Stargaze: A LEO Constellation Emulator for Security Experimentation

Patrick Tser Jern Kon
pk37@rice.edu
Rice University
Houston, TX, USA

Diogo Barradas
diogo.barradas@uwaterloo.ca
University of Waterloo
Waterloo, ON, Canada

Ang Chen
angchen@rice.edu
Rice University
Houston, TX, USA

Abstract

Low-earth orbit (LEO) satellite constellations are a special type of cyber-physical systems. Their meteoric rise has led to the proposition of many novel use cases and applications. Recent research has also highlighted the broad and unique threat landscape afflicting LEO constellations. However, the CPS security community lacks an experimentation platform to thoroughly identify and explore attacks and their corresponding defenses. We report our experience in building such a platform and perform initial case studies.

CCS Concepts

• **Security and privacy** → **Network security**.

Keywords

LEO constellations, emulation, satellite security

ACM Reference Format:
Patrick Tser Jern Kon, Diogo Barradas, and Ang Chen. 2022. Stargaze: A LEO Constellation Emulator for Security Experimentation. In *Proceedings of the 4th Workshop on CPS & IoT Security and Privacy (CPSIoTSec '22), November 7, 2022, Los Angeles, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3560826.3563382

1 Introduction

"New space" industries are on the rise. The decline of satellite launch costs, enabled by reusable rockets, has significantly lowered the barrier of entry. Several major players, including Starlink [33], Kuiper [50], and Telesat [74], are actively working towards building LEO constellations composed of hundreds to tens of thousands of satellites [33] to grasp this opportunity to deploy novel network services that take advantage of satellites' global, continuous, cost-efficient, and high-capacity connectivity. Starlink, for instance, already provides satellite internet access to 250k users today [68], and aims to reach 42k satellites [33]. In addition, defense agency programs, such as DARPA Blackjack [27], also capitalize on this trend for military purposes.

LEO networks provide substantial benefits over existing networking capabilities. For instance, they achieve much lower latency than existing satellite networks operating in geostationary orbit (GEO), and compete with terrestrial fiber internet in many markets, both in terms of latency [30] and coverage (e.g., serving internet to warzones disconnected from the terrestrial network, as shown in practice in the context of the recent armed conflict between Russia and Ukraine [13]). Further, LEO satellites enable space-native tasks like satellite image processing [44]. These trends have in turn garnered significant interest from academia, resulting in a line of work on LEO computing [4, 6, 64], networking [5, 31, 48], and applications [20, 70].

LEO constellations are a special type of CPS infrastructure and, as such, high-value assets. Just like for critical terrestrial infrastructures – such as electric grids [16, 67] and datacenters [7, 36] – the security of LEO constellations is paramount as they will be prime targets for attacks. With compute, networking, storage, and sensory systems equipped to each satellite, LEO constellations exhibit a similar range of attack vectors. In fact, security concerns are amplified due to the unique characteristics of LEO constellations. Mobility across geographic regions including potentially hostile countries, and the lack of physical perimeters in terrestrial deployments (e.g., datacenter warehouses) lead to further complications. LEO attacks are also harder to detect and remedy, since physical access, intervention, and maintenance are difficult. LEO attacks also threaten to cause disproportionately higher damage, as constellation resources are more scarce and costly compared to terrestrial counterparts (e.g., cloud datacenters). Combined, LEO attacks are a very immediate concern: researchers have demonstrated their feasibility in the lab [29] and attacks have been recently reported in the wild [51].

We believe that the security community has much to offer in protecting the emerging LEO ecosystem. With thorough experimentation and analysis, researchers will be able to adapt tried-and-true defenses to space and rapidly test novel ideas for building solutions against newfound attacks. However, our community is currently handicapped by the lack of a proper security experimentation platform that can accurately emulate LEO constellations, including their orbital properties, networks, and compute capabilities. The utility of emulators for scientific advance is clear from the widespread usage of terrestrial simulation and emulation platforms—e.g., Mininet [46], ns3 [60] or SimGrid [8]. For LEO constellations, however, existing platforms [38, 76] only perform *network-level* simulation. Emulation up the stack, in particular for LEO security experimentation, is not addressed by existing work.

We have been developing a LEO constellation emulator, named Stargaze, for high-velocity security experimentation. By "velocity" [17], we refer to the rapidity of developing and experimenting with attack and defense techniques in an automated and reproducible manner, as well as the easy incorporation of new features to further increase emulation fidelity. By building this emulator, our ultimate goal is to enable two classes of experimentation: a) emerging LEO attacks and defenses (i.e., security for LEO constellations), and b) unique security protections afforded by these satellites (i.e., LEO constellations for security).

Stargaze is a software emulator constructed from familiar and commercial-off-the-shelf (COTS) components (e.g., Kubernetes [42]),

CPSIoTSec '22, November 7, 2022, Los Angeles, CA, USA

ACM ISBN 978-1-4503-9876-3/22/11...$15.00
https://doi.org/10.1145/3560826.3563382

while capturing two unique properties of this CPS infrastructure—*mobility* and *reconfigurability*. First, LEO satellites orbit the earth at ≈ 27,000km/h, and their relative speed difference to the Earth's rotation makes them a mobile infrastructure. A geographic location (e.g., ground stations or user terminals) will be served by a constantly shifting fleet of satellites. Thus, Stargaze needs to continuously calculate the orbital trajectories and geometries of a desired constellation [33, 50, 74]. We develop a technique called *slice emulation* to precisely construct parts of the constellation that serve particular geographic regions, at a given time and as time evolves. This eschews the overhead of emulating entire constellations, making Stargaze a suitable candidate for resource-constrained testbeds.

