A System to Detect Forged-Origin BGP Hijacks

Thomas Holterbach[*], Thomas Alfroy[*], Amreesh Phokeer[†], Alberto Dainotti[‡], Cristel Pelsser[§]
[*]*University of Strasbourg,* [†]*Internet Society,* [‡]*Georgia Tech,* [§]*UCLouvain*

Abstract

Despite global efforts to secure Internet routing, attackers still successfully exploit the lack of strong BGP security mechanisms. This paper focuses on an attack vector that is frequently used: *Forged-origin hijacks*, a type of BGP hijack where the attacker manipulates the AS path to make it immune to RPKI-ROV filters and appear as legitimate routing updates from a BGP monitoring standpoint. Our contribution is DFOH, a system that *quickly* and *consistently* detects forged-origin hijacks in the *whole* Internet. Detecting forged-origin hijacks boils down to inferring whether the AS path in a BGP route is legitimate or has been manipulated. We demonstrate that current state-of-art approaches to detect BGP anomalies are insufficient to deal with forged-origin hijacks. We identify the key properties that make the inference of forged AS paths challenging, and design DFOH to be robust against real-world factors (e.g., data biases). Our inference pipeline includes two key ingredients: *(i)* a set of strategically selected features, and *(ii)* a training scheme adapted to topological biases. DFOH detects 90.9% of the forged-origin hijacks within only ≈5min. In addition, it only reports ≈17.5 suspicious cases every day for the whole Internet, a small number that allows operators to investigate the reported cases and take countermeasures.

1 Introduction

On 3 February 2022, the cryptocurrency platform KLAYswap was targeted by hackers who stole $1.9 million worth of digital assets [59]. More recently, on 17 August 2022, an attack to cBridge—a crypto-asset bridge—affected 32 victims, who lost $235,000 [4]. Both attacks were the result of a *forged-origin BGP hijack*, a type of routing hijack where the attackers announce forged AS paths towards a victim prefix by prepending the victim's origin AS number in order to make them appear legitimate. Clearly, BGP hijacking attacks are not a surprise anymore. They repeatedly make the headlines [1,2] and are known as attack vectors to steal cryptocurrency [8], obtain bogus certificates [15], or deanonymize Tor users [62]. The vulnerability they exploit is simply the result of BGP being designed without security in mind: An attacker can manipulate every attribute in a BGP message (including the AS path and its origin AS) and illegitimately announce a prefix owned by its victim so as to divert the traffic to its network.

Proactive solutions against BGP hijacks are being gradually deployed. However, forged-origin hijacks have been left uncovered by such solutions—despite these attacks being actively used in the wild. In fact, network operators attempt to proactively thwart BGP hijacks by configuring their routers to filter hijacked routes [46] using *(i)* RPKI-based Route Origin Validation (ROV) and *(ii)* data from Internet Routing Registries (IRR). Unfortunately, RPKI-ROV filters do not help to detect forged-origin hijacks, since the forged origin in the AS path is actually valid, while IRR-based filters are known to be inaccurate, incomplete [25], and far too often missing given the increasing number of observed BGP hijacks [7]. Today, network operators do not have many options left other than waiting for the deployment of new security extensions to BGP to consistently prevent forged-origin hijacks [44]. Such deployment—if it will happen at all—might take more than a decade, as in the case of RPKI-ROV [21].

In this paper, we present DFOH, the first *locally-deployable* system that *widely*, *quickly*, and *accurately* **D**etects **F**orged-**O**rigin **H**ijacks on the Internet. With a single deployment of DFOH on a commodity server, any attacker performing a forged-origin hijack is likely to be quickly detected, the hijack publicly reported, and the victim immediately notified. Being aware of the attack, the victim can apply countermeasures and the community can take actions to prevent similar attacks from happening again. Additionally, DFOH can detect past attacks, allowing the community to measure the frequency of such attacks or profile forged-origin hijackers.

DFOH is a passive system that processes the AS paths observed in publicly collected BGP routes to detect forged-origin hijacks. The problem of detecting forged-origin hijacks can be reduced to identifying whether a link between two ASes is real or fake. Unfortunately, there are multiple reasons why two ASes might connect, whereas there is no simple

property that indicates whether they are not supposed to. In fact, while *link prediction* in the AS graph might appear as a similar problem, approaches presented in literature do not translate well to revealing forged-origin hijacks [29, 68]. Alternatively, the only detection method specifically designed for forged-origin hijacks (ARTEMIS [56]) is solely capable of detecting hijacks targeting the network deploying it.

With DFOH, we address this challenge by first identifying and studying a rich set of domain-specific properties. Then, we derive classification features from them that—when carefully combined into an inference pipeline—enable us to effectively distinguish fake AS links from real ones. We observe that the key factors (e.g., data biases and discriminating power of classification features) that come into play when building the inference model, significantly vary depending on the characteristics of the two ASes whose hypothetical link is under examination. We take this observation into account when training DFOH to build a system that performs well in realistic settings. With DFOH, all attack scenarios (including adversarial inputs) are covered and legitimate cases are consistently not reported. DFOH is a solid line of defense against forged-origin hijacks: It detects 90.9% of them within ≈5min and only reports ≈17.5 suspicious cases every day, a small number compared to the size of the Internet.

Main contributions. Our main contributions are:

- An identification of the key factors to consider when designing a forged-origin hijack detection system that is effective in real-world scenarios (§4).
- The design of the DFOH inference pipeline, which quickly and accurately detects *any* forged-origin hijacks on the *whole* Internet (§5 and §6).
- An open-source implementation of DFOH, which we also run as a service (`https://dfoh.uclouvain.be/`).
- An evaluation of DFOH on synthetic and real data, which demonstrates that it is an effective line of defense against forged-origin hijacks and that its design is sound (§7).

2 Background

BGP Hijacks. As BGP was originally designed without security in mind, an AS can announce a prefix that it does not own with the result of diverting traffic towards itself, either accidentally or intentionally. This is referred to as a BGP hijack and results in some ASes adopting the false route when it is perceived to be better than the legitimate one. As a result, the traffic originated by many Internet users and destined to the hijacked prefix is sent to the hijacker AS instead of the AS owning the hijacked prefix. Consider the scenario in Fig. 1, which depicts an AS-level topology with nine ASes, one Content Delivery Network (CDN) and one Internet Exchange

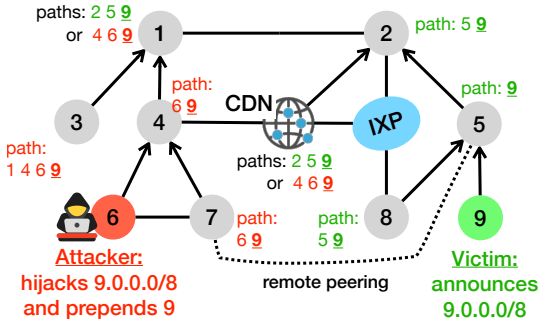


Figure 1: An example of an AS-level topology and the effect of a forged-origin hijack on the traffic destined to `9.0.0.0/8`. Nodes are ASes, straight arrows are customer to provider links and straight lines are peer to peer links.

Point (IXP) and assume that they all configure routing policies following the Gao-Rexford model [28]. Assume that *AS6* hijacks `9.0.0.0/8` (without prepending the origin AS). Without defenses, the hijacked route is used by four ASes (*AS1*, *AS3*, *AS4* and *AS7*), which thus send traffic for that prefix to the hijacker AS. These hijacks are commonly called misorigin hijacks or "Type-0" hijacks following ARTEMIS's convention [56]. Observe that *AS6* could also hijack a more specific prefix, e.g., `9.0.100.0/24`, in which case *all* the ASes divert the traffic destined to `9.0.100.0/24` to the hijacker, since routes towards more specific prefixes are always preferred.

Defenses against misorigin hijacks. Misorigin hijacks are often accidental and can be detected by looking at the origin of their AS path. Detecting them is effective today because there are many BGP vantage points from RIS [55] and RouteViews [51] that collect BGP routes received by many ASes and make them available to users. Sermpezis et al. show that the current public BGP monitoring infrastructure is able to observe all the hijacks with significant impact [56]. Thus, many hijack detection systems examine the collected BGP routes and raise an alarm when the origin AS is invalid [19, 35, 42, 56, 65]. Operators can also configure their routers to automatically deny routes with a wrong origin AS by using the Resource Public Key Infrastructure (RPKI) [43]. While RPKI-ROV is gradually being deployed [21], it only protects against misorigin hijacks and cannot help to detect AS-path manipulations.

3 Attack Model

Forged-origin hijack. Imagine that *AS6* in Fig. 1 is a clever attacker and knows that RPKI-ROV is increasingly adopted by operators [21]. After reading the state of the art about routing attacks in the literature [30], the attacker decides to prepend *AS9* in the AS path of the hijacked route (as shown in Fig. 1). Rapidly after applying this configuration, a significant part of the Internet traffic destined to `9.0.0.0/8` is deflected to

the attacker's AS. The attack is successful because the origin of the hijacked route is valid. Consequently, the RPKI-based filters miss the hijacked route, which also remains invisible from all the hijack detection systems that solely look at the origin AS. For instance, *AS4* in Fig. 1 receives a route for `9.0.0.0/8` with AS path *AS6-AS9* and accepts it given that its origin (*AS9*) is valid. Observe that *AS1* and the CDN receive the two routes for `9.0.0.0/8` with the same AS path length, in which case their local preferences determine whether they use the legitimate route or the hijacked one.

Definition: A forged-origin hijack is a BGP hijack where an attacker AS announces a route for an IP prefix that it is not authorized to originate and with an AS path that the attacker purposely manipulates so that the origin AS is valid.

The flip side of a forged-origin hijack is that it makes the AS path longer, which results in fewer ASes using the hijacked route compared to a misorigin (Type-0) hijack.

More specific prefixes are not better off. Despite RPKI, forged-origin hijacks can also succeed towards more specific prefixes. Operators oftentimes set a loose RPKI MaxLength [31] (sometimes confusingly [21]), allowing the attacker to launch successful forged-origin hijacks on more specific prefixes—a particularly harmful attack, since all traffic from *all* ASes is directed to the attacker [30, 32].

More advanced path manipulations are possible. An attacker can prepend more than one AS number. For instance, the attacker *AS6* in Fig. 1 could prepend *5 9*, in which case the origin is valid but the attacker AS appears in the third position from the origin. In this paper, we align with the state-of-art taxonomy [56] and define a Type-X hijack as follows:

Definition: A Type-X hijack is a forged-origin hijack where $X \geq 1$ indicates the position of the attacker's AS in the forged AS path, with the first AS (the origin) being at position zero.

In the (most common) cases where each AS number appears once in the AS path, X indicates the number of prepended ASes. Prepending *5 9* thus results in a Type-2 hijack. Intuitively, the higher is the number of prepended ASes, the lower is the impact of the attack (since the hijacked route reaches fewer ASes, as its AS path is longer).

4 The Case for DFOH

In this section, we highlight why DFOH is practically relevant (§4.1), and identify its key challenges and requirements (§4.2).

4.1 Lack of Defenses

DFOH is practically relevant because there is currently no effective mechanism to detect forged-origin hijacks.

Existing proactive defenses are not bulletproof. Because RPKI-ROV does not prevent forged-origin hijacks, the main proactive defense for network operators is to check whether the announcements of their customers are correct, i.e., each customer legitimately holds the AS numbers and IP address space they announce. This is achieved using route filters. In theory, these filters, recommended by MANRS [46], prevent an AS to propagate incorrect routing information when they are properly configured. Yet, in practice, they are often missing, inaccurate, or controllable by an attacker.

Missing filters: In April 2020, ROSTELECOM hijacked several prefixes, which impacted at least Amazon and Akamai. After investigation, it appears that at least two ASes, namely Rascom amd Cogent Co, did not configure proper filters that would have reduced the spread of the hijack [47]. Although an increasing—but still rather low—number of operators agree to follow routing security norms [26], we still observe many hijacks that widely propagate through the Internet, which indicates that these filters are often missing [48].

