Improving 802.11 Fingerprinting of Similar Devices by Cooperative Fingerprinting

Clémentine Maurice[1,2], Stéphane Onno[1], Christoph Neumann[1], Olivier Heen[1], Aurélien Francillon[2]

[1] *Technicolor, Rennes, France*

[2] *Eurecom, Sophia Antipolis, France*

first_name.last_name@technicolor.com, aurelien.francillon@eurecom.fr

Keywords: 802.11 Fingerprinting, Wireless Network Security, Anti-Forgery.

Abstract: Fingerprinting 802.11 devices has been proposed to identify devices in order to mitigate IEEE 802.11 weaknesses. However, important limitations prevent any real deployment. On the first hand, fingerprinting has a low accuracy when the devices have similar hardware and software. On the second hand, attackers may forge signatures to impersonate devices. We propose *Diversity*, a cooperative fingerprinting approach that improves accuracy of existing fingerprinting methods while relying only on off-the-shelf hardware. *Diversity* improves fingerprinting up to the reliable individual identification of identical 802.11 devices. This approach modifies the signature of devices by modifying slightly their traffic attributes. We evaluate *Diversity* with both a simulation and an implementation, achieving a false positive rate of 0% with a dataset including identical devices. Finally, we complement *Diversity* by mechanisms for detecting attackers that try to forge signatures.

1 INTRODUCTION

Popular versions of IEEE 802.11 (IEEE, 1997) have limitations when it comes to providing a secure wireless network. Management and control frames are unauthenticated and unencrypted in 802.11, due to backward compatibility, even when Wi-Fi Protected Access is used. Such frames are the source of most attacks on 802.11 (Bellardo and Savage, 2003). Aside from protocol weaknesses, there are situations where the wireless authentication tokens may leak. For example in an enterprise network, on which we focus in this work, the access control is mainly based on the couple login and password. However, credentials are often lent to a guest for an ephemeral network access. Credentials can also be leaked by attacks relying on rogue Access Points (APs) or phishing. These scenarios are common and they harm network security. Non-legitimate devices could indeed use leaked credentials to reconnect abusively and launch more severe attacks, or publish the authentication tokens. Some non-cryptographic techniques thereby emerge on the sidelines of the standard to overcome its limits (Zeng et al., 2010): control of sequence numbers, localization or fingerprinting. These techniques are meant to be used in conjunction with cryptography, not to replace it.

We focus on 802.11 fingerprinting, that is the identification of a device by extracting some externally observable characteristics. Fingerprinting results in a signature and a classification of each device. It is made feasible by the variety of implementations of the 802.11 standard.

We aim to tackle three main challenges: (i) provide unique signatures, i.e., two different devices should always have two different signatures even if the devices use the same hardware and software, (ii) be resistant to signature forgery attacks, (iii) rely only on off-the-shelf hardware.

Existing fingerprinting methods do not meet the above challenges. For instance, fingerprinting methods that identify drivers or Network Interface Cards (NICs) (Cache, 2006; Franklin et al., 2006) cannot differentiate two different devices using the same driver or NIC. Methods that identify unique devices using physical layer fingerprinting (Hall et al., 2004; Brik et al., 2008) achieve a very low false positive rate, but they require dedicated and expensive hardware (e.g., signal analyzer). In contrast, we rely on commercial off-the-shelf NIC in monitoring mode. Finally, to our knowledge, attacks on the fingerprinting process itself – like signature forgery – have only been studied in physical layer fingerprinting (Danev et al., 2010).

Contributions

We propose a fingerprinting system that detects illegitimate devices connected to 802.11 networks, even if the devices use legitimate but compromised connection credentials. The system enables two different devices to have distinct signatures, even if the two devices are equivalent. In particular, we present:

1. An approach we call *Diversity* that improves unique devices identification by slightly changing some traffic attributes of the fingerprintees, and thus their signatures. *Diversity* can be implemented as a daemon installed on the cooperative fingerprintees. *Diversity* handles existing fingerprinting methods as black boxes and does not modify them.
2. A proof of concept of *Diversity* relying on a fingerprinting method inspired by (Cache, 2006), based on 802.11 duration fields. The evaluation of our implementation on 9 devices shows that it achieves a false positive rate of 0% – compared to a false positive rate of 78% *without Diversity*.
3. An anti-forgery mechanism that raises the level of difficulty to forge fingerprints for an attacker. The mechanism prevents traffic replay attacks and separates forged signatures from authentic ones.

