SatIQ: Extensible and Stable Satellite Authentication using Hardware Fingerprinting

JOSHUA SMAILES, Department of Computer Science, University of Oxford, Oxford, United Kingdom of Great Britain and Northern Ireland
SEBASTIAN KÖHLER, Department of Computer Science, University of Oxford, Oxford, United Kingdom of Great Britain and Northern Ireland
SIMON BIRNBACH, Department of Computer Science, University of Oxford, Oxford, United Kingdom of Great Britain and Northern Ireland
MARTIN STROHMEIER, armasuisse, Science + Technology, Zurich, Switzerland
IVAN MARTINOVIC, Department of Computer Science, University of Oxford, Oxford, United Kingdom of Great Britain and Northern Ireland

As satellite systems become a greater part of critical infrastructure, they have become a significantly more appealing target for attacks. The availability of cheap off-the-shelf radio hardware has made signal spoofing and physical layer attacks more accessible than ever to a wide range of adversaries, from hobbyists to nation-state actors. Legacy systems are particularly vulnerable due to their lack of cryptographic security, and cannot be patched to support novel security measures.

In this article, we use radio transmitter fingerprinting to authenticate satellite downlinks, using characteristics of the transmitter hardware expressed as impairments on the physical layer radio signal. Our SatIQ system employs a Siamese neural network and an autoencoder to extract an efficient encoding of message headers that preserves identifying information. We focus on high sample rate fingerprinting, making device fingerprints difficult to forge without similarly high sample rate transmitting hardware.

We collected 10 290 000 messages from the Iridium satellite constellation at 25 MS/s, and demonstrate that the SatIQ model trained on this data maintains performance over time without retraining, and can be used on new transmitters with no impact on performance. We analyze the system's robustness against weather and signal factors, and demonstrate its effectiveness under attack, achieving an Equal Error Rate of 0.072 and ROC AUC of 0.960. We conclude that our techniques are useful for building fingerprinting systems that are effective at authenticating satellite communication, maintain performance over time and across satellite replacement, and provide robustness against spoofing and replay by raising the required budget for attacks.

CCS Concepts: • **Security and privacy** → **Mobile and wireless security**; **Authentication**; **Systems security**; • **Hardware** → *Wireless devices*; • **Computer systems organization** → *Sensor networks*; *Embedded and cyber-physical systems*; • **Computing methodologies** → Machine learning;

This article is an extension of an earlier conference paper, referenced as [1].

Authors' Contact Information: Joshua Smailes, Department of Computer Science, University of Oxford, Oxford, United Kingdom of Great Britain and Northern Ireland; e-mail: joshua.smailes@cs.ox.ac.uk; Sebastian Köhler, Department of Computer Science, University of Oxford, Oxford, United Kingdom of Great Britain and Northern Ireland; e-mail: sebastian.kohler@cs.ox.ac.uk; Simon Birnbach, Department of Computer Science, University of Oxford, Oxford, United Kingdom of Great Britain and Northern Ireland; e-mail: simon.birnbach@cs.ox.ac.uk; Martin Strohmeier, armasuisse, Science + Technology, Zurich, Switzerland; e-mail: martin.strohmeier@armasuisse.ch; Ivan Martinovic, Department of Computer Science, University of Oxford, Oxford, United Kingdom of Great Britain and Northern Ireland; e-mail: ivan.martinovic@cs.ox.ac.uk.

ACM 2471-2566/2025/11-ART2
https://doi.org/10.1145/3768619

Additional Key Words and Phrases: Satellite communication, fingerprinting, radio frequency, autoencoder, machine learning, siamese network, physical layer security

ACM Reference Format:
Joshua Smailes, Sebastian Köhler, Simon Birnbach, Martin Strohmeier, and Ivan Martinovic. 2025. SatIQ: Extensible and Stable Satellite Authentication using Hardware Fingerprinting. *ACM Trans. Priv. Sec.* 29, 1, Article 2 (November 2025), 35 pages. https://doi.org/10.1145/3768619

1 Motivation

Recent years have seen a dramatic rise in the availability of cheap radio hardware, particularly **Software-Defined Radios** (**SDRs**). Not only have these devices become more widely available, but their capabilities have increased, with devices like the *HackRF One* (340 USD, Adafruit) able to transmit and receive in frequencies ranging from 1 MHz to 6 GHz [2]. As a result, the ability to carry out physical layer attacks (particularly spoofing, jamming, and replay), once exclusive to large-budget organizations and nation-state actors, is now within reach for even motivated hobbyists. This poses a particular threat to satellite systems, many of which were built under the assumption that tampering with signals would be prohibitively expensive for the vast majority of attackers.

Physical layer attacks have been widely explored in wireless systems – attackers equipped with an SDR can overshadow legitimate communications or spoof messages outside normal communication. Many widely used systems are vulnerable to these attacks, including the ADS-B avionics protocol [3], the LTE telephony system [4, 5], and satellite systems including GPS [6]. Due to the critical nature of satellite systems, it is vital that operators can prevent or detect spoofing attacks to protect the systems and applications that rely on them.

There are a wide range of techniques for detection and prevention of spoofing attacks, the foremost of which is cryptography – a properly implemented cryptosystem with associated key management provides robust authentication, making spoofing attacks near-impossible. However, there are a number of reasons why cryptography may not be desirable (or possible) in the context of satellite systems. Firstly, there are a huge number of legacy satellites currently in orbit. Many of these do not implement cryptography, and cannot be retrofitted to do so due to their limited onboard processing power. However, the data collected by these satellites is immensely useful in both scientific and private use cases – they are used for monitoring forest fires, land usage, population density, flooding, and more [7–10]. These satellites are often bespoke designs which would be prohibitively expensive to replace; it is important to ensure systems like these can be used for their entire projected lifespan (and beyond).

There are also a number of satellite systems which were initially built with cryptography, but which have become insecure post-launch due to leaked keys [11] or outdated cryptosystems [12]. Some of these satellites cannot be patched due to a lack of over-the-air update capabilities, so other methods must be used to authenticate their telemetry data.

Finally, some attacks can be carried out without violating any cryptographic properties of the system. The authors of [13] show that precisely timed message replays can cause **Global Navigation Satellite Systems** (**GNSS**) to misreport the location of the receiver to an attacker-specified location. Since these attacks are carried out by simply introducing delay to messages rather than altering message contents, conventional cryptography does not protect against them. We still want to be able to detect and prevent these attacks, so we must turn to non-cryptographic techniques for message authentication. In particular, we investigate radio transmitter fingerprinting, in which radio signal characteristics are used to identify the transmitters. This is achieved by identifying impairments on the signal which are created by small differences in the radio transmitter hardware. These impairments are unique to transmitters and consistent over time, as we will show for the satellite case.

Contributions

In this article we present SatIQ, a novel approach to fingerprinting satellite signals. We work with signals at a high sample rate to counteract problems with spoofing at lower sample rates. This makes the techniques more useful in a security context, requiring attackers to use more expensive radio hardware that works at high sample rates in order to successfully impersonate a device – thus excluding a large number of low-budget adversaries. In doing this we can provide an additional level of confidence in the authenticity of the origin of satellite signals, particularly in systems where cryptography is either unavailable or ineffective. We verify that the system can detect replay attacks by replaying captured messages using an SDR.

We also use a *Siamese model* – unlike conventional classifiers, these compare two signals and produce a distance metric representing the likelihood of two messages having been sent from the same transmitter. This technique enables *one-shot learning*: new transmitters can be introduced without requiring the system to be retrained, and can be used immediately with only a small number of examples. This is particularly useful in the context of satellites in **Low Earth Orbit** (**LEO**), which must be replaced more frequently.

In Section 2 we explore the background of SDRs, countermeasures to physical-layer radio attacks, introduce the concept of fingerprinting, and explore related work in the area. In Section 3 we outline the threat model, describing the goals and capabilities of the attacker. In Section 4 we describe our large, high sample rate dataset of real-world data from the *Iridium* satellite constellation, composed of 10 290 000 messages from 3 different locations and hardware configurations. We use this data to train and test the SatIQ system, showing its ability to distinguish legitimate transmitters from one another, and exploring methods to enhance performance. In Section 5, we evaluate various useful properties of SatIQ, including its stability over time and extensibility to new transmitters without requiring retraining, correlations between performance and weather/signal factors, and transferability between different receiver configurations. We also evaluate SatIQ under a replay attack case, showing that it can easily distinguish an attacker-controlled radio from legitimate transmitters, and compare to the existing state-of-the-art. Finally, we explore interesting areas of future work in Section 6 and conclude in Section 7.

To faciliate this, our full dataset and code has been made freely available: links have been provided at the end of the article.

2 Background

In this section, we discuss key concepts in radio and digital signal processing. We also explore existing countermeasures to spoofing attacks, and look at existing research in radio fingerprinting in general. Finally, we discuss related work that focuses on radio fingerprinting of satellites, as well as related work that considers fingerprinting in a security context.

2.1 Software-defined Radios

SDRs allow the use of software for signal processing tasks traditionally done using dedicated hardware. This is achieved by sampling raw signals into a digital form and sending the samples to a computer, where they can be processed further. This provides significant versatility to signal processing over traditional radio hardware, at the cost of additional processing power.

A concept at the core of digital signal processing is *IQ Sampling*. By taking a *carrier signal* at a given frequency and sampling the components of the incoming signal that are in phase (I) and out of phase (quadrature, Q) with the carrier before sampling, the incoming signal is downsampled, shifting the carrier frequency to 0 Hz [14]. This significantly reduces the sample rate required.

One benefit of sampling in this way is that samples can be represented as complex numbers, and plotted on the complex plane. In this representation, distance from the origin represents amplitude and angle from the horizontal axis represents phase relative to the carrier signal. This makes it a particularly useful representation of ***Phase Shift Keying*** (**PSK**), a form of signal modulation which encodes data in the phase of the signal. In this case, symbols appear as distinct points on the complex plane, producing a *constellation diagram*.

We only see distinct points on the constellation diagram if we sample at the exact symbol rate of the modulation scheme. If we instead *oversample* the signal by using a significantly higher sample rate, we start to see the points between the symbols, as the transmitter hardware modulates between them. This can be seen later on in Figure 9 (Section 4.3). The appearance of this interpolation between points is affected by a number of factors including atmospheric noise and multipath distortion, but also by small variations in the transmitter hardware – these can be used to fingerprint the transmitter.

2.2 Spoofing Countermeasures

In Section 3 we explore the threat of an attacker equipped with a software-defined radio, looking in particular at spoofing attacks. When implemented properly, modern cryptographic authentication can solve this problem, but it is also possible to make these systems more secure without the use of cryptography. As discussed in Section 1, this is desirable in legacy systems without cryptography, or systems where cryptographic authentication is undesirable or has been compromised. Additionally, some systems provide open data by design, leaving out cryptography on downlinked communications by choice.

There are a diverse range of approaches for the authentication of downlinked transmissions without the use of cryptography. These can be partitioned into *data inspection*, *timing analysis*, and *waveform analysis*.

Data Inspection. The integrity of data can be verified by receiving the same transmission at multiple locations, and comparing the data or its hash between receivers. This requires an attacker to be physically present at each location in order to successfully carry out spoofing attacks, where they would have previously only needed to be at a single location. This is particularly useful in cases where there are already large numbers of community-operated ground stations, such as with NASA's Direct Readout Laboratory (168 operated worldwide at the time of writing) [15]. However, it is likely to be infeasible for smaller organizations to set up multiple ground stations.

Timing Analysis. These techniques involve looking at the timing of signals in order to verify their legitimacy. At the simplest level, this could be ensuring the signal is received at a time the satellite was known to be transmitting – this does not provide much real security. More advanced techniques in this area include **Time Difference of Arrival** (**TDOA**) analysis, looking at the time difference between multiple receivers to verify the transmitter's location against where the satellite is known to be [16]. This is an effective technique, forcing any potential attackers to be physically present at every ground station, but is once again only feasible for larger organizations capable of operating many ground stations.

Waveform Analysis. Alongside data and timing analysis, the waveform itself can be inspected to detect attacks. A spoofed signal is likely to have different properties from the legitimate signal, particularly when spoofing requires overshadowing an ongoing transmission. These properties include amplitude, SNR, doppler shift, and signal distortion [17, 18]. With appropriate radio hardware it is possible to verify these parameters, making attacks more difficult to execute – the adversary must replicate the measured properties in order to successfully spoof messages.

Fingerprinting also falls into this category – by looking at unique impairments on the raw waveform we can identify the transmitter. When spoofing or replaying messages the attacker's radio will impart a different fingerprint on the signal, allowing a fingerprinting system to detect when this has occurred. In order for the adversary to circumvent this system they will need to replicate the fingerprint of the legitimate transmitter. Depending on how the fingerprinter has been designed, this may raise the required budget to carry out attacks.

2.3 Fingerprinting

Radio fingerprinting is a mature field, with a large base of research looking at a wide range of techniques on many different systems – [19] provides a good overview of existing research. Fingerprinting techniques can be partitioned into two key areas: *transient fingerprinting* and *steady-state fingerprinting*.

2.3.1 Transient Fingerprinting. The *transient* of a radio signal occurs when the transmitter first powers on, or changes power levels following a signal lock. Various properties of the transient, such as its duration or the number of peaks which occur in the carrier signal, are characteristic to the transmitter and can be used to identify it, if properly extracted. The majority of historical fingerprinting research makes use of transient analysis, since almost all radio transmission involves a transient, and the transient typically exhibits the same characteristics every time the device powers up. Much of the work in transient fingerprinting revolves around novel techniques for precisely identifying the start and end of the transient [20, 21], or processing the transient to extract useful identifying features [22].

Transient fingerprinting has seen some use in security contexts – in [23], transient fingerprinting is used to identify devices even when all other identifying information has been removed, and to detect wormhole and device cloning attacks (these attack types are explored further in Section 3).

