StarryNet
Empowering Researchers to Evaluate Futuristic Integrated Space and Terrestrial Networks

Zeqi Lai, Hewu Li, Yangtao Deng, Qian Wu, Jun Liu, Yuanjie Li, Jihao Li, Lixin Liu, Weisen Liu, Jianping Wu

Presenter: Yangtao Deng
The Future is Up in the Sky

Satellite Internet constellations are under heavy development
The Future is Up in the Sky

Satellite Internet constellations are under heavy development

Thousands of broadband satellites in low earth orbit (LEO)

- 4408 satellites in 5 shells
- 1671 satellites in 2 shells
- 3236 satellites in 3 shells
Integrating LEO satellites with existing terrestrial Internet (ISTN)

Inter-Satellite Link (ISL)  Space backbone network (satellite routers)

Ground-Satellite Link (GSL)

Ground facilities (ground stations, satellite terminals ...)

Integrated Space and Terrestrial Network
**Integrated Space and Terrestrial Network**

Integrating LEO satellites with existing terrestrial Internet (ISTN)

- **Inter-Satellite Link (ISL)**
- **Space backbone network (satellite routers)**
- **Ground-Satellite Link (GSL)**

Ground facilities (ground stations, satellite terminals ...)

Remote Service  |  Rural Education  |  Airplane  |  Global IoT  |  Maritime

Provide pervasive, low-latency, high-bandwidth Internet service
Unique Characteristics of ISTN

Satellites move at a high velocity in the outer space resulting in high LEO dynamics and NEW challenges on the networking stack.

[*] "Internet in Space" for Terrestrial Users via Cyber-Physical Convergence, HotNets'21
Unique Characteristics of ISTN

Satellites move at a high velocity in the outer space resulting in high LEO dynamics and NEW challenges on the networking stack.

Researcher may propose NEW networking technologies to tackle those challenges (e.g. a new ground-satellite integration scheme).

[CDF graph showing differences between Starlink, Kuiper, OneWeb, and Telesat]
Unique Characteristics of ISTN

Satellites move at a high velocity in the outer space resulting in high LEO dynamics and NEW challenges on the networking stack.

Researchers may propose NEW networking technologies to tackle those challenges (e.g., a new ground-satellite integration scheme).

How can researchers build an experimental network environment (ENE) to test, evaluate and understand their new ideas?

[*] "Internet in Space" for Terrestrial Users via Cyber-Physical Convergence, HotNets’21
## ENE Requirements for ISTN Experiments

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>① Constellation Consistency</strong></td>
<td>Spatial and temporal characteristics of a real constellation</td>
</tr>
<tr>
<td><strong>② System and Networking Stack Realism</strong></td>
<td>Run user-defined system codes and network functionalities like in a real system</td>
</tr>
<tr>
<td><strong>③ Flexible and Scalable Environment</strong></td>
<td>Flexibly support various network topologies and diverse test requirements</td>
</tr>
</tbody>
</table>
Problems with Existing ENE Approaches

1. **Constellation Consistency**
   - Spatial and temporal characteristics of a real constellation
   - ✔️

2. **System and Networking Stack Realism**
   - Run user-defined system codes and network functionalities like in a real system
   - ✔️

3. **Flexible and Scalable Environment**
   - Flexibly support various network topologies and diverse test requirements
   - ❌

**Approach I: conducting experiments in a live satellite network**
- Flexibility and scalability are limited
- End-host test only, and it is difficult to conduct various what-if experiments

Benchmarking my new routing protocol upon 4400 LEO satellites? Emm ...
Problems with Existing ENE Approaches

① Constellation Consistency
Spatial and temporal characteristics of a real constellation

② System and Networking Stack Realism
Run user-defined system codes and network functionalities like in a real system

③ Flexible and Scalable Environment
Flexibly support various network topologies and diverse test requirements

Approach II: network simulators
- Realism is limited, since it runs abstractions instead of real applications