The remainder of this paper is organized as follows. Section 2 describes the principle of *Diversity*. Section 3 presents an instantiation of *Diversity* on a fingerprinting method relying on duration fields. Section 4 contains the evaluation and results. Section 5 discusses design issues. Section 6 exposes a mechanism to prevent the forgery of signatures. Section 7 describes the related work and Section 8 concludes.

2 PRINCIPLE OF DIVERSITY

In this section, we introduce our application setup, an enterprise network, and present our attacker model. We then describe the *Diversity* approach, which tackles the issue of uniqueness of signatures for each device.

2.1 Context and Attacker's Model

We illustrate our approach with the case of an enterprise network. In such a network, the identification is often based on the couple login and password with an additional access control based on MAC addresses. Fingerprinting can be used as an additional mitigation technique to detect unauthorized devices that bypassed the latter security mechanisms. The fingerprinters are typically deployed on the APs that can continuously observe the connected devices.

Fingerprinting methods generally rely on supervised learning and contain two different phases. The learning phase generates a base of known devices' signatures, the signatures being a compact representation of the externally observed traffic characteristics. The prediction phase lets the fingerprinter identify unknown devices relatively to the base of known ones. In an enterprise, the administration of devices is often centralized by the IT department, which can deploy defensive mechanisms on all devices or prepare the devices before handing them to users. The learning phase can thus be performed for each new device, enriching the base of known signatures with the signatures of authentic devices. Consistently with other fingerprinting approaches, we assume there is no attacker during this learning phase. The prediction phase is handled by the fingerprinters on the APs.

The devices used in enterprises are most of the time standard, similar and off-the-shelf since enterprises generally work with a single and standard device supplier. This makes classical fingerprinting approaches quite inefficient because many devices exhibit the same fingerprint. In addition, an attacker that uses similar hardware can bypass the fingerprinting protection. *Diversity* is meant to solve this issue. It can be implemented as a daemon installed on all authorized devices by the IT department.

We suppose that an attacker is able to impersonate MAC addresses and may have access to credentials to connect to the wireless network, e.g., borrowing credentials or obtaining them through phishing. The attacker has hardware similar to that of a legitimate user both for fingerprinting and for connecting to the network, making him more difficult to detect. The attacker is also able to capture and replay signatures extracted from wireless traffic. We assume the attacker has the *Diversity* daemon. However, we do not consider an attacker that is able to use special hardware to, e.g., perform physical layer fingerprinting. Finally, we suppose an attacker that has no access to the device of his victim, whether it is physically or remotely.

2.2 Design

Diversity consists, for a given fingerprinting method, in a slight and uninformed modification of the traffic attributes of the fingerprinted devices. The modification is slight so that normal traffic is not perturbed, but sufficiently important to change the signatures of the fingerprinted devices. The modification is also uninformed so that no additional secret needs to be de-

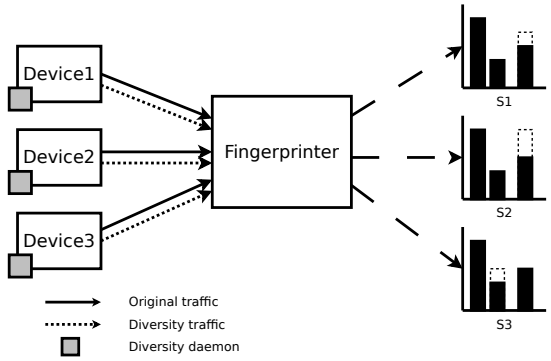

Figure 1: Device1, Device2 and Device3 are similar and run the *Diversity* daemon. *Diversity* slightly changes the traffic attributes and thus differentiates the signatures S1, S2 and S3.