2.3.2 Steady-state Fingerprinting. In contrast to transient fingerprinting looking at a very brief portion of the signal, steady-state fingerprinting instead looks at the modulated portion of the signal. The features in the steady-state portion of the signal are different from the transient, often looking at how the IQ constellation is affected by hardware impairments. These impairments include quadrature errors, self-interference, amplitude clipping, and frequency offsets in the various stages of signal modulation (DAC, mixer, filter, upconverter, amplifier) [24]. Additionally, the signal is affected by properties of the wireless channel, including background noise, free space path loss, and multipath distortion, which are likely to change over time as the environment changes. These wireless channel properties are significantly more prominent in satellite communication, due to the long-distance radio links involved. Very good performance has been achieved using fingerprinting techniques to authenticate devices using over-the-wire communication [25], but these have much lower levels of noise on the channel, making the problem significantly easier.

There is some variety in the techniques used, including observing features of the constellation at a low sample rate [26], analyzing features in the frequency domain [27], and looking at high sample rate signals to observe high frequency impairments [28]. Machine learning techniques are more commonly used to aid steady-state fingerprinting, particularly in approaches working at a high sample rate, in order to pull out features which may not be immediately obvious. Manual feature engineering is also used, but is less common than in transient analysis.

The majority of steady-state fingerprinting looks at a single protocol or class of devices at a time, but with the recent rapid increase in machine learning capabilities this is no longer a requirement, and there has been some work into generalizable fingerprinting techniques which do not require retraining to apply to new contexts [29].

There are also some interesting techniques extending the concept of fingerprinting – the authors of [30] train a convolutional neural network to identify SDRs at 5 MS/s, then intentionally introduce signal impairments at the transmitter in order to further increase classification accuracy.

This achieves incredible accuracy (greater than 0.995), but fingerprint forgery is not considered in this context.

2.4 Related Work

We have explored the existing body of radio fingerprinting research in the previous section. However, there is also some research that focuses specifically on satellite fingerprinting and fingerprinting in a security context.

2.4.1 Satellite Fingerprinting. Unlike signals from terrestrial devices, satellite signals have to travel hundreds of kilometers through the atmosphere, causing significant signal attenuation and channel noise. This adds additional challenge to fingerprinting in this context, particularly since many techniques rely on minimal presence of background noise. There are some works looking at fingerprinting in the presence of noise, either by adding noise to clean signals during model training (effectively training models to remove/ignore the noise) [31], or by smoothing out long signals at low sample rates to obtain average symbol positions [32]. In the context of satellites it is difficult to obtain signals without noise, and smoothing does not work with high sample rate signals (since important detail is lost) – we must find other methods of reducing or ignoring noise. We discuss this further in Section 4.

The authors of [26] design “PAST-AI”, analyzing heatmaps of low sample rate transmissions from the *Iridium* constellation to classify satellites. This technique achieves an accuracy of approximately 0.85, increasing to 1.00 for small subsets of the constellation. Although this technique achieves high accuracy, it is not as useful from a security context – the classifier works by processing large batches of consecutive messages, making it more difficult to detect individual message spoofing. Furthermore, fingerprinting at a very low sample rate (1 sample per symbol) makes fingerprint forgery significantly easier, since the attacker does not need to replicate as many features of the waveform. The resulting bandwidth of 25 kHz is significantly less than the Iridium channel spacing of 41.667 kHz [33], so transmitter characteristics at higher frequencies will be discarded. The work claims to be able to discriminate SDR-equipped attackers from legitimate satellites by solving the harder problem of discriminating between satellites with the same hardware. This is true, however, the assumption only holds at the sample rate used by the fingerprinting system. As PAST-AI operates at 1 sample per symbol, it cannot protect against SDR-based attacks at higher sample rates. We demonstrate this experimentally in Section 5.7.

There has also been some fingerprinting work looking at other satellite systems – the authors of [24] make use of manual feature extraction to identify spoofing of GPS localization satellites. This technique is effective, but manual feature extraction is less likely to transfer easily to other satellites or constellations.

2.4.2 Security. Most fingerprinting is done for the purpose of security, preventing spoofing and replay attacks, but some techniques have been specifically proposed to provide better security properties. For instance, the authors of [28] address the problem of identifying devices which have not been seen in the training dataset by separating the feature extraction and classification components of the model, using clustering techniques on the extracted features to identify transmitters. This allows new transmitters to be introduced without retraining the feature extraction component. We take a similar approach in our work, using an autoencoder to produce the fingerprints and a Siamese model in the place of clustering. The SatIQ system is fully described in Section 4.

There has also been work assessing attacks on fingerprinting systems – it has been shown that an arbitrary waveform generator with a sufficiently high sample rate can be used to impersonate devices, fooling fingerprinting systems [34]. Hardware that can achieve these sample rates is prohibitively expensive for the vast majority of adversaries (we discuss budget further in Section 3), so

Table 1. Summary of Related Work in Satellite Fingerprinting

	Fingerprinting Technique	Satellites	High Sample Rate	Stable over Time	Extensible to New Transmitters	Transferable to New Receivers	No Manual Features	Single Message
Rasmussen et al. [23]	Transient		✓	–	✓	–		✓
Tekbaş et al. [31]	Transient			–	–	–		✓
Kennedy et al. [27]	Steady State			–	–	–		✓
Bassey et al. [28]	Steady State		✓	–	(✓)[A]	–	✓	✓
DeepRadioID [29]	Steady State			–	(✓)[B]	–		✓
ORACLE [30]	Steady State		✓	–		–	(✓)[C]	✓
Wang et al. [32]	Steady State			–		–	(✓)[D]	✓
PAST-AI [26]	Steady State	✓		–		–	✓	
Spotr [24]	Steady State	✓	✓	✓	–	✓		✓
SatIQ	Steady State	✓	✓	✓	✓	✓	✓	✓

Ticks (✓) denote support, dashes (—) indicate that the feature was not discussed or tested.
A: Claimed to be extensible but not experimentally verified.
B: Requires transmitter modifications.
C: Some manual features, optional transmitter modifications.
D: Manual filtering/smoothing, neural network classifier.

we do not consider this to be a major issue – although our techniques are likely vulnerable to impersonation at a very high sample rate, preventing spoofing below a certain transmitter sample rate is sufficient to exclude the vast majority of lower-budget attackers. In prior work, we have also looked at denial of service attacks on the fingerprinting system itself, showing that the introduction of fingerprinting does not necessarily make the overall system more vulnerable to jamming attacks [35].

2.4.3 Our Approach. Table 1 gives an overview of the features supported in the related works discussed. We see that our approach provides many desirable features, not all of which are present in other related works. SatIQ is designed to work in the high-noise environment of satellite communication, and works at a high sample rate to prevent spoofing and replay attacks using cheap SDRs from forging the fingerprint. Performance is stable over time, and the system can be trivially extended to support new transmitters – this is critical in satellite systems, where at any time new satellites may be introduced or existing satellites replaced, and gathering enough data on the new transmitters to retrain the model would be time-consuming. Furthermore, it does not rely upon any manual feature extraction, and operates on individual incoming messages, making it easy to deploy and use. In later sections, we describe the architecture of SatIQ that makes these features possible. We also demonstrate its features through experimentation, confirming its performance under attack and its ability to function over time and as new transmitters are introduced.

3 Threat Model

Goals. In this article, we concentrate on attacks involving spoofing and message replay. In the case of spoofing, the adversary's goal is to broadcast messages appearing to come from a satellite such that the ground system processes them alongside legitimate messages. Alternatively, the attacker may delay or advance messages (jamming the original and replaying a recording) to affect timing-based systems such as GPS [13]. Similarly, they could carry out "wormhole attacks", in which messages are captured at one location and tunneled to another location, from where they are broadcast – this is also effective against GPS and other localization systems. Unlike spoofing, these attacks can be performed even on signed or encrypted messages, since they do not affect message contents.

Capabilities. The adversary's capabilities and budget will vary depending on the type of adversary. We can reasonably expect any attacker to have access to an off-the-shelf SDR (as well as the appropriate amplifiers and antennas) enabling them to transmit messages within the vicinity of a single ground station. An equipment setup similar to the one given in Section 4.3 could be used to carry out attacks, costing approximately 6,600 USD. By using a cheaper SDR with a lower sample rate and a different antenna (with a suitable amplifier), the budget can be significantly lowered to approximately 500 USD.[1] Alternatively, with a greater budget the attacker could afford a more powerful amplifier enabling messages to be broadcast from a greater distance, or multiple copies of the hardware to attack multiple ground stations simultaneously.

It has been demonstrated in [34] that device fingerprinting is vulnerable to signal replay attacks, provided the attacker has access to a high-end arbitrary waveform generator capable of transmitting signals with a sufficiently high sample rate.[2] For this reason we do not consider attackers with nation-state level capabilities, since they are always capable of purchasing hardware that can fool a fingerprinting system.

In this work, we are particularly interested in lower-budget attacks; a simple spoofing or replay attack with a cheap SDR can have a potentially devastating effect on improperly secured satellite systems. Through robust high-sample-rate device fingerprinting, we aim at defeating these attacks by making it impossible to forge the fingerprint on spoofed signals using only cheap COTS radio hardware. In doing so, we increase the budget of attacks such that they can no longer be carried out by lower-budget attackers.

Of course, we do not expect fingerprinting techniques to be able to prevent all attacks; such a goal is unrealistic for this scope, requiring robust cryptography. As discussed in Section 1, there are a large number of satellites with outdated or no cryptography and no capability for in-flight patching, and we argue that it is still crucial to secure these satellites as well as possible despite this, so they can still be used to serve customers and for the advancement of science. We discuss in more detail in Section 5 how the threat model changes with the introduction of robust signal fingerprinting, addressing remaining concerns and potential avenues for future research.

We also do not consider attacks from within a compromised system, in which compromised satellites are communicating with other devices in the network. This further means that spoofing attacks launched from satellites within the constellation that our system protects are out of scope for this work. Existing systems such as PAST-AI [26] are already well-suited to tackle this type of attacker. However, as these existing systems are based on low-sample rate data, they cannot protect against SDR-based attacks – the focus of this work.

We are primarily concerned with attackers overshadowing messages on the downlink – whilst attacks on the uplink are possible, they have not been extensively explored due to the greater hardware cost of a suitable amplifier and directional dish. Furthermore, device fingerprinting on the space segment is currently infeasible, requiring large amounts of computational power, and cannot be carried out aboard the legacy satellites with which we are primarily concerned. Further work may consider fingerprinting the uplink of legacy satellite systems by capturing signals in-transit, but this is a very limited use case and is out of scope for this research.

4 System Design

Our system comprises two primary components: data collection and the SatIQ machine learning fingerprinting system. A representation of the end-to-end system can be seen in Figure 1 – messages are collected by an SDR and decoded into message contents, and the raw samples are ran through an encoder network (explained below), and compared against known example messages from the

[1] For example, a HackRF One can be used as the radio (340 USD, Adafruit), with a suitable power amplifier and passive antenna.

[2] The authors approached a reputable manufacturer, and received a quote for approximately 125 000 USD at academic institution rates. We therefore conservatively assume that in the best case this hardware would cost no less than 60000 USD.

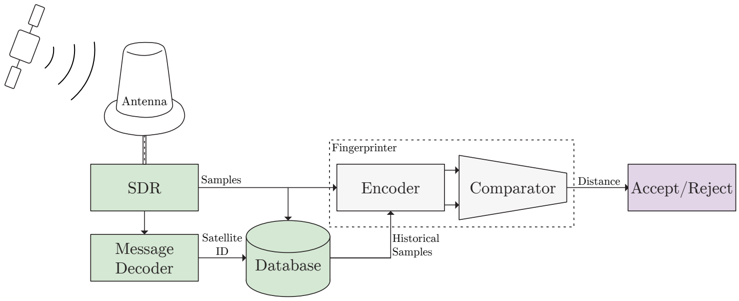


Fig. 1. An overview of the end-to-end fingerprinting process used by SatIQ. Satellite signals are received, decoded, processed into fingerprints, and compared to historical fingerprints to determine a distance metric, which is used to accept or reject the message.

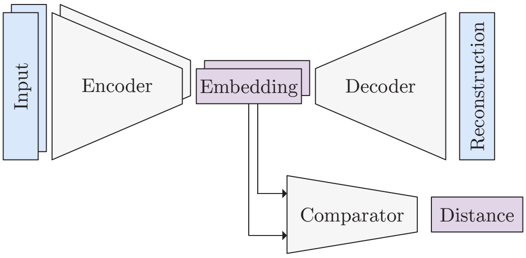


Fig. 2. An overview of the Siamese autoencoder architecture used in SatIQ's fingerprinting model. Two inputs are passed into the encoder with identical weights, and the encodings are compared using the comparator (angular distance) to generate a distance metric.

same transmitter. This produces a distance metric – low distances indicate the message is likely to be legitimate. This is used to accept or reject the message.

All code and datasets will be made openly available on publication. The source code for the original SatIQ models can be found at https://github.com/ssloxford/SatIQ, and the dataset and model weights at https://zenodo.org/record/8220494 and https://zenodo.org/record/8298532 respectively [1].

4.1 Design Decisions

Our fingerprinting model, illustrated in Figure 2, uses an autoencoder combined with a Siamese model to compare two input waveforms and produce a distance metric between the two inputs. In the following, we describe autoencoders and Siamese networks in general, and we explain why we chose them as the basis for our fingerprinting system.

Autoencoders. An autoencoder is a type of neural network which is used to learn an efficient encoding of data. This is achieved by simultaneously training an *encoder* and *decoder*, validating the accuracy of the encoding by comparing the output of the decoder to the input (reconstruction accuracy). The output of the encoder is restricted in size, thus forcing data to pass through a bottleneck at this portion of the network. This forces the model to discard less important information, producing an efficient encoding. This technique is particularly useful for dimensionality reduction, since the encoder produces an embedding of the input in a significantly lower-dimensional space. The layout of an autoencoder can be seen in Figure 3. Prior works have shown autoencoders to be effective in fingerprinting contexts, both in terrestrial systems [36, 37] and with satellites specifically [26, 38].