Inaccurate filters: Network operators often rely on peering information from the IRRs [39] to infer their customer cone, i.e., the set of ASes that can be reached only using customer links—a necessary information to build accurate filters. However, IRRs are known to contain inaccurate information [24, 25, 53], which inevitably results in inaccurate filters.

Controllable filters: Because the data from the IRRs is not verified [49], an attacker can intentionally pollute it to take control over which filters are added or removed in its provider's routers. More precisely, an attacker could pollute its `as-set` object, which is the IRR object that specifies a set of ASNs through which traffic can be routed, and is the object often used to automatically generate route filters [12]. This is supposedly what happened no later than in August 2022: An attacker (AS209243) added an Amazon's AS number into its `as-set` object before it started to announce an Amazon's prefix with an Amazon's AS number purposely prepended at the origin of the AS path [5].

Existing reactive defenses are narrowly focused. ARTEMIS is a system that detects forged-origin hijacks [56]. However, it is narrowly focused since it only detects hijacks that pertain to the AS deploying it. The key mechanism that ARTEMIS relies on to detect forged-origin hijacks consists in classifying new AS links as legitimate when observed in both directions, whereas it reports all the others as possible fake links. The problem is that ARTEMIS' approach cannot be directly extended to monitor *all prefixes*. Link bidirectionality can rule out a significant fraction of AS paths with new links when *they pertain to prefixes originated by the AS running ARTEMIS*. In fact, ARTEMIS likely observes both directions of the *neighboring* links as it combines BGP views from public collectors with local BGP views. However, when targeting attacks toward *any prefix* on the Internet, and using only public router collectors as vantage points, we find that bidirectionality is observable for only a small fraction of new links (see §6.2.4). Therefore, in the practical scenario that DFOH targets, solely relying on link bidirectionality results in a poor accuracy, as we show in §E.1.

4.2 Requirements

We now highlight four key challenges and requirements that DFOH must address to be effective and practical.

Requirement 1: A forged-origin hijack detection system must be fast and accept real-time and historical queries.

A fast detection (within a few minutes) enables mitigating the hijacks swiftly to limit their impact. Additionally, detecting past events is useful to e.g., lay hands on serial hijackers [64].

Requirement 2: A forged-origin hijack detection system must be accurate, both for pinpointing actual hijacks and avoiding triggering false alarms.

Accuracy matters because every detected hijack is likely to be manually investigated by the victim AS to confirm the attack and apply the appropriate mitigation scheme. A high number of false alarms would thus overwhelm network operators and limit the practicality of the system. Besides, the detection system should not miss actual hijacks.

Requirement 3: A forged-origin hijack detection system must be robust against missing, inaccurate and polluted data.

A forged-origin hijacks detection system must rely on a probabilistic inference as cryptographic-based AS paths verification is still missing today (BGPsec is not deployed at all [44]). The inferences are made using the data collected in various datasets including public peering information (PeeringDB and the IRR [39, 52]), which are known to often be inaccurate and incomplete [25, 53]. Worse, these datasets can be polluted by attackers, as they depend on voluntary and unverified contributions [49]. Consequently, care must be taken when designing the system to ensure robustness against any possible adversarial inputs or missing data.

Requirement 4: A forged-origin hijack detection system must be accurate in all attack and peering scenarios.

Internet routing involves various sort of players (e.g., ASes, CDNs, IXPs, as summarized in Fig. 1). Thus, there are many possible attack scenarios. Each induces a different fingerprint on the collected data and may require a different detection scheme. Typically, a densely-connected Tier1 AS that announces many prefixes (among which some on behalf of its customers) and hijacks a prefix owned by a stub AS might be harder to detect than a single-homed stub AS hijacking the prefix owned by another stub AS. Similarly, there are many legitimate peering scenarios (e.g., a remote peering session between two ASes located in different regions of the world) and none should exhibit a high level of false alarms.

5 Overview

We describe DFOH's workflow (§5.1) as well as the functions that it provides (§5.2) and how users can leverage them (§5.3).

5.1 Workflow

DFOH's workflow comprises the following three main components that are executed on a daily basis.

Zooming on the new AS link (§6.1). DFOH processes the publicly available BGP routes and pinpoints AS links that appear for the first time in their AS paths. DFOH zooms on those links because a forged-origin hijack is likely to trigger the appearance of a new AS link—typically the forged AS link that connects the attacker and the victim in case of a Type-1 hijack (we discuss the corner cases in §6.1). While a new AS link is an indicator to detect forged-origin hijacks, triggering an alert whenever a new link appears would result in a high number of inopportune alarms. In fact, legitimate factors such as new peering agreements or backup path activation can result in new AS links being visible too.

Unfortunately, we demonstrate in §7.4 that link prediction algorithms such as SEAL [68] do not translate well to discriminating the malicious new AS links from the real ones. These algorithms fail because they are generic whereas revealing malicious AS links is a problem with specific properties. Thus, DFOH includes the following components that aim to discriminate the fake new AS links from the legitimate ones.

Computing features (§6.2). DFOH uses a set of features that we carefully selected based on the key requirements identified in §4.2 and that are computed with security in mind: They remain correct even with adversarial input. We divide these features into the following four categories:

Topological features allow to quantify the change induced by a new link on the AS topology (following the reasoning in [63]) and pinpoint the suspicious ones (e.g., that do not follow the typical hierarchical structure of the AS topology).

Peering features are peering characteristics such as points of presence within a logical or geographical region that are computed on a per-AS basis. Intuitively, two ASes that exhibit similar peering characteristics have a higher chance to peer.

AS-Path-pattern features indicate whether the AS paths observed from vantage points and that include a new link are relevant based on the supposedly configured routing policies.

Bidirectionality features indicate whether an AS link is observed in both directions, which is a sign of legitimacy.

DFOH uses these features for the following two reasons. First, they enable to consistently (i.e., in all attack scenarios) discriminate whether a new AS link is a legitimate interconnection or caused by a forged-origin hijack (see §7.1). Second, to ensure that DFOH works even when some data is missing and some feature values cannot be precisely computed. We evaluate the relevance of the different categories of features in §7.4, and confirm that when one category is not relevant for a particular attack scenario or missing, the others compensate.

Inferring forged-origin hijacks (§6.3). Finally, DFOH builds an inference model that takes as input an AS link and its computed features and infers whether this AS link is the result of a

forged-origin hijack. DFOH builds the model using the typical training pipeline used for *link prediction* in a graph [45, 68]. A set of existing (legitimate cases) and nonexistent (malicious cases) links are sampled from the AS topology and used as ground truth to train the inference model. Unfortunately, naively using the training pipeline of existing *link prediction* frameworks falls short for detecting forged-origin hijacks in all attack scenarios (we confirm this in §7.4). The problem is that they sample the AS links without taking into consideration the biases observed in the AS topology caused by its hierarchical structure. In fact, they sample the links uniformly at random, which returns samples in which stub-to-stub links are overrepresented and links connecting the highly-connected ASes are underrepresented. This skewed effect is particularly critical when generating the negative sample, i.e., a set of nonexistent links labeled as malicious and used in the training pipeline. Following the recommendation in [66], DFOH builds a balanced negative sample, representative of all possible malicious AS links, to ensure that all the attack scenarios are covered.

5.2 Software functionalities

Ease of deployment and usage. DFOH is open source and can run on a commodity server or a VM. Once installed, DFOH first downloads the different datasets and saves the parsed data in a database, which we make publicly available at `https://dfoh.uclouvain.be/`. It then processes the data and builds inference models on a daily basis.

Real-time and historical detection. DFOH uses the precomputed inference models to detect past and real-time forged-origin hijacks. Upon detection of a new AS link, DFOH only needs to compute a few features before running the inference, which is swift because it relies on a simple model (a random forest). The most time-consuming operation is bootstrapping DFOH, because it needs to build the database and the inference models for many days (for historical detection).

Wide and public detection at no cost for users. DFOH detects and reports forged-origin hijacks for all possible attackers and victims on the Internet, and for any time period. Users can examine the list of suspicious cases and apply filters on them (e.g., to focus on one AS number) at no cost as we publicly disclose them at `https://dfoh.uclouvain.be/`. We describe a few interesting cases in §A.

5.3 Planned usages

Leveraging DFOH locally. Network operators can use DFOH to detect forged-origin hijacks targeting their prefixes. Our evaluation (§7.3) shows that in the median case, a given AS is only involved in zero or one suspicious case in a month, given network operators to ability to manually check each reported case and take the proper countermeasures if necessary. This usage is similar to how users use ARTEMIS [56]. However, unlike ARTEMIS which relies on a list of neighboring ASes provided by the user as well as feeds from local routers, DFOH does not require users to install any software and configure it.

Leveraging DFOH globally. DFOH enables a global detection of forged-origin hijacks, which is useful for the scientific and operational community. For instance, DFOH can help researchers to characterize forged-origin hijacks (e.g., their frequency, scope, or to profile serial hijackers [64]). Additionnally, DFOH helps to globally monitor Internet routing and is complementary to global BGP monitoring systems that detect misorigin (Type-0) hijacks (e.g., [35]), traffic delays, and disconnections (e.g., [37, 38]). Finally, we envision the output of DFOH to be used in the BGP decision process as an alternative to BGPSec and ASPA [9, 44], which may take years to be deployed. Network operators could deprioritize (or drop) suspicious routes over legitimate ones to prevent them from propagating to other networks.

6 Design

In this section, we motivate and explain the design of the DFOH's internal components and algorithms.

Terminology. Throughout this section, we consider the undirected graph $G_{d,k} = (V_{d,k}, E_{d,k})$ as the AS topology inferred at a given day d from the AS paths collected during the k days prior day d. $V_{d,k}$ is the set of ASes, and $E_{d,k} \subseteq V_{d,k} * V_{d,k}$ are the links between the ASes. We also consider that a new link (v_1, v_2) appears at day d, and is visible in a set A of AS paths.

Datasets. We collect BGP routes using BGPKIT [11] from 287 RIS [55] and RouteViews [51] vantage points that we carefully select using MVP [6]. We collect the AS paths that CAIDA uses to build its AS relationship dataset [3]. For now, we collect the data from one IRR (RADb) as it is the only one that makes available archives of its database. We collect daily snapshots of PeeringDB from the CAIDA website [17]. We explain how we clean and combine these datasets in §B.

6.1 Zoom on new AS links

We start by explaining how DFOH detects new AS links.

6.1.1 Under ideal conditions

Consider that an attacker controlling AS a launches a Type-1 hijack on a prefix owned by the victim AS v, with $a, v \in V_{d,k}$. Rapidly after, the BGP vantage points start to observe the hijacked routes and record their AS path. They likely observe different AS paths because they are scattered everywhere on the Internet. Yet, they all observe an AS path that ends with the attacker-to-victim link, here $(a-v)$, which is a new AS link, i.e., $(a-v) \notin E_{d,k}$. This is the case in Fig. 1, where the BGP

routes induced by the forged-origin hijack launched by *AS6* all have an AS path that ends with *AS6-AS9*. DFOH follows ARTEMIS' approach to detect new AS links: It considers the AS topology $G_{d,k}$, with $k = 300$, i.e., sufficiently high to avoid missing existing links, and classifies an observed link (x, y) as new if $(x, y) \notin E_{d,k}$. We detail how DFOH builds $G_{d,k}$ in §B.

6.1.2 In the real world

DFOH has to deal with real world factors, e.g., attackers could advertise carefully manipulated BGP updates to thwart DFOH. We now list the scenarios in which a forged-origin hijack does not create a new AS link, and show that either DFOH includes mechanisms to avoid them or that they occur only when the impact of the attack is greatly limited.

Scenario 1: The attacker hijacks the prefix owned by an AS with which it legitimately peers.

This attack scenario is akin to a route leak. For instance, an attacker announcing a route learned from one provider to another provider is defined as Type-1 route leak according to RFC 7908 [60]. Route leaks are outside the scope of DFOH because there already exists tools that aim to detect them in the wild [10,23,61]. Besides, thwarting DFOH by legitimately peering with its victim makes the attack harder to perform as it requires *(i)* additional resources for the attacker (e.g., being present in a peering facility where its victim is present too) and *(ii)* to convince the victim to peer with it (unless if the victim peers with the route server of an IXP [54]).