The need for reconfigurability, on the other hand, stems from the use of inter-satellite links (ISLs) [85]. ISLs enable direct Sat-to-Sat communication, without bouncing data back and forth through intermediate ground stations in what used to be a "bent pipe" architecture [5]. With free-space laser, each satellite can typically establish four high-speed ISLs with neighboring satellites [5, 82]. Moreover, in contrast to standard topologies in cloud datacenters (e.g., Clos [14], or Fat-trees) that are symmetric and static, ISL topologies can be reconfigured [10, 12, 69], and permit a wider range of design choices (e.g., +grid, motif) [5] that can be further morphed to suit different workloads or service regions. ISLs are also a central topic of study in the security realm, where researchers have demonstrated the effectiveness of denial-of-service attacks by congesting selected ISLs [29]. Experimentation with the LEO constellation diversity would thus enable a deeper exploration for LEO security.

While Stargaze is an ongoing effort requiring further development, our current prototype supports experimentation on *security for LEO constellations*, and its source code is available online [41]. A list of potential future work is also included in Section 5.

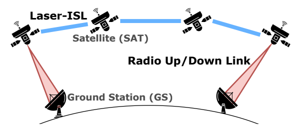
2 Overview and Motivation

With a wide range of use cases, LEO constellations have become critical CPS infrastructure. LEO satellites operate at an altitude orders of magnitude lower compared to their GEO counterparts (<2000 km for LEO vs. 30000 km+ for GEO). The closer range enables LEO satellites to deliver lower-latency services, but at the same time they must orbit the Earth at higher speeds, with a much shorter orbital period, and a much smaller cone of coverage [4]. This results in a mobile infrastructure where each user is only continuously served by a satellite for several minutes. Moreover, each LEO satellite only covers a small fraction of service regions compared to GEO satellites. Thus, a constellation of LEO satellites are needed for full coverage, which are organized in orbital *shells* consisting of many "parallel" *planes*, as shown in Figure 1. A full construction would also equip satellites with ISLs, so that they can communicate with each other without relaying data through ground stations (GSs) in a bent pipe architecture. This enables efficient data transfer, as GS-to-Sat links suffer from tropospheric attenuation, lower bandwidths and higher latency and latency variation [31]. So far, three classes of use cases have been proposed:

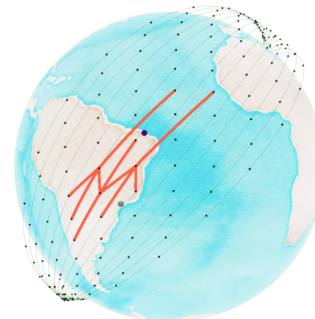
Networking. Delivering network services to underserved (e.g., parts of Africa), unpopulated (e.g., maritime activities), or the battlefield, where terrestrial fiber deployments are difficult. Even for well-connected regions, LEO satellites can provide lower latency than terrestrial fiber, as fiber deployments are usually far longer than geodesic distances and light travels ~47% faster in free space [30].

Num Orbital Planes	72
SATs per Orbital Plane	22
Num SATs	1584
Inclination	53.0°
Altitude	550 km

(a) Constellation param.



(b) GS/Sat communication



(c) Starlink's 1st shell (subset)

Figure 1: (a) Select parameters from Starlink's first Shell. (b) Communication model: GS-to-Sat uses radio links, Sat-to-Sat uses laser links. (c) Subset of Starlink orbital planes (green lines). A possible ISL configuration for some satellites is shown (orange lines) in +grid: a Sat forms two links with its immediate neighbours in the same plane and two links to Sats in adjacent planes. Two GSs: São Paulo (gray), and Fortaleza (purple).

Compute. Equipping satellites with computing hardware and software [4, 24] further enables edge services in LEO constellations, e.g., for processing images and videos captured in space without having to relay them back to the GS. This saves GS-to-Sat link bandwidth and delivers faster reactions to events of interest (e.g., maritime surveillance).

General services. LEO infrastructure further enables general applications, ranging from federated learning on satellite data [70] to wartime communication [27] and remote sensing [20].

Thus, enduing LEO constellations with security properties is paramount as both industry and academic communities explore the benefits of deploying space services.

2.1 The need for security experimentation

We hope to enable research in both *security for LEO constellations*, experimenting with LEO attacks and defenses, and *LEO constellations for security*, exploring the use of LEO constellations for offering better security protections to terrestrial services. Given the nascent state of LEO technologies, this section does not aim to curate a comprehensive list of useful experiments but rather provide example first steps that could foster the community's interest and lead to further developments in the field.

Security for LEO constellations. Recent work has started to distill the broad attack surface of LEO constellations [63, 78]. Amongst existing examples of vulnerability exploitations in LEO constellations [56, 66], a noteworthy endeavour is Icarus [29], a customized DDoS attack that exploits the topology and path structure of LEO constellations to cause traffic congestion at specific ISLs and severely impact the dissemination of legitimate traffic. Icarus can be regarded as the "space equivalent" to what is known as a link-flooding attack [37, 72] in terrestrial networks. Experimentation is needed for further exploration, as Icarus has merely scratched the surface regarding the design of resilient routing schemes in LEO constellations. Briefly, Icarus (a) does not consider emerging capabilities of LEO networks like ISL runtime reconfiguration; (b) focuses on a static +grid ISL topology across the LEO constellation, but ignores alternative yet popular topologies like motif [5];

and (c) does not study effective countermeasures against DDoS attacks, neither in steady state, nor in the presence of satellite failures [80] or intermittent satellite operation [20]. The ability to emulate application-level behavior is important as it increases the range of security experiments we can construct, such as application-level, asymmetric DoS attacks and defenses [11, 19]. We elaborate on these aspects in Section 2.2.

LEO constellations for security. Space-based Earth observation platforms have long provided monitoring and surveillance capabilities that enable both military and civilian agencies to tackle security challenges arising from illegal immigration, transnational crime, maritime piracy, and emergency response [21, 25, 53]. Traditionally, the collection of high-resolution Earth imagery was constrained by limited on-satellite storage, and by the number of opportunities that satellites had to beam images down to ground stations when they come into close range [59, 81]. With ISLs, LEO constellations can bridge this gap by routing satellite imagery towards satellites that are within the range of a ground station [35, 79]. This allows satellite surveillance systems to continuously collect data while speeding up the transmission of satellite imagery to terrestrial datacenters where sensitive surveillance data can be processed and archived. This requires experimentation on (a) the effects of different constellation and ISL topologies on the processing and propagation speed of satellite imagery; (b) throughput variations across different topologies in the event of random and/or correlated satellite failures (e.g., solar flares disabling portions of the constellation); and (c) how QoE may be affected during transient ISL link disruptions due to maneuvering (e.g., to avoid space debris) or satellite power intermittence [20].