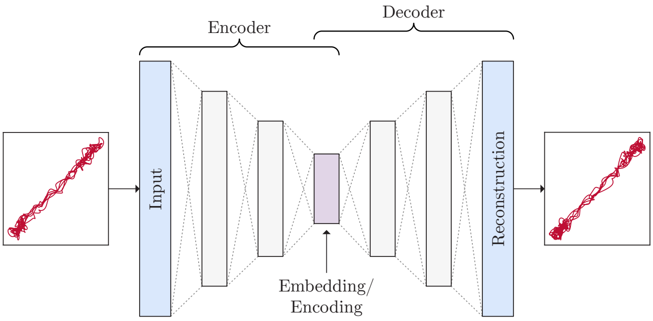


Fig. 3. The layout of an autoencoder. Input is passed through an encoder to produce a low-dimensional encoding, and then through a decoder to produce output of the same dimension as the input. The model is trained to reconstruct the input as best as possible.

Siamese Neural Networks. Siamese neural networks are designed to be effective for one-shot classification. To this end, they generate a similarity score between two inputs [39]. This is achieved by passing each input through the same "encoder" network to generate an embedding of the inputs in some feature space, followed by a comparison function to generate a distance metric in the feature space – the lower the distance, the more similar the samples are. The weights of the encoder network are shared between the two inputs.

We chose this approach due to some of its advantages over a simple classifier, particularly in the context of fingerprinting:

- The number of classes is not fixed – new classes can be introduced after training by comparing to examples from the new class.
- The one-shot (or in some cases few-shot) nature of the model means a new class can be identified using only a very small number of examples.
- The distance threshold can be raised to increase the acceptance rate of legitimate messages at the cost of increased false positives (or vice versa), granting fine-grained control of the level of protection granted by fingerprinting.

Past work has shown these models to be effective in a wide range of use cases, particularly in classification problems with huge numbers of classes such as malware detection or gait recognition [40, 41]. The architecture has also been demonstrated to be successful at detecting spoofing and replay attacks on other systems, such as face recognition and voice biometrics [42, 43]. Siamese networks have also seen use in radio systems, shown to be effective in areas such as automatic modulation classification [44]. Finally, some research has shown promise in using Siamese networks for fingerprinting radio transmitters [45, 46].

Our work builds upon these in a number of key aspects. Firstly, we deal with a more difficult scenario than the majority of terrestrial fingerprinting cases – the great distance of satellite transmitters introduces large amounts of atmospheric noise and multipath distortion, which dwarf the hardware impairments on the signal. Secondly, we consider in particular the security implications of radio fingerprinting, assessing the level of protection against spoofing and replay attacks granted by our approach, and the expected budget required to circumvent these techniques. Finally, we use an autoencoder alongside the Siamese network to provide encodings that better capture meaningful features in the input – prior work has shown this to be effective in areas such as signature verification, but to our knowledge we are the first to apply this architecture to radio fingerprinting [47].

In a non-adversarial classification setting, simple classifiers typically exhibit higher performance than Siamese networks, since they partition the entire input space into fixed categories. However,

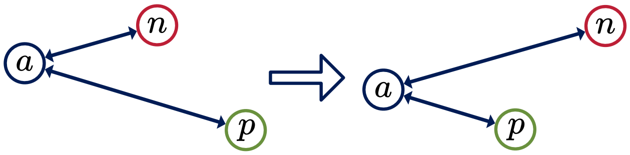


Fig. 4. An illustration of the triplet loss function. This function takes an anchor a, a positive sample p of the same class as the anchor, and a negative sample n of a different class. Optimizing this loss function minimizes the distance between the anchor and positive samples, and maximizes the distance between the anchor and negative samples.

this is less effective in adversarial settings, since malicious transmitters will still be given a legitimate label. Our results in Section 5 show that Siamese architectures are particularly effective at detecting replay attacks, even with the levels of noise observed in satellite signals.

Siamese models can also require more data during training, since they must learn a distance metric that works for all transmitters. Furthermore, independently of the model architecture, greater levels of noise mean more training data is required. Our results show that our dataset was sufficient for this work, but more data may yield further performance improvements.

4.2 Fingerprinting Model

The Siamese network of SatIQ uses the encoder portion of the autoencoder to produce embeddings of two inputs, which are then compared to one another using an angular distance function. A triplet loss term encourages the model to produce embeddings that are close to one another for messages from the same transmitter, and different for messages from different transmitters:

$$d(u,v) = 1 - \frac{u \cdot v}{\|u\|\|v\|}$$
$$L(a,p,n) = \max\left(d(a,p) - d(a,n) + \alpha, 0\right).$$

This takes an anchor a, a positive example p belonging to the same class as the anchor, and a negative example n of a different class, and encourages the distance from a to p to be less than a to n. A margin α ensures that effort is not wasted on optimizing triplets for which this is already the case. This is illustrated in Figure 4.

Convolutional layers are used in the encoder to enable identification of position-invariant features and reduce the overall size of the model. For the comparator, we compute the angular distance between the embeddings – this tends to work better than L2 distance for high-dimensional data. A diagram of the fingerprinting model architecture used in this article can be seen in Figure 5.

The inclusion of convolutional layers reduces the model's size, and allows it to extract position-invariant features from the waveform. We also use separate layers for the in-phase (I) and quadrature (Q) portions of the signal – although the components are tightly coupled to one another, we find that they express different features and the model is able to perform better when the two are separated. Following the convolutional layers, we concatenate the outputs and flatten, before using a fully connected layer to reduce the output to the correct size. The decoder uses a very similar architecture to the encoder, composed of alternating upsampling and convolutional layers.

4.3 Data Collection

For a fingerprinting model to be effective, a good dataset is essential. Community projects such as *SatNOGS* (an open-source network of ground stations [48]) provide data from a wide range of satellites. However, in this article, we collect our own data for a couple of important reasons: firstly,

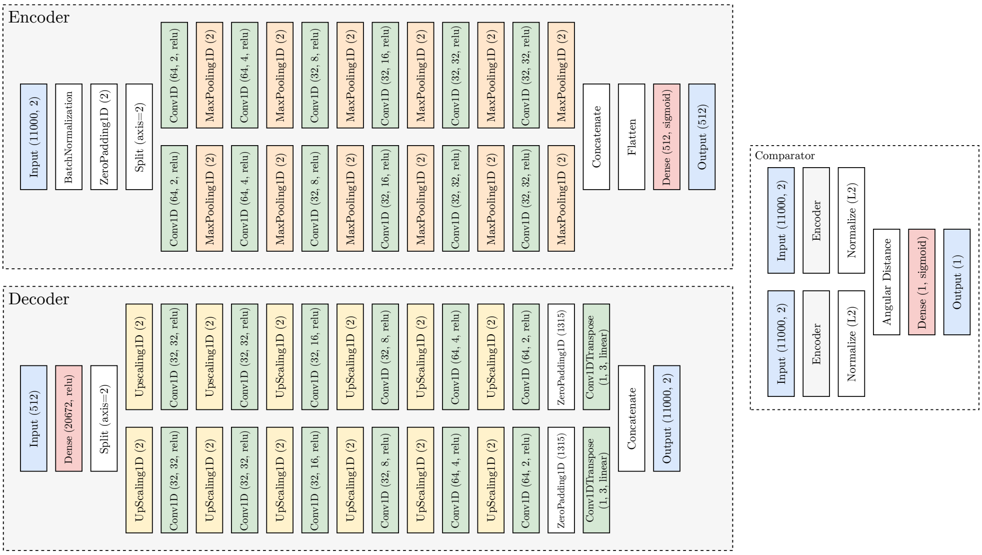


Fig. 5. The layers of the Siamese neural network used in SatIQ. The encoder uses separate convolutional and max-pooling layers for the *I* and *Q* portions of the signal, before producing a final embedding using a dense layer. Similarly, the decoder uses separate convolutional and upsampling layers. The comparator uses two copies of the encoder with identical weights, computing a difference score between the outputs.

collecting our own data enables us to capture at a significantly higher sample rate than existing datasets, providing a good foundation for a fingerprinting system. We also capture signals from a specific constellation rather than a collection of individual satellites; by doing this, we ensure that the signal modulation and protocol does not vary between messages. We can therefore guarantee that the message header is always the same between messages, so differences in the captured waveform will be caused only by hardware differences and channel noise – the contents are always identical at the bit level. This gives us a consistent baseline upon which a fingerprinting model can be built.

We focus on the *Iridium* constellation, used in telecommunications. This constellation has a number of useful properties:

- The constellation contains a large number of satellites (66, each with 48 transmitters [33]), providing sufficient variety within the dataset;
- the transmitter hardware on each satellite is identical, so a fingerprinter will need to distinguish satellites only through differences introduced at time of manufacture, rather than distinguishing between entirely different components;
- the communication protocols are known and well documented [33], so no reverse engineering is required;
- downlinked transmissions can be received using cheap and widely available COTS hardware, as it transmits at a frequency below 6 GHz.

Other constellations are available; for instance, the *ORBCOMM* and *Globalstar* constellations have open-source decoders available but fewer satellites, and *Starlink* and Planet's *Dove* constellations have more satellites but operate at high frequencies and (at the time of writing) have no open-source decoder available. Some of these constellations may be useful for future evaluations of the versatility of SatIQ. Furthermore, existing research in fingerprinting also uses Iridium satellites [26], providing evidence that some fingerprinting is plausible in this context and providing a baseline to

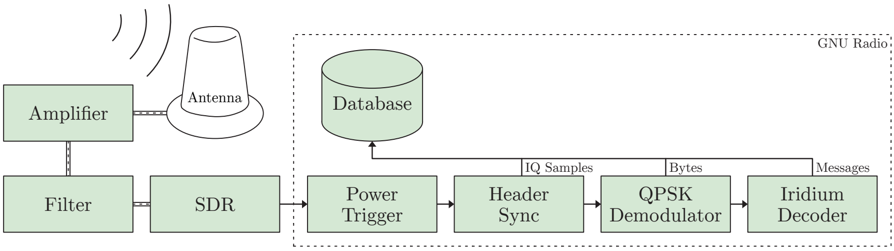


Fig. 6. An overview of the data collection system. Signals are captured by the SDR and sent to GNURadio for processing. Raw IQ samples, demodulated bytes, and fully decoded message data are all sent to a database.

which fingerprinting systems can be compared. We argue that testing on Iridium is sufficient to demonstrate and validate the effectiveness of SatIQ, as the properties of the signal used by SatIQ are not constellation-specific.

For the primary data collection setup, we use the following hardware[3]:

- Iridium Beam active antenna (1,245 USD, Beam Communications)
- Mini-Circuits ZKL-33ULN-S+ low-noise amplifier (209 USD, Mini-Circuits)
- NooElec DC block (20 USD, Amazon)
- Mini-Circuits VBF-1560+, 1500–1620 MHz band pass filter (44 USD, Mini-Circuits)
- USRP N210 SDR (3,354 USD, Ettus Research)
- UBX 40 USRP daughterboard (1,732 USD, Ettus Research)

The SDR is connected to a computer running *GNU Radio*, a software library to aid digital signal processing. We use components from the *gr-iridium* library, created and maintained by the Chaos Computer Club München e.V., to demodulate and decode messages [49]. Figure 6 illustrates our full data collection and processing pipeline.

Iridium downlink messages have a number of different message types, one of which is the ***Iridium Ring Alert*** (**IRA**) message – this is the only type of message we collect. These messages contain diagnostic information about the satellite, including the satellite ID, beam ID (identifier of the current transmitter), position, and altitude. They also contain the anonymous identifiers of subscribers currently receiving incoming paging calls. The messages are openly broadcast and do not contain any personally identifiable information, so we can decode and collect them without raising any ethical concerns.

We save the raw IQ samples from the message headers in a database, alongside demodulated bytes and decoded message contents. This gives us a consistent dataset of raw message headers which can be used for fingerprinting, using the decoded messages to label the data. All data was collected at 25 MS/s, giving us an oversampling rate of 1000 (Iridium messages have a symbol rate of 25 000 symbols per second). This allows us to capture the high-frequency features characteristic to a transmitter for effective fingerprinting.

Our data collection was spread across multiple locations, enabling analysis of the effect of receiver hardware and physical location on the performance of a trained fingerprinter. The bulk of the data was collected on the roof of a building in Oxford, UK, with additional hardware in Würzburg, Germany, and Thun, Switzerland. Each had an antenna located on the roof of a building with a good view of the sky in all directions. Table 2 gives some information about each location's data collection hardware, and the dataset collected – note that in Germany a cheaper patch antenna was used, resulting in fewer messages per day and lower signal quality. Additionally, Figure 7

[3] Prices are as recorded on 2023-04-21, and may not reflect current prices.

Table 2. Hardware and Dataset Information for the Data Collected at Each Location

	Oxford, UK	Thun, Switzerland	Würzburg, Germany
SDR	USRP N210	USRP N210	USRP B205mini-i
Antenna	Beam RST740	Comant CI-490-490	Tallysman TW4600
Amplifier	Mini-Circuits ZKL-33ULN-S+	Qorvo SPF5189Z	Nooelec LaNA
Filter	Mini-Circuits VBF-1560+	Mini-Circuits VBF-1560+	Mini-Circuits VBF-1560+
Start Date	2023-05-11	2024-08-09	2023-03-14
End Date	2024-04-29	2024-09-01	2023-04-13
Days Active	248	20	40
Messages Collected	9 695 000	450 000	145 000

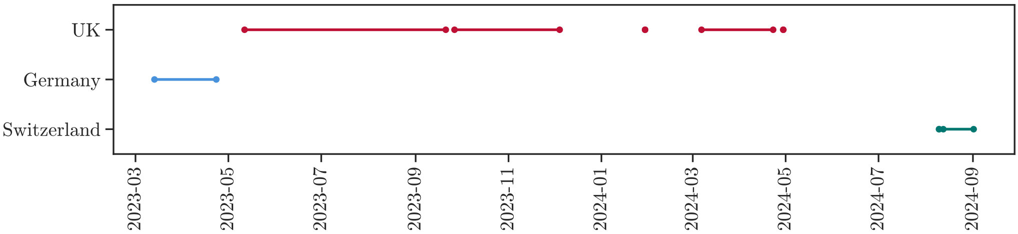


Fig. 7. Timeline of data collection in each location.