Scenario 2: The attacker pollutes DFOH's database by advertising legitimate routes but with fake AS paths.

Past AS path manipulations carried in legitimate routes could pollute the graph $G_{d,k}$ by adding fake AS links, preventing DFOH to classify them as new AS links any longer. DFOH thwarts this scenario by filtering out links that it inferred as fake from past inferences, i.e., these links are not in $G_{d,k}$. Observe that a link incorrectly inferred as supicious can be recurrently inferred as supicious over time, polluting DFOH's output. To prevent incorrect inferences from piling up over time, DFOH only considers past inferences for up to one month, after which fake links are added in $G_{d,k}$ and not considered as new link anymore. This one-month delay gives enough time for operators to examine the suspicious cases and protect their network against a potential future attack. Additionally, DFOH's website provides filters for users to omit these recurrent cases so as to prevent them from polluting the output.

Scenario 3: The attacker announces a fake path that comprises an existing path between the victim and the attacker.

The attacker could prepend a path (ideally the shortest one) from the victim AS to its own AS that exists in the AS topology inferred from the routes collected by the BGP vantage points. For instance, in Fig. 1, the attacker could prepend *AS7 AS5 AS9* to avoid triggering a new link. However, prepending more ASes increases the length of the AS path, inducing a trade-off between *visibility* and *impact* of the attack (which we highlight in §C) and compelling the attacker to significantly reduce the impact of its attack. In fact, Type-1 hijacks are *impactful* (31.3% of the ASes are polluted in the median case) but also *visible* by DFOH (100% induce a new link in the median case) whereas Type-2 are slightly less *visible* (98.8% induce a new link in the median case) but also less *impactful* (1.3% of the ASes are polluted in the median case).

Scenario 4: The attacker ensures that its fake announcements bypass the public BGP route collectors.

Previous works show that this is possible (unless DFOH relies on some private collectors) [50]. However, the attacker must prepend additional AS numbers in the AS paths. As shown in previous works [50,56] (and confirmed in §C), this significantly reduces the impact of the attack by diverting less traffic to the attacker. For instance, the impact of hijack types larger than Type-1 is very limited or negligible (e.g., less than 10% of the ASes see the hijacked route for Type-2 hijacks [56]).

6.2 Features computation

Upon detection of a new link or an explicit user query, DFOH computes feature values. DFOH uses as input the new link and a set of AS paths that include this new link. These AS paths are inferred from the public BGP routes or directly provided by the user. We now explain how DFOH computes the feature values for the four types of features.

6.2.1 Topological features

The topological features aim to quantify how the new link changes the AS topology [63]. Table 2 (Appendix) lists the topological features that DFOH uses to capture different dimensions of the topological changes. The topological features are either relative to a node (node-based) or a pair of nodes (pair-based). DFOH uses seven node-based features that we classify into three categories. The first one quantifies how central and connected a node is in the graph; the second quantifies how connected are the neighboring nodes; and the third quantifies the topological patterns (e.g., triangles) that include the node. We classify the four pair-based features into two categories. The first one measures how close are two nodes based on their neighboring nodes whereas the second measures how close they are using their shortest distance. We omit other topological features as they are either redundant with the selected ones or too slow to compute.

Computing the feature values. DFOH computes the difference induced by the new link on the feature scores. More formally, assume a set F_n of node-based features and a set F_p of pair-based features. The feature values computation

differs depending on the feature type. Note that for each type of feature, DFOH uses $k = 300$ to build the AS topology $G_{d,k}$. *Node-based features:* Consider feature $f_i \in F_n$ and $f_i(x, G_{d,k})$ its score for node x on $G_{d,k}$, with i the feature index in Table 2. The feature value $v(f_i, d, v_1)$ is the difference induced by the new link (v_1, v_2) on the score of feature f_i for node v_1 on day d, and DFOH computes it using the following equation.

$$v(f_i, d, v_1) = f_i(v_1, G_{d,k}) - f_i(v_1, G'_{d,k})$$

$G'_{d,k} = (E'_{d,k}, V'_{d,k})$ is the graph $G_{d,k}$ that includes link (v_1, v_2), that is $E'_{d,k} = E_{d,k} \cup (v_1, v_2)$. DFOH computes the feature values for both nodes v_1 and v_2. Given that there are seven node-based features, the resulting 14-dimensional feature vector $T_{node\_based}(d, v_1, v_2)$ is the following:

$$T_{node\_based}(d, v_1, v_2) = [v(f_0, d, v_1), v(f_0, d, v_2), \ldots, v(f_6, d, v_1), v(f_6, d, v_2)]$$

Pair-based features: Consider feature $f_i \in F_p$ where $f_i(x, y, G_{d,k})$ is its score for the pair of nodes x, y, with i the feature index in Table 2. The feature value $v(f_i, d, v_1, v_2)$ is the difference induced by the new link (v_1, v_2) on the feature score f_i for the pair of node v_1, v_2 at day d, and DFOH computes it using the following equation.

$$v(f_i, d, v_1, v_2) = f_i(v_1, v_2, G_{d,k}) - f_i(v_1, v_2, G'_{d,k})$$

Given that there are four pair-based features, the resulting 4-dimensional feature vector $T_{pair\_based}(d, v_1, v_2)$ is:

$$T_{pair\_based}(d, v_1, v_2) = [v(f_7, d, v_1, v_2), \ldots, v(f_{10}, d, v_1, v_2)]$$

6.2.2 Peering features

The peering features evaluate the likelihood that two ASes peer based on peering information collected from PeeringDB [52] and BGPView [13]. DFOH considers the five peering information listed in Table 1. The first three features stem from the fact that two ASes registered in the same country, connected to the same IXP, or present in the same facility are more likely to peer. The last two features stem from the fact that ASes that are not present in the same facilities but that have point of presence that are geographically close (e.g., same city) are more likely to peer. Of course, these intuitions are not always true and an obvious counterexample is remote peering. Fortunately, the different categories of features compensate between each other so that DFOH remains accurate even when one is less relevant (see §7).

Dealing with adversarial inputs and polluted data. The peeringDB data is sometimes missing because participation is voluntary. Besides, the integrity of the data is unverified and an attacker could populate deceitful peering information. DFOH addresses those two problems with the following strategy. Instead of computing the feature scores for a hypothetical attacker *ASX*, it computes the scores for the neighboring ASes, for which *ASX* has no control over the peering information. In fact, an operator can only update the peering information relative to its own organization. Besides, as ASes often have several neighbors (the average node degree of the AS topology is 12 and the median is 2), focusing on the neighboring ASes helps find relevant peering information even if a few of them do not add peering information into peeringDB.

Index	Description
1	The countries where *ASX*'s neighbors are registered
2	The IXPs to which *ASX*'s neighbors are connected to
3	The facilities to which *ASX*'s neighbors are present
4	The cities of the facilities to which *ASX*'s neighbors are present
5	The countries of the facilities to which *ASX*'s neighbors are present

Table 1: List of peering features used by DFOH along with their description. We consider features computation for *ASX*.

Computing the feature values. Consider the vector $f_{v,i,d}$ that contains information about feature i for node v at day d. For each feature i, DFOH builds two vectors $f_{v_1,i,d}$ and $f_{v_2,i,d}$ based on peering information collected at day d. On September 19, 2022, for features 1 and 5, the vectors have 271 dimensions and each dimension corresponds to one of the 271 countries found in peeringDB. Similarly, for feature 2, the vectors have 944 (number of IXPs) dimensions, whereas for feature 3 they have 3558 (number of facilities) dimensions, and for feature 4 they have 1482 (number of cities) dimensions. The value of $f_{v,i,d}$ at index j is the number of v's neighbors that are in the country/IXP/facility/city that corresponds to index j.

DFOH then normalizes the two vectors $f_{v_1,i,d}$ and $f_{v_2,i,d}$ such that they become comparable even if v_1 and v_2 have a different number of neighbors (normalization operation ∇), and removes indexes for which the values in both vectors are zero (feature reduction operation $\ominus$). Finally, DFOH computes the feature value for the link (v_1, v_2) and a feature i by computing the cosine distance between the two vectors, which quantifies how similar the two vectors are (operation α). We use the cosine distance because we are interested in the direction of these vectors, not in their actual values, which depend on the number of neighbors, an irrelevant information for DFOH. For a given link (v_1, v_2) and a day d, DFOH computes the following 5-dimensional feature vector.

$$P(d, v_1, v_2) = [\alpha(\ominus(\nabla(f_{v_1,1,d}), \nabla(f_{v_2,1,d}))), \ldots, \alpha(\ominus(\nabla(f_{v_1,5,d}), \nabla(f_{v_2,5,d})))]$$

6.2.3 *AS-path-pattern* feature

DFOH uses the AS paths of the hypothetically observed hijacked routes to compute the *AS-path-pattern* features. More precisely, DFOH checks whether the sequence of AS degree

and customer cone size in an observed AS path $p \in A$ is relevant given the following two assumptions. First, as the AS topology exhibits a hierarchical pattern with Tier1 ASes at the top, we expect that ASes higher in the hierarchy exhibit a higher AS degree and customer cone size. Second, because the majority of the inter-domain routing policies follow the Gao-Rexford model [33], we expect the AS paths to have a valley-free pattern. When these two assumptions are valid, the sequence of AS degree and customer cone size in an AS path follows a strong and identifiable up-and-down pattern. For instance, between two stub ASes, we expect the AS degree and customer cone size to increase until the path reaches the Tier1 ASes, and then to decrease until it reaches the destination AS.

Inferring the suspicious AS paths. Unsurprisingly, these two assumptions do not always hold. For instance, a CDN may have a higher node degree than some of its providers. DFOH thus trains an inference model that computes the probability that a sequence of AS degree or customer cone size is legitimate or caused by a forged-origin hijack, based on historical (for the legitimate cases) and artificial (for the hijack cases) data used as ground truth. More precisely, we select a set of existing and nonexistent AS links. The existing links are selected randomly whereas the nonexistent links are selected following the sampling scheme described in §6.3, which ensures that the distribution of the nonexistent links follows the distribution of the existing links. Then, for each existing link, we randomly pick an AS path that includes this AS link and where one end of the link is at the origin. For the nonexistent links, we randomly define the attacker and the victim and pick an existing AS path for which the origin is the hijacker AS and add the victim as a new origin.

Computing the feature values. DFOH trains a random forest on sequences of AS degree and customer cone size inferred from the created AS paths. DFOH finds the best parameters of the random forest using a cross-validated grid search over a parameter grid. The degree of an AS is computed from the AS topology that DFOH builds on a daily basis, and the customer cone size is obtained from ASRank [18]. DFOH computes the following 3-dimensional feature vector.

$$J(d, v_1, v_2, p) = [\rho(d, v_1, v_2, p), \sigma(d, v_1, v_2, p), \tau(d, v_1, v_2, p)]$$

where ρ is the result of the fake AS path inference considering the AS degree for link (v_1, v_2), AS path $p \in A$ and day d. Similarly, σ is the result of the inference for the customer cone size, and τ is the result of the inference when AS degree and customer cone size are combined. We find that combining both AS degree and customer cone size improves the inference on some scenarios where, e.g., CDNs are involved. Note that each feature value is computed using its own independently-trained inference model that DFOH updates every day.

6.2.4 The *bidirectionality* feature

Identifying an AS link in both directions is a strong sign that it is legitimate [56]. However, checking for link bidirectionality is more challenging for DFOH compared to local detection systems such as ARTEMIS [56]. This is because the routes collected by the public BGP vantage points only allow to observe a small fraction of the new AS links as bidirectional. DFOH improves state-of-the-art techniques by combining the information from the public BGP data and the IRR to observe more AS links in both directions. We explain our methodology and demonstrate its *safety* against adversarial inputs and *benefits* in §D.2 due to space constraints. Computing the bidirectionality feature results in the following 1-dimensional feature vector.

$$B(d, v_1, v_2) = [bidir(d, v_1, v_2)]$$

Where $bidir(d, v_1, v_2) = 1$ if the link (v_1, v_2) is bidirectional at day d, else it is equal to zero.