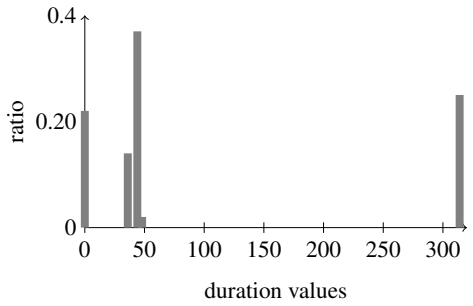

Figure 2: Duration fields signature of an Atheros AR5B95 chipset without *Diversity*.

ployed. *Diversity* requires the cooperation of the fingerprintees that run the *Diversity* daemon. Figure 1 illustrates the principle. The modifications are derived from a seed chosen randomly by each device at installation time of the daemon. Therefore, even two identical devices will have a different signature.

Applying *Diversity* requires three steps:

1. Choice and study of a given fingerprinting method. We need to understand the calculation of the signatures and the factors that influence them. This fingerprinting method will be used as a black box.
2. Choice of a traffic attribute that we can modify. The modifications of this traffic attribute must have an impact on the signatures.
3. Choice of the magnitude of the modifications. The modifications must lead to a significant change in the signatures to improve the results of the fingerprinting method. However, the modifications should not disrupt the behavior of the device.

We stress that there is no shared secret between the fingerprinter and the fingerprintees. The seed is internal to each fingerprintee, and is not known by the fingerprinter that only observes the result on the signatures. The modifications also suppose that the defender has a certain control over the legitimate devices, which is the case in an enterprise setup.

3 DIVERSITY IN PRACTICE

In this section, we instantiate the three steps discussed in Section 2.2.

In the first step, we consider a fingerprinting method based on *duration* fields that is inspired by J. Cache's method (Cache, 2006). We first recall the method. The 802.11 standard sets the duration field to announce how long in microseconds the device intends to keep the channel. This field is a 16-bit value, which represents up to $65,535$ µs, whereas the standard does not admit values over $32,767$. Cache observes that the field generally only takes a few discrete values, and that some implementations differ with distinctive values – sometimes illegal ones. He uses this field to fingerprint drivers. A signature is a set of pairs (*dur*, *ratio*), where *dur* is the duration value and *ratio* is the ratio for the duration *dur* relative to the total number of packets. A signature can be represented as a histogram (see Figure 2). We choose this fingerprinting method to test our approach because it passively fingerprints devices, using a standard NIC. Moreover, several parameters seem to influence the signatures, therefore we have some freedom to modify them. To compare unknown signatures to known ones, the learning algorithm uses a distance measure. The prediction is the closest known signature. We use the Jaccard distance measure $J_\delta(A,B)$, where two signatures A and B are considered as a set of duration values. This distance is defined by: $J_\delta(A,B) = 1 - \frac{|A \cap B|}{|A \cup B|}$. Two signatures are then considered to be "close" if they share many duration values. This distance does not use the duration field ratios because we found that they do not improve the results of driver identification.

In the second step, we choose the attributes that can be modified in the signatures. Figure 2 presents a signature represented by a histogram. To differentiate the signatures of devices that have the same driver, we choose to inject frames with specific duration values chosen randomly as described in detail below. Figure 3 presents the slightly modified signature and the impact of the injected frames.

In the third step, we choose the intensity of perturbations. Adding excessive perturbations to durations may impact the normal behavior of the network.

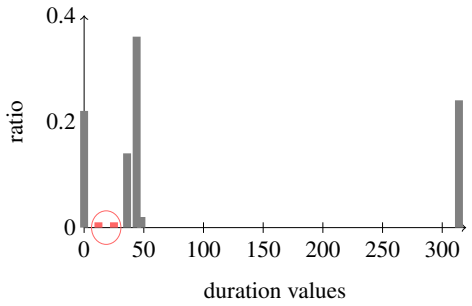

Figure 3: Duration fields signature of an Atheros AR5B95 chipset with *Diversity*: the durations 12 and 25 are added to the original signature.