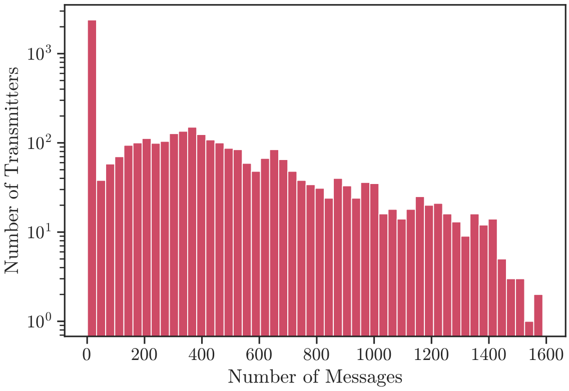


Fig. 8. The distribution of the number of Iridium messages received per transmitter.

illustrates the number of messages collected over time. This data extends the original dataset of 1 705 202 messages collected over 40 days used to train the original SatIQ model [1].

For each transmitter in the UK dataset, as many as 1588 messages were received, with a mean of 260 messages per transmitter. The distribution of message count per transmitter can be seen in Figure 8.

An example of the collected data can be seen in Figure 9. We can see the signal encodes 8 QPSK symbols, corresponding to the bit sequence for the Iridium message header: `11 00 00 11 11 00 11 00`. However, unlike a constellation plot at 1 sample per symbol, we can see the movement between the two symbols and the impairment on the signal. It is clear that there is significant impairment; this is likely caused by a combination of channel noise, multipath distortion, and hardware characteristics of the transmitter. Our goal is to isolate the last of these, ignoring the channel noise and other characteristics – these will be the same between transmitters, and not useful in this context.

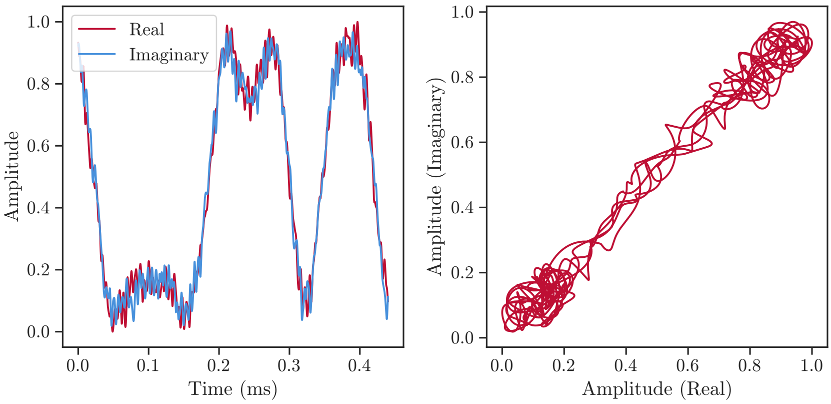


Fig. 9. A message header received from an Iridium satellite, shown in the time domain (left) and as a constellation plot (right).

4.3.1 Data Preprocessing. We opt to use a minimal amount of preprocessing to avoid destroying data which might be useful for fingerprinting. On top of the band-pass filtering and phase synchronization performed by *gr-iridium*, we scale each waveform in the dataset so values range between -1-i and 1+i. This removes amplitude as a factor the model needs to learn to adjust for, and makes visualization easier. We also remove all messages which do not decode as valid IRA messages – although this removes a large number of messages, it ensures that all data is labeled and eliminates the noisiest messages which do not properly decode. This leaves us with messages that are the most likely to contain meaningful identifying factors. In Section 5.2 we look into this further, identifying environmental and signal features that affect performance.

Finally, we process the data into "TFRecord" files – this format is optimized for use in TensorFlow, storing data as raw protocol buffers. This enables us to read data directly from disk as needed, minimizing RAM usage with only a small buffer to reduce read latency. This is vital for a dataset this large; even on machines with significant amounts of RAM (≥500 GB) the dataset cannot be loaded into RAM in its entirety.

4.3.2 Dataset Construction. After data preprocessing, the entire dataset is split up into files of 5000 entries each, split up by location and organized by date.[4] This is split into training, validation, and testing datasets as follows: for each date, the first file (i.e., the first 5000 entries) is used for testing, the second file for validation, and the remaining files are used for training. This resulted in a training:validation:testing split of $74:13:13$ for our dataset. This ensures we have sufficient data for validation and testing without significantly reducing the size of the training dataset, whilst maintaining an even distribution of timestamps across all three datasets.

Since the triplet loss function is used, the batch generator is configured to produce batches of inputs with 32 messages (8 batches of 4 messages each from the same transmitter). This ensures the loss function has a large number of triplets that can be selected from each batch.

4.4 Model Training

Model training was performed on a machine with the following hardware:

- Intel Xeon Gold 6134 CPU (32 threads, 3.20 GHz)

[4] Previous versions of the dataset shuffled the data at this point, but this proved impossible once the dataset grew sufficiently large. Instead, the data is shuffled by the TensorFlow data pipeline as needed.

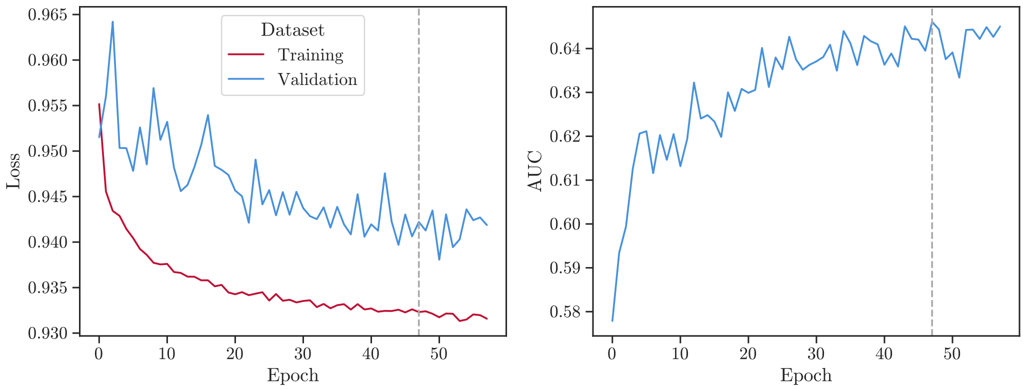


Fig. 10. Training and validation loss curves for one training run of SatIQ, and the corresponding validation AUC. The dashed line indicates the epoch with the best-performing AUC.

- Nvidia TITAN V GPU (1,455 MHz, 12 GB VRAM)
- 512 GB DDR4 RAM

On this hardware, 2 model configurations can be trained simultaneously.[5] All models are trained for up to 200 epochs, with early stopping if no improvement is seen within 10 epochs. The training and validation loss curves for one training run can be seen in Figure 10, demonstrating how early stopping prevents overfitting: as soon as validation AUC stops increasing, training is stopped and the model is rolled back to its best-performing weights. Training on the full dataset took approximately 10 days, with most other training runs taking closer to 24 hours – each epoch runs through the full dataset, so larger datasets result in longer training runs.

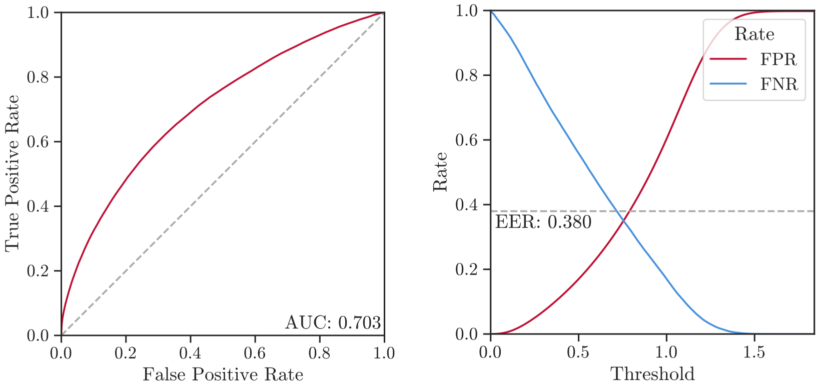
Running a trained model takes significantly fewer resources, taking a fraction of a second to verify messages – this makes it possible to run SatIQ in real-time, verifying messages as they arrive. Furthermore, much less RAM and disk space is required, as the training dataset is not needed. Hardware requirements are discussed further in Section 5.8.

4.5 Model Optimization

We start by assessing the base performance of the SatIQ model, trained on the full dataset from a single location and tested on the problem of differentiating satellites from different transmitters in the dataset. Note that this differs from the most likely attack case, in which a ground-based attacker transmits messages via an SDR. However, it does enable us to get a good idea of initial performance and fine-tune the system, and enables us to work with a large dataset when performing analysis later on. It is also much harder to distinguish between nearly identical transmitters than to distinguish a ground-based SDR, so we can be confident that good performance in this evaluation will likely translate to good performance in a more realistic attack case. We back this up in Section 5.6 with an evaluation of SatIQ under simple replay attacks.

Figure 11 shows the base performance of SatIQ. In this section, models are trained and tested on the problem of differentiating satellites in the dataset, in order to achieve better performance later on when identifying adversaries in an attack scenario. Results in this section are primarily to assess relative performance and fine-tune the system. When assessing performance, we focus on two key metrics:

[5] The primary constraint is VRAM, with the model requiring approximately 6 GB to train. The vast majority of the system's RAM is not used, since the data is streamed from files stored on disk.



(a) ROC curve

(b) FPR against FNR

Fig. 11. The performance of the base SatIQ model, evaluated by comparing messages from different pairs of satellites without analyzing the attack case.

- *Equal Error Rate (EER):* the error rate when the **false positive rate** (**FPR**) and **false negative rate** (**FNR**) are equal.
- *Area Under Curve (AUC):* the area under the **Receiver Operating Characteristic** (**ROC**) curve, obtained by plotting the **true positive rate** (**TPR**) against the FPR. This can be intuitively thought of as the probability that the system can distinguish between two inputs of different classes [50].

To compute each of these, we vary the distance threshold below which we accept two messages as being from the same transmitter. By raising this threshold, we accept a greater number of legitimate messages, but open the system up to easier spoofing attacks. Conversely, by lowering the threshold the system is made more secure by rejecting more attacker messages, but legitimate messages are more likely to be erroneously rejected.

Our base model has an AUC of 0.703 and EER of 0.351. This is sufficient performance to demonstrate the feasibility of our techniques, particularly in the more difficult case of distinguishing satellite transmitters with identical hardware. In Section 5 we go on to demonstrate that performance is significantly better in a replay attack scenario, confirming its usefulness in a security context. For the rest of this section we continue to assess performance on the original (non-attacked) dataset.

4.5.1 Multiple Anchors. To improve performance over the base system, we can compare each incoming message to a larger number of "anchors" (known messages from that transmitter), taking the mean distance between the message and each anchor in the embedding space. The results of this are shown in Figure 12. By taking 32 anchors for each incoming message, we can achieve an EER of 0.277 and AUC of 0.795 – a significant improvement! In practice, this can be implemented by saving a larger set of messages from each known transmitter, or by comparing multiple consecutive messages to the same set of anchors. Both of these techniques are practical – our observations suggest that during an Iridium phone call or web connection approximately 11 packets are exchanged per second, so an attacker will need to spoof many packets to have a meaningful impact on the victim. Such an attack would certainly be picked up by SatIQ, even if multiple consecutive messages are compared. We explore attack scenarios further in Section 5.

4.5.2 Model Variations. We also tested a number of variations on the model architecture and hyperparameters. To save time and enable further iteration, these were all tested on a reduced

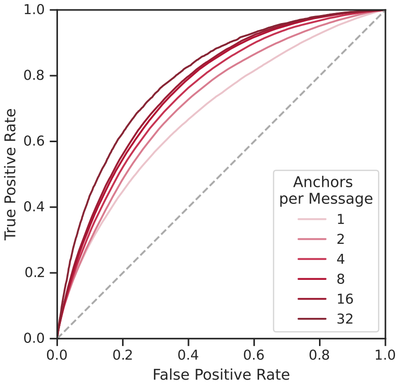


Fig. 12. ROC curves as we compare each incoming message to a larger number of anchors, taking the mean distance.

Table 3. Performance of SatIQ models (ROC AUC, Equal Error Rate) Trained on 16 Days of Data, as we Adjust the "Margin" Term on the Triplet Loss Parameter

Margin	AUC	EER
2.000000	0.673596	0.377031
1.000000	0.673623	0.377184
0.500000	0.676968	0.373603
0.100000	0.677803	0.372970
0.050000	0.668686	0.379689
0.010000	0.664890	0.383152

portion of the dataset with only 16 days of data, and compared to an equivalent base model trained on the same dataset. The base model has an AUC of 0.674 on the file used for testing (5000 messages).

We first trained a model with the autoencoder removed. This model achieves an AUC of 0.676 on the same dataset – a very small change, well within random variation. We may therefore be able to save training time by removing the autoencoder, although this would remove the ability to reconstruct message headers from their fingerprints. This capability is not used in this work, but may be useful if adversarial ML techniques are used. We discuss this in Section 6.

We also tested the effect of switching the autoencoder for a "variational autoencoder" [51] – in these models, the encoder maps the input to a probabilistic latent space (in this case, a multivariate Gaussian distribution) instead of mapping directly to the latent space. This provides benefits in avoiding overfitting and can sometimes produce more meaningful features in the latent space. However, our trained model on this architecture had an AUC of 0.499, essentially equivalent to random guessing. Variational autoencoders are noted to be quite sensitive to certain hyperparameters, so it may be possible to achieve good performance on this architecture via tuning.

Finally, we adjust the "margin" of the triplet loss parameter used in model training. By increasing this, the separation of the anchor and negative samples during training can be increased further before it stops yielding further gains, and vice versa. Table 3 shows the performance of models trained on different margin values – we can see that changing the margin has nearly no effect, so we do not need to worry about fine-tuning this parameter to maximize performance.

Further optimizations to the model are likely possible – there are a wealth of machine learning techniques not explored in this article, and model/hyperparameter optimization is a known difficult problem. We have demonstrated that SatIQ is effective for fingerprinting satellites, and operators may choose to fine-tune the model further to achieve greater performance, either on the dataset provided or on a new constellation.