6.3 Inference

We now explain how DFOH runs (§6.3.1) and trains (§6.3.2) its inference model using balanced samples (§6.3.3).

6.3.1 Detecting forged-origin hijacks

After computing the feature values for a new link (v_1, v_2) and the observed AS path $p \in A$ that includes the new link, DFOH concatenates the resulting feature vectors and obtains the following 27-dimensional feature vector.

$$F(d, v_1, v_2, p) = T_{node\_based}(d, v_1, v_2) \oplus T_{pair\_based}(d, v_1, v_2) \\ \oplus P(d, v_1, v_2) \oplus J(d, v_1, v_2, p) \oplus B(d, v_1, v_2)$$

Where $\oplus$ is the concatenation operation. DFOH uses this feature vector as input to its inference model, which is a supervised binary classifier. The classifier relies on a random forest as this algorithm returns a slightly better performance compared to others (e.g., neural networks, decision tree or SVM), is easier to understand, and is fast to train and query.

DFOH refines its inference using many vantage points. A new AS link is often visible from different BGP vantage points, and the AS paths that include this new link may be different. DFOH computes the *AS-path-pattern* features for all these AS paths, runs inferences for this new link and for every observed AS path using the computed *AS-path-pattern* features, and triggers an alarm if half or more of the inferences detect a forged-origin hijack. Observe that DFOH performs well even if only one AS path is used ($|A| = 1$), which is what we use to evaluate DFOH in §7.1.

6.3.2 Training the classifier

DFOH trains its classifier following a supervised training approach used in state-of-the-art link prediction frameworks [36,

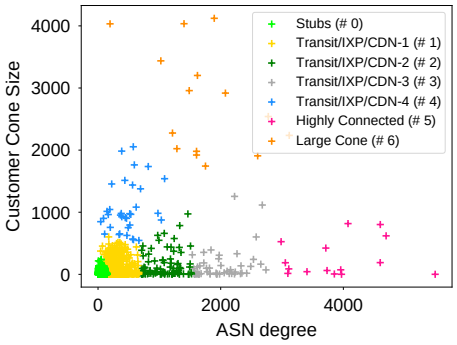


Figure 2: Computed clusters of ASes on April 30, 2022.

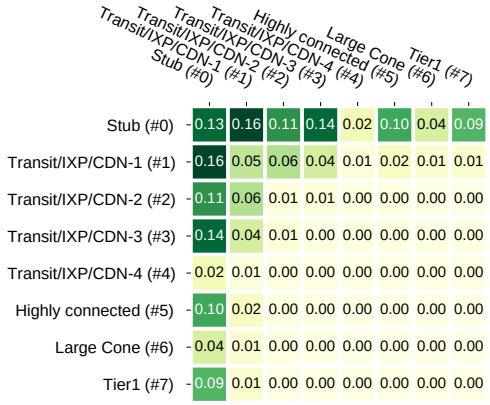


Figure 3: Link distribution within and between clusters. Each cell indicates the proportion (green means high proportion).

68]. From the graph $G_{d,k}$, DFOH samples a set of existing and nonexistent links, i.e., node pairs from the graph which have no link connecting them. DFOH samples 60 000 existing and nonexistent links that it uses to train the random forest. The AS paths presumably observed by the BGP vantage points and that include the new AS link (required to compute the *AS-path-pattern* feature) are generated using the same technique as in §6.2.3. For the training, DFOH estimates the best parameters of the random forest classifier using a cross-validated grid search over a parameter grid on 25% of the sampled links. Then, it trains the random forest classifier using the remaining 75% sampled links and with the optimal parameters. Note that DFOH uses different samples of links to train this classifier and the one used to generate the *AS-path-pattern* features (§6.2.3). DFOH trains and saves a random forest classifier on a daily basis so that it remains accurate over time.

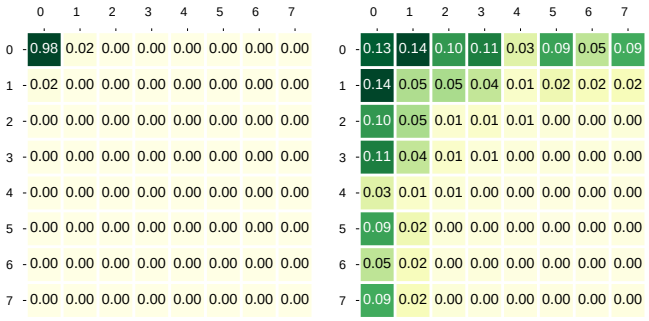
DFOH trains its classifier on representative samples. A key factor for a classifier to be accurate is the quality of the sample used for the training. A good sample should be a representative subset of the scenarios that the model to train will encounter in practice. There is no exception with link prediction frameworks, for which the quality of the samples of existing and nonexistent links plays a key role [67]. In the case of DFOH, a randomly generated sample of existing links turns to be a representative sample (e.g., stub-to-stub links are similarly represented in the initial and sampled set). However, randomly selecting a few nonexistent links returns a sample containing mostly stub-to-stub links and only a few links that involve e.g., transit networks. The sample is skewed because the AS topology is hierarchical, with many more lowly-connected ASes (e.g., stub ASes) than highly-connected ASes (e.g., Tier1 ASes). Thus, DFOH applies a balanced sampling scheme that aims to build a sample of nonexistent links that is representative of all the attacker scenarios.

6.3.3 Sampling all attack scenarios

An attack scenario is defined by the attacker and the victim AS and the link that appears between these two ASes. There exists many categories of ASes (e.g., stub or transit ASes) and within each category, the ASes exhibit different properties (e.g., high/low degree, or a different position in the topology). Sampling all types of attack scenarios is a challenge that DFOH addresses using the following two-steps approach.

Step #1: DFOH clusters ASes based on their degree and customer cone size. DFOH computes the degree of every AS based on the AS topology $G_{d,k}$ and obtains their customer cone size from ASRank [18]. It then uses K-means to cluster the ASes on a daily basis. We omit the Tier1 ASes listed in the CAIDA AS-level map [3] in the clustering but manually class them into their own dedicated "Tier1" cluster. We configure K-means to use the highest number of clusters as long as each cluster contains at least ten ASes. Fig. 2 shows the result of the clustering on April 30, 2022. There are seven clusters to which we assign different labels. For instance, we label the ASes with a low degree and customer cone size (the cluster in the bottom-left corner) as "Stub" ASes. ASes with a high degree but a customer cone size that is not as high are labeled as "Highly Connected" ASes and often correspond to CDNs. "Large Cone" ASes correspond to ASes at the top of the Internet hierarchy. Finally, the remaining ASes can be Transit, IXP, or CDNs and are divided into four clusters.

Step #2: DFOH samples links based on their distribution within and between clusters. Fig. 3 shows the link distribution, within and between clusters, observed in the AS topology. The number at the i-th line and j-th column indicates the proportion of links that connect an AS in cluster i with an AS in cluster j (the grid is symmetric). For instance, 10% of the links are connecting a "Stub" AS with a "Highly connected" AS. While a random sample of the *existing* links returns an inter-cluster link distribution similar to the one obtained when considering all the existing links (i.e., similar to the one in Fig. 3), a random sample of the *nonexistent* links returns a skewed inter-cluster link distribution. This skewed sampling is visible in Fig. 4, which shows the inter-cluster link distribution of the nonexistent links obtained depending on the sampling algorithm used. Most of the randomly sam-



(a) Random sample. (b) DFOH's sample.

Figure 4: Link distribution within and between clusters for the **nonexistent** links sampled using a random (4a) or DFOH's balanced (4b) sampling. The proportion of sampled links is written in each cell. Greener is a cell higher is the proportion.

pled nonexistent links connect two stub ASes because stubs are largely dominant in the AS topology (Fig. 4a). With this skewed sampling, DFOH (and also the state-of-the-art link prediction frameworks) detect forged-origin hijacks involving stub ASes, but performs poorly for other attack scenarios (see §7.4). Consequently, DFOH uses a sampling distribution for the *nonexistent* links that is proportional to power $3/4$ of the inter-cluster distribution of existing links, as suggested by Yang et al. [67]. The resulting sample includes nonexistent links for various inter-cluster scenarios (Fig. 4b)—enabling DFOH to work in all attack scenarios.

7 Evaluation

In this section, we show that DFOH is accurate (§7.1)—even on remote peering sessions (§7.2.1)—and practical (§7.3). We also demonstrate that the design choices of DFOH are relevant (§7.4). In the appendix, we show that DFOH is fast (§E).

Taxonomy. Throughout this section, we focus on the accuracy of DFOH in terms of *True Positive Rate* (TPR) and *False Positive Rate* (FPR). The TPR is the proportion of actual forged-origin hijacks (i.e., *Positive* cases) that are detected by DFOH, whereas the FPR is the proportion of legitimate (i.e., *Negative*) cases incorrectly inferred as malicious.

7.1 Accuracy

The challenge to evaluate the accuracy of DFOH is the lack of ground truth. In fact, while forged-origin hijacks occur in practice, they may not be publicly disclosed, as they negatively impact the victim's reputation. To address this challenge, we evaluate DFOH on both synthetic data and real cases.

Methodology. We measure the TPR and FPR of DFOH using synthetic data. As we have (synthetic) ground truth, we can measure the overall, and specific, performance of DFOH upon various attack scenarios. The synthetic data is built identically to the training data used in §6.3.2, i.e., it is constituted of a sample of (existing and nonexistent) links different from those in the training set. We also run DFOH on the two forged-origin hijacks reported in the news [5, 59] and which are interesting case studies as they involve different types of ASes.

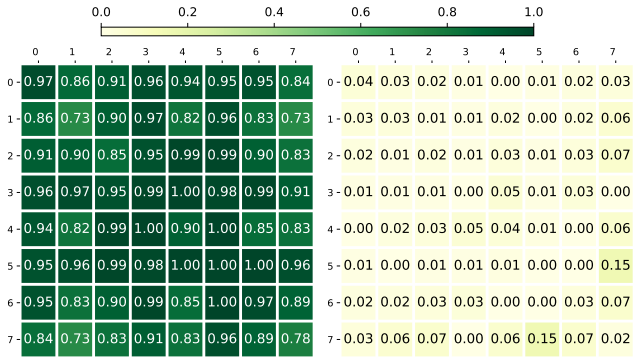
DFOH is accurate in any attack scenario. We measure the accuracy of DFOH on 9000 existing and nonexistent links selected using our balanced sampling (§6.3.3). We use DFOH's inference model computed on April 30, 2022 to align with Fig. 2. DFOH correctly detects 8181 synthetic forged-origin hijacks (TPR=0.909) and incorrectly inferred as forged-origin hijack 171 legitimate AS links (FPR=0.019). We further tested DFOH on various dates (including in 2023) and always found a comparable accuracy. Observe that only one AS path is considered for every inference with the synthetic data. In practice, this scenario happens when only one BGP vantage point sees the hijacked route. DFOH could refine its inference and possibly exhibit an higher TPR and lower FPR with more AS paths, i.e., if more vantage points see the attack (e.g., see §7.2.1).

We now investigate the accuracy of DFOH for specific classes of attack scenarios in Fig. 5. We consider the clusters of ASes as computed in §6.3.3 to classify the attack scenarios and measure the TPR and FPR within and between the clusters. Here, we sample 100 links for every attack scenario, unless fewer than 100 exist, in which case we take all of them (the minimum number of links for a scenario is 58). The number at the i-th line and j-th column indicates the TPR or FPR when the attacker is in cluster i and the victim in cluster j. DFOH is accurate in any attack scenario: The lowest TPR for an attack scenario is 0.73, and occurs when an AS in the "Tier1 (#7)" cluster hijacks another AS in the "Transit/IXP/CDN-1 (#1)" cluster, or when an AS in the "Transit/IXP/CDN-1 (#1)" hijacks an AS either in the same cluster or a "Tier1 (#7)" AS. The highest FPR is only 0.15 and occurs when a "Tier1 (#7)" AS hijacks an AS in the "Highly Connected (#5)" cluster.