Indeed, a duration indicates the reservation time of the channel, so it should not be too long. While it would have been possible to set duration field values from 0 µs to 2^{16} µs (or 2^{15} to be standard compliant) we choose to be very conservative and only use values that are commonly found in 802.11 implementations. Therefore, the seed initializes a pseudorandom number generator that returns one duration value or more in the interval $]0,44[\cup]44,50]$.[1] Moreover, if the packet type requires an answer it must be coherent with the state of the device. We choose to inject probe requests that can legitimately be sent by a device associated or not. RTS/CTS are also eligible. As such, the packet type does not impact the signature, since we do not take types into account.

4 EVALUATION

In this section, we expose the methodology and the results of our evaluation.

4.1 Methodology

We evaluate our approach against two datasets: *Homogeneous* and *Heterogeneous*. The dataset *Heterogeneous* is composed of 17 unique tuples {machine, NIC model, driver} that cover 9 different drivers. The dataset *Homogeneous* is composed of 9 identical Netgear WNA1100 USB adapters. The adapters use an Atheros AR9271 chipset with `ath9k_htc` driver, and support frame injection. The host runs Ubuntu, kernel 2.6.38-13. The dataset *Heterogeneous* (respectively the *Homogeneous*) characterizes the ability of fingerprinting methods to identify drivers (respectively unique devices). For both datasets, the monitoring device runs `tcpdump` with an Intel NIC and its native driver. While the size of the datasets might seem small, this is not unusual in the fingerprinting field. Indeed, datasets are not easy to build and, as we need the ground truth for the drivers and unique devices, we cannot use existing traces.[2] In comparison, Franklin et al. (Franklin et al., 2006) use a dataset of 17 drivers, Cache (Cache, 2006) a dataset of 14 devices, and Bratus et al. (Bratus et al., 2008) a dataset of 5 devices.

We use two different trace files for the dataset *Homogeneous*: one without *Diversity* and one with. We collect signatures one by one during 15 minutes for each trace. For the trace *without Diversity*, the device is not associated for the first 10 minutes, then it associates with a specific access point for the rest of the experiment. After 2 minutes, the device performs 2 `wget` requests and finally stays idle for the remaining time. This procedure allows to obtain a significant number of frames of each type, thus a complete signature. For the trace *with Diversity*, we inject diversified frames (see Section 4.2) before the `wget` requests. We then split our trace files to separate the training set from the test set.

We evaluate the False Positive Rate (FPR) as we try to reduce the number of false alarms in fingerprinting systems. The lower the better, since false positives represent the fraction of devices wrongly identified as intruders. We also use the True Positive Rate (TPR) to evaluate the ability of our approach to classify correctly the devices. The greater the better, since true positives represent the fraction of devices correctly identified as authorized.

Table 1: Simulation on the dataset *Homogeneous*.

Frames injected	0	1	2	10
TPR	100%	100%	100%	100%
FPR	78%	9%	0%	0%

Table 2: Implementation results.

(TPR, FPR)	without *Diversity*	with *Diversity*
Homogeneous	(100%, 78%)	(100%, 0%)
Heterogeneous	(100%, 34%)	–

4.2 Results

Without Diversity, we obtain the following results. On the dataset *Heterogeneous*, we obtain a TPR of 100% and a FPR of 13% for driver identification; we obtain a TPR of 100% and a FPR of 34% for unique devices identification. On the dataset *Homogeneous*, we ob-

[1] Removing durations 0 and 44 allows to improve fingerprinting reliability because these values are very commonly used by 802.11 devices.

[2] For example, Sigcomm traces: `http://www.cs.umd.edu/projects/wifidelity/sigcomm08_traces/`.

tain a TPR of 100%, but a FPR of 78% for unique device identification. Table 2 summarizes the results regarding unique device identification. In the remainder we only consider the dataset *Homogeneous*, as it characterizes the ability of a fingerprinting method to identify unique devices.

