Infiltrating the Sky: Data Delay and Overflow Attacks in Earth Observation Constellations

Xiaojian Wang[1], **Ruozhou Yu**[1], Dejun Yang[2], Guoliang Xue[3]

[1] North Carolina State University

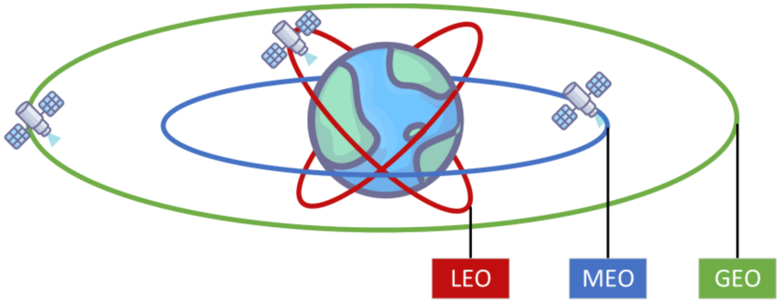
[2] Colorado School of Mines

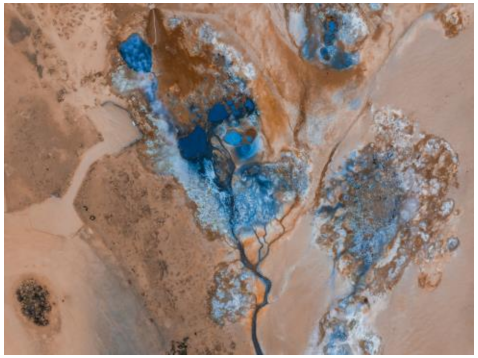
[3] Arizona State University

NC STATE UNIVERSITY

Low Earth Orbit Earth Observation Constellation

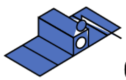
- Low Earth Orbit (LEO) Satellite
  - Near Earth, altitudes < 2000km
- Earth Observation (EO) Satellite
  - Monitoring Earth's surface
- LEO EO Constellation
  - Enabling continuous imaging of the entire Earth's surface
- Applications
  - Agriculture, forestry, urban planning, and disaster management





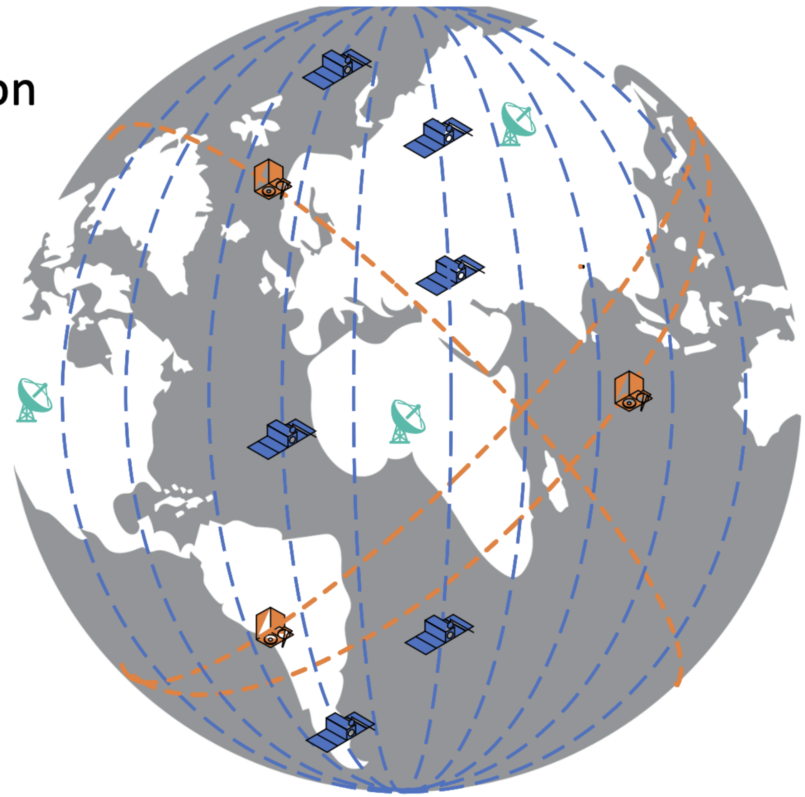
Ref:https://earthobservatory.nasa.gov/blogs/earthmatters/2024/09/17/september-puzzler-10/