2.2 Security threats to LEO constellations

Despite the growing importance of LEO satellite constellations, research has yet to catch up in terms of the identification, evaluation, and mitigation of their security threats. Indeed, while current LEO constellations are prone to a similar range of threats like those affecting terrestrial cyber-physical systems or even other types of satellites (e.g., GEO satellites), these threats' details are markedly distinct. Such differences stem from LEO constellations' mobility (e.g., frequent passes through potentially hostile countries), physical inaccessibility that makes human intervention impossible, comparative lack of standardized regulations guiding satellite cybersecurity [62], increaasingly diverse use cases due to private industry involvement, and the increasingly common use of COTS equipment in these satellites (e.g., giving attackers the ability to analyze vulnerabilities in such equipment). These differences chip away at previous assumptions on space security [45, 55, 62, 77]. The following is a non-comprehensive but important sampling of these potential threats:

Denial-of-service attacks. An attacker could perform signal jamming (i.e., overpowering the signal of a particular frequency with a higher powered signal at the same frequency) to disrupt legitimate radio up/down-link traffic. Real-world examples include Indonesia jamming signals of satellites due to a dispute over orbital slot access with Hong Kong [22], and jamming from non-state actors such as the Tamil Tigers [28]. Other forms of DDoS include volumetric attacks where a massive number of globally distributed bots overwhelm victim endpoints (e.g., ground stations), and link congestion attacks (as demonstrated by Icarus [29]). Such attacks are possible because: satellite positions are public knowledge; bots deployed anywhere can still reach the victim, and; because routing is predictably shortest-path given that path diversity is relatively low in the current constellation ecosystem.

Remote hijacking and malware attacks. In contrast to previous closed-source satellite systems, recent LEO satellites leverage COTS components that have not been explicitly conceived and hardened for deployment in critical infrastructures. This opens up the possibility for attackers to target and exploit vulnerabilities in these hardware and software components, making remote hijacking attacks highly feasible. Further, increasing monetary incentives driven by a deluge of private-industry involvement, which in turn have triggered a cascade of novel space-based applications each with their own potential vulnerabilities, make this form of attack highly likely. Examples of such attacks include hackers commanding unauthorized maneuvers of NASA Satellites [54], the German-US ROSAT x-ray telescope inexplicably changing its orientation to direct its optical sensors at the sun which leads to irreparable hardware damage [23], reuse for adversarial purposes [18, 28, 45], privilege escalation to send flight control commands from payload software applications [15], and conducting replay attacks to reissue intercepted satellite maneuvering commands [62]. Compromised software supply-chains involving software with malware implants, bugs, or vulnerabilities are another well-known threat vector [34] which may compromise satellites with preinstalled software (pre-launch) or software updates (post-launch).

Spoofing attacks. An attacker establishes itself as a trusted user/-client [26] and spoofs packets. This has been demonstrated in the past on satellites providing GPS built with COTS components [49, 75] where GPS receivers were sent spoofed signals, or through attacks that exfiltrate data from compromised computer systems [73].

Physical attacks. Attack vectors include nuclear detonation, directed energy weapons [61, 65] and ground site attacks (e.g. transient loss of control of a ground station) [3, 34]. For example, the 2007 Chinese anti-satellite missile test [58] destroyed a FY-1C weather satellite which resulted in the largest increase in space debris (> 150,000 debris particles) in history. This is especially dangerous in the context of LEO constellations; for instance, a sudden spike in space debris, affecting even a single LEO satellite, could trigger a chain reaction that decimates the constellation [47], if no adequate mitigations are put in place.

2.3 Existing simulators are inadequate

An ideal LEO constellation emulator would be able to a) model the threats outlined in the previous section, b) provide sufficient realism to capture the effect these threats have on network traffic and on actual applications (or arbitrary system-level code) running on any constellation (e.g., in terms of latency, bandwidth, availability, and consistency), c) deploy and evaluate different mitigation strategies, and d) be released in open source to enable researchers to conduct their own experiments. Unfortunately, no existing simulator fulfills all these four requirements.

Orbital simulators: Tools like GMAT [57] and STK [2] are meant for space mission design and navigation, and provide accurate orbit trajectory determination for spacecraft. SaVi [83] can additionally generate orbital coverage of a satellite constellation in 3-D. However, all three do not provide network simulation capabilities, in terms of topology, link-interconnect or measurements.

Satellite network simulators: SNS3 [52] is a simulator built on top of ns-3 [60] that models GEO satellite communication channels, but it cannot model LEO satellites and their ISL topology. The same applies to OpenSAND [71]. The simulator proposed by Henderson and Katz [32] is limited to modeling polar constellations, neglecting non-polar constellations that compose the bulk of existing LEO constellations. On the other hand, the work by Handley [30] is able to simulate a constellation's path trajectories and latency measurements, but lacks the ability to simulate network packet-level behavior, and their software is closed-source.

Hypatia [38] is a LEO constellation network-level simulator and visualizer that is capable of modeling satellite characteristics and the dynamism of space. While it enables network-level simulation effectively, it lacks the ability to deploy real applications or system-level code, as network traffic is simulated and not generated by actual programs. It is also incapable of capturing system-level effects (e.g., the extent to which an adversary-induced fault can have on application throughput) that could be crucial in real-time deployments. Another simulator proposed by Denby and Lucia [20] also suffers from similar drawbacks as Hypatia, and further, it focuses on edge computing on very small power-constrained nano-satellites that eschew the need for online coordination or cross-link communication, and hence is not designed to simulate the intricacies of LEO constellations.