STK

GMAT

Hypatia [IMC’20]
StarPerf [ICNP’20]
Problems with Existing ENE Approaches

- **① Constellation Consistency**
  Spatial and temporal characteristics of a real constellation
  - X

- **② System and Networking Stack Realism**
  Run user-defined system codes and network functionalities like in a real system
  - ✓

- **③ Flexible and Scalable Environment**
  Flexibly support various network topologies and diverse test requirements
  - ✓

**Approach III: network emulators**
- VM- or container-based emulation
- Existing emulators can not mimic dynamic behaviors of LEO constellations
- Some of them are also difficult to scale to very large constellation emulation (e.g. thousands of LEO satellites)

**Mininet**

```
> sudo mn
```

DieCast[TOCS’11]: VM-based emulation
Etalon[NSDI’20]: container-based emulation
<table>
<thead>
<tr>
<th>Our Goal</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>① Constellation Consistency</strong></td>
</tr>
<tr>
<td>Spatial and temporal characteristics of a real constellation ✔</td>
</tr>
<tr>
<td><strong>② System and Networking Stack Realism</strong></td>
</tr>
<tr>
<td>Run user-defined system codes and network functionalities like in a real system ✔</td>
</tr>
<tr>
<td><strong>③ Flexible and Scalable Environment</strong></td>
</tr>
<tr>
<td>Flexibly support various network topologies and diverse test requirements ✔</td>
</tr>
</tbody>
</table>

Can we build an ENE simultaneously satisfying all the above requirements?

CHALLENGE ACCEPTED
Our Approach

**StarryNet**: a new evaluation framework for ISTN experiments

**Key idea**: building a data-driven, hybrid ENE

### Public information from real satellite Internet constellations

- **Regulator**: Federal Communications Commission, Consultative Committee for Space Data Systems
- **Satellite operator**: Starlink, project kuiper, OneWeb
- **Ground station operator**
- **User statistics**

### Combining model-based simulation, emulation and satellite hardware

- **Constellation-model-based simulation**
- **Large-scale emulation cluster**
- **Satellite hardware (e.g. low-power processor)**
StarryNet Architecture

System overview

Constellation Observer

Real-World Facilities
- Satellites
- GStations
- Terminals

Constellation Orchestrator

Abstraction (APIs)
- Env-APIs
- Sat-APIs

Constellation Synchronizer

Resource Manager
- Multi-host Resource Allocation
- Efficient State Updater

Physical Machines (Worker Cluster)

Constellation Model
- GS Model
- Network Model

Computation Model

constellation-driven data collection

community-driven data collection

inter-active traffic

setup

result

Researcher

User-Defined Functionalities

Constellation Configurations
StarryNet Design Details

- **Constellation Observer**
  - Crowd-sourcing approach to collect public information
  - Databases to store constellation-relevant data (e.g. constellation elements)
  - Exploiting multidimensional, realistic data to support ENE creation

### Real-World Facilities
- Constellation Observer
  - Satellite Database
  - Ground Station Database
  - User Terminal Database

### Constellation Model
- Constellation Synchronizer
  - Constellation Model
  - GS Model
  - Network Model

### Computation Model
- Efficient State Updater

### Resource Manager
- Multi-host Resource Allocation

### Physical Machines
- (Worker Cluster)

**Community-driven data collection**

**Setup**
- Abstraction (APIs)
  - Env-APIS
  - Sat-APIS

**Result**
- Constellation Orchestrator
  - inter-active traffic

**Databases**
- Crowd-sourcing approach to collect public information
- Databases to store constellation-relevant data (e.g. constellation elements)

**Exploiting multidimensional, realistic data to support ENE creation**
**StarryNet Design Details**

**Constellation Synchronizer**

- **Building a series of models** to characterize ISTN network features
- **Driven by realistic constellation information** and **user-defined experiment requirements** to calculate spatial and temporal behaviors