Examples of EO constellations



Dove constellation

Continuous board-area monitoring
Medium resolution
~200 satellites
> 120TB per day



SkySat constellation

User scheduled AoI imaging
High resolution
21+ satellites
Flexible task scheduling

EO constellations operated by Planet Labs

New Attack Surface

- Satellite downlink bottleneck
  - Limited number and location of ground stations
  - Small transmission windows
  - Limited transmission bandwidth
- Constellations collaboration and competition
  - Share limited downlink resources
- Opportunity for attack
  - Users can schedule high-priority imaging and downlinking tasks at dedicated times and locations, causing intentional downlink competition with low-priority constellation

Exploiting downlink competition to disrupt a low-priority satellite's data downlink

Motivation and Main Idea

- Data delay attack
  - delaying the downlink of target data
- Data overflow attack
  - dropping target data

Motivation

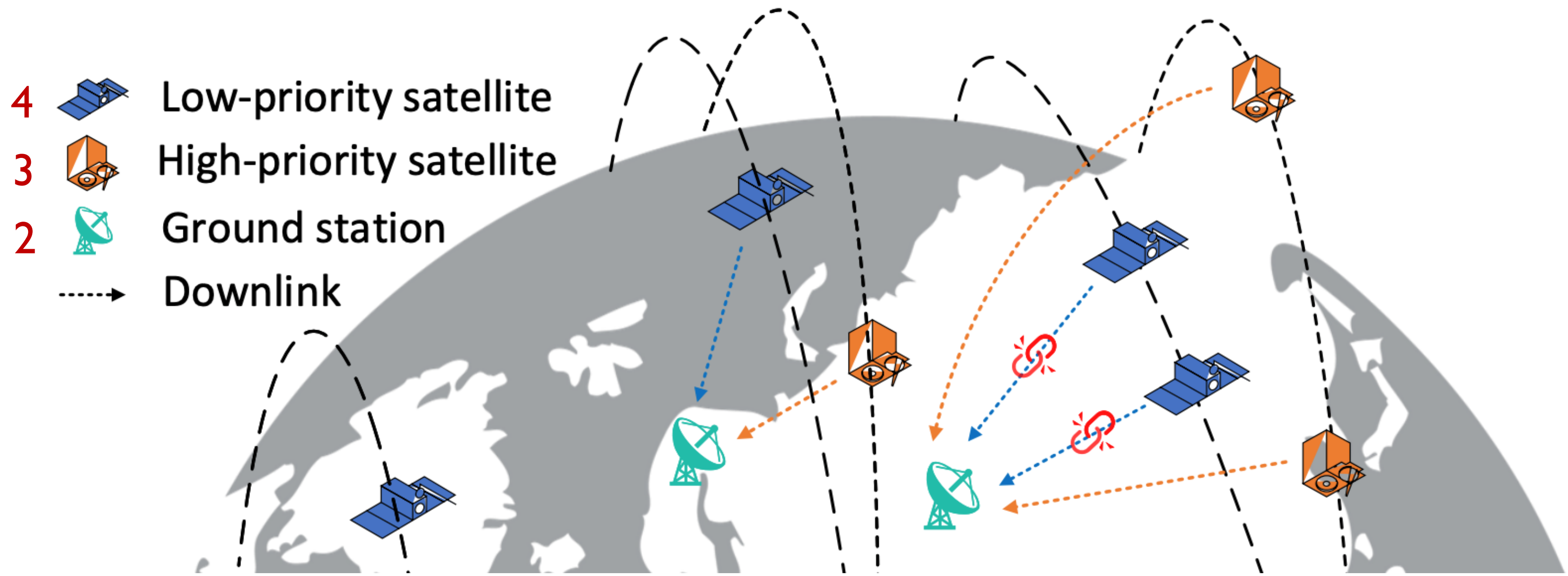
- Prevent downlink and analysis of sensitive information
  - warfare strategies
  - illegal operations

Strategy

- An attacker can ***inject high-priority requests*** to preempt low-priority data downlink windows.
- By utilizing predictable satellite dynamics, an attacker can intelligently target ***critical data*** from low-priority satellites.