It is worth reemphasizing the importance of thoroughly analyzing threats and testing security improvements to LEO constellations, *prior* to their deployment, given the scale, monetary value, and proliferation of LEO constellations. Such experimentation is relevant not only for current threats but also for future threats that may arise. Furthermore, an emulator that models the characteristics of a wide range of constellation configurations would open up the possibility to develop solutions that are either constellation-specific, or compatible across groups of constellations.

3 The Stargaze Emulator

Stargaze is a security emulator for LEO constellations that fulfills the aforementioned requirements. Our emulator is constructed from COTS software components (e.g., Kubernetes, Linux `tc` [40]) and its architecture is depicted in Figure 2. Stargaze is easily deployable in academic testbeds for high-velocity experimentation. Given a configuration script, Stargaze automatically constructs constellation *slices* that are sufficient for the experimentation. Stargaze also provides a modular platform, where new features could be added to enhance the emulation (e.g., link failure models and signal-to-noise ratio models). To showcase its emulation capability, we perform case studies with ISL attacks and defenses using these features (Section 4). While our current experiments are limited to *security for LEO constellations*, we believe that *LEO constellations for security* experimentation boils down to the use of a similar range of features provided in Stargaze.

3.1 Modular configuration of the emulator

Users of Stargaze provide a *constellation initialization script*, which configures a LEO environment. The configuration includes two types of devices: space devices (satellites), and ground devices (user terminals and ground stations). It also exposes various user-tunable parameters, including selection of satellite constellations (e.g., Starlink's first shell), ISL topologies (e.g., +grid), link bandwidth capacities, geographic locations, emulation timescales, and compute

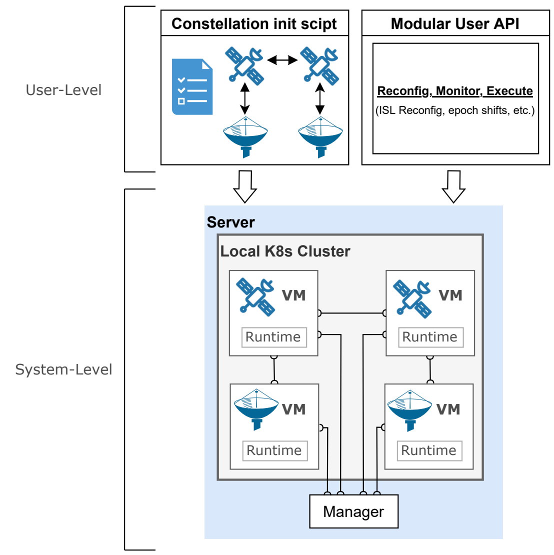


Figure 2: Overview of Stargaze's architecture.

capabilities (e.g., CPU count and available memory). For highly customizable experimentation, users also have the option of providing a configuration of finer granularity, instead of constellation-wide parameters. For example, users can also configure the altitude, inclination, orbit number, number of ISLs and forwarding behaviour of each satellite, and the cone of coverage of each ground device. For our experiments, we obtain the orbital parameters of constellations we emulate from Hypatia (which has in turn obtained them from the FCC or ITU filings made by the respective operators).

Given the above configuration, Stargaze generates the state of each satellite using the two-line element format (a space industry standard) [39], and the GS-to-Sat and Sat-to-Sat connectivity. Stargaze also precomputes, using tools from [38], all variations of link latency (propagation speed is `c`, the speed of light in a vacuum), bandwidth and connectivity before start-up, according to the configured emulation timescale and timestep granularity (which essentially updates on a per time-slot basis). This offline computation improves emulation fidelity by ensuring that computational overhead during the emulation itself is not affected by recomputations of orbital properties.

3.2 Modular constellation slice emulation

To make emulation tractable for large LEO constellations, Stargaze develops the ability to perform *slice emulation* within constrained resources. Based on the user's choice of geographic locations and emulation timescales, Stargaze only emulates a target subset of devices within that constellation with guaranteed emulation fidelity. In other words, constellation slices that are not "visible" to the user are obviated from the emulation. Stargaze also provides a modular user API to accept user-defined reconfiguration and monitoring tasks. The architecture of Stargaze adopts a modular organization, so that new emulation features can be easily developed and added to the platform over time. For instance, Stargaze currently uses separate runtime reconfiguration modules for ISL topology, latency and bandwidth; Stargaze also implements a modular telemetry module for traffic monitoring at each emulated node. Under the hood, the emulated environment mainly consists of the components that are described below:

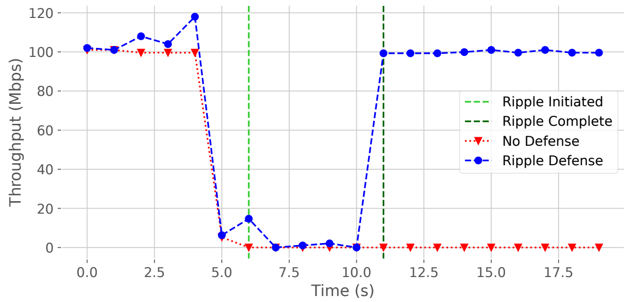


Figure 3: Throughput of benign traffic (100 Mbps) passing through Paris and Madrid, with and without the load dispersion defense (i.e., Ripple).

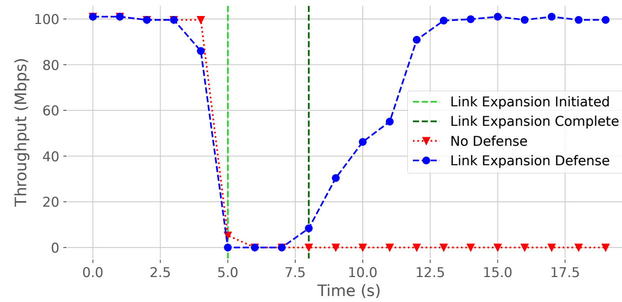


Figure 4: The link expansion defense (same setup).