DFOH detects actual forged-origin hijacks. DFOH detects[1] the Type-1 hijack launched by AS209243 on AS14618 (one of Amazon's ASes) [5]. Both the victim and the attacker ASes are classified as "Stub" ASes. We use the only available reported AS path: 34854 1299 209243 14618. The fact that DFOH detects this hijack despite the attacker manipulating its IRR data to presumably fake a peering link with AS14618 confirms its robustness against adversarial inputs.

DFOH also detects[2] the Type-1 hijack launched by AS6461 on AS9457 [59]. We used the four distinct AS paths reported in [59] as input. This use case highlights the ability of DFOH to detect forged-origin hijacks that involve central ASes (the attacker is a Tier1 AS and attacked an AS classified as "Stub"), which can be more challenging to detect compared to others.

[1] https://dfoh.uclouvain.be/cases/2022-08-17_14618_209243
[2] https://dfoh.uclouvain.be/cases/2022-02-03_6461_9457



(a) TPR. (b) FPR.

Figure 5: Performance of DFOH upon various attack scenarios. The y- and x-axis indicate the cluster in which the attacker and the victim are classified, respectively.

7.2 Robustness

We measure the accuracy of DFOH on cases that are intuitively harder to classify. We first focus on remote peering sessions (§7.2.1) and then on peering sessions through IXP (§7.2.2).

7.2.1 Remote peering sessions

New *remote* peering sessions are inherently harder to infer as legitimate for DFOH as they exhibit different characteristics than the more common local peering sessions. We now show that DFOH remains accurate in two remote peering scenarios: tunneled sessions and remote peering at IXPs.

BGP tunnels. We measure the accuracy of DFOH on tunneled BGP sessions using a four-steps approach: *(i)* We take four tunnel brokers (Securebit AG, Serveperso, iFog, August Internet); *(ii)* We collect the AS links that they have with their downstream ASes (based on the dataset from `bgpview.io`); *(iii)* We intersect these links with the new AS links observed in 2022; and *(iv)* we run DFOH on the resulting new AS link events (155 events)—each one thus likely corresponds to a new tunneled BGP session. DFOH only reports 12.9% of them as suspicious and all others as legitimate events (FPR=0.129).

We also tested the ability of DFOH to detect forged-origin hijacks launched by ASes connected to the Internet through BGP tunnels. We randomly took 100 victim ASes in every cluster computed by DFOH (see §6.3.2). For every victim, we randomly pick one AS that is a downstream of a tunnel broker and assume it is the attacker. We then take the AS paths observed from the BGP vantage points and for which the assumed attacker is at the origin, and artificially create a forged-origin hijack by adding the victim AS at the origin. DFOH detected 97.5% of these attacks (TPR=0.975).

Finally, we tested DFOH against forged-origin hijacks launched by BGP tunnel providers. We follow the same approach to synthetically generate attacks and found that DFOH detects 72% of them (TPR=0.72).

Remote peering at IXPs. The ground truth in [34] enabled us to identify 42 new remote peering sessions that appeared at several IXPs between January and March 2018. We ran DFOH on them and found that it only classifies eight as suspicious (FPR=0.19). We also measured the ability of DFOH to detect forged-origin hijacks launched by ASes that remotely peer at an IXP. We follow the same approach as with the ASes peering through BGP tunnels (see above) and found that DFOH finds 87.7% of the attacks (TPR=0.877).

7.2.2 Fake peering sessions at IXPs

Fake AS links between two ASes that have an IXP in common are challenging for DFOH as they appear legitimate from a topological and peering perspective.

Methodology. We measure the accuracy of DFOH on 1000 fake links connecting two ASes sharing an IXP but that do not peer through this IXP. We build a fake link using the following methodology. We select an IXP using a weighted random selection where the weight depends on the number of participants of the IXP (i.e., largest IXPs are prioritized). Then, we randomly select two of its participants that do not appear directly connected from a BGP standpoint and define one as the victim and the other as the attacker. Finally, we create a fake AS path similarly as in §7.1, i.e., we take an existing AS path where the attacker is at the origin and add the victim AS as the new origin.

DFOH detects 65.6% of these attacks. We find that it is the *AS-path-pattern* features that enable DFOH to detect most of the attacks. In fact, when omitting this feature category, the accuracy drops to 26.2% whereas when omitting another feature category, DFOH is still able to detect ≈65% of the attacks. This is an intuitive observation as fake AS paths resulting from these attacks are likely to violate the expected routing policies, which the *AS-path-pattern* features can detect. On the contrary, *topological* and *peeringDB* features tend to misclassify fake links at IXPs as two ASes sharing a common IXP could legitimately peer from a topological or peering perspective.

More AS paths improves the accuracy. In fact, when building ten AS paths (instead of one) for every case, the accuracy increases to 73%. This is a logical observation as more paths increase the chance to infer some as being fake.

7.3 Practical use

We now show that DFOH is usable in practice: It pinpoints suspicious events without reporting too many incorrect alarms.

DFOH reports $\approx 13.1\times$ ***fewer suspicious cases than a naive approach.*** We ran DFOH on 2022 and compare it to a naive approach that simply reports all the AS links that appeared during this time frame and have not been observed during the

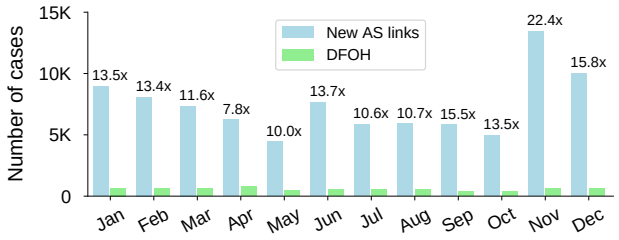


Figure 6: Number of new AS links and reported cases by DFOH for every month of 2022. We indicate the reduction factor at the top of the bars.

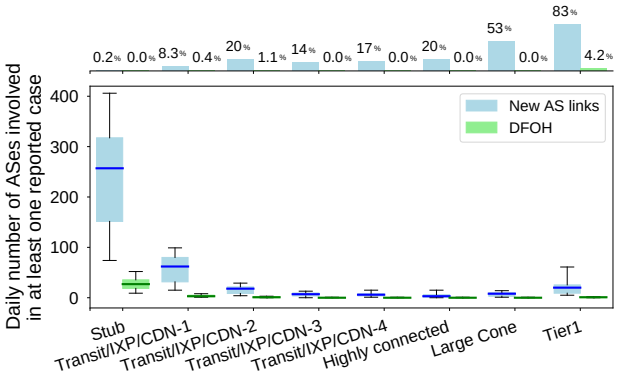


Figure 7: Number of ASes involved in at least one event for every AS category. We show at the top the proportion of ASes involved in at least one reported case every day (median).

last 300 days (to align with DFOH's design choices, see §6.1). As shown in Fig. 6, the number of suspicious cases reported by DFOH remains stable over time. In the median case (resp. 90th percentile), 180 (resp. 390) new links appeared *every day*, and DFOH only classifies 17.5 (resp. 31) of them as being suspicious. In 2022, the reduction factor (i.e., the ratio between the new AS links and the cases reported by DFOH) is 13.1 and ranges between 7.8 (April) and 22.4 (November).

DFOH is practical for each category of ASes. Fig. 7 shows the daily number of ASes involved in at least one event, as a function of their cluster (see §6.3.3). The line in a box depicts the median value; the whiskers show the 5 and the 95th percentile. Regardless of the category of ASes, DFOH always reduces by at least 8.4× the number of reported cases. For instance, in a single day, the naive approach reports 257 stub ASes involved in at least one new link, whereas DFOH only reports 27 stub ASes (median case). Besides, only a tiny fraction of the ASes (max. 4.2% for "Tier1" ASes) are involved in at least one reported case with DFOH every day. In July 2022, we count that the highest number of alarms that an operator can receive is 15, i.e., one every two days. The vast majority (99.8%) would receive zero or one alarm only during the month.

Unsurprisingly, the naive approach detects a higher number of cases involving stub ASes as they are overrepresented compared to others. Yet, when looking at the proportion of ASes involved in at least one reported case (top part), the naive approach seems to perform particularly poorly for the "Large Cone" and "Tier1" ASes as a significant portion of these ASes (53% and 83%, respectively) are involved, every day, in at least one reported case. On the contrary, DFOH exhibits high accuracy for every legitimate peering scenario.

DFOH pinpoints possible forged-origin hijacks. We manually inspected some of the detected suspicious cases and found cases that strongly seem malicious. In §A, we describe four of them and provide all the reported cases in 2022. However, it is impossible to precisely measure the FPR of DFOH when run on real BGP updates because of the lack of ground truth. What makes it even harder is that AS path manipulations are not limited to forged-origin hijacks. Among others, one can poison an AS path by prepending an AS number to prevent a BGP route to go through that AS or can remove AS numbers from an AS path to attract more traffic. DFOH may detect any sort of AS path manipulations that triggers the appearance of a new—but fake—AS link. Since these AS path manipulations often have malicious purposes, we argue that detecting them is useful for the hypothetical victim.

7.4 Relevance of the design choices

In this section, we show that our design choices are relevant.

Every feature category is useful but none is vital. We evaluate DFOH with all the four feature categories activated but one. The resulting TPR and FPR for a few different attack scenarios are reported in Table 3 and further discussed in §E.1. In short, each feature category helps improving the TPR or FPR by a significant factor in at least one attack scenario. We also observe that the accuracy of DFOH never appears particularly flawed when one feature category is missing. Thus, these feature categories compensate each other, allowing DFOH to remain accurate when there is missing or polluted data.

Training DFOH on a balanced sample is necessary. We trained DFOH on a randomly generated sample of existing and nonexistent links instead of using our balanced sampling scheme (r-DFOH in the following) and measured its performance on two synthetic datasets. The first dataset is built using a random sample of existing and nonexistent links, whereas the second is built using our balanced sampling presented in §6.3.3. Unsurprisingly, r-DFOH is accurate on the randomly generated dataset (the TPR is 97.3% and the FPR is 0.3%). However, it performs poorly when tested on the balanced dataset. In fact, the overall TPR is only 49.07%, and the FPR is 0.17%. Besides, r-DFOH is particularly inaccurate for the attack scenarios that are underrepresented in the random sample used for the training, e.g., between "Transit/IXP/CDN-1" ASes and "Tier1" ASes, the TPR is only 1.5%. When trained on a balanced sample, DFOH is accurate regardless of on which dataset it is tested (see §7.1), which demonstrates that

our balanced sampling scheme is a key ingredient to obtain high accuracy in every attack scenario.

DFOH outperforms state-of-the-art link prediction algorithms. We compared DFOH to SEAL, a state-of-the-art link prediction framework that could be an alternative to infer whether a new AS link is legitimate or fake [68]. We run SEAL on the AS topology inferred on August 1, 2022, and configure it to use 20 000 existing and nonexistent links for the training. We then evaluate it on 10 000 other existing and nonexistent links that are selected using our balanced sampling scheme. SEAL returns a TPR of 19.3%, and an FPR of 5.5%, a better accuracy than a random classifier that would return ≈50% of TPR *and* FPR, but a significantly lower accuracy than DFOH.

The accuracy of SEAL turns out to be very skewed (as we show in §E.2) because it uses a random sampling scheme for the training. In fact, the attack scenarios for which SEAL poorly performs are the ones involving the highly-connected ASes, i.e., the ones that are underrepresented in a random sample. We thus implemented *b*-SEAL, a modified version of SEAL that uses our balanced sampling scheme for the training instead. With a TPR of 80.6% and a FPR of 30.8%, *b*-SEAL is still significantly less accurate that DFOH. Yet, we find that its accuracy is more consistent across all attack scenarios. Thus, we conclude that *(i)* the balanced sampling is necessary for a consistent accuracy, regardless of the inference model used, and *(ii)* our selection of features based on domain-specific properties is relevant. We give more details in §E.2.

8 Related work

Misorigin (Type-0) hijacks detection. Prior works that analyze control-plane information to detect MOAS hijacks [19, 35, 42, 58, 65] can detect accidental hijacks but not the malicious ones induced by forged-origin hijacks. Prior works that detect hijacks from data-plane information [16, 70, 71] often can only be deployed per AS, precluding global analysis.