EO Constellation In Operation

❑ EO Constellations in Low Earth Orbit

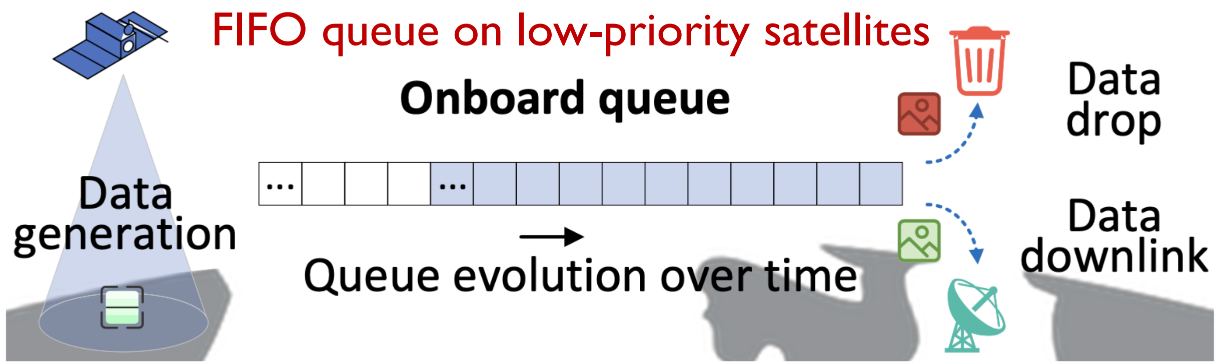


Assumptions of attack scenario:

- High-priority satellites can **preempt** low-priority ones during shared downlinks.
- Attacker (high-priority users) can **schedule** tasks for specific location and time.
- Attacker has **knowledge** of low-priority queue policy & (possibly noisy) dynamics.