VM-based emulation: Stargaze runs on a local Kubernetes (k8s) cluster, where each VM (managed by KVM) corresponds to a device (e.g., a LEO satellite, user device, or GS). k8s instantiates the nodes when bootstrapping based on the configuration. Each VM has vCPUs that are pinned to distinct non-overlapping cores, and to a single non-uniform memory access (NUMA) node whenever possible. We use our own CNI-plugin [43] (responsible for implementing the k8s overlay network, allocating and configuring network interfaces) to accommodate non-standard and dynamic device topologies.

Mobile constellations: Unlike GEO satellites, LEO satellites are in constant motion relative to the Earth, which results in variability in both GS-to-Sat and Sat-to-Sat connectivity. To address this, the VM-to-VM topology mirrors the ISL and GS-to-Sat link topologies defined in the initialization script. Each link consists of a virtual interface between two VMs, and they are connected to corresponding TAP interfaces managed by KVM and a single Linux virtual bridge. Mobility of satellites in the space environment is abstracted away from the user, i.e., the user only needs to provide constellation parameters in the initialization script, and Stargaze will automatically supply fluctuations in link availability, latency, and bandwidth that vary across time. Under the hood, Linux `tc` is used to periodically and dynamically set these values based on the aforementioned precomputed values.

Constellation reconfiguration and monitoring: Stargaze provides a modular user API that performs reconfigurations and monitoring. For instance, users can initiate runtime ISL topology reconfigurations. This includes installing or tearing down links dynamically, while ensuring that the properties of these new links (e.g., latency) are updated correctly in subsequent timesteps. Since this is abstracted as an API function call, users can create scripts that automate the process of reconfiguring arbitrary groups of links for experimentation purposes. (This feature is utilized in our case study described in Section 4.) Similarly, they can extract metrics, e.g., interface traffic, from the monitor through the modular user API for their own use or for use by satellite applications.

Constellation manager: Stargaze exposes a constellation-wide management interface at `eth0` of each node within the cluster, for reconfigurations that do not affect the space environment directly, e.g. setting emulation epochs, turning `tc` off or restarting it for debugging purposes, instantiating new system containers, updating VM configuration files, and retrieving system log files. The constellation manager also ensures the correctness of our emulation at the initial bootstrap stage, and during subsequent runtime reconfigurations; these are specified by the user as a set of modular rules, e.g., only ISLs that continuously (during the entire emulation timescale) stay at or above a certain altitude are permissible.

4 Emulating constellation DDoS attacks

This section showcases the capabilities of Stargaze in a case study that aims to analyze the effects of a DDoS attack in a LEO constellation, as well as to evaluate two countermeasures based on unique features of current LEO constellations.

4.1 Emulating ISL link-flooding attacks

Our case study first considers Icarus' [29] single shortest-path routing attack targeted at single ISL links. In this attack, the adversary starts by gathering public knowledge about a constellation's satellite positions [9] and assumes a +grid ISL topology, to create a connectivity graph. Armed with this information, the adversary then crafts traffic patterns that are able to disrupt an ISL's connectivity, while minimizing the amount of traffic required for a successful attack. In our evaluated scenario, using slice emulation, we emulate two ground stations in Paris and Madrid, respectively, with a cluster of six satellites above them, obtained from Starlink's first shell. 100 Mbps of benign traffic is then transmitted between the two ground stations via the LEO constellation. The attacker initiates an ICMP echo request flooding attack; in the absence of link-flooding defenses, she is able to congest the target link with the malicious traffic, reducing the throughput of benign traffic to close to zero (Figure 3 and 4).

4.2 Emulating ISL link-flooding defenses

Icarus implicitly assumes a static +grid ISL topology, where the configuration/connection between satellites is fixed. Using the Stargaze emulator, we explore two defenses against Icarus that leverage flexible constellation topologies and satellite-based ISL telemetry information: *load-aware dispersion*, as demonstrated in Ripple [84] and *dynamic link expansion*, a new defense. By choosing these defenses, we demonstrate the flexibility of the Stargaze platform in supporting experimentations with known and new defense techniques. Both defenses work by removing congestion at the target links so that the link flooding becomes less effective.

Load dispersion: Ripple [84] is a rerouting-based defense that disperses extra traffic from the congested links to other places in the network, thereby relieving the congestion and reducing attack effectiveness. We emulate this defense by installing local monitors at each satellite, which continuously monitor traffic loads in the last epoch, and communicates the latest results to a designated central node (satellite), which will compute the least-utilized paths to other satellites to avoid a statically configured route. Figure 3 shows the effectiveness of the defense. As observed in the plot, the defense is activated at t=6s and the normal traffic throughput is successfully brought back to 100 Mbps.

Link expansion: This defense dynamically reallocates an extra ISL link to the attack target, therefore expanding the link and doubling its bandwidth. The newly expanded link is no longer a bandwidth bottleneck on this topology, so this drastically degrades the attack strength. The monitoring modules are similar as in the previous defense, but the defense actions leverage the topology reconfiguration capability of the emulator. It doubles the bandwidth that is required for launching a successful attack, by taking away bandwidth from other parts of the network. As seen in Figure 4, at t=5s, the defense is activated and link expansion is initiated. This triggers an ISL reconfiguration by establishing a new ISL link between the two end-nodes (satellites) of the link, while disconnecting an existing link that is not under attack. The number of ISLs per satellite remains fixed at an upper bound of four. We set the link expansion setup delay at three seconds (similar to delays observed in [10, 69]). We observe that this brings down the attack strength to a similar degree as in Ripple, recovering normal traffic throughput.

5 Conclusions

LEO constellations comprise an emerging type of cyber-physical systems. Unlike terrestrial CPS, LEO constellations have unique properties, such as infrastructure mobility and network reconfigurability. In this paper, we described our ongoing work on Stargaze, a LEO constellation emulator for security experimentation.