5 Evaluation

In the previous section, we designed and trained the SatIQ system to distinguish between different legitimate satellite transmitters. In this section we evaluate the key properties that enable SatIQ to be deployed in the real world as a component of a satellite authentication system. We evalaute the system's stability over time, look at how performance correlates with weather and signal quality factors, and measure extensibility to new satellite transmitters and receiver configurations. We next evaluate the robustness of SatIQ against replay attacks over a wire, demonstrating a dramatic increase in effectiveness at distinguishing an attacker-controlled SDR from legitimate transmitters. Finally, we discuss practical questions surrounding a real-world deployment of SatIQ, including hardware requirements, handling rejected messages, and keeping anchors up to date.

5.1 Time Stability

We start by evaluating the stability of SatIQ over time. Existing works in radio fingerprinting observe a decrease in performance when there is a time gap between the training and testing data [52, 53]. This is thought to be caused by changes in the conditions of the wireless channel over time. Our dataset is sufficiently large that we can easily establish the extent of this phenomenon, and explore methods of counteracting it without compromising the security of the system.

We assess the effect in two ways. Firstly, we train a number of models on increasing portions of the dataset, starting with just 1 day of data and going up in multiples of 2, up to 64 days followed by the full dataset. We also ensure the start time of each of these dataset slices is 1 month after the start of data collection, so we can look backward in time as well as forwards. Next, we test each model by computing its AUC and EER on 24-hour slices of the dataset, taking the anchors from the same dataset as the testing samples. This enables us to see how, or indeed if, performance drops off over time as the collected data becomes newer than the training data.

Secondly, we assess the effect of using older anchors in the testing data. We use the same trained models as above, and the same testing methodology, but instead of taking the anchors and testing samples from the same 24-hour slice, we offset the anchor by a number of days (ranging from 1 to 32 days). This enables us to see how accuracy drops off if anchors are not replaced, and establish how frequently they must be updated.

The results of the first experiment can be seen in Figure 13. We can see that overall performance improves with dataset size, and does not drop off with newer testing data when fresh anchors are used. This is a promising result, indicating that it will not be necessary to retrain the model repeatedly over time; it is sufficient to train the model once and then keep updating the anchors. We also see that larger datasets start to yield diminishing performance returns – this is unsurprising, as all systems demonstrate this behavior, but it's promising that this occurs with only weeks to months of training data. We can therefore be confident that a system trained on this amount of data will perform well and maintain its accuracy as time goes on.

The results of the second experiment are shown in Figure 14, for a subset of the training dataset sizes – aggregated metrics for all dataset sizes are shown in Figure 15. We can see that when SatIQ is trained on only 1 day of data there is a significant drop in performance the moment anchors deviate from the testing data – with just 1 day of difference the average AUC drops dramatically from 0.667 to 0.515. When training on larger datasets the drop in performance is still present, but less pronounced: on 64 days of data the AUC drops from 0.695 to 0.568. We recommend for Iridium

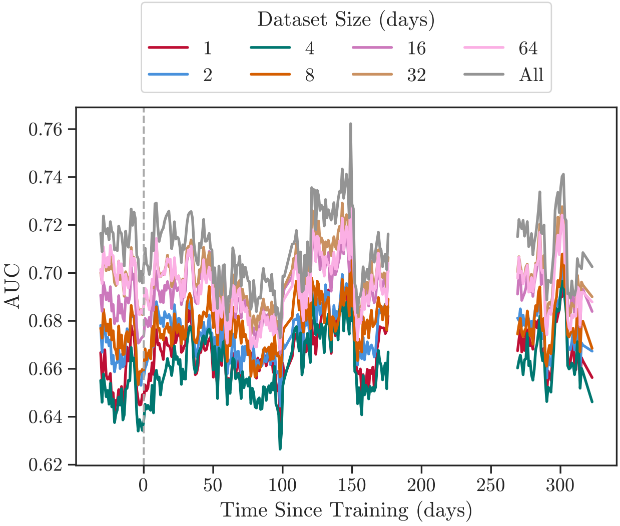


Fig. 13. Graph showing the performance of SatIQ on incoming messages over time, relative to the start of the training data, for different training dataset sizes. The gap is caused by a break in the training data.

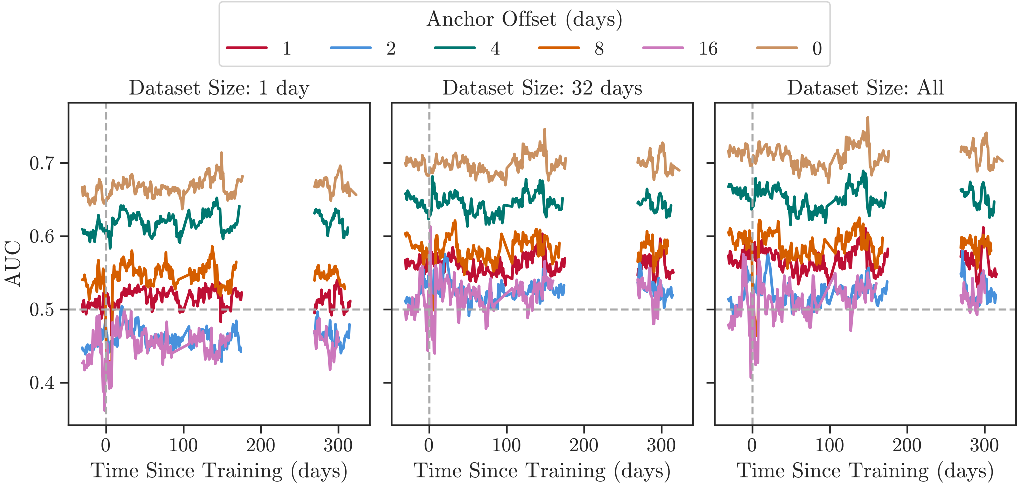
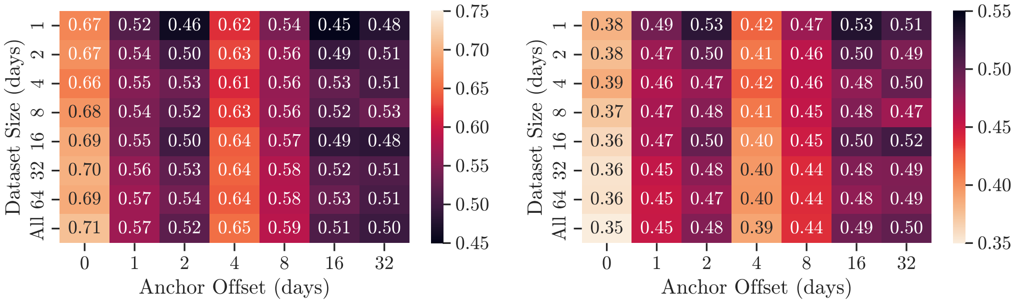


Fig. 14. Graph showing the performance of SatIQ on incoming messages over time, relative to the start of the training data, with anchors collected at an offset from the testing messages. The gap is caused by a break in the training data.

that data collection should run for a minimum of 32 days, and that anchors are refreshed every 8 days or fewer to ensure freshness and maximize performance.

Interestingly, there is a spike in performance at an offset of 4 days. Figure 16 shows a heatmap of how many transmitter IDs overlap for different offset values – we can see there is a non-uniform distribution, with some IDs disappearing for a while before reappearing. It is likely that this is largely due to the sequential nature in which data was collected – each 1-day slice of the testing data represents only a small amount of time, with approximately 24-hour spacing, and the same periodicity may not be present in the full dataset.



(a) ROC AUC

(b) EER

Fig. 15. Average performance (AUC and EER) of models trained on each dataset size, as the time difference between the anchors and testing samples is increased.

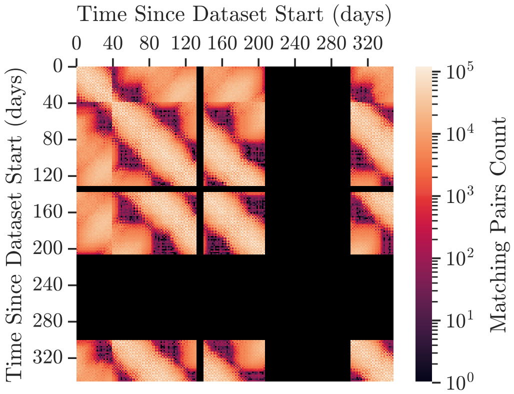


Fig. 16. Heatmap showing the number of pairs of messages with the same transmitter ID in each day of the testing dataset. Black sections represent gaps in the collected data.

Note that refreshing anchors presents some security issues of its own: if an attacker manages to insert their own messages into the dataset as anchors in place of legitimate messages, then the entire authentication system may be compromised, with legitimate messages rejected and the attacker's messages accepted. Caution must be exercised to ensure anchors are securely refreshed. This could be achieved by periodically verifying messages through other means – for instance, by precisely measuring angle of arrival, or making use of other physical measurements.

5.2 Correlations

Next we evaluate the performance of the fingerprinting model compared to other factors. We look at characteristics of the waveform: noise level and magnitude reported by the Iridium decoder, amplitude of the waveform, confidence of the decoder, and so on. We also look at weather data and distance from the satellite to the receiver.[6] For each of the tested variables, we filter the dataset into slices comprising 10% of the testing dataset, sorted by that variable. We then compute the model's statistics as normal to see results.

Figure 17 displays the results of this analysis. We can see that for some weather properties (cloud cover, solar radiation, temperature) there does not appear to be any correlation between the

[6] Weather data provided by Visual Crossing [54].

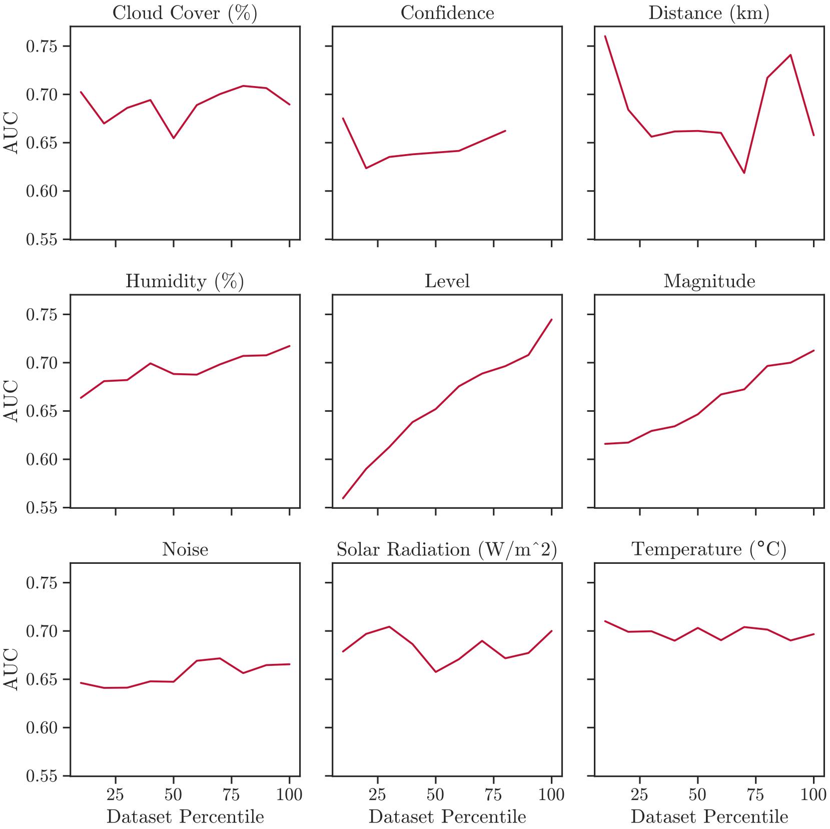


Fig. 17. Performance of the SatIQ model on slices of 10% of the dataset, sorted by various properties output by the decoder and data collection system.

property and the performance of SatIQ. However, we do see a strong correlation for the "level" and "magnitude" properties output by the decoder, which roughly correspond to signal strength. This is unsurprising – cleaner and louder signals are easier for the fingerprinter to identify, and contain more identifying information. We also see a weak negative correlation between distance and performance – this is also unsurprising, as messages sent from satellites closer to the receiver will have been subject to less atmospheric attenuation.

To see if any weather effects emerged over time, we also compute the average of each weather-related variable for each 24-hour slice of the testing dataset. This can be seen in Figure 18 – even looking at the entire dataset, no clear correlations emerge. This indicates that weather does not have a significant effect on the performance of SatIQ, and suggests that weather features are unlikely to be incorporated into the transmitter fingerprint.

Some of the tested properties could be used to filter the dataset to improve performance, but operators must take care when performing this kind of filtering. If only a small fraction of messages are accepted, a denial of service attack can be performed on the system by affecting measured factors – for instance, by adding noise to the channel. Furthermore, if the data is filtered using factors affected by weather, a deployed system might naturally go through an extended period with no suitable messages. Not only would this prevent the satellite from being authenticated, but if anchors cannot be refreshed frequently enough then authentication will be impacted going forward.

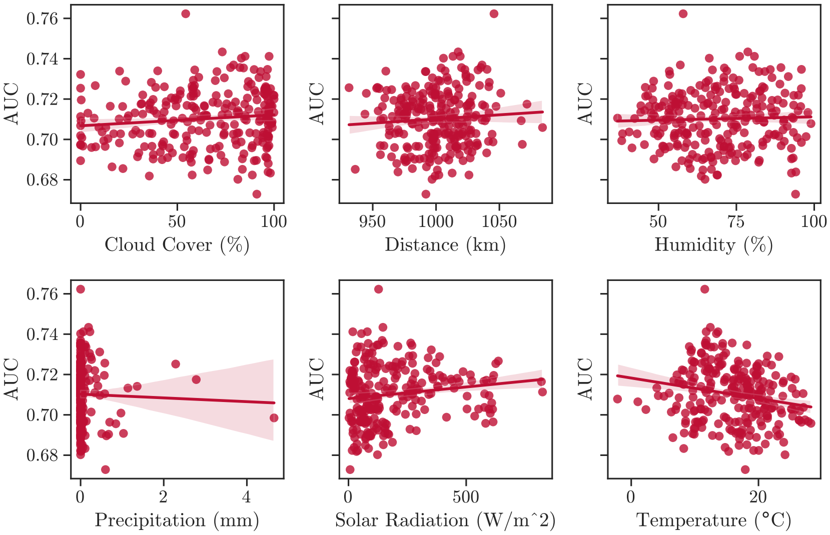


Fig. 18. Performance of the SatIQ model on 1-day slices of the dataset, plotted against the physical conditions during data collection at that time.