Threat Model and Attack Goals

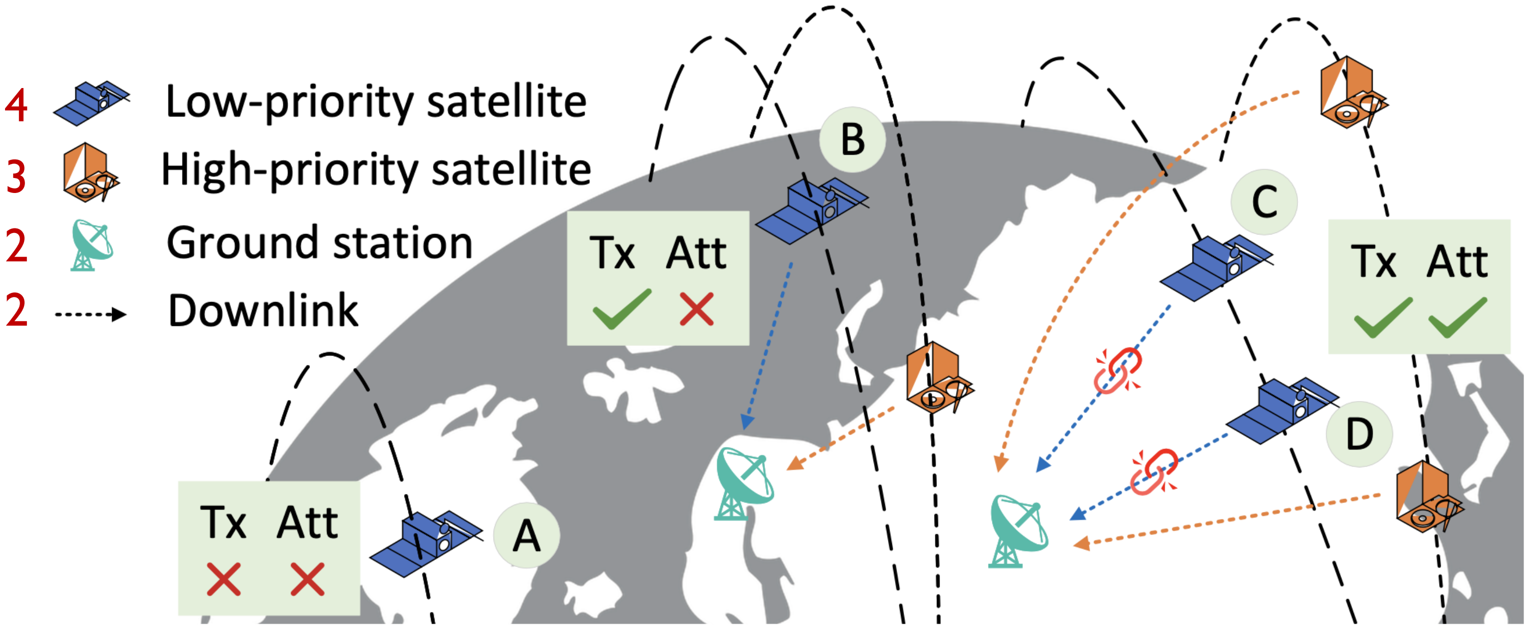
Limited onboard storage capacity



- Attack target: one or more data units s^* on low-priority satellite.
  - A series of images, video fragments, etc.
- Attack goal: **delay** or **drop** the target data before it downlinks to the ground.
- Attacker ability: utilize *legitimate task scheduling* on high-priority satellites with shared ground communications.
- Attacker strategy $\mathcal{Y}_{s^*}$
  - A set of **attackable** time slots for which the attacker schedules high-priority tasks
- Attacker cost $\rho_{s^*}(t)$
  - Depends on number of **attacked** slots, and high-priority service pricing
- Attacker objective
  - Successfully delay / drop target data
  - Minimize attack cost or time frame