Future work. We envision the development of additional emulation modules that will enable practitioners to leverage Stargaze to experiment with a myriad of advanced security scenarios in LEO satellite constellations. Examples include a) a crash fault module, responsible for emulating satellite crash faults, ISLs failures and/or disconnection [80]; b) a Byzantine fault module, responsible for emulating rogue satellites, e.g., targeted by malware infections [63], and; c) a satellite maneuvering module, responsible for emulating satellite steering capabilities to dodge space debris [1] or re-arrange a satellite's position in orbit [12]. These modules will allow the community to experiment with, for instance, new fault-tolerance models tailored for a range of LEO constellations deployments, or to study the effects of satellite maneuvering on ISL reliability.

We also plan to commence work on *LEO constellations for security* (Section 2.1). One direction would enable experimentation on the performance and resilience of satellite surveillance systems. We could for instance experiment with different LEO constellation topologies that can accelerate the transmission of satellite imagery, or fault tolerance mechanisms that rely on the dynamic reconfiguration of LEO constellations. As a specific example, suppose a fraction of the constellation is knocked out by a solar flare. To maintain adequate situational awareness, an operator may wish to reconfigure the surviving satellites in multiple ways, such as: a) maximize global coverage even if with brief periods of interruption (e.g., by uniformly distributing satellites across orbits), or; b) ensure continuous surveillance over a particular region of interest for a time period (e.g., packing the satellites in a given orbit closer together for uninterrupted visibility over some region of the globe).

Acknowledgments. This work is partially supported by NSF under the grant CNS-1942219.

References

[1] Yehia A Abdel-Aziz. 2013. An analytical theory for avoidance collision between space debris and operating satellites in LEO. *Applied Mathematical Modelling* 37, 18-19 (2013), 8283–8291.

[2] Ansys. 1989. AGI STK. https://www.agi.com/products/stk.

[3] Brandon Bailey, RJ Speelman, PA Doshi, NC Cohen, and WA Wheeler. 2019. Defending spacecraft in the cyber domain. *Aerospace Corp. TR OTR202000016, El Segundo, CA* (2019).

[4] Debopam Bhattacherjee, Simon Kassing, Melissa Licciardello, and Ankit Singla. 2020. In-orbit computing: An outlandish thought experiment?. In *proc. HotNets.*

[5] Debopam Bhattacherjee and Ankit Singla. 2019. Network topology design at 27,000 km/hour. In *Proc. CoNEXT.*

[6] Vaibhav Bhosale, Ketan Bhardwaj, and Ada Gavrilovska. 2020. Toward Loosely Coupled Orchestration for the LEO Satellite Edge. In *Proc. HotEdge.*

[7] Tom Bienkowski. 2018. No Sooner Did the Ink Dry: 1.7 Tbps DDoS Attack Makes History. https://www.netscout.com/blog/security-17tbps-ddos-attack-makes-history.

[8] Henri Casanova. 2001. Simgrid: A toolkit for the simulation of application scheduling. In *Proc. CCGrid.* IEEE.

[9] CelesTrak. 1985. NORAD two-line element sets current data. https://celestrak.com/NORAD/elements/.

[10] Aizaz U. Chaudhry and Halim Yanikomeroglu. 2021. Laser Intersatellite Links in a Starlink Constellation: A Classification and Analysis. *IEEE Vehicular Technology Magazine* 16, 2 (2021), 48–56. https://doi.org/10.1109/MVT.2021.3063706

[11] Ang Chen, Akshay Sriraman, Tavish Vaidya, Yuankai Zhang, Andreas Haeberlen, Boon Thau Loo, Linh Thi Xuan Phan, Micah Sherr, Clay Shields, and Wenchao Zhou. 2016. Dispersing Asymmetric DDoS Attacks with SplitStack. In *Proc. HotNets.*

[12] Xiaoyan Chen and Chao Han. 2011. Reconfiguration of communications constellation with ISLs: fix the net. In *Proc. CECNet.*

[13] Mark Scott Christopher Miller and Bryan Bender. 2022. UkraineX: How Elon Musk's space satellites changed the war on the ground. https://www.politico.eu/article/elon-musk-ukraine-starlink/.

[14] Charles Clos. 1953. A Study of Non-Blocking Switching Networks. *Bell System Technical Journal* (1953).

[15] Nicholas Cohen, Wayne A Wheeler, Roberta Ewart, and Joseph Betser. 2016. Spacecraft embedded cyber defense-prototypes & experimentation. In *AIAA SPACE 2016.* 5231.

[16] The United States Senate Republican Policy Committee. 2021. Infrastructure Cybersecurity: The U.S. Electric Grid. https://www.rpc.senate.gov/policy-papers/infrastructure-cybersecurity-the-us-electric-grid.

[17] Michael Dalton, David Schultz, Jacob Adriaens, Ahsan Arefin, Anshuman Gupta, Brian Fahs, Dima Rubinstein, Enrique Cauich Zermeno, Erik Rubow, James Alexander Docauer, et al. 2018. Andromeda: Performance, isolation, and velocity at scale in cloud network virtualization. In *Proc. NSDI.*

[18] Luca del Monte. 2013. Towards a cybersecurity policy for a sustainable, secure and safe space environment. In *Proceedings of the 64th International Astronautical Congress (IAC).*

[19] Henri Maxime Demoulin, Tavish Vaidya, Isaac Pedisich, Robert DiMaiolo, Jingyu Qian, Chirag Shah, Yuankai Zhang, Ang Chen, Andreas Haeberlen, Boon Thau Loo, Linh Thi Xuan Phan, Micah Sherr, Clay Shields, and Wenchao Zhou. 2018. DeDoS: Defusing DoS with Dispersion Oriented Software. In *Proc. ACSAC.*

[20] Bradley Denby and Brandon Lucia. 2020. Orbital Edge Computing: Nanosatellite Constellations as a New Class of Computer System. In *Proc. ASPLOS.*

[21] Gil Denis, Hélène de Boissezon, Steven Hosford, Xavier Pasco, Bruno Montfort, and Franck Ranera. 2016. The evolution of Earth Observation satellites in Europe and its impact on the performance of emergency response services. *Acta Astronautica* 127 (2016), 619–633.