---

**Real-World Facilities**

- Satellites
- GStations
- Terminals

**Constellation Observer**

- Satellite Database
- Ground Station Database
- User Terminal Database

**Constellation Orchestrator**

- Multi-host Resource Allocation
- Efficient State Updater
- Resource Manager

**Physical Machines**

- (Worker Cluster)

**Community-driven data collection**

**Abstraction (APIs)**

- Env-APIs
- Sat-APIs

**User-Defined Functionalities**

**Constellation Configurations**

**Researcher**

**Setup**

**Result**

**Constellation Synchronizer**

- Constellation Model
- GS Model
- Network Model

**Computation Model**

**Inter-active traffic**

**Experiment config**
StarryNet Design Details

- **Constellation Orchestrator**
  - **Container-based** emulation on physical machines
  - Each container mimic a satellite/ground-station/terminal
  - Support **flexible computation and network capability** in each node

**Abstraction (APIs)**
- Env-APIs
- Sat-APIs

**Constellation Observer**
- Satellite Database
- Ground Station Database
- User Terminal Database

**Real-World Facilities**
- Satellites
- Ground Stations
- Terminals

**community-driven data collection**

**Constellation Model**
- GS Model
- Network Model
- Computation Model

**Constellation Observer**

**Resource Manager**
- Multi-host Resource Allocation
- Efficient State Updater

**Constellation Synchronizer**

**Constellation Configurations**

**setup**

**result**

**inter-active traffic**
StarryNet Design Details

**Constellation Orchestrator**

- **Multi-machine extension** for large-scale mega-constellation
- Leverage VLAN-based traffic isolation to build correct network topology

Incorrect topology without traffic isolation

Correct topology with traffic isolation
Framework Usage: An Example

① Self-defined program

```python
# geo_routing.py
from lib_starrynet import *

def geocast_next_hop(dst_addr):
    # Obtain adjacent satellites info
    n_sats = sn_get_sat_neighbors()
    # Find the sat closest to dst
    for sat in n_sats:
        if dis(sat, dst) < dis(next_sat, dst):
            next_sat = sat
    return next_sat
```

② Configuration file

```json
"starlink": [  
    "name": "SL-Phase-I-shell-I",
    "altitude": "550km",
    "inclination": "53.0",
    "plane_count": "72",
    "satellites_per": "22" ]
```

③ Shell commands

```bash
# listen on manager machine
@manager:/$ sn manager init --m-addr=192.168.0.1
# on each worker machine, join the framework
@worker: /$ sn worker join --m-addr=192.168.0.1
# on manager machine, load manifest files and create the ENE
@manager:/$ sn create --name sl_cons -c 'starlink.json' -gs 'gs.json'
# start the ENE for 3600 seconds
@manager:/$ sn start sl_cons --duration=3600
# run user-specific program in all satellites in the first orbit
@manager:/$ sn cmd sl_cons.orbit[0] python geo_routing.py
```
Evaluation Setup

**StarryNet implementation**
- Eight high-performance DELL R740 servers in a cluster. Each one with 2*Intel Xeon 5222 (4 cores @ 3.8GHz), and 8*32GB DDR RAM
- Based on Docker Container, OpenvSwitch, tc, etc.

**Open data**
- CeleTrak[1] (orbital information), FCC filing ... etc.

**Evaluation and Use Case**
- Ability to satisfy various experimental requirements for ISTNs
- Fidelity analysis
- Case studies

[1] https://www.celestrak.com/
### Various Constellation Configurations

STARRYNET is **flexible** to scale to various constellation configurations with different network topologies.