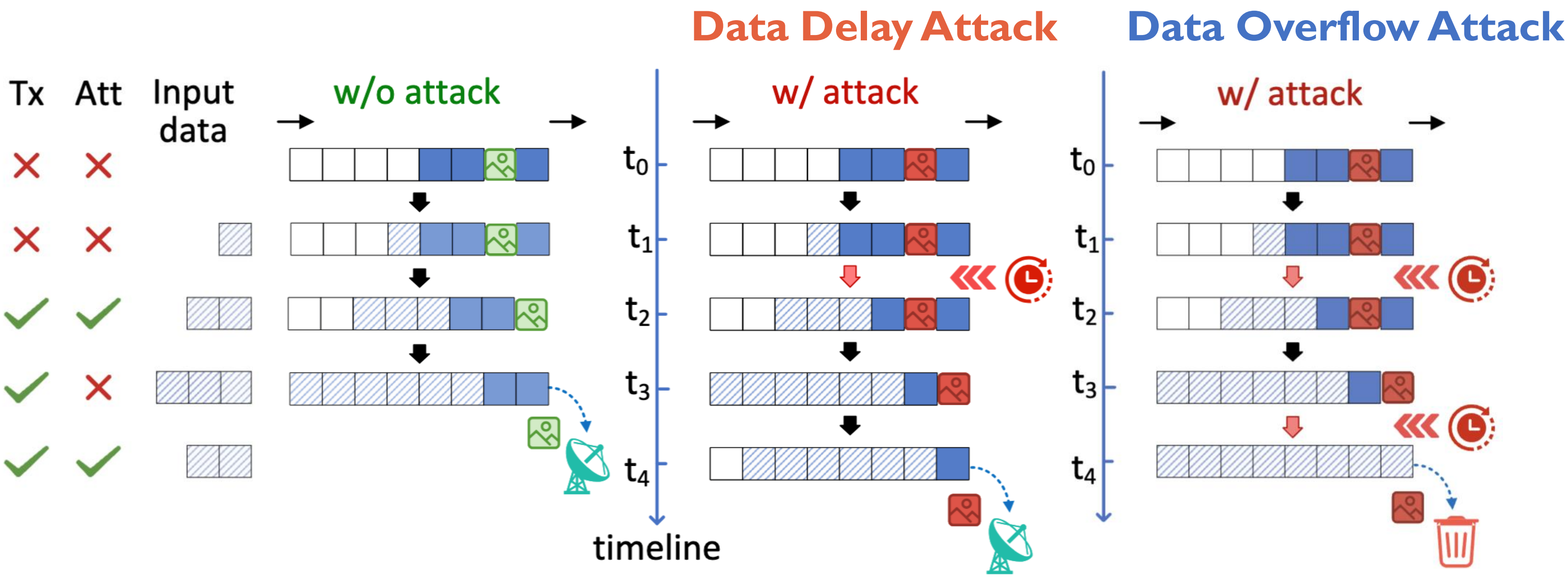
Transmissible and Attackable Windows

- Transmissible and attackable time slot



Tx - transmissible indicator **Att** - attackable indicator

Data Delay Attack & Data Overflow Attack



Remark: The attacks can start **long before the target data is generated** on the target satellite, under mild queue conditions.

Example: Planned or long-term scheduling of high-priority services by a nation-state attacker to elicit and keep satellites queues in desired states for rapid launching of targeted attacks.

Attack Algorithms

- Given knowledge of queue dynamics and orbital dynamics

Algorithm 1: Data Delay Attack

Input: Target data Θ, target downlink time $t^*(\tilde{\tau})$, EO constellations state $\{(\mathcal{X}_{s^*}, \mathcal{A}_{s^*}, \mathcal{Q}_{s^*}(t_0, \emptyset), \mathcal{Q}_{s^*}(\tau, t_0, \emptyset), c_{s^*}), \tau \in \Theta\}$, attack cost set $\boldsymbol{\rho}_{s^*}$

Output: Attack strategy $\mathcal{Y}_{s^*}$

$\mathcal{Y}_{s^*} \leftarrow \emptyset$;
for $\tau \in \Theta$ **do**
  $t_e(\tau, \mathcal{Y}_{s^*}) \leftarrow \min_t\{t | Q_{s^*}(\tau, t, \mathcal{Y}_{s^*}) = 0\}$;
  $t_{lb}(\tau, \mathcal{Y}_{s^*}) \leftarrow \max\{t_0, \max_t\{t | Q_{s^*}(t, \mathcal{Y}_{s^*}) = c_{s^*}$ and $t_0 \leq t < t_e(\tau)\}\}$;
  if *data unit* τ *is dropped* **then return** $\mathcal{Y}_{s^*}$;
  while $t_e(\tau, \mathcal{Y}_{s^*}) - t_e(\tau, \emptyset) \leq t^*(\tilde{\tau}) - t_e(\tilde{\tau}, \emptyset)$ **do**
    $\hat{T}_\tau \leftarrow \{t | t_{lb}(\tau, \mathcal{Y}_{s^*}) < t \leq t_e(\tau, \mathcal{Y}_{s^*})$ and $t \in \mathcal{A}_{s^*}$ and $t \notin \mathcal{Y}_{s^*}\}$;
    if $\hat{T}_\tau = \emptyset$ **then return** Attack Fail;
    $\hat{t} \leftarrow \arg\min_{t \in \hat{T}_\tau}\{\rho_{s^*}(t)\}$;
    $\mathcal{Y}_{s^*} \leftarrow \mathcal{Y}_{s^*} \cup \{\hat{t}\}$;
    $t_e(\tau, \mathcal{Y}_{s^*}) \leftarrow \min_t\{t | Q_{s^*}(\tau, t, \mathcal{Y}_{s^*}) = 0\}$;
    if $t_e(\tau, \mathcal{Y}_{s^*}) = \infty$ *(τ is dropped)* **then**
      return Attack strategy $\mathcal{Y}_{s^*}$;
    $t_{lb}(\tau, \mathcal{Y}_{s^*}) \leftarrow \max\{t_0, \max_t\{t | Q_{s^*}(t, \mathcal{Y}_{s^*}) = c_{s^*}$ and $t_0 \leq t < t_e(\tau, \mathcal{Y}_{s^*})\}\}$;
return Attack strategy $\mathcal{Y}_{s^*}$;