[22] Mahmood Enayat. 2012. Satellite Jamming In Iran: A War Over Airwaves. *Small Media Report, Kasım* (2012).

[23] Keith Epstein and Ben Elgin. 2008. Network security breaches plague nasa. *Business Week Online–http://www. businessweek. com/magazine/content/08_48* (2008).

[24] Darrell Etherington. 2019. OrbitsEdge partners with HPE on orbital data center computing and analytics. https://techcrunch.com/2019/12/03/orbitsedge-partners-with-hpe-on-orbital-datacenter-computing-and-analytics/.

[25] European Commission. 2016. Space Strategy for Europe. COM(2016) 705.

[26] Gregory Falco and Nicolo Boschetti. 2021. A security risk taxonomy for commercial space missions. In *ASCEND 2021.* 4241.

[27] Stephen Forbes. 2017. DARPA's Blackjack Program. https://www.darpa.mil/program/blackjack.

[28] Jason Fritz. 2013. Satellite hacking: A guide for the perplexed. *Culture Mandala* 10, 1 (2013), 5906.

[29] Giacomo Giuliari, Tommaso Ciussani, Adrian Perrig, and Ankit Singla. 2021. ICARUS: Attacking low Earth orbit satellite networks. In *Proceedings of the*

USENIX Annual Technical Conference. 317–331.
[30] Mark Handley. 2015. Delay is not an option: Low latency routing in space. In *Proceedings of the 14th ACM Workshop on Hot Topics in Networks*. 85–91.
[31] Yannick Hauri, Debopam Bhattacherjee, Manuel Grossmann, and Ankit Singla. 2020. Internet from Space without Inter-Satellite Links. In *Proc. HotNets*.
[32] Thomas Henderson and Randy Katz. 2000. Network simulation for LEO satellite networks. In *18th International Communications Satellite Systems Conference and Exhibit*. 1237.
[33] Caleb Henry. 2019. SpaceX submits paperwork for 30,000 more Starlink satellites. https://spacenews.com/spacex-submits-paperwork-for-30000-more-starlink-satellites/.
[34] Kyle W Ingols. 2017. Design for security: Guidelines for efficient, secure small satellite computation. In *Proc. IMS*. IEEE.
[35] Xiaohua Jia, Tao Lv, Feng He, and Hejiao Huang. 2017. Collaborative Data Downloading by Using Inter-Satellite Links in LEO Satellite Networks. *IEEE Transactions on Wireless Communications* 16, 3 (2017), 1523–1532.
[36] A. L. Johnson. 2016. Mirai: what you need to know about the botnet behind recent major DDoS attacks. https://www.symantec.com/connect/blogs/mirai-what-you-need-know-about-botnet-behind-recent-major-ddos-attacks.
[37] Min Suk Kang, Virgil D Gligor, and Vyas Sekar. 2016. SPIFFY: Inducing Cost-Detectability Tradeoffs for Persistent Link-Flooding Attacks.. In *Proc. NDSS*.
[38] Simon Kassing, Debopam Bhattacherjee, André Baptista Águas, Jens Eirik Saethre, and Ankit Singla. 2020. Exploring the Internet from space with Hypatia. In *Proc. IMC*.
[39] Amiko Kauderer and Kim Dismukes. 2011. NASA HSF: Definition of Two-line Element Set Coordinate System.
[40] Michael Kerrisk. 2001. tc(8) — linux manual page. https://man7.org/linux/man-pages/man8/tc.8.html.
[41] Patrick Tser Jern Kon, Diogo Barradas, and Ang Chen. 2022. Stargaze Source Code. https://github.com/patrickkon/Stargaze. (2022).
[42] Kubernetes. 2014. Kubernetes. https://kubernetes.io/.
[43] Kubernetes. 2022. Kubernetes Network Plugins. https://kubernetes.io/docs/concepts/extend-kubernetes/compute-storage-net/network-plugins/.
[44] Planet Labs. 2010. Planet FAQ. https://www.planet.com/faqs/.
[45] Daria Lane, Enrique Leon, Dexter Solio, Daniel Cunningham, Dmitrie Obukhov, and Francisco C Tacliad. 2017. High-assurance cyber space systems for small satellite mission integrity. (2017).
[46] Bob Lantz and Brandon Heller. 2010. Mininet. http://mininet.org/.
[47] S. Le May, S. Gehly, B.A. Carter, and S. Flegel. 2018. Space debris collision probability analysis for proposed global broadband constellations. *Acta Astronautica* 151 (2018), 445–455. https://doi.org/10.1016/j.actaastro.2018.06.036
[48] Yuanjie Li, Hewu Li, Lixin Liu, Wei Liu, Jiayi Liu, Jianping Wu, Qian Wu, Jun Liu, and Zeqi Lai. 2021. Internet in Space for Terrestrial Users via Cyber-Physical Convergence. In *Proc. HotNets*.
[49] Henrik Lied. 2017. Gps freaking out? maybe you're too close to putin. *NRK beta, September* 18 (2017).
[50] Kuiper Systems LLC. 2020. Application of Kuiper Systems LLC for Authority to Launch and Operate a Non-Geostationary Satellite Orbit System in Ka-band Frequencies.
[51] Stephen Losey. 2022. A top Pentagon official said SpaceX Starlink rapidly fought off a Russian jamming attack in Ukraine. https://www.defensenews.com/air/2022/04/20/spacex-shut-down-a-russian-electromagnetic-warfare-attack-in-ukraine-last-month-and-the-pentagon-is-taking-notes/.
[52] Magister Solutions Ltd. 2011. SNS3. http://sns3.org/content/home.php.
[53] Holger Lueschow and Roberto Pelaez. 2020. *Satellite Communication for Security and Defense*. Springer International Publishing, 779–796.
[54] Bill Malik. 2019. Attack Vectors in Orbit: The Need for IoT and Satellite Security. https://www.rsaconference.com/Library/presentation/USA/2019/attack-vectors-in-orbit-the-need-for-iot-and-satellite-security.