To simulate *Diversity* for a given device, we add frames with a specific duration from its MAC address in the traffic trace *without Diversity*. We want to estimate the number of diversified frames required to distinguish the 9 adapters. We run the simulation ten times with 1, 2 and 10 different durations. Table 1 shows the simulation results. The TPR remains of 100%: the injected frames do not degrade the ability of the classifier to identify the devices. Moreover, the FPR decreases with the number of durations injected, and it is significant with a single duration: from 78% without injection to 9%; with 2 durations, we get a FPR of 0%.

The implementation requires frame injection. We use the libraries `lorcon2` and `Net-Lorcon2`[3] for Perl to inject raw 802.11 packets. We inject probe requests frames whose durations are in the same interval as the simulation. Given the simulation results, we inject 2 different durations for each USB adapter. In some cases, we observe that some of the frames are lost due to natural 802.11 faults. We send 4 frames for each duration in order to avoid these. Table 2 shows our implementation results. We obtain a TPR of 100% and a FPR of 0% with the 9 identical USB adapters. Two different durations are therefore discriminating enough to correctly identify them with no false alarm.

5 DISCUSSION

In this section, we discuss the scalability of *Diversity*. We also argue that it has no impact on 802.11 performance. Then we consider the use of an alternative fingerprinting method. Finally, we briefly mention privacy issues.

5.1 Scalability

Diversity is scalable in the sense that the number of devices that need to be diversified can be adjusted. A small portion of devices covered by *Diversity* is sufficient to improve globally the identification results. However, we recommend covering a large portion so that the choice of spoofable devices is the smallest possible for an attacker. For the portion covered, we must ensure that each device generates a modification of its signature that is different from all other devices.

[3] `http://search.cpan.org/~gomor/Net-Lorcon2-2.02/`

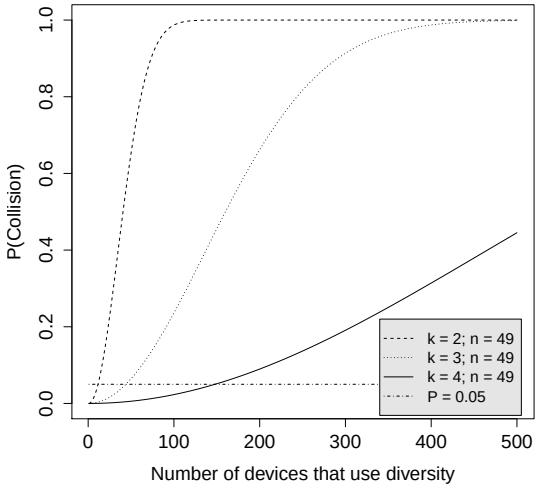

Figure 4: Probability that two devices choose the same set of duration values as a function of the number of devices.

Orthogonally, *Diversity* is scalable in the sense that it can support hundreds of devices. To generate unique signatures, the set of duration values chosen by each device must be different from all other devices. We can calculate the theoretical number of devices supported by *Diversity* with duration fields as follows. Each device chooses randomly k different duration values out of n ($k = 1, 2$ and 10 in our simulations and $k = 2$ in our experiments; $n = 49$ in our setting). The number of possible combinations is $\binom{n}{k}$ and we consider the choice to be i.i.d. for each device. Using the birthday paradox, we can then calculate the probability $P(Collision)$ of having at least two devices that choose the same set of duration values, as a function of d the number of devices being deployed, as follows: $P(Collision) = 1 - d!\binom{\binom{n}{k}}{d} / \binom{n}{k}^d$. Figure 4 shows that up to hundreds of devices can be supported by increasing k accordingly while $P(Collision) \leq 0.05$. With $k = 4, n = 49$ up to 147 devices yield $P(Collision) \leq 0.05$.

5.2 Impacts on 802.11 Performance

We argue that *Diversity* does not alter 802.11 performance in terms of duration and number of frames injected. While the standard (IEEE, 1997) states that all devices have to process the duration field to update their Network Allocation Vector (NAV), it also states that the duration of broadcast frames must be 0. To be able to evaluate the behavior of devices in the presence of forged broadcast probe requests with non-zero durations, we inject a probe request whose duration is 32,767 (roughly 32 ms) every 30 ms. A standard

reaction for other devices would be the inability to send their own frames. However, this is not the case. Devices thus seem to ignore the duration field of the injected probe requests. This is a behavior we also noticed in other traffic captures. Bellardo et al. confirmed these observations when trying to attack the virtual carrier sense mechanism with RTS/CTS (Bellardo and Savage, 2003).