Algorithm 2: Data Overflow Attack

Input: Target data Θ, EO constellations state $\{(\mathcal{X}_{s^*}, \mathcal{A}_{s^*}, \mathcal{Q}_{s^*}(t_0, \emptyset), \mathcal{Q}_{s^*}(\tau, t_0, \emptyset), c_{s^*}), \tau \in \Theta\}$

Output: Attack strategy $\mathcal{Y}_{s^*}$

$\mathcal{Y}_{s^*} \leftarrow \emptyset$;
for $\tau \in \Theta$ **do**
  $t_e(\tau, \mathcal{Y}_{s^*}) \leftarrow \min_t\{t | Q_{s^*}(\tau, t, \mathcal{Y}_{s^*}) = 0\}$;
  $t_{lb}(\tau, \emptyset) \leftarrow \max\{t_0, \max_t\{Q_{s^*}(t_0) = c_{s^*}$ and $t_0 \leq t < t_e(\tau, \mathcal{Y}_{s^*})\}\}$;
  Sort set $\{t | t \geq t_e(\tau, \mathcal{Y}_{s^*})$ and $t \in \mathcal{A}_{s^*}\}$ in ascending order as T_n;
  Sort set $\{t | t_0 \leq t < t_e(\tau, \mathcal{Y}_{s^*})$ and $t \in \mathcal{A}_{s^*}\}$ in descending order as T_p;
  $t_n \leftarrow \mathsf{T}_n.pop()$, $t_p \leftarrow \mathsf{T}_p.pop()$;
  while *τ is not be dropped* **do**
    if $t_n \leq t_e(\tau, \mathcal{Y}_{s^*})$ **then**
      $\mathcal{Y}_{s^*} \leftarrow \mathcal{Y}_{s^*} \cup \{t_n\}$;
      $t_e(\tau, \mathcal{Y}_{s^*}) \leftarrow \min_t\{t | Q_{s^*}(\tau, t, \mathcal{Y}_{s^*}) = 0\}$;
      $t_n \leftarrow \mathsf{T}_n.pop()$;
    else if *$T_p \neq \emptyset$ and $t_p > t_{lb}(\tau, \mathcal{Y}_{s^*})$* **then**
      $\mathcal{Y}_{s^*} \leftarrow \mathcal{Y}_{s^*} \cup \{t_p\}$;
      $t_e(\tau, \mathcal{Y}_{s^*}) \leftarrow \min_t\{t | Q_{s^*}(\tau, t, \mathcal{Y}_{s^*}) = 0\}$;
      $t_{lb}(\tau, \mathcal{Y}_{s^*}) \leftarrow \max\{t_0, \max_t\{t | Q_{s^*}(t, \mathcal{Y}_{s^*}) = c_{s^*}$ and $t_0 \leq t < t_e(\tau)\}\}$;
      $t_p \leftarrow \mathsf{T}_p.pop()$;
    else return Attack Fail;
return Attack strategy $\mathcal{Y}_{s^*}$.

- Feasible data delay attack with minimum cost (Algorithm 1)
- Feasible data overflow attack with minimum attack period (Algorithm 2)

(Both proofs in extended arXiv version)