[55] Mark Manulis, Christopher P Bridges, Richard Harrison, Venkkatesh Sekar, and Andy Davis. 2021. Cyber security in new space. *International Journal of Information Security* 20, 3 (2021), 287–311.
[56] Wei Meng, Kaiping Xue, Jie Xu, Jianan Hong, and Nenghai Yu. 2018. Low-latency authentication against satellite compromising for space information network. In *Proc. MASS*.
[57] NASA. 2007. NASA General Mission Analysis Tool (GMAT). https://sourceforge.net/projects/gmat/.
[58] Brendan Nicholson. 2007. The Age: World fury at satellite destruction. https://www.theage.com.au/national/world-fury-at-satellite-destruction-20070120-ge416d.html.
[59] Pat Norris. 2008. *Surveillance Satellites in War and Peace*. Springer Praxis Books, 1–220.
[60] ns3 development team. 2011. The ns3 simulator. https://www.nsnam.org/.
[61] Jesse Oppenheim. 2012. Danger at 700,000 Feet: Why the United States Needs to Develop a Kinetic Anti-Satellite Missile Technology Test-Ban Treaty. *Brook. J. Int'l L.* 38 (2012), 761.
[62] James Pavur. 2021. *Securing new space: on satellite cyber-security*. Ph. D. Dissertation. University of Oxford.
[63] James Pavur and Ivan Martinovic. 2020. SOK: Building a Launchpad for Impactful Satellite Cyber-Security Research. *arXiv preprint arXiv:2010.10872* (2020).
[64] Tobias Pfandzelter, Jonathan Hasenburg, and David Bermbach. 2021. Towards a Computing Platform for the LEO Edge. In *Proc. EdgeSys*.
[65] Peter Pindják. 2016. A Stronger EU in Cosmos: Embracing the Concept of Space Security. *INCAS Bulletin* 8, 3 (2016), 91.
[66] Frederick Rawlins, Richard Baker, and Ivan Martinovic. 2022. Death By A Thousand COTS: Disrupting Satellite Communications using Low Earth Orbit Constellations. *arXiv preprint arXiv:2204.13514* (2022).
[67] Michael Riley. 2022. What Happens When Russian Hackers Come for the Electrical Grid. https://www.bloomberg.com/news/features/2022-01-26/what-happens-when-russian-hackers-cyberattack-the-u-s-electric-power-grid.
[68] Michael Sheetz. 2022. SpaceX adds $25 monthly fee for users to temporarily change Starlink locations. https://www.cnbc.com/2022/05/05/spacex-adds-25-portability-monthly-fee-for-starlink-service.html.
[69] Berry Smutny, Hartmut Kaempfner, Gerd Muehlnikel, Uwe Sterr, Bernhard Wandernoth, Frank Heine, Ulrich Hildebrand, Daniel Dallmann, Martin Reinhardt, Axel Freier, et al. 2009. 5.6 Gbps optical intersatellite communication link. In *Free-space laser communication technologies XXI*, Vol. 7199. International Society for Optics and Photonics, 719906.
[70] Jinhyun So, Kevin Hsieh, Behnaz Arzani, Shadi Noghabi, Salman Avestimehr, and Ranveer Chandra. 2022. FedSpace: An Efficient Federated Learning Framework at Satellites and Ground Stations. *arXiv preprint arXiv:2202.01267* (2022).
[71] Thales Alenia Space. 2006. OpenSAND. https://opensand.org/.
[72] Ahren Studer and Adrian Perrig. 2009. The Coremelt attack. In *Proc. ESORICS*.
[73] Stefan Tanase. 2015. Satellite turla: Apt command and control in the sky. *SecureList. https://securelist.com/satellite-turla-apt-command-and-control-in-the-sky/72081/* (2015).
[74] Telesat. 2022. Telesat satellite operator. https://www.telesat.com/.
[75] Nils Ole Tippenhauer, Christina Pöpper, Kasper Bonne Rasmussen, and Srdjan Capkun. 2011. On the requirements for successful GPS spoofing attacks. In *Proc. CCS*.
[76] Deepak Vasisht, Jayanth Shenoy, and Ranveer Chandra. 2021. L2D2: Low Latency Distributed Downlink for Low Earth Orbit Satellites. In *Proc. SIGCOMM*.
[77] Ted Vera. 2016. Cyber security awareness for SmallSat ground networks. (2016).
[78] Edward Verco. 2021. Satellites are Cyber Insecure: We Need Regulation to Avoid a Disaster. *ANU Journal of Law and Technology* 2, 2 (2021), 57–94.
[79] Luyao Wang and Kwan-Wu Chin. 2018. On Emptying Small Satellite Networks with In-Network Data Aggregation. In *Proc. ITNAC*.
[80] Shaoqing Wang, Youjian Zhao, and Hui Xie. 2019. Pkn: Improving survivability of LEO satellite network through protecting key nodes. In *Proc. CoNEXT, Poster.*
[81] Yu Wang, Min Sheng, Weihua Zhuang, Shan Zhang, Ning Zhang, Runzi Liu, and Jiandong Li. 2018. Multi-Resource Coordinate Scheduling for Earth Observation in Space Information Networks. *IEEE Journal on Selected Areas in Communications* 36, 2 (2018), 268–279.
[82] Lloyd Wood. 2001. *Internetworking with satellite constellations*. Ph.D. thesis – University of Surrey (United Kingdom).
[83] Lloyd Wood. 2002. Satellite constellation visualization (SaVi). https://savi.sourceforge.io/.
[84] Jiarong Xing, Wenqing Wu, and Ang Chen. 2021. Ripple: A Programmable, Decentralized {Link-Flooding} Defense Against Adaptive Adversaries. In *Proc. USENIX Security*.
[85] Ramish Zafar. 2020. SpaceX Successfully Tests Inter-Satellite Starlink Connectivity Via Lasers. https://wccftech.com/spacex-starlink-satellite-laser-test/.