<table>
<thead>
<tr>
<th>Constellation</th>
<th>Metrics</th>
<th>Height (km)</th>
<th>Constellation Size (number of satellites)</th>
<th>Creation Time (min) Nodes/Links/Total</th>
<th>Avg. CPU (%) Interval = 1/2/3 (s)</th>
<th>Avg. Memory (%) Interval = 1/2/3 (s)</th>
<th>Minimum # of Required Workers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Starlink S1 (72°22, 53°)</td>
<td>550</td>
<td>1584</td>
<td>5.9 4.6 10.5</td>
<td>7.2% 7.0% 6.3%</td>
<td>3.9% 3.5% 3.4%</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Starlink S2 (72°22, 53.2°)</td>
<td>540</td>
<td>1584</td>
<td>5.9 4.6 10.5</td>
<td>7.2% 7.0% 6.3%</td>
<td>3.9% 3.5% 3.4%</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Starlink S3 (36°20, 70°)</td>
<td>570</td>
<td>720</td>
<td>3.0 2.1 4.9</td>
<td>1.2% 1.1% 1.0%</td>
<td>2.7% 2.6% 2.6%</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Starlink S4 (6°58, 97.6°)</td>
<td>560</td>
<td>348</td>
<td>1.9 1.3 3.2</td>
<td>1.0% 1.0% 1.0%</td>
<td>2.7% 2.6% 2.4%</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Starlink S5 (4°43, 97.6°)</td>
<td>560</td>
<td>172</td>
<td>1.6 1.2 3.2</td>
<td>1.0% 1.0% 1.0%</td>
<td>2.3% 2.3% 2.3%</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Starlink Full (4408 satellites)</td>
<td>hybrid</td>
<td>4408</td>
<td>13.3 7.9 21.2</td>
<td>39.6% 37.0% 34.3%</td>
<td>10.4% 9.1% 8.9%</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>Kuiper K1 (34°34, 51.9°)</td>
<td>630</td>
<td>1156</td>
<td>4.4 3.8 8.2</td>
<td>2.6% 2.4% 2.3%</td>
<td>3.8% 3.5% 3.2%</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Kuiper K2 (36°36, 42°)</td>
<td>610</td>
<td>1296</td>
<td>4.7 4.2 8.9</td>
<td>3.9% 3.6% 3.2%</td>
<td>4.0% 3.6% 3.5%</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Kuiper K3 (28°28, 33°)</td>
<td>590</td>
<td>784</td>
<td>3.2 2.4 5.6</td>
<td>1.3% 1.2% 1.2%</td>
<td>2.7% 2.6% 2.6%</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Kuiper Full (3236 satellites)</td>
<td>hybrid</td>
<td>3236</td>
<td>5.7 4.8 10.5</td>
<td>24.6% 23.9% 23.2%</td>
<td>6.3% 6.2% 6.2%</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>Telesat T1 (27°13, 98.98°)</td>
<td>1015</td>
<td>351</td>
<td>1.9 1.3 3.2</td>
<td>1.0% 1.0% 1.0%</td>
<td>2.6% 2.5% 2.4%</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Telesat T2 (40°53, 50.88°)</td>
<td>1325</td>
<td>1320</td>
<td>4.8 4.2 9.0</td>
<td>3.9% 3.7% 3.3%</td>
<td>4.0% 3.6% 3.5%</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Telesat Full (1671 satellites)</td>
<td>hybrid</td>
<td>1671</td>
<td>3.1 2.4 5.5</td>
<td>7.2% 7.0% 6.4%</td>
<td>4.2% 3.7% 3.6%</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>
Fidelity Analysis

Network performance under the same bent-pipe topology compared with live Starlink and other simulation tools.

- **Live Starlink**
- **StarryNet**
- **StarPerf**
- **Hypatia**

**RTT (ms)**
- Average
- 50th
- 70th
- 90th

**Throughput (Mbps)**
- Uplink
- Downlink
StarryNet achieves acceptable fidelity

- **Similar latency performance** to live Starlink measurements
- Accurately emulating the **bandwidth** of a live ISTN
Fidelity Analysis

At this time it is difficult to measure real ISL performance. We analyze the results as compared with other simulators. Similar results but involve additional system-level overhead.

Network performance with ISLs compared with other two simulation tools.