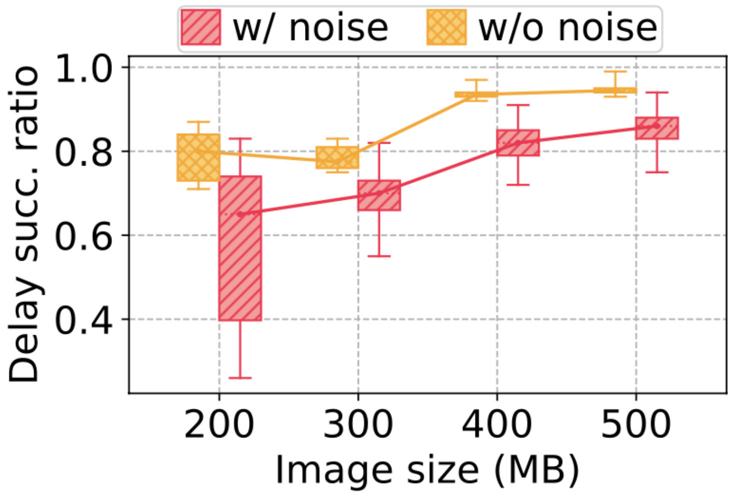
Simulation of the Two Attacks

- PlanetLab Dove (low-priority, 118 satellites) and SkySat (high-priority, 21-50 satellites)
  - 12 shared ground stations, each with 4 antennas
- **Simulation parameters:** image sizes & rate, onboard storage, downlink rate, (synthetic) costs
- **Noisy knowledge:** randomly perturb image size, data rate and/or initial queue size