Forged-origin hijacks detection. ARTEMIS detects forged-origin hijacks involving the AS deploying the tool, but cannot be extended for global detection (see §4.1). Cho et al. introduce algorithms based on the AS hegemony [27] to classify reported hijacks as forged-origin hijacks [20]. The proposed global hegemony feature is similar in essence to our *AS-path-pattern* feature. Yet, without our key ingredients, this technique alone results in a low and skewed accuracy when used for *globally detecting* forged-origin hijacks (see §7.4). This is confirmed by the authors themselves, who acknowledge that their algorithm fails to classify hijacks that involve highly-connected ASes such as in the KlaySwap incident [59]. Kruegel et al. propose to detect anomalous BGP updates by combining geographical and topological information about the ASes in the path [40]. However, little is known about how this technique would work to detect forged-origin hijacks.

Link prediction applied to the AS topology. SEAL is a framework for link prediction in a graph but it does not apply well to detect fake AS links (see §7.4) [68]. Giakatos et al. compare link prediction algorithms based on graph-based prediction models on Internet routing data [29]. More precisely, they compute a set of features for every AS and feed them either into a GNN model or a graph embedding model such as bgp2vec [57]. The authors acknowledge that the AS topology and its hierarchical structure is challenging for a GNN or a graph embedding model, and their inference models do not translate well to detecting fake AS links. Finally, Shapira et al. proposed a deep-learning approach with a recurrent neural network based on node embedding computed using bgp2vec [57]. Yet, the performance of the proposed solution is evaluated on the small and biased dataset used in [20].

New protocols and architectures. BGPSec is an extension to BGP where routers cryptographically verify the validity of the AS path [44]. However, it is not deployed *at all*, as it requires expensive cryptographical operations in the routers. ASPA is a proposal to extend RPKI and use it for AS path validation but it is not extensively deployed [9]. Finally, new secure inter-domain protocols and architectures such as SCION [69] are challenging to widely deploy.

9 Conclusion

We present DFOH, the first system that consistently detects forged-origin hijacks on the Internet. DFOH only reports ≈17.5 cases every day, a small number that allows operators to manually investigate each case and take the proper countermeasures. We believe DFOH triggers interesting follow-up works, such as measuring the frequency of these events, profiling the forged-origin hijackers, or analyzing how often the data traffic is diverted to the supposed attacker.

10 Acknowledgments

We are grateful to the NSDI anonymous reviewers for their insightful comments. We are also grateful to our shepherd, Italo Cunha, for its detailed feedback which helped us to improve the quality of the paper. We thank the InetLab platform from the ICube laboratory for providing us a VM which we used to run DFOH.

Thomas Holterbach is partially funded by the Internet Society (through the MANRS Initiative) and by Région Grand Est. Thomas Alfroy is funded by ArtIC project "Artificial Intelligence for Care" (grant ANR-20-THIA-0006-01) and co funded by Région Grand Est, Inria Nancy - Grand Est, IHU of Strasbourg, University of Strasbourg and University of Haute-Alsace. This project has been made possible in part by a grant from the Cisco University Research Program Fund, an advised fund of Silicon Valley Foundation.

References

[1] Amazon once again lost control (for 3 hours) over the IP pool in a BGP Hijacking attack. https://research.securitum.com/amazon-once-again-lost-control-for-3-hours-over-the-ip-pool-in-a-bgp-hijacking-attack.

[2] BGP Hijack of Amazon DNS to Steal Crypto Currency. https://medium.com/oracledevs/bgp-hijack-of-amazon-dns-to-steal-crypto-currency-a90dd29cb3ab.

[3] CAIDA AS relationships. https://catalog.caida.org/dataset/as_relationships_serial_2. 2022-12-1.

[4] Celer Bridge incident analysis. https://www.coinbase.com/blog/celer-bridge-incident-analysis.

[5] Yet another BGP hijacking towards AS16509. https://mailman.nanog.org/pipermail/nanog/2022-August/220320.html.

[6] Thomas Alfroy, Thomas Holterbach, and Cristel Pelsser. MVP: Measuring internet routing from the most valuable points. In *IMC'22*.

[7] BGP, RPKI, and MANRS: 2020 in review. https://blog.apnic.net/2021/02/05/bgp-rpki-and-manrs-2020-in-review/.

[8] Maria Apostolaki, Aviv Zohar, and Laurent Vanbever. Hijacking bitcoin: Routing attacks on cryptocurrencies. In *Security and Privacy*, 2017.

[9] Alexander Azimov, Eugene Bogomazov, Randy Bush, Keyur Patel, Job Snijders, and Kotikalapudi Sriram. BGP AS_PATH Verification Based on Autonomous System Provider Authorization (ASPA) Objects. Internet-draft, 2023.

[10] Marcelo Bagnulo, Alberto García-Martínez, Stefano Angieri, Andra Lutu, and Jinze Yang. Practicable route leak detection and protection with ASIRIA. *Computer Networks*, 2022.

[11] Fast, Extensible, On-premise Global BGP Monitoring. https://bgpkit.com/.

[12] bgpq4 - BGP filtering automation tool. https://github.com/bgp/bgpq4.

[13] BGPview. https://bgpview.io/".

[14] Bias in Internet Measurement Infrastructure. https://ripe84.ripe.net/archives/video/768/.

[15] Henry Birge-Lee, Yixin Sun, Anne Edmundson, Jennifer Rexford, and Prateek Mittal. Bamboozling certificate authorities with BGP. In *USENIX Sec*, 2018.

[16] Tobias Bühler, Alexandros Milolidakis, Romain Jacob, Marco Chiesa, Stefano Vissicchio, and Laurent Vanbever. Oscilloscope: Detecting BGP Hijacks in the Data Plane, 2023.

[17] Daily snapshots of historic PeeringDB data. https://publicdata.caida.org/datasets/peeringdb/.

[18] CAIDA AS Rank. http://as-rank.caida.org/.

[19] Massimo Candela. BGPAlerter, 2019.

[20] Shinyoung Cho, Romain Fontugne, Kenjiro Cho, Alberto Dainotti, and Phillipa Gill. BGP hijacking classification. In *TMA*, 2019.

[21] Taejoong Chung, Emile Aben, Tim Bruijnzeels, Balakrishnan Chandrasekaran, David Choffnes, Dave Levin, Bruce M. Maggs, Alan Mislove, Roland van Rijswijk-Deij, John Rula, and Nick Sullivan. RPKI is Coming of Age: A Longitudinal Study of RPKI Deployment and Invalid Route Origins. In *IMC*, 2019.

[22] CIDR REPORT for 12 Oct 22. https://www.cidr-report.org/as2.0/".

[23] How we detect route leaks and our new Cloudflare Radar route leak service. https://blog.cloudflare.com/route-leak-detection-with-cloudflare-radar/.

[24] B Du, G Akiwate, C Testart, A Snoeren, k claffy, K Izhikevich, and S Rao. IRRegularities in the Internet Routing Registry. In *IMC'23*.

[25] Ben Du, Gautam Akiwate, Thomas Krenc, Cecilia Testart, Alexander Marder, Bradley Huffaker, Alex C. Snoeren, and KC Claffy. IRR Hygiene in the RPKI Era. In *PAM*, 2022.

[26] Ben Du, Cecilia Testart, Romain Fontugne, Gautam Akiwate, Alex C. Snoeren, and kc claffy. Mind Your MANRS: Measuring the MANRS Ecosystem. In *IMC*, 2022.

[27] Romain Fontugne, Anant Shah, and Emile Aben. "The (Thin) Bridges of AS Connectivity: Measuring Dependency Using AS Hegemony". In *PAM*, 2018.

[28] Lixin Gao and Jennifer Rexford. Stable Internet Routing without Global Coordination. ACM SIGMETRICS, 2000.

[29] Dimitrios Panteleimon Giakatos, Sofia Kostoglou, Pavlos Sermpezis, and Athena Vakali. Benchmarking Graph Neural Networks for Internet Routing Data, 2022.

[30] Yossi Gilad, Avichai Cohen, Amir Herzberg, Michael Schapira, and Haya Shulman. Are We There Yet? On RPKI's Deployment and Security. In *NDSS*, 2017.

[31] Yossi Gilad, Sharon Goldberg, Kotikalapudi Sriram, Job Snijders, and Ben Maddison. The Use of maxLength in the Resource Public Key Infrastructure (RPKI). RFC 9319, October 2022.

[32] Yossi Gilad, Omar Sagga, and Sharon Goldberg. MaxLength Considered Harmful to the RPKI. In *CoNEXT*, 2017.

[33] Phillipa Gill, Michael Schapira, and Sharon Goldberg. A Survey of Interdomain Routing Policies. *ACM SIGCOMM CCR*, 2014.

[34] Vasileios Giotsas, George Nomikos, Vasileios Kotronis, Pavlos Sermpezis, Petros Gigis, Lefteris Manassakis, Christoph Dietzel, Stavros Konstantaras, and Xenofontas Dimitropoulos. O Peer, Where Art Thou? Uncovering Remote Peering Interconnections at IXPs. *IEEE/ACM ToN*, 2021.

[35] GRIP. https://grip.inetintel.cc.gatech.edu/.

[36] Aditya Grover and Jure Leskovec. node2vec: Scalable Feature Learning for Networks, 2016.

[37] Internet Health Report. https://ihr.iijlab.net/ihr/en-us.

[38] IODA. https://ioda.inetintel.cc.gatech.edu/.

[39] Internet Routing Registry. https://www.irr.net/.

[40] Christopher Kruegel, Darren Mutz, William Robertson, and Fredrik Valeur. Topology-Based Detection of Anomalous BGP Messages. In *Recent Advances in Intrusion Detection*, 2003.

[41] Craig Labovitz, Abha Ahuja, Abhijit Bose, and Farnam Jahanian. Delayed internet routing convergence. *ACM SIGCOMM CCR*, 2000.

[42] Mohit Lad, Daniel Massey, Dan Pei, Yiguo Wu, Beichuan Zhang, and Lixia Zhang. PHAS: A Prefix Hijack Alert System. In *USENIX Sec*, 2006.

[43] Matt Lepinski and Stephen Kent. An Infrastructure to Support Secure Internet Routing. RFC 6480, February 2012.

[44] Matt Lepinski and Kotikalapudi Sriram. BGPsec Protocol Specification. RFC 8205.

[45] David Liben-Nowell and Jon Kleinberg. The link prediction problem for social networks. In *CIKM*, 2003.

[46] Mutually Agreed Norms for Routing Security (MANRS). https://www.manrs.org/about/.

[47] Not just another BGP Hijack. https://www.manrs.org/2020/04/not-just-another-bgp-hijack/.

[48] MANRS blogpost: BGP security in 2021. https://www.manrs.org/2022/02/bgp-security-in-2021/.

[49] Do we still need the IRR? An analysis and comparison of IRR data across databases. https://ripe85.ripe.net/wp-content/uploads/presentations/71-10-RIPE85-IRRAnalysis.pdf.

[50] Alexandros Milolidakis, Tobias Bühler, Kunyu Wang, Marco Chiesa, Laurent Vanbever, and Stefano Vissicchio. On the Effectiveness of BGP Hijackers That Evade Public Route Collectors. *IEEE Access*, 2023.

[51] University of Oregon. Route Views Project, 2016. www.routeviews.org/.

[52] The Interconnection Database. https://www.peeringdb.com/.

[53] Richard Steenbergen. Examining the validity of IRR data. https://archive.nanog.org/meetings/nanog44/presentations/Tuesday/RAS_irrdata_N44.pdf.

[54] Philipp Richter, Georgios Smaragdakis, Anja Feldmann, Nikolaos Chatzis, Jan Boettger, and Walter Willinger. Peering at Peerings: On the Role of IXP Route Servers. In *IMC'14*.

[55] RIPE RIS Raw Data, 2016. https://www.ripe.net/data-tools/stats/ris/.

[56] Pavlos Sermpezis, Vasileios Kotronis, Petros Gigis, Xenofontas Dimitropoulos, Danilo Cicalese, Alistair King, and Alberto Dainotti. ARTEMIS: Neutralizing BGP Hijacking Within a Minute. *IEEE/ACM ToN*, 2018.

