayed as aireye_office %p%s%s%s%s%n	By using enough new line characters, the strange suffix is pushed down and outside of the visible display
Non-printable	2	aireyex1c_office is displayed as aireye_office	We tested this with multiple non-printable characters

Microsoft Windows

Version tested: Windows 10 Pro Build 19042.1083

Char	Type	Example	Comments
Tab	1	aireye_officetttttt%p%s%s%s%s%n is displayed as aireye_office	Tabs create a large white space which is wrapped. Once wrapping to the third line the rest of the name is not displayed
New Line	1	aireye_officennn%p%s%s%s%s%n is displayed as aireye_office	Only two lines are displayed per network name. The rest of the characters are removed from the display

Android

Version tested:

Char	Type	Example	Comments
Null	2	aireye_office is displayed as aireye_office	
Tab	2	Aireyet_office as aireye_office	A single tab character is ignored in the display

Ubuntu

Version tested: 20.04.2

Char	Type	Example	Comments
Null	2	aireye_office is displayed as aireye_office	

This entry was posted in SSID Stripping and tagged research, SSID, vulnerability, Wireless Networks.

AMICHAI SHULMAN

‹ The problem with NSA’s public wireless guidance

AirEye

Contact Us

Tel Aviv, Israel

Tel: +972-52-4268488

info@aireye.tech

Network Airspace Security
Airspace Control
Airspace Protection
NACP Capabilities

Platform
Overview
Unauthorized Access
Device Hijacking
Data leakage

About
Who We Are
In the News
Trademarks

Free Tools
Hide ‘n Seek

SCHEDULE A DEMO