We showed in Section 5.1 that *Diversity* can support hundreds of devices when injecting as few as 4 different duration values. The impact of the number of frames injected depends on the frequency of *Diversity*: such an injection is negligible in a normal 802.11 traffic if it happens every hour or minute. However, the frequency of *Diversity* also impacts the time needed to recognize a device. Probe requests may induce a number of probe responses that is of the same order of magnitude as the number of probe requests.

5.3 Alternative Fingerprinting Method

Diversity is generic in the way that it applies to other fingerprinting methods. Indeed, we also tested the approach with Franklin et al. method (Franklin et al., 2006). This method uses the active scanning period to passively fingerprint drivers. The active scanning procedure is known to lack precise timing definition in the 802.11 standard. This fingerprinting method measures the time between probe requests in a burst in the same channel, as well as the time needed to cycle through the different channels (which forms peaks in the signatures). On a dataset *without Diversity* of one hour for each device non-associated, we obtain a FPR of 56%. To test the *Diversity* approach, we randomly change the time spent on a channel for each device. *With Diversity*, we obtain a FPR of 44%: the improvement is not statistically significant for such a dataset. It seems that the limiting factor is the intensity of the perturbation. If the modifications are too intense, the signatures and the fingerprinting process itself are affected, resulting in bad fingerprinting results. It also disrupts the very functioning of the device: the whole active scanning procedure takes more than a minute, which is too long. One solution would be to find another way to modify signatures for Franklin et al. method, by changing the modified attributes. Future work should address this issue.

5.4 Privacy

We reckon that privacy is not part of our design goals for the following reasons. First, it can be argued that *Diversity* decreases privacy, as our goal is to achieve a better identification of devices. While this is true, we remind the reader that MAC addresses already present a risk for privacy as stated in (Gruteser and Grunwald, 2005). Moreover, Pang et al. (Pang et al., 2007) show the existence of privacy leaks by implicit identifiers even when users change their MAC addresses every hour. Second, we propose to restrict the use of *Diversity* to networks that specifically need it. For instance, it can be enabled in enterprise network and disabled in home network, based on SSIDs. In this case, the situation regarding privacy in home network is left unchanged, that is to say heterogeneous devices with fixed MAC addresses and possibly different signatures.

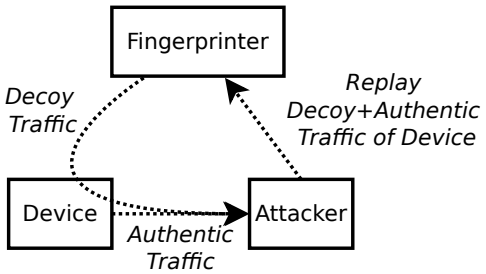

Figure 5: Principle of anti-forgery. The fingerprinter sends additional decoy frames while spoofing the MAC address of the device. The attacker does not distinguish authentic from decoy traffic.

6 ANTI-FORGERY

Diversity protects from an attacker that uses similar hardware and knows the credentials of his victim. However, it does not protect from an attacker able to capture and replay signatures extracted from wireless traffic. Unfortunately, it is not difficult for an attacker to inject or alter its own frames to forge signatures. To our knowledge, such forgery attacks have only been studied in physical layer fingerprinting (Danev et al., 2010). We propose an anti-forgery system that tackles the issue of an attacker being able to capture and replay signatures – including *Diversity*– from his victim.

In this system, the fingerprinter sends decoy frames spoofing the MAC address of a protected device to trap an attacker that forges signatures of legitimate devices. The principle is illustrated in Figure 5. The surrounding devices, including the attacker, cannot differentiate the authentic frames of the legitimate device from the decoy frames of the fingerprinter. The attacker that captures and replay the traffic of his victim would also capture and replay these decoy frames. His signature as computed by the fingerprinter will

consequently be different from that of his victim. The fingerprinter will not match the two different signatures, and the attacker would be detected. The decoy frames are chosen and sent in order to modify slightly the signature of the legitimate device from an external viewpoint. They must be taken into account by the fingerprinting method.