[57] Tal Shapira and Yuval Shavitt. BGP2Vec: Unveiling the Latent Characteristics of Autonomous Systems. *IEEE TNSM*, 2022.

[58] Xingang Shi, Yang Xiang, Zhiliang Wang, Xia Yin, and Jianping Wu. Detecting Prefix Hijackings in the Internet with Argus. In *IMC'22*, 2012.

[59] Aftab Siddiqui. KlaySwap – Another BGP Hijack Targeting Crypto Wallets. https://www.manrs.org/2022/02/klayswap-another-bgp-hijack-targeting-crypto-wallets/.

[60] Kotikalapudi Sriram, Doug Montgomery, Danny R. McPherson, Eric Osterweil, and Brian Dickson. Problem Definition and Classification of BGP Route Leaks. RFC 7908, June 2016.

[61] Shen Su, Beichuan Zhang, Lin Ye, Hongli Zhang, and Nathan Yee. Towards real-time route leak events detection. In *IEEE ICC*, 2015.

[62] Yixin Sun, Anne Edmundson, Laurent Vanbever, Oscar Li, Jennifer Rexford, Mung Chiang, and Prateek Mittal. RAPTOR: Routing Attacks on Privacy in Tor. In *USENIX Sec*, 2015.

[63] Mattia Tantardini, Francesca Ieva, Lucia Tajoli, and Carlo Piccardi. Comparing methods for comparing networks. *Scientific Reports*, 2019.

[64] Cecilia Testart, Philipp Richter, Alistair King, Alberto Dainotti, and David Clark. Profiling BGP Serial Hijackers: Capturing Persistent Misbehavior in the Global Routing Table. In *IMC*, 2019.

[65] He Yan, Ricardo Oliveira, Kevin Burnett, Dave Matthews, Lixia Zhang, and Dan Massey. BGPmon: A Real-Time, Scalable, Extensible Monitoring System. In *Cybersecurity Applications Technology Conference for Homeland Security*, 2009.

[66] Zhen Yang, Ming Ding, Chang Zhou, Hongxia Yang, Jingren Zhou, and Jie Tang. Understanding Negative Sampling in Graph Representation Learning, 2020.

[67] Muhan Zhang and Yixin Chen. Weisfeiler-Lehman Neural Machine for Link Prediction. In *KDD '17*, 2017.

[68] Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks. In *Advances in Neural Information Processing Systems*, 2018.

[69] Xin Zhang, Hsu-Chun Hsiao, Geoffrey Hasker, Haowen Chan, Adrian Perrig, and David G. Andersen. SCION: Scalability, Control, and Isolation on Next-Generation Networks. In *S&P*, 2011.

[70] Zheng Zhang, Ying Zhang, Y. Charlie Hu, Z. Morley Mao, and Randy Bush. ISPY: Detecting IP Prefix Hijacking on My Own. *IEEE/ACM Trans. Netw.*, 2010.

[71] Changxi Zheng, Lusheng Ji, Dan Pei, Jia Wang, and Paul Francis. A Light-Weight Distributed Scheme for Detecting IP Prefix Hijacks in Real-Time. In *ACM SIGCOMM*, 2007.

Appendix

A A sample of the most suspicious cases

We now describe three suspicious cases that DFOH detected in 2022. These cases illustrate that DFOH is exploitable, and that reporting the suspicious cases and notifying the operators is beneficial. Note that we did not manually inspect all the reported cases in 2022. Thus, even more suspicious cases might exist.

January 1, 2022:[3] AS267548, a small Peruvian AS, appears between Sprint (AS1239), a Tier1 AS, and AS199524, a large content provider. However, AS267548 is not supposed to

[3] https://dfoh.uclouvain.be/cases/2022-01-01_1239_267548

provide transit between these two ASes. This is suspicious as AS267548 is not connected to any IXP and not present in any facility according to PeeringDB. Besides, it is only connected to Brazilian ASes according to bgp.tools.

January 30, 2022:[4] This case involves AS138263, a small India AS, which according to an observed AS path provided transit between AS32934 (Meta) and AS1828 (a large global software-defined network) for Meta's prefix `31.13.79.0/24`. This case is suspicious for two reasons. First, the hypotetical victim (Meta) does not share any IXP with the hypothetical attacker (AS32934). Second, its unlikely that a small AS provides transit for two large ASes such as Meta and AS1828.

April 1, 2022:[5] New BGP routes for prefix `14.0.48.0/24`, owned by AS54994, have been observed from ten BGP vantage points. In their AS path, we observe that the (valid) origin AS54994, a CDN based in the US, appears to peer with AS132116, a rather small Indian ISP that only peers with Indian networks. Even though the two ASes are both connected to DE-CIX in Mumbai, these routes are a solid sign of either AS-path manipulation or route leak.

B Datasets

In this section, we give more details on which datasets DFOH downloads, cleans, and combines to build the main data structures that it uses for its inferences. We make publicly available the parsed data that DFOH uses as input for its inferences at the following URL: `https://dfoh.uclouvain.be/`.

Collecting and cleaning up the public BGP data. We use 287 BGP vantage points from RIS [55] and RouteViews [51] to collect BGP updates. These vantage points are selected using MVP [6] to maximize the utility of the collected routes, which is useful as our Ubuntu 20.04 server with 16 CPUs and 64GB of RAM does not have enough resources to collect BGP routes from all the $\approx$ 2500 vantage points in a timely manner. Observe that collecting BGP routes from all the vantage points would improve the accuracy of DFOH. We also use the AS paths that CAIDA uses to update its AS relationships dataset every month and makes publicly available [3]. DFOH applies the following actions to clean up the BGP routes before using them for inferences. First, it removes the redundant and identical AS numbers that appear consecutively due to AS path prepending. Second, it removes private AS numbers. Finally, it removes the AS numbers owned by an IXP. We obtain (and update on a daily basis) the list of IXPs from PeeringDB.

Building the AS topology. DFOH builds the AS topology (the undirected graph $G_{d,k}$ presented in §6) by combining three datasets. First, it builds the graph from AS links observed in the AS paths of all the BGP updates collected from the 287 vantage points during the ten months ($k = 300$ as in ARTEMIS [56]) preceding day k. DFOH takes the data from previous days to include the past and transient links in the AS topology, which prevents incorrectly classifying them as new AS link. Then, it also takes the AS paths observed in the RIB of every one of these 287 vantage points on the first day of the month at which the graph is built. Finally, it also takes the AS paths that CAIDA uses to update its AS relationships dataset [3]. For instance, the AS topology $G_{d,k}$ with d =April 15, 2023, and $k = 30$ (instead of 300 to simplify) is the combination of the AS paths observed between March 15, 2023 and April 14, 2023 in the BGP updates, the AS paths observed in the RIBs on April 1, 2023, and the AS paths used by CAIDA to build their AS relationship dataset on April 1, 2023.

Finding new AS links. To find the new AS links for the day d, DFOH collects the BGP updates from the 287 BGP vantage points observed at day d, parses the AS paths, and checks whether every AS link in the AS paths is in the AS topology $G_{d,k}$. AS links not in $G_{d,k}$ are classified as new.

[4] https://dfoh.uclouvain.be/cases/2022-01-30_32934_138263
[5] https://dfoh.uclouvain.be/cases/2022-02-24_54994_132116

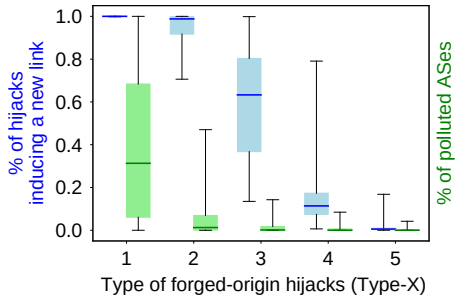


Figure 8: There is a tradeoff between hijack *visibility* (blue boxplots) and *impact* (green boxplots). Most of the impactful forged-origin hijacks are visible to DFOH. A polluted AS is an AS that prefers the hijacked route over the legitimate one.

C Tradeoff between visibility and impact

Aside from prepending the victim AS number in the AS path of the hijacked route, it is hard for an attacker to further manipulate the AS path to thwart DFOH (e.g., scenarios 3 and 4 in §6.1.2). In fact, prepending more ASes increases the length of the AS path, inducing a trade-off between *visibility* and *impact* of the attack, which we highlight in Fig. 8. This figure shows, for the different Type-X hijacks with X ranging from 1 to 5, the proportion of forged-origin hijacks that induce a new AS link (*visibility*) and the number of polluted ASes, i.e., that use the hijacked route (*impact*). Observe that computing the *visibility* of a Type-X hijack boils down to computing the proportion of ASes that are reachable from the attacker's AS in X hops or less, whereas computing the *impact* boils down to measuring for how many ASes the path to the attacker AS is shorter than the path to the victim AS. We compute *visibility* and *impact* for 5000 attacker ASes randomly selected, and for each, we take a random sample of 2000 victims. The results

Type	Categorie	Name	Index	Description
Node-based	Centrality Metrics	Degree centrality	0	Fraction of nodes connected to v
		Closeness centrality	1	Average length of the shortest path between v and all other nodes
		Harmonic centrality	2	Sum of the reciprocal of the shortest path distances from all nodes to v
	Neighborhood Richness	Average neighbor degree	3	Average degree of all the neighbors of v
		Eccentricity	4	Max distance from v to all other nodes
	Topological Pattern	Number of Triangles	5	Number of triangles that include v
		Clustering	6	Fraction of possible triangles including v that exist
Pair-based	Closeness Metrics	Jaccard	7	Similarity between the neighbors of v_1 and v_2
		Adamic Adar	8	Closeness of v_1 and v_2 based on their shared neighbors
		Preferential attachment	9	Likelihood of v_1 and v_2 to be connected based on their degree
	Distance	Shortest Path	10	Length of the shortest path between v_1 and v_2

Table 2: List of topological features used by DFOH along with their description. In the description, we consider for the node-based features a node v in the AS topology whereas we consider two nodes v_1 and v_2 for the pair-based features.

are aggregated in the box plots. Clearly, there is no sweet spot where an attack has high *impact* and low *visibility*. For Type-1 and 2, *impact* is high but *visibility* is high too, and vice versa for Type-3, 4 and 5. An attacker launching a forged-origin hijack thus often cannot prevent the AS path of the hijacked route to include a new AS link—giving DFOH the ability to detect it.

D Features computation (extension)

D.1 Topological features

Table 2 describes the ten topological features that DFOH uses in its inference model. We explain how DFOH computes the feature values in §6.2.

D.2 Bidirectionality feature

Observing an AS link in each direction is a strong sign that it is legitimate. In fact, consider the forged-origin-hijacked route with the AS path $x_1,\ldots,x_n,v_1,v_2$ where v_1 is the attacker and v_2 the forged origin. v_1 can only forge the upstream part of the AS path (i.e., the part on the right side of v_1), and has no control over the downstream part. Note that v_1 could prepend v_2,v_1 on a route to another prefix that it owns to fake a bidirectional link. However, the AS path would contain a loop and would be either denied by BGP routers or easily detectable by BGP monitoring systems. A challenge when assessing the bidirectionality of the AS links is that the AS topology derived from the AS paths in the BGP routes is incomplete (e.g., backup links can be missing). Thus, only a small fraction of the links (≈25 000 , i.e., ≈4.8% of the visible links) are visible in both directions.

Using the IRR data to supplement the BGP routes. DFOH parses the IRR data to infer more peering links that are not visible from the collected BGP routes. More precisely, DFOH parses the `aut-num` objects of every AS in the routing registries. For now, DFOH only uses RADb as it is the only one that makes available archive of its database. However, we envision to use all the registries listed in [39] for real-time detection. An `aut-num` object related to *ASX* may include (partial) information about the export and import policies of *ASX*. These policies generally indicate the AS number or an `as-set` objects to/from which *ASX* is exporting/importing routes. In the case of an `as-set` object, DFOH recursively parses the object (an `as-set` object can include other `as-set` objects) until it finds all the ASes in this `as-set`. With the IRR data, DFOH infers peering information that when combined with BGP data, allows identifying ≈10 000 more bidirectional links compared to with BGP data only.