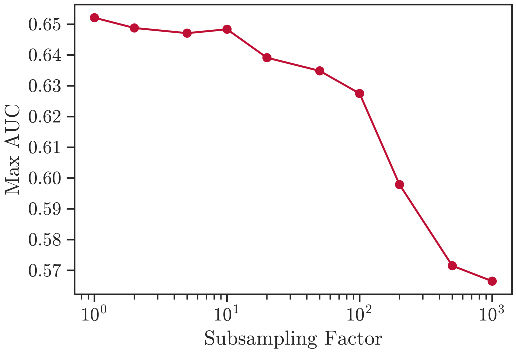


Fig. 19. Performance of the SatIQ model trained on subsampled data.

Nevertheless, these factors may be useful alongside the confidence of the SatIQ fingerprinter to gain a good understanding of the signal state.

5.3 Subsampling

We also look at models trained on data that has been subsampled from the original 25 MS/s, to see to what extent it is necessary to operate at such a high sample rate, and to verify that system performance is not held back via the inclusion of unnecessary data. For this evaluation, we trained models on the reduced dataset of 16 days of data, and subsampled to a factor of N by setting all but every Nth sample to 0, for different values of N. We then take the highest validation AUC reached during training. We can see from the results in Figure 19 that performance is highest with no subsampling (i.e., a factor of 1), remains somewhat stable until a subsampling factor of 10, and begins to drop quite quickly above this value. This suggests that SatIQ may be effective when using data collected at a sample rate as low as 2.5 MS/s, without excessively limiting performance of the trained model.

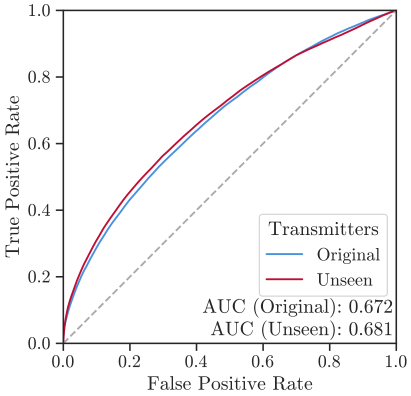


Fig. 20. ROC curve showing the system's performance on transmitters it has never seen before.

5.4 Extensibility

Next we look at how well SatIQ performs on transmitters it has never seen before. This is of particular importance in satellite constellations, where satellites may need to be replaced at any time, and we want to minimize time and effort spent retraining the model. To test the extensibility of SatIQ, we trained the fingerprinting model on a dataset with some of the transmitters in the testing dataset removed. Specifically, we sorted the transmitter IDs in the testing files by how frequently they appeared, and removed every other ID from training. This resulted in the removal of 422 transmitters. To minimize training time, we once again trained the model on a reduced dataset with only 16 days of data.

The results of this analysis are shown in Figure 20. As expected, the base performance roughly matches the testing performance of our original model trained on this dataset, with an AUC of 0.672. When tested on the transmitters removed from the training data, the performance actually increases slightly to an AUC of 0.681! This is far better than existing classifier-based systems, which require retraining each time a satellite is launched or replaced – in modern systems this can be very frequent, with Starlink launching 1984 satellites across 63 launches in 2024 [55]. From this result we can be confident that SatIQ can easily be extended to new transmitters without requiring any retraining, resulting in a system that can continue to be used indefinitely as transmitters are added and replaced.

5.5 Transferability

We have already shown that SatIQ can authenticate transmitters it has never seen before, but we also need to ensure that it can be used across different receiver configurations. All our training and evaluation so far has looked at data collected from a single source, but this does not provide a useful system if the data collection hardware differs from that of the deployed system – even if the hardware is identical, the receiver may be imparting its own fingerprint, or additional factors like multipath distortion may be affecting the signal. We therefore perform analyses on the additional data collected in Germany and Switzerland, in addition to the main dataset collected in the UK.

We start by taking our base model (trained on the UK data only) and assess its performance on datasets from the other two locations – we evaluate a model trained on the full dataset, and a model trained on a reduced dataset comparable in size to the data collected in Switzerland.[7] Next, we train

[7] The reduced dataset contained only 8 days of collected data.

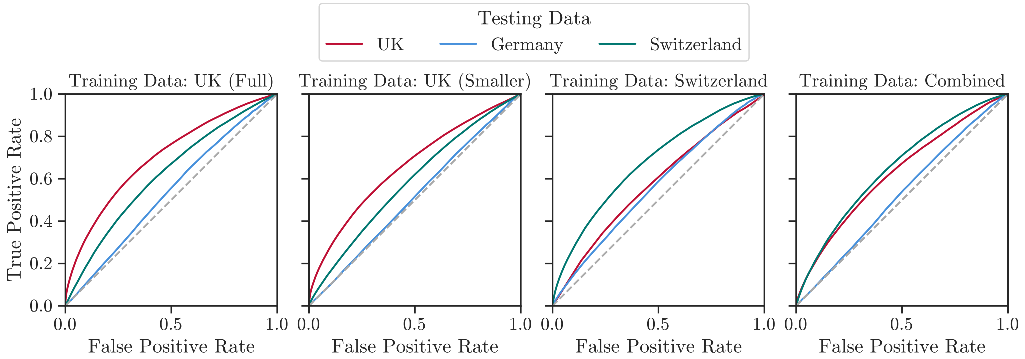


Fig. 21. ROC curves for models trained on datasets from each location, tested against different locations.

Table 4. Performance of Models Trained on Datasets from Each Location When Tested Against the Other Locations

Training Data	Testing Data	AUC	EER
UK (Full)	UK	0.703	0.351
	Germany	0.536	0.473
	Switzerland	0.620	0.410
UK (Smaller)	UK	0.660	0.385
	Germany	0.512	0.492
	Switzerland	0.585	0.440
Switzerland	UK	0.577	0.444
	Germany	0.564	0.458
	Switzerland	0.678	0.372
Combined	UK	0.627	0.408
	Germany	0.524	0.481
	Switzerland	0.652	0.392

models on the Switzerland dataset and test it on the other two datasets, to see if the same effects are present. Finally, we train using a combined dataset from both the UK and Switzerland.[8] This provides potential for interesting analysis – not only can we see whether training on two datasets is a good technique for performance across both datasets, but we can also see how well this model can transfer to the third dataset.

The results of these experiments are shown in Figure 21, with statistics for each model on each dataset in Table 4. In each case we see a drop in performance when using data from other locations, but not a complete loss of ability to distinguish transmitters. This is an encouraging result, indicating that SatIQ is looking at characteristics inherent to the transmitter even across different locations and receiver configurations, rather than at unrelated properties. We also see that the model trained on data collected in the UK does not show good performance on the data collected in Germany – this data was collected using cheaper hardware, and as a result has lower signal quality, so this is not surprising. Interestingly, the model trained on the Switzerland dataset shows better performance on

[8] We do not include the data collected in Germany due to the smaller number of messages and low signal quality.

the Germany dataset. This could be due to the lower signal quality of the Switzerland data resulting in a greater ability to withstand noise. We also note that the smaller UK dataset performs similarly to the Switzerland dataset, suggesting that good performance will be possible with a larger dataset from any location or hardware configuration.

When training a model on the combined dataset, good performance is achieved on both the UK and Switzerland datasets, with the results on the data from Germany somewhere between the two single-dataset models. The training dataset for this analysis was relatively small compared to the full dataset, but it is sufficient to show the transferability of the architecture. From these results we can also be reasonably confident that with a larger dataset that mixes data from more hardware configurations and physical locations, good performance can be achieved even on unseen hardware. This will dramatically increase the deployability of SatIQ, as it can be deployed in new locations on any receiver hardware without requiring a new dataset or model retraining.

It may also be possible to transfer a trained SatIQ model to a completely new satellite constellation, since many of the signal impairments will be common between hardware configurations. With a small amount of retraining, similar performance might be achieved across a wide range of satellite systems. Such analysis is beyond the scope of this article, as it will require a new dataset for the second constellation.

5.6 Security

We next evaluate the security properties of SatIQ, assessing its performance under an attack scenario. The simplest such scenario involves merely swapping the identifiers of transmitters in our existing dataset. This models an attack from within a compromised system, in which an attacker has gained control of a satellite currently in use – this matches our training scenario and results. As discussed in Section 3, this scenario is somewhat unrealistic, and not our primary concern.

The more interesting case is to evaluate SatIQ's robustness under real-world replay attacks. We can evaluate this in a realistic setting by replaying recorded messages over a wire. By capturing and replaying real-world messages, we ensure signal characteristics like background noise, path loss, and attenuation are as realistic as possible, since they are already a part of the recorded data. Furthermore, by replaying the messages over a wire, the signal is not being further degraded by the inclusion of channel impairments a second time over. However, it is still impacted by the features of the attacker's SDR, so the final signal will include the fingerprint of the original transmitter distorted by the new features of the attacker's transmitter. A similar result could be achieved through the use of an RF-shielded box, although this introduces additional complexities due to reflections and other effects caused by the wireless channel. Our evaluation using the wired channel is sufficient to demonstrate the effectiveness of SatIQ, but future work may consider adapting our experimental setup to a wireless environment for a slightly more realistic analysis, provided its shortcomings can be mitigated.

Our experimental setup is as shown in Figure 22. We first capture Iridium messages at 25 MS/s, saving the raw IQ samples to a file – this provides us with a dataset of samples identical to what the SDR would normally receive. We then replay these samples over a wire connected to the "victim" SDR, feeding the captured messages into the fingerprinting system. By this strategy, 253 messages were collected.

For each replayed message, we take a number of "known good" messages from the same transmitter from our testing dataset. We randomly select a number of these messages to be our anchors using a "shuffle split" strategy. We compare the anchors to the replayed messages to obtain the false positive rate, and to the other known good messages to get the true positive rate. The results of this experiment are shown in Figure 23. We can see that SatIQ performs significantly better in this scenario, with a base AUC of 0.927. When we compare each message to 16 anchors this

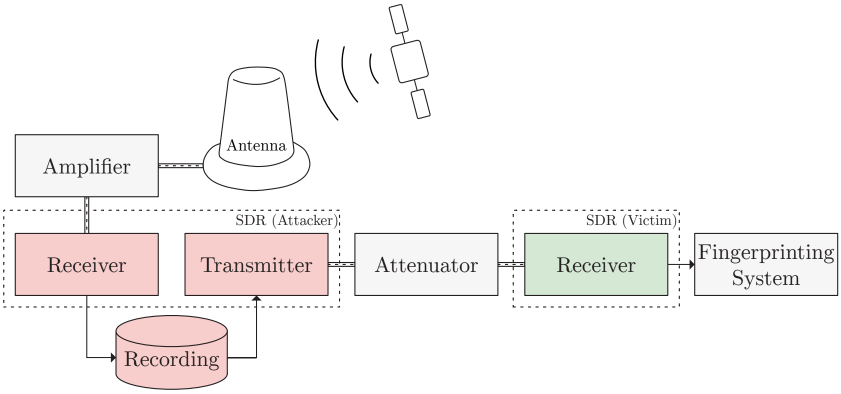


Fig. 22. Hardware setup for the replay attacks. Raw samples are captured using an SDR, then replayed directly into the fingerprinting system's SDR over a cable.

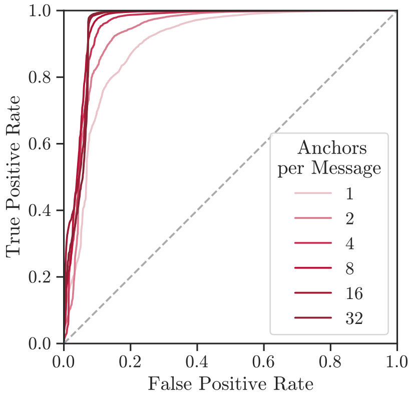


Fig. 23. ROC curves showing the system's performance when detecting replayed messages. Performance is significantly better than our training results, achieving a maximum AUC of 0.960.

performance increases even further, with an AUC of 0.960 and an EER of 0.072! [9] This indicates that the attacker's SDR has introduced its own fingerprint, distorting the message and altering its features.

Furthermore, this performance is good enough to deploy in a real-world system – by adjusting the acceptance threshold we can achieve a high true acceptance rate while minimizing the number of spoofed messages that are accepted. These results are summarized in Table 5. By setting the threshold such that 99% of legitimate messages are accepted, we accept only 10% of the attacker's messages. This performance is good enough to use in a real-world setting, particularly if we continuously fingerprint messages over the course of a communication session, taking the average acceptance rate over time as an indicator of attack – in order to have a meaningful impact the attacker will need to spoof multiple messages, which significantly raises the likelihood of detection.

Figure 24 shows the distribution of fingerprint distances from the replay dataset over time. We can see that there are clusters of consecutive messages all with similar distances in fingerprint

[9] Note that performance starts to drop off slightly for larger numbers of anchors – this is not an indication of the strategy's ineffectiveness, but of the limited size of the replay dataset. If more replayed messages were collected then this artifact would disappear.

Table 5. True Positive (True Accept) Rates and False Positive (False Accept) Rates for Key Threshold Values, Tested on Replayed Messages

TPR	FPR	Threshold
0.999	0.257	1.238
0.990	0.107	1.092
0.950	0.074	0.970
0.900	0.072	0.907
0.989	0.100	1.081
0.524	0.050	0.703
0.327	0.010	0.624
0.148	0.000	0.538

Messages are tested against 32 anchors, and the mean distance is taken.