End to end RTT through inter-satellite links (ISL)

- At this time it is difficult to measure real ISL performance
- We analyze the results as compared with other simulators
- Similar results but involve additional system-level overhead
Emerging satellites are equipped with evolved computation capabilities to support various on-board applications.

Orbital edge computing (OEC) uses Jetson TX2 to enable on-board AI capability.

European Space Agency (ESA) uses low-power Raspberry Pi for on-board missions.

StarryNet can be configured to mimic various computation capabilities on-demand.
**Fidelity Analysis**

StarryNet can be configured to mimic various computation capabilities on-demand.
Case Study I: Interconnecting LEO Satellites and Terrestrial Facilities

Exploring the design-space for various space-ground integration methodologies

SRLA: satellite relays for last-mile accessibility

SRGS: satellite relays for ground station networks

GSSN: ground station access for satellite networks

DASN: satellite networks directly accessed by terrestrial users
Case Study I: Interconnecting LEO Satellites and Terrestrial Facilities

StarryNet supports **realistic routing and data transmission** for mega-constellations

<table>
<thead>
<tr>
<th>Design</th>
<th>Average end-to-end latency and its breakdown (ms)</th>
<th>Reachability</th>
<th>Frequent Address Update</th>
<th>Operating Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Inter-Satellite</td>
<td>Space-Ground</td>
<td>Ground</td>
<td>Total</td>
</tr>
<tr>
<td>SRLA</td>
<td>0</td>
<td>76.25</td>
<td>107</td>
<td>183.25</td>
</tr>
<tr>
<td>SRGS</td>
<td>0</td>
<td>313.39</td>
<td>0</td>
<td>313.39</td>
</tr>
<tr>
<td>GSSN</td>
<td>48.46</td>
<td>38.45</td>
<td>20</td>
<td>106.91</td>
</tr>
<tr>
<td>DASN</td>
<td>48.46</td>
<td>37.65</td>
<td>0</td>
<td>86.11</td>
</tr>
</tbody>
</table>

**Conclusions**
- An obvious latency reduction accomplished by ISLs
- Reachability discrepancy caused by handovers and uneven GS distributions
- Deployment and costs vary a lot
Case Study II: Hardware-in-the-loop Testing

STARRYNET supports a hybrid deployment and evaluates real system effects for user-defined functionalities.

- A number of virtual, emulated nodes + 1 real prototype
- StarryNet Virtual Satellites on a R740 cluster

Evaluate system-level effects of a new ISTN network protocol or functionality:

- Link advertisement overhead of a new routing protocol
- Power consumption
- CPU usage
- Memory overhead ... ...

Interactive ISTN Traffic

Power Monitor

CubeSat

On-board Processor

Power measurement

<table>
<thead>
<tr>
<th>State</th>
<th>Idle</th>
<th>Routing convergence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transmission rate (Mbps)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>250</td>
<td>500</td>
</tr>
<tr>
<td>Power (W)</td>
<td>2.83</td>
<td>3.22</td>
</tr>
</tbody>
</table>
CONTENTS

1 Motivation
2 StarryNet Design
3 Evaluation and Use Case
4 Conclusion
Conclusion

- Existing tools fail to guarantee realism, flexibility, and low-cost simultaneously

- StarryNet is able to achieve the goal by
  - Integrating real constellation-relevant information, orbit analysis, etc.
  - Container-based large-scale emulations
  - Low-cost usage and open APIs

- Evaluation results show that StarryNet
  - Achieves high-fidelity to real measurements
  - Supports various ISTN experiments flexibly
Thank you!

Q&A

Zeqi Lai, Hewu Li, Yangtao Deng, Qian Wu, Jun Liu, Yuanjie Li, Jihao Li, Lixin Liu, Weisen Liu, Jianping Wu

Contact
- https://github.com/SpaceNetLab/StarryNet
- zeqilai@tsinghua.edu.cn
- yangtaodeng@gmail.com

Read our paper!