The bidirectionality feature is beneficial. Even after parsing the IRR data, the number of bidirectional links remains small compared to the total number of AS links. Yet, they are worth the effort because they help DFOH to correctly classify new links as legitimate in some particular peering scenarios. In fact, as the location of BGP collectors is typically skewed with many of them located in the core of the Internet [14], many bidirectional links pertain to highly-connected ASes. We observe the same effect on the bidirectional links inferred from the IRR data, as network operators of the highly-connected ASes tend to populate their IRR data more frequently than others. The bidirectional feature thus improves the accuracy of DFOH upon peering scenarios that involve the highly-connected ASes—scenarios that are hard to accurately classify with the other features (see §E.1).

The bidirectionality feature is safe. The IRR data has two drawbacks: It can be *inaccurate* and it is *unverified*. This is not an issue for DFOH for two reasons. First, the number of possible attacker and victim pairs is $C^2_{74000} \approx 2.74$ billions (74 000 is the number of ASes [22]) whereas the number of inconsistencies in the IRR is by far lower [25]. Consequently, the bidirectional links incorrectly inferred because of these in-

	w/o AS-Path-based		w/o Bidirectionality		w/o Topological		w/o PeeringDB		All Features	
	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR
Stubs - Stubs	85.0 %	4.2 %	97.0 %	4.2 %	95.0 %	4.2 %	87.0 %	2.8 %	**97.0 %**	**4.2 %**
Stubs - Tier1	74.0 %	3.1 %	82.0 %	4.1 %	83.0 %	3.1 %	76.0 %	6.2 %	**83.0 %**	**3.1 %**
Transit/IXP/CDN-2 - Transit/IXP/CDN-3	82.0 %	1.0 %	95.0 %	1.0 %	94.0 %	1.0 %	92.0 %	2.0 %	**95.0 %**	**1.0 %**
Transit/IXP/CDN-3 - Highly Connected	96.2 %	1.0 %	98.1 %	1.0 %	98.1 %	2.0 %	98.1 %	1.0 %	**98.1 %**	**1.0 %**
Transit/IXP/CDN-4 - Tier1	58.0 %	15.6 %	83.0 %	6.2 %	83.0 %	12.5 %	82.0 %	12.5 %	**84.0 %**	**6.2 %**
Large Customer Cone - Tier1	35.0 %	13.3 %	89.0 %	13.3 %	88.0 %	6.7 %	94.0 %	13.3 %	**89.0 %**	**6.7 %**
All types of Links	74.0 %	2.1 %	90.9 %	2.0 %	90.6 %	2.3 %	86.2 %	2.9 %	**90.9 %**	**1.9 %**

Table 3: Accuracy of DFOH for a few selected attack scenarios when all the features but one are used in the inference pipeline.

consistencies only have very little impact on the performance of DFOH. Second, as an attacker can only change the IRR data for its own organization, she can only fake one direction of an AS link (attacker $\rightarrow$ victim)—the same direction as when prepending the victim's AS number in a BGP announcement.

Computing the feature values. Upon a historical request for a day d, DFOH infers link bidirectionality from the AS topology computed on day d (i.e., $G_{d,k}$) combined with the BGP and IRR data collected during the days *following* d (up to 30 days). Considering the following days allows DFOH to find more bidirectional links that only appear e.g., once BGP converges. Upon a real-time query on day d, DFOH considers the graph $G_{d,k}$, and the IRR data collected on day d. Observe that for the bidirectionality feature, $G_{d,k}$ is a *directed* graph. Computing the bidirectionality feature results in the following 1-dimensional feature vector.

$$B(d, v_1, v_2) = [bidir(d, v_1, v_2)]$$

Where $bidir(d, v_1, v_2) = 1$ if the link (v_1, v_2) is bidirectional at day d, else it is equal to zero.

E Detection speed

DFOH automatically downloads all the data and trains the inference model on a daily basis. Upon launching an inference, DFOH uses the inference model trained the day before. Thus, the detection speed depends on *(i)* the time to compute the feature values for the link and AS paths given as input, and *(ii)* the time to run the inference. The inference is fast (<1s) because it relies on a simple random forest. However, the time to compute the feature values depends on whether DFOH is used for real-time detection or to detect past forged-origin hijacks. We now differentiate the two cases.

DFOH detects past forged-origin hijacks in a few seconds. We measured the time needed by DFOH to compute the feature values for all the 18 000 synthetic cases used to evaluate the overall performance of DFOH in 7.1. We use an Ubuntu 20.04 LTS version server with 16 cores and 64 GB of memory. DFOH needs 7510 seconds to compute the feature values (i.e., $\leq 1s$ for a single case). The topological features are the most time consuming to compute (7155 seconds). Observe that the *AS-path-pattern* features are fast to compute, thus DFOH remains fast even when it must compute these features for many different AS paths.

DFOH detects new forged-origin hijacks in a few minutes. The only difference when running DFOH in real time pertains to the bidirectionality feature. In fact, a new peering interconnection may be visible in both directions from the BGP routes only when BGP has converged. As the BGP convergence typically takes a few minutes [41], DFOH waits a few minutes (five, identically to ARTEMIS [56]) to let BGP converge before computing the bidirectionality feature.

E.1 Discriminate power of classification features

In this section, we examine the discriminatory power of the classification features used in DFOH. We show that our selection is sound: Every feature is useful in at least one attack scenario. Besides, none of the features alone is able to detect forged-origin hijack consistently and for all attack scenarios.

Every feature category is useful. Table 3 shows the performance of DFOH when one feature category is deactivated

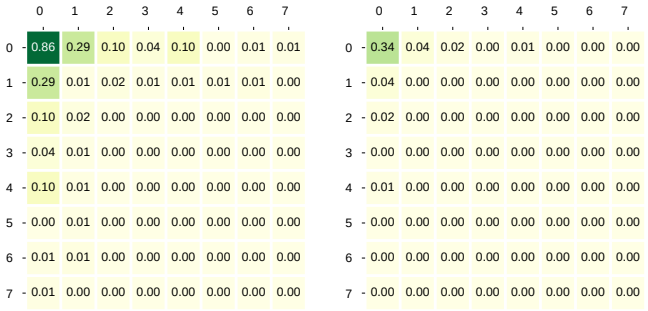
	AS-Path-based		Bidirectionality		Topological		PeeringDB		All Features	
	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR
Stubs - Stubs	83.0 %	2.8 %	0.0 %	0.0 %	79.0 %	5.6 %	84.0 %	4.2 %	**97.0 %**	**4.2 %**
Stubs - Tier1	70.0 %	10.3 %	0.0 %	0.0 %	46.0 %	9.3 %	65.0 %	1.0 %	**83.0 %**	**3.1 %**
Transit/IXP/CDN-2 - Transit/IXP/CDN-3	88.0 %	2.0 %	0.0 %	0.0 %	56.0 %	0.0 %	61.0 %	0.0 %	**95.0 %**	**1.0 %**
Transit/IXP/CDN-3 - Highly Connected	98.1 %	1.0 %	0.0 %	0.0 %	94.2 %	1.0 %	88.5 %	0.0 %	**98.1 %**	**1.0 %**
Transit/IXP/CDN-4 - Tier1	65.0 %	15.6 %	0.0 %	0.0 %	63.0 %	31.2 %	37.0 %	15.6 %	**84.0 %**	**6.2 %**
Large Customer Cone - Tier1	81.0 %	13.3 %	0.0 %	0.0 %	52.0 %	20.0 %	18.0 %	0.0 %	**89.0 %**	**6.7 %**
All types of Links	79.7 %	2.8 %	0.0 %	0.0 %	56.0 %	3.8 %	65.4 %	2.4 %	**90.9 %**	**1.9 %**

Table 4: Accuracy of DFOH for a few selected attack scenarios when only one feature is used in the inference pipeline.

and for a few attack scenarios. We can see that every feature matters: Removing one feature category from the inference pipeline significantly reduces the accuracy of DFOH for at least one attack scenario. The *AS-Path-based* features appear to be the most useful: They are useful in every scenario. The *peeringDB* features also significantly improve the accuracy in many scenarios. For instance, in the "Stubs - Tier1" scenario, they divide the FPR by two. The *topological* features reduces the FPR in the scenarios "Transit/IXP/CDN-3 - Highly connected" and "Transit/IXP/CDN-4 - Tier1" by a factor of two. Finally, the bidirectionality feature is mainly useful to reduce the FPR upon scenarios that involve highly-connected ASes. For instance, it reduces the FPR from 13.3% to 6.7% in the "Large Customer Cone - Tier1" scenario.

DFOH remains accurate when one feature category is missing. Unsurprisingly, removing one feature category from the inference pipeline reduces the accuracy. Yet, the accuracy of DFOH never appears particularly flawed when one feature category is missing. In fact, the minimum FPR when one feature is missing across all the possible scenarios is 15.8%, and the TPR is only below 50% for four out of the $36 * 4 = 144$ scenarios (36 is the number of attack scenarios and 4 is the number of feature categories). Consequently, these feature categories compensate each other, allowing DFOH to remain accurate when there is missing or polluted data.

None of the features alone is enough. Table 4 shows that none of the feature is able to accurately detect forged-origin hijacks when used individually. The *AS-path-pattern* feature returns the best accuracy when used individually, yet performs rather poorly in some cases. For instance, using this feature only returns 15.6% of FPR in the scenario that involves a



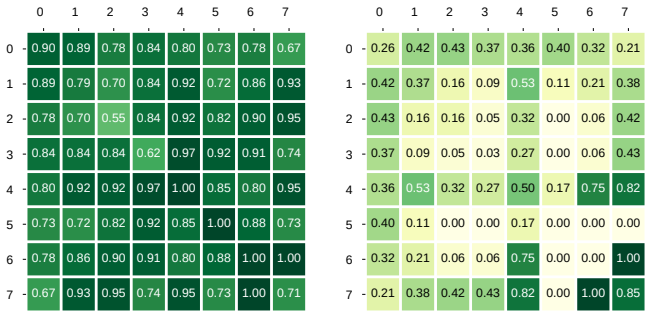
(a) TPR. (b) FPR.

Figure 9: Performance of SEAL on every attack scenario when trained on a randomly generated sample of links.

"Transit/IXP/CDN-4" and a "Tier1" AS. Observe that the *bidirectionality* feature always returns 0% of TPR and FPR. In fact, when used individually in the DFOH inference pipeline, the resulting classifier simply classifies all the links under examination as legitimate. This is the consequence of *(i)* the bidirectionality feature being helpful to assess the legitimacy of a link, but not its maliciousness, and *(ii)* the tiny fraction of AS links observed in both directions (only 5-10% of all the links are bidirectional, see §6.2.4).

E.2 Performance of SEAL for revealing forged-origin hijacks

In this section, we provide more details about the performance of SEAL [68] to detect fake AS links. Fig. 9 shows the performance (in terms of TPR and FPR) of SEAL for every attack



(a) TPR. (b) FPR.

Figure 10: Performance of *b*-SEAL on every attack scenario when trained on a sample of links generated with the balanced sampling scheme used by DFOH.

scenario when trained on a *randomly* generated set of existent and nonexistent links (the default implementation of SEAL). The number at the *i*-th line and *j*-th column indicates the TPR or FPR for the cases that involve an AS in cluster *i*, and an AS in cluster *j*. The grids are symmetric, and the greener a cell higher the rate. Clearly, the accuracy of SEAL is skewed: It is only capable of detecting fake links that involve two "Stub" ASes (the TPR is 0.86 for this attack scenario). For the majority of the other attack scenarios, SEAL classifies the links as legitimate, resulting in a low TPR and FPR.

Fig. 10 shows the performance of *b*-SEAL, a modified version of SEAL where it is trained on a set of existent and nonexistent links generated using our *balanced* sampling scheme (see §6.3.3). The performance of *b*-SEAL is more balanced: The minimum TPR is 0.55, which illustrate that our balanced sampling is useful. Yet, the FPR is also high for many of the attack scenarios, which illustrates that *b*-SEAL does not translate well to revealing fake AS links.