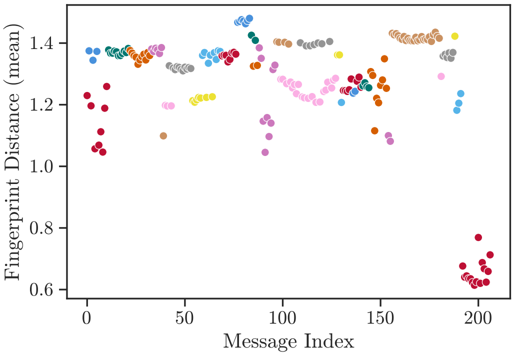


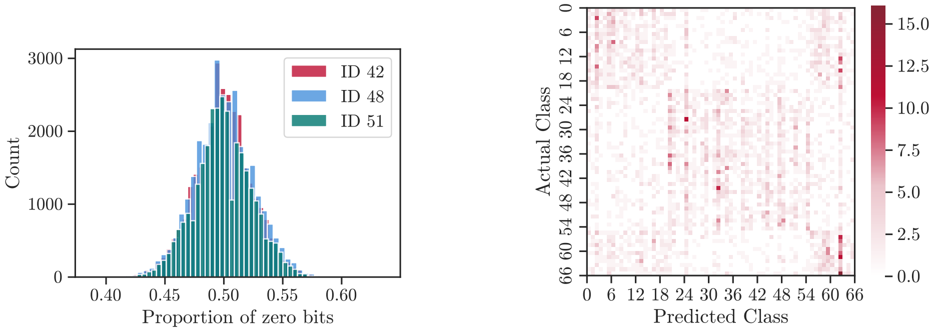
Fig. 24. Distance in fingerprint space between attacker-replayed messages and legitimate messages over time. Transmitter ID is represented by message color; repeated clusters of the same color belong to different transmitters.

space, and further that these all belong to the same transmitter ID. We draw from this result that some transmitters may be easier to spoof than others. The overall false acceptance rate is still low, however, so the risk is limited.

This attack scenario assumes a well-equipped adversary with access to a high-end SDR, and eliminates all the difficulties of over-the-air replay attacks. Despite all these concessions, we are still able to detect the attack in the majority of cases. With an even higher budget (to transmit at an even higher sample rate) and careful effort to eliminate noise introduced by the radio, it will certainly be possible to circumvent this system [34], but our results show that it will take a concerted effort to do so – simple message replay is not enough. We can therefore exclude a large proportion of attackers with all but the highest budgets, granting a real-world security benefit to ground systems.

5.7 Comparison with Existing Systems

We also compare the performance of SatIQ against the similar "PAST-AI" system discussed in Section 2.4 [26]. Instead of looking at the signal at a high sample rate, this system instead combines a large number of incoming messages from the same transmitter into a single heatmap (with signals captured at 1 sample per symbol), using an image classifier to identify transmitters. This is a much simpler system, but it comes with a number of caveats: perhaps most importantly, that it requires a



(a) Example of the different distributions of zero bits between transmitters in the dataset.

(b) Confusion matrix from a Support Vector Machine trained to classify transmitters from the data distribution alone, resulting in an accuracy of 3.70%.

Fig. 25. Evaluation of the distribution of bits in the PAST-AI dataset, and how this may skew results.

large number of messages (approximately 100, following the implementation in the article) to build each heatmap, so authentication cannot be done on a per-message basis. Attacks that take a small number of messages to execute are therefore unlikely to be detected. Furthermore, the system is a simple classifier with a closed set of identifiers, so it is impossible for the attacker to be identified as anything other than a legitimate transmitter – the only possible outcome is misclassification.

We demonstrate this with a quantitative comparison between PAST-AI and SatIQ. The dataset used in the article has been made open access [56], and we gained access to the code by contacting the authors directly. The training data and code are therefore identical to those used in their article.

We compare the two systems under three different types of attack:

- We first look at the “label swap” attack evaluated in Section 4.5, in which messages are sent from a legitimate transmitter but the identifier has been falsified. This is the least realistic attack, as it requires the adversary to take over a satellite from the original set.
- We next look at the “continuous SDR” case, in which an attacker is continuously sending spoofed messages from an SDR.
- Finally, we look at the “single message SDR” case, in which the attacker sends a single spoofed message from the SDR.

For the SatIQ results, we use the same results established earlier in this article. Note that when using SatIQ the latter two attacks are considered to be the same, since we are primarily fingerprinting individual incoming messages (although performance may be increased by tracking accept/reject rates over time for consecutive messages). On the other hand, PAST-AI requires many incoming messages to build a single image for classification, so single message spoofing events are much harder to detect.

We also note the differences between PAST-AI and SatIQ for the label swap attack. Firstly, PAST-AI is only capable of classifying entire satellites, and cannot identify individual beams within the satellite as SatIQ can – this is a much harder problem due to the larger number of classes involved, and naturally results in a moderate drop in accuracy. Secondly, we note that PAST-AI does not attempt to mask out identifying information present in messages, creating its input images from the whole message, including transmitter ID. This affects the distribution of symbols present in the message, and opens up the possibility for the model to learn the underlying distribution. Figure 25 illustrates the differing distribution of symbols between messages from two different transmitters, and shows the output of a Support Vector Machine trained to classify transmitters

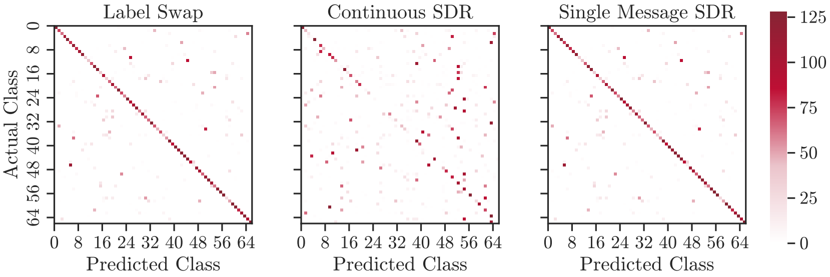


Fig. 26. Heatmaps showing the predicted vs actual class under each type of attack, using the PAST-AI model. Note that in the first attack, the attacker's goal is for transmitters to be misclassified, whereas in the second and third their goal is for replayed messages to be classified as legitimate.

Table 6. Success Rate of the Attacker in Three Different Types of Attack, Comparing Between PAST-AI and SatIQ (Lower Numbers Indicate the System is Better at Detecting the Attack)

	PAST-AI	SatIQ
Label Swap	0.287	0.277
Continuous SDR	0.278	0.072
Single Message SDR	0.713	0.072

All results are taken from experiments in this article.

based on the distribution of symbols alone. We achieve a maximum validation accuracy of 3.70% (random guessing would yield 1.51%), demonstrating the presence of identifying information in the underlying distribution of symbols. This casts doubt on the performance figures quoted by the authors of PAST-AI, as it seems likely the model is improving its performance by looking at the distribution of symbols in a group of messages. On the other hand, SatIQ operates on message headers alone, which do not contain any identifying information which could skew the results.

For the SDR-based attacks, we must create a replay dataset in the same format as used by PAST-AI. We achieve this using a hardware configuration similar to the one shown in Figure 22, replaying messages from the original PAST-AI dataset through an SDR transmit-receive loop over a wire. Using this dataset, we construct images composed of solely replayed messages (for the "continuous SDR" attack), and heatmaps derived primarily from the original dataset, but with a single message replayed (for the "single message SDR" case). We classify these images using the trained PAST-AI model. The results of this are shown in Figure 26, with performance figures in Table 6. We can see that under the "label swap" attack, performance is very similar between SatIQ and PAST-AI, despite the fact that SatIQ is authenticating a much larger set of transmitters.[10] We also see that under the "continuous SDR" attack, PAST-AI incorrectly classifies the replayed messages as the legitimate transmitter with a rate of 0.278, compared to SatIQ's 0.072. Performance is worse when only a single message is replayed: the message is incorrectly classified as legitimate in 0.713 of cases. We therefore conclude that SatIQ is more suitable for deployment as a countermeasure to spoofing attacks, both continuous and on individual messages.

[10] We note that our experimental reproduction of PAST-AI (using the exact code and dataset used in their article) yielded worse performance than reported in the article [26], with a baseline accuracy of 0.82, leading to an attacker success rate of 0.18.

5.8 Deployment Considerations

Finally, we discuss what it would look like to deploy SatIQ in a real-world system. As we mention in Section 4.4, a trained SatIQ model can be run in real-time, verifying messages as they arrive. We verify this by running the model on lightweight hardware. Our data collection observed roughly 0.5 ring alert messages per second, or 6 to 7 messages per second across all message types. This gives us approximately 150 ms to validate each incoming message. On the Intel Xeon CPU used for training (disabling the GPU), the model takes 4.85s to validate a batch of 543 messages (8.9 ms per message), with a maximum RAM usage of 1.7 GB – this is more than fast enough to validate all incoming messages.

We also ran SatIQ on a Raspberry Pi 4B single-board computer, with 4 GB of RAM and a 1.8 GHz quad-core processor. On this hardware, it took 93.25 s to validate the same 543 messages (171 ms per message), using 2.0 GB of RAM. This demonstrates that SatIQ can run on lightweight hardware – although slightly too slow to validate all messages in real time, this can certainly validate incoming ring alert messages, and could run even faster with the addition of a dedicated AI accelerator.

We must also consider how deployed devices can capture high sample rate message fingerprints. Our current setup requires a moderately powerful computer to perform the necessary high sample rate message synchronization and decoding, but this can be reduced significantly with the use of a dedicated FPGA demodulator. Combined with the good performance on low-power CPUs shown above, it will be possible to integrate SatIQ into even lightweight devices – however, some instances may require the addition of hardware.

Additionally, it is important to decide what action is taken when a (potentially legitimate) message is rejected by SatIQ. The action taken depends on the level of security required. For some systems it may be sufficient to simply notify the user, but still accept the message, providing increased awareness of potential attacks without risking any impact to service through rejected legitimate messages. Alternatively, higher security applications may choose to reject all messages flagged by SatIQ, accepting the increased packet loss in exchange for greater security against spoofing and replay attacks. It may be possible for some systems to request message retransmission in this case, to ensure messages are still delivered – this may occur at the transport or application layer, or systems may be modified to allow retransmission requests (if this has not already been implemented).

Finally, we must consider the logistics of such a deployment. As discussed in Section 5.1, we propose that SatIQ-based systems periodically refresh their anchor messages to maximize performance. An additional benefit of this approach is that no centralized database of fingerprints is required, significantly reducing the complexity of deploying SatIQ – there is no need to gather or maintain a database of fingerprints, and devices do not need to connect to a server to update their databases or verify incoming messages. SatIQ-based systems can therefore also be developed by either the satellite operator or the receiver manufacturer, or by a third party building devices to integrate with existing ground systems.

6 Future Work

Our research has revealed several promising avenues of future research. Firstly, it is likely that better performance can be achieved through fine-tuning, using the same base model architecture as SatIQ. We show that by training a model on data collected from multiple receiver configurations can produce a system that transfers more readily to different ground systems; with a larger dataset from more locations, this would likely be even more versatile.

It would also be useful to implement some of the other fingerprinting techniques, particularly in assessing the extent to which a trained system can be transferred to another constellation, and what degree of retraining is required. Furthermore, it would be particularly beneficial to standardize the other analyses such that they could be applied to other fingerprinting techniques. We currently have no method of empirically comparing fingerprinting techniques to each other in terms of their security properties – a standard suite of tests would remedy this. We consider a comparison between fingerprinting systems to be out of scope for this article, as each fingerprinting technique relies on different kinds of data. For example, PAST-AI uses full messages rather than just message headers [26], and transient fingerprinting requires precise transient synchronization.

Another promising area of research would be to assess the effectiveness of fingerprinting in conjunction with other methods of spoofing detection, such as assessing SNR or distortion. Multiple fingerprinting methods could also be used in concert, providing even greater effectiveness than any model alone. However, some methods are likely to use the same signal properties as each other (providing no mutual benefit), so a full analysis is needed in order to understand which methods are effective together.

Finally, it would be useful to assess the effectiveness of fingerprinting in systems which already have some amount of authentication. For instance, such an analysis could evaluate fingerprinting as a preventative measure against GNSS message delay/advancement attacks [13]. This would demonstrate that fingerprinting is not just useful in legacy systems, but has concrete usefulness even in new satellite systems.

7 Conclusion

In this article we have contributed new methods toward radio signal fingerprinting in the context of satellite transmitters, providing novel techniques which can be used to build high sample rate fingerprinting systems. We have succeeded in demonstrating that satellite signals at high sample rates contain sufficient identifying information, and confirmed that our techniques combining autoencoders and Siamese models are feasible for fingerprinting. We have also provided a large dataset of captured message headers from Iridium satellites, which can be used for further research and testing in satellite transmitter fingerprinting.

We have performed detailed analysis showing that the accuracy of the system remains stable over time, demonstrated that it is not affected by weather and other conditions, and shown that it can be extended to new transmitters with no drop in performance. We have also confirmed that the system can be used across different receiver hardware with a suitable dataset. Finally, we have demonstrated excellent performance detecting replay attacks in the scenario of a well-equipped attacker sending messages via a wired channel, achieving an Equal Error Rate of 0.072 and ROC AUC of 0.960.

Our work shows that high sample rate fingerprinting is feasible to secure satellite communication, does not degrade over time or on new trasmitters, and can significantly improve security against spoofing and replay attacks.

Acknowledgments

We would like to thank armasuisse Science + Technology for their support during this work. Joshua was supported by the **Engineering and Physical Sciences Research Council** (**EPSRC**). Sebastian was supported by the Royal Academy of Engineering and the Office of the Chief Science Adviser for National Security under the UK Intelligence Community Postdoctoral Research Fellowships programme. Simon was supported by the Government Office for Science and the Royal Academy of Engineering under the UK Intelligence Community Postdoctoral Research Fellowships scheme.

Open Access

To facilitate future research, all code and data have been made openly available. The code is available at https://github.com/ssloxford/SatIQ. The dataset and model weights can be found at the following URLs:

- Dataset (UK): https://doi.org/10.7910/DVN/P5FUAW
- Dataset (Germany): https://doi.org/10.7910/DVN/RXWV1M
- Dataset (Switzerland): https://doi.org/10.7910/DVN/OSSJ68
- Trained model weights: https://doi.org/10.7910/DVN/GANMDZ

References

[1] Joshua Smailes, Sebastian Köhler, Simon Birnbach, Martin Strohmeier, and Ivan Martinovic. 2023. Watch this space: Securing satellite communication through resilient transmitter fingerprinting. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security.* Association for Computing Machinery, New York, NY, USA, 608–621. DOI: http://doi.org/10.1145/3576915.3623135

[2] Great Scott Gadgets. 2021. HackRF One. (2021). Retrieved September 30, 2025 from https://greatscottgadgets.com/hackrf/one/

[3] Martin Strohmeier, Vincent Lenders, and Ivan Martinovic. 2015. On the security of the automatic dependent surveillance-broadcast protocol. *IEEE Communications Surveys Tutorials* 17, 2 (2015), 1066–1087. DOI: http://doi.org/10.1109/COMST.2014.2365951

[4] Marc Lichtman, Roger Piqueras Jover, Mina Labib, Raghunandan Rao, Vuk Marojevic, and Jeffrey H. Reed. 2016. LTE/LTE-A jamming, spoofing, and sniffing: Threat assessment and mitigation. *IEEE Communications Magazine* 54, 4 (2016), 54–61.