One difference between the fingerprinter and the fingerprintee is the location. One attacker that would be able to localize precisely both the fingerprintee and the fingerprinter could distinguish the decoy frames and remove them when replaying his victim's traffic. Nevertheless, accurate localization is not easy. For example, Haeberlen et al. need at least a minute of traffic from the devices to analyze their signal strength in (Haeberlen et al., 2004). We do not provide as many opportunities to determine the fingerprinter localization, as we only inject a few frames as decoy frames. Another strategy for an attacker would be to identify the source of the packets by performing physical layer fingerprinting. However, this requires signal analyzers, and as we mentioned in Section 2.1, such attackers are out of the scope of this paper.

7 RELATED WORK

Driver or NIC fingerprinting exploit differences in implementation of the standard. Franklin et al. use the active scanning period to fingerprint passively drivers (Franklin et al., 2006). This procedure lacks precise timing definition in 802.11 standard about the frequency of probe requests in a specific channel, and the time needed to cycle through the different channels. Driver implementations then vary. Bratus et al. develop an active method to fingerprint the driver or NIC (Bratus et al., 2008). They send non-standard frames and chart the responses in decision trees. These methods do not intend to differentiate two different devices using the same driver or NIC. They are good candidates to the *Diversity* approach.

For unique devices fingerprinting, two ways are envisioned: using the clock characteristics, or global traffic characteristics. Desmond et al. follow the work of (Jana and Kasera, 2008) and (Arackaparambil et al., 2010) on APs (Desmond et al., 2008). They discuss the possibility to enhance time skew to devices by clustering probe requests inter-arrival time. They remark slight time variations that depend on the time skew. However, gathering a sufficient number of frames for a good time skew estimation takes more than an hour. Neumann et al. propose to identify unique devices using the frame inter-arrival time (Neumann et al., 2012). This reveals global traffic characteristics that depend on a mix of wireless cards, driver features, and on the application generating the data.

Physical layer fingerprinting exploits the characteristics of the radio transceivers that sends and receives data in 802.11 devices. They also have unique characteristics, often based on minute hardware imperfections. Hall et al. establish a transient-based approach that extracts the transient portion of the signal (Hall et al., 2004). Brik et al. develop a modulation-based approach that extracts features from the part of the signal that has been modulated, i.e., the data (Brik et al., 2008). These two approaches require dedicated and expensive signal analyzer whereas we rely on off-the-shelf devices. Danev et al. have considered the ease with which an attacker is able to forge physical layer fingerprints (Danev et al., 2010). They found that transient-based techniques are more difficult to reproduce.

The work of Prigent et al. focus on TCP/IP stack identification (Prigent et al., 2010). Using fingerprint deception, they modify the signatures of the fingerprintees as we do in *Diversity*. However, their goal is opposite: instead of better identifying the devices, they want to hide the true identities of the fingerprintees to be less appealing to attackers.

Castelluccia and Mutaf use frame injection and identity spoofing as we do in the anti-forgery system (Castelluccia and Mutaf, 2005). They build a pairing protocol, in which two devices agree on a n-bit secret sending n packets with the source field set to one of their two identities (depending on the desired bit 0 or 1). The devices need to be shaken to achieve spatial indistinguishability. An eavesdropper cannot retrieve the secret since she cannot figure out which device actually sent the packet.

Finally, the 802.11w amendment targeted the protection of management frames, so one can argue that fingerprinting is less needed. However, Ahmad and Tadakamadla found that 802.11w is still vulnerable to some known attacks, in addition to three new ones (Ahmad and Tadakamadla, 2011). In common, theses attacks leverage the ability to spoof MAC addresses. More generally, we believe that as long as it is possible to spoof MAC addresses, fingerprinting will be useful. We also stress that fingerprinting can be used with legacy devices.