[5] Gyuhong Lee, Jihoon Lee, Jinsung Lee, Youngbin Im, Max Hollingsworth, Eric Wustrow, Dirk Grunwald, and Sangtae Ha. 2019. This is your president speaking: Spoofing alerts in 4G LTE networks. In *Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services.* 404–416.

[6] João Gaspar, Renato Ferreira, Pedro Sebastião, and Nuno Souto. 2020. Capture of UAVs through GPS spoofing using low-cost SDR platforms. *Wireless Personal Communications* 115, 4 (2020), 2729–2754.

[7] NASA. 2022. Fire Information for Resource Management System (FIRMS). (2022). Retrieved September 30, 2025 from https://earthdata.nasa.gov/earth-observation-data/near-real-time/firms

[8] Esri. 2022. Esri Releases Updated Land-Cover Map with New Sets of Global Data. (2022). Retrieved September 30, 2025 from https://www.esri.com/about/newsroom/announcements/esri-releases-updated-land-cover-map-with-new-sets-of-global-data/

[9] Meta. 2022. High Resolution Population Density Maps. (2022). Retrieved September 30, 2025 from https://dataforgood.facebook.com/dfg/tools/high-resolution-population-density-maps

[10] Cloud to Street. 2022. Cloud to Street. (2022). Retrieved September 30, 2025 from https://www.cloudtostreet.ai/

[11] sam210723. 2020. Receiving Images from Geostationary Weather Satellite GEO-KOMPSAT-2A. (2020). Retrieved September 30, 2025 from https://vksdr.com/xrit-rx

[12] sam210723. 2018. COMS-1 LRIT Key Decryption. (2018). Retrieved September 30, 2025 from https://vksdr.com/lrit-key-dec

[13] Maryam Motallebighomi, Harshad Sathaye, Mridula Singh, and Aanjhan Ranganathan. 2022. Cryptography is not enough: Relay attacks on authenticated GNSS signals. arXiv:2204.11641. Retrieved from https://arxiv.org/abs/2204.11641

[14] Marc Lichtman. 2021. IQ sampling. In *PySDR: A Guide to SDR and DSP Using Python.* https://pysdr.org/content/sampling.html

[15] NASA. 2022. X-Band Direct Readout Sites Worldwide. (2022). Retrieved September 30, 2025 from https://directreadout.sci.gsfc.nasa.gov/?id=dspContent&cid=78

[16] Eric Jedermann, Martin Strohmeier, Matthias Schäfer, Jens Schmitt, and Vincent Lenders. 2021. Orbit-based authentication using TDOA signatures in satellite networks. In *Proceedings of the 14th ACM Conference on Security and Privacy in Wireless and Mobile Networks.* 175–180.

[17] Mohsen Riahi Manesh, Jonathan Kenney, Wen Chen Hu, Vijaya Kumar Devabhaktuni, and Naima Kaabouch. 2019. Detection of GPS spoofing attacks on unmanned aerial systems. In *Proceedings of the 2019 16th IEEE Annual Consumer Communications and Networking Conference.* IEEE, 1–6.

[18] Damian Miralles, Aurelie Bornot, Paul Rouquette, Nathan Levigne, Dennis M. Akos, Yu-Hsuan Chen, Sherman Lo, and Todd Walter. 2020. An assessment of GPS spoofing detection via radio power and signal quality monitoring for aviation safety operations. *IEEE Intelligent Transportation Systems Magazine* 12, 3 (2020), 136–146.

[19] Naeimeh Soltanieh, Yaser Norouzi, Yang Yang, and Nemai Chandra Karmakar. 2020. A review of radio frequency fingerprinting techniques. *IEEE Journal of Radio Frequency Identification* 4, 3 (2020), 222–233.

[20] Jeyanthi Hall, Michel Barbeau, and Evangelos Kranakis. 2003. Detection of transient in radio frequency fingerprinting using signal phase. *Wireless and Optical Communications* 9 (2003), 13–18. https://scholar.google.com/scholar?cluster=14122771568642364504&hl=en&as_sdt=7,39)

[21] Lianfen Huang, Minghui Gao, Caidan Zhao, and Xiongpeng Wu. 2013. Detection of Wi-Fi transmitter transients using statistical method. In *Proceedings of the 2013 IEEE International Conference on Signal Processing, Communication and Computing.* IEEE, 1–5.

[22] K. J. Ellis and N. Serinken. 2001. Characteristics of radio transmitter fingerprints. *Radio Science* 36, 4 (2001), 585–597. DOI: http://doi.org/10.1029/2000RS002345

[23] Kasper Bonne Rasmussen and Srdjan Capkun. 2007. Implications of radio fingerprinting on the security of sensor networks. In *Proceedings of the 2007 3rd International Conference on Security and Privacy in Communications Networks and the Workshops-SecureComm 2007.* IEEE, 331–340.

[24] Mahsa Foruhandeh, Abdullah Z. Mohammed, Gregor Kildow, Paul Berges, and Ryan Gerdes. 2020. Spotr: GPS spoofing detection via device fingerprinting. In *Proceedings of the 13th ACM Conference on Security and Privacy in Wireless and Mobile Networks.* Association for Computing Machinery, 242–253. DOI: http://doi.org/10.1145/3395351.3399353

[25] Simon Birnbach, Joshua Smailes, Richard Baker, and Ivan Martinovic. 2023. Adaptable hardware fingerprinting for radio data links and avionics buses in adversarial settings. In *Proceedings of the 2023 IEEE/AIAA 42nd Digital Avionics Systems Conference.* IEEE, 1–10.

[26] Gabriele Oligeri, Savio Sciancalepore, Simone Raponi, and Roberto Di Pietro. 2022. PAST-AI: Physical-layer Authentication of Satellite Transmitters via Deep Learning. *IEEE Transactions on Information Forensics and Security* 18 (2022), 274–289. https://ieeexplore.ieee.org/abstract/document/9936663

[27] Irwin O. Kennedy, Patricia Scanlon, Francis J. Mullany, Milind M. Buddhikot, Keith E. Nolan, and Thomas W. Rondeau. 2008. Radio transmitter fingerprinting: A steady state frequency domain approach. In *Proceedings of the 2008 IEEE 68th Vehicular Technology Conference.* IEEE, 1–5.

[28] Joshua Bassey, Damilola Adesina, Xiangfang Li, Lijun Qian, Alexander Aved, and Timothy Kroecker. 2019. Intrusion detection for IoT devices based on RF fingerprinting using deep learning. In *Proceedings of the 2019 4th International Conference on Fog and Mobile Edge Computing.* IEEE, 98–104.

[29] Francesco Restuccia, Salvatore D'Oro, Amani Al-Shawabka, Mauro Belgiovine, Luca Angioloni, Stratis Ioannidis, Kaushik Chowdhury, and Tommaso Melodia. 2019. DeepRadioID: Real-time channel-resilient optimization of deep learning-based radio fingerprinting algorithms. In *Proceedings of the 20th ACM International Symposium on Mobile Ad Hoc Networking and Computing.* 51–60.

[30] Kunal Sankhe, Mauro Belgiovine, Fan Zhou, Luca Angioloni, Frank Restuccia, Salvatore D'Oro, Tommaso Melodia, Stratis Ioannidis, and Kaushik Chowdhury. 2020. No radio left behind: Radio fingerprinting through deep learning of Physical-Layer Hardware Impairments. *IEEE Transactions on Cognitive Communications and Networking* 6, 1 (2020), 165–178. DOI: http://doi.org/10.1109/TCCN.2019.2949308

[31] ÖH Tekbaş, Oktay Üreten, and Nur Serinken. 2004. Improvement of transmitter identification system for low SNR transients. *Electronics Letters* 40, 3 (2004), 182–183.

[32] Weidong Wang and Lu Gan. 2022. Radio frequency fingerprinting improved by statistical noise reduction. *IEEE Transactions on Cognitive Communications and Networking* 8, 3 (2022), 1444–1452.

[33] Dan Veeneman. 2021. Iridium: Technical Details. (2021). Retrieved September 30, 2025 from http://www.decodesystems.com/iridium.html

[34] Boris Danev, Heinrich Luecken, Srdjan Capkun, and Karim El Defrawy. 2010. Attacks on physical-layer identification. In *Proceedings of the 3rd ACM Conference on Wireless Network Security.* 89–98.

[35] Joshua Smailes, Edd Salkield, Sebastian Köhler, Simon Birnbach, Martin Strohmeier, and Ivan Martinovic. 2024. Sticky fingers: Resilience of satellite fingerprinting against jamming attacks. In *Proceedings of the Workshop on Security of Space and Satellite Systems.*

[36] Jiabao Yu, Aiqun Hu, Fen Zhou, Yuexiu Xing, Yi Yu, Guyue Li, and Linning Peng. 2019. Radio frequency fingerprint identification based on denoising autoencoders. In *Proceedings of the 2019 International Conference on Wireless and Mobile Computing, Networking and Communications.* IEEE, 1–6.

[37] Joshua Bassey, Xiangfang Li, and Lijun Qian. 2020. Device authentication codes based on RF fingerprinting using deep learning. arXiv:2004.08742. Retrieved from https://arxiv.org/abs/2004.08742

[38] Qi Jiang and Jin Sha. 2024. Radio frequency fingerprint identification based on variational autoencoder for GNSS. *IEEE Geoscience and Remote Sensing Letters* 21 (2024), 1–4. https://ieeexplore.ieee.org/abstract/document/10556578

[39] Davide Chicco. 2021. Siamese neural networks: An overview. *Artificial Neural Networks* (2021), 73–94. https://link.springer.com/protocol/10.1007/978-1-0716-0826-5_3

[40] Jinting Zhu, Julian Jang-Jaccard, and Paul A. Watters. 2020. Multi-loss siamese neural network with batch normalization layer for malware detection. *IEEE Access* 8 (2020), 171542–171550.

[41] Cheng Zhang, Wu Liu, Huadong Ma, and Huiyuan Fu. 2016. Siamese neural network based gait recognition for human identification. In *Proceedings of the 2016 IEEEE International Conference on Acoustics, Speech and Signal Processing* . IEEE, 2832–2836.

[42] Mingtao Pei, Bin Yan, Huiling Hao, and Meng Zhao. 2023. Person-specific face spoofing detection based on a siamese network. *Pattern Recognition* 135 (2023), 109148.

[43] Kaavya Sriskandaraja, Vidhyasaharan Sethu, and Eliathamby Ambikairajah. 2018. Deep siamese architecture based replay detection for secure voice biometric. In *Proceedings of the Interspeech*. 671–675.

[44] Yu Mao, Yang-Yang Dong, Ting Sun, Xian Rao, and Chun-Xi Dong. 2021. Attentive siamese networks for automatic modulation classification based on multitiming constellation diagrams. *IEEE Transactions on Neural Networks and Learning Systems* 34, 9 (2021), 5988–6002.

[45] Louis Morge-Rollet, Frédéric Le Roy, Denis Le Jeune, and Roland Gautier. 2020. Siamese network on I/Q signals for RF fingerprinting. In *Proceedings of the Actes de La Conférence CAID 2020*. 152.

[46] Zachary Langford, Logan Eisenbeiser, and Matthew Vondal. 2019. Robust signal classification using siamese networks. In *Proceedings of the ACM Workshop on Wireless Security and Machine Learning*. 1–5.

[47] Kian Ahrabian and Bagher BabaAli. 2019. Usage of autoencoders and siamese networks for online handwritten signature verification. *Neural Computing and Applications* 31 (2019), 9321–9334.

[48] SatNOGS. 2022. SatNOGS: Open Source global network of satellite ground-stations. (2022). Retrieved September 30, 2025 from https://satnogs.org/

[49] Tobias Schneider and Stefan Zehl. 2022. gr-iridium: GNU Radio Iridium Out Of Tree Module. Chaos Computer Club München. (Sept. 2022).

[50] Sarang Narkhede. 2018. Understanding AUC - ROC Curve. *Towards Data Science* 26, 1 (2018), 220–227.

[51] Diederik P. Kingma and Max Welling. 2013. Auto-encoding variational bayes. arXiv:1312.6114. Retrieved from https://arxiv.org/abs/1312.6114

[52] Bechir Hamdaoui and Abdurrahman Elmaghbub. 2022. Deep-learning-based device fingerprinting for increased LoRa-IoT security: Sensitivity to network deployment changes. *IEEE Network* 36, 3 (2022), 204–210. DOI: http://doi.org/10.1109/MNET.001.2100553

[53] Amani Al-Shawabka, Francesco Restuccia, Salvatore D'Oro, Tong Jian, Bruno Costa Rendon, Nasim Soltani, Jennifer Dy, Stratis Ioannidis, Kaushik Chowdhury, and Tommaso Melodia. 2020. Exposing the fingerprint: Dissecting the impact of the wireless channel on radio fingerprinting. In *Proceedings of the IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*. 646–655. DOI: http://doi.org/10.1109/INFOCOM41043.2020.9155259

[54] Visual Crossing Corporation. 2024. Visual Crossing Weather. (2024). Retrieved September 30, 2025 from https://www.visualcrossing.com/

[55] Jonathan McDowell. 2024. Starlink Statistics. (2024). Retrieved September 30, 2025 from https://planet4589.org/space/con/star/stats.html

[56] Gabriele Oligeri and Savio Sciancalepore. 2022. Physical Layer Data Acquisition of IRIDIUM Satellites Broadcast Messages. (2022). Retrieved September 30, 2025 from https://data.mendeley.com/datasets/xcxspv8c2r/1

Received 23 October 2024; revised 30 June 2025; accepted 18 August 2025