8 CONCLUSIONS

Fingerprinting has a limited accuracy in the case where devices are similar. We designed a generic approach called *Diversity* that, given a fingerprinting

method, improves the identification of unique devices even if they have equivalent {machine, NIC model, driver}. This approach does not need a shared secret between the fingerprinter and the fingerprintee. We implemented and evaluated it against a dataset composed of 9 identical USB adapters, which corresponds to a worst case for traditional fingerprinting methods. We used a fingerprinting method based on duration fields and we obtain a TPR of 100% and a FPR of 0%. Finally, we proposed a mechanism that prevents the forgery of fingerprints. Further studies are needed to determine if *Diversity* could be applied to other fingerprinting methods with success; we identify two candidate methods: (Neumann et al., 2012) and (Desmond et al., 2008).

REFERENCES

Ahmad, M. S. and Tadakamadla, S. (2011). Short Paper: Security Evaluation of IEEE 802.11w Specification. In *WiSec'11*, pages 53–58.

Arackaparambil, C., Bratus, S., Shubina, A., and Kotz, D. (2010). On the reliability of wireless fingerprinting using clock skews. In *WiSec'10*.

Bellardo, J. and Savage, S. (2003). 802.11 denial-of-service attacks: Real vulnerabilities and practical solutions. In *USENIX Security Symposium*.

Bratus, S., Cornelius, C., Kotz, D., and Peebles, D. (2008). Active behavioral fingerprinting of wireless devices. In *WiSec'08*.

Brik, V., Banerjee, S., Gruteser, M., and Oh, S. (2008). Wireless device identification with radiometric signatures. In *MobiCom'08*.

Cache, J. (2006). Fingerprinting 802.11 implementations via statistical analysis of the duration field. *Uninformed. org*, 5.

Castelluccia, C. and Mutaf, P. (2005). Shake them up!: a movement-based pairing protocol for cpu-constrained devices. In *MobiSys'05*.

Danev, B., Luecken, H., Capkun, S., and El Defrawy, K. (2010). Attacks on physical-layer identification. In *WiSec'10*.

Desmond, L. C. C., Yuan, C. C., Pheng, T. C., and Lee, R. S. (2008). Identifying unique devices through wireless fingerprinting. In *WiSec'08*.

Franklin, J., McCoy, D., Tabriz, P., Neagoe, V., Randwyk, J. V., and Sicker, D. (2006). Passive data link layer 802.11 wireless device driver fingerprinting. In *USENIX Security Symposium*.

Gruteser, M. and Grunwald, D. (2005). Enhancing location privacy in wireless lan through disposable interface identifiers: a quantitative analysis. *Mobile Networks and Applications*, 10(3):315–325.

Haeberlen, A., Flannery, E., Ladd, A. M., Rudys, A., Wallach, D. S., and Kavraki, L. E. (2004). Practical robust localization over large-scale 802.11 wireless networks. In *MobiCom'04*.

Hall, J., Barbeau, M., and Kranakis, E. (2004). Enhancing intrusion detection in wireless networks using radio frequency fingerprinting. In *Conference on Communication, Internet and Information Technology*.

IEEE (1997). IEEE Std 802.11-1997 - Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications.

Jana, S. and Kasera, S. K. (2008). On fast and accurate detection of unauthorized wireless access points using clock skews. In *MobiCom'08*.

Neumann, C., Heen, O., and Onno, S. (2012). An empirical study of passive 802.11 device fingerprinting. In *IEEE ICDCS Workshop on Network Forensics, Security and Privacy*.

Pang, J., Greenstein, B., Gummadi, R., Seshan, S., and Wetherall, D. (2007). 802.11 user fingerprinting. In *MobiCom'07*.

Prigent, G., Vichot, F., and Harrouet, F. (2010). Ipmorph: Fingerprinting spoofing unification. *Journal in computer virology*, 6(4):329–342.

Zeng, K., Govindan, K., and Mohapatra, P. (2010). Non-cryptographic authentication and identification in wireless networks. *IEEE Wireless Communications Magazine*, 17(5):56–62.