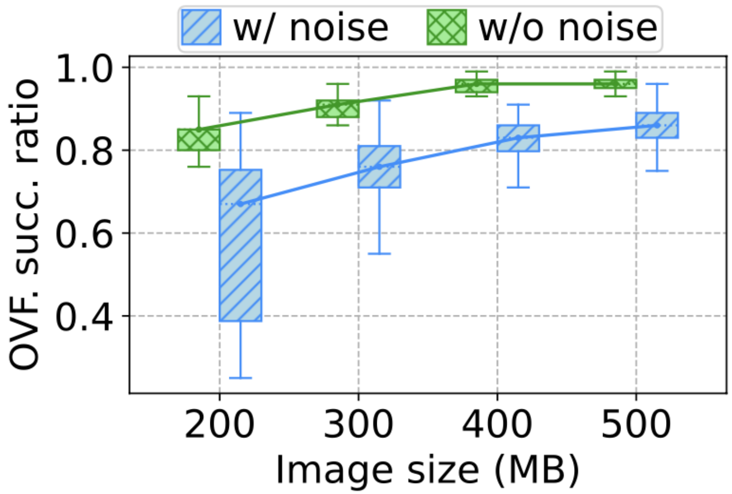
---

Selected results

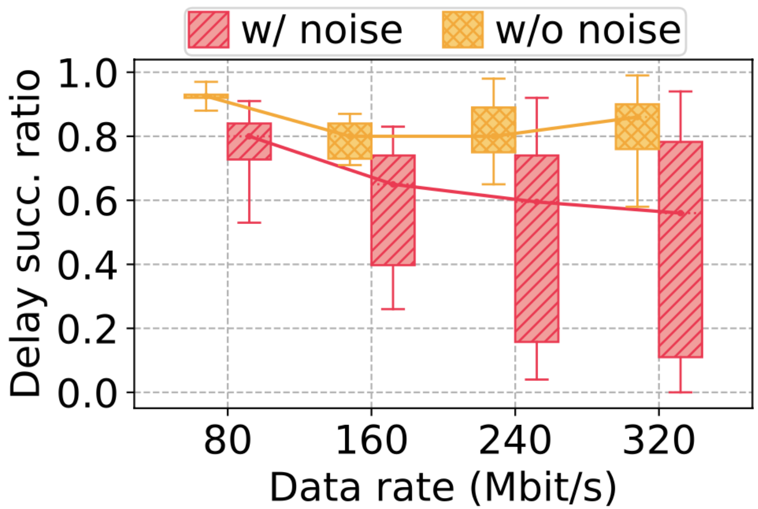
Delay Attack

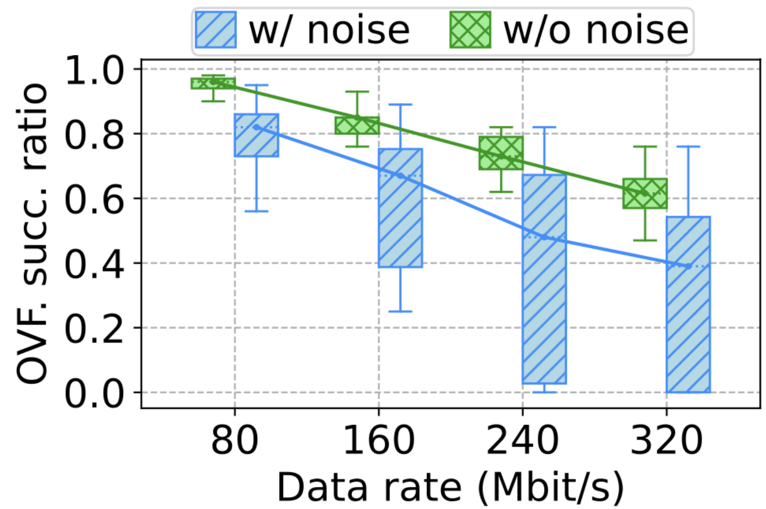


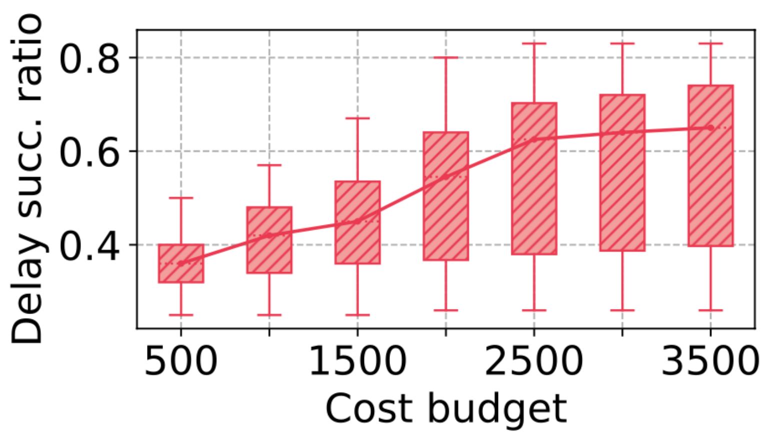
Overflow Attack

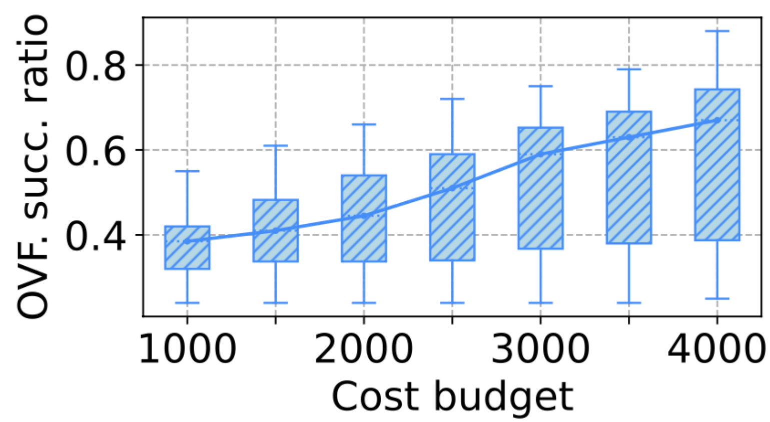


More Simulation Results









Countermeasures and Other Thoughts

- *Q: Can enlarging on-board storage or downlink stop the attacks?*
  - *Storage:* possibly for overflow, not for delay.
  - *Downlink capacity:* yes, but costly → race between application & capacity.

- *Q: Can different queue scheduling policies help?*
  - *Deterministic (like LIFO):* not really; attacker can adjust.
  - *Random on low-priority:* makes it harder for attacker to profile the policy.
  - *Random between low- and high-priority:* can help make attack less successful, but degrades high-priority QoS.

- *Q: Can user access control help?*
  - ✓ Detect and suspend abnormal user activities on high-priority services.
  - ✓ Utilization-based resource / service pricing → increase attacker cost.
  - *More complex security games between attacker and EO operator* ☺

Thank you very much!

Q&A?

Data Delay Attack

Algorithm 1: Data Delay Attack

Input: Target data Θ, target downlink time $t^*(\tilde{\tau})$, EO constellations state $\{(\mathcal{X}_{s^*}, \mathcal{A}_{s^*}, \mathcal{Q}_{s^*}(t_0, \emptyset), \mathcal{Q}_{s^*}(\tau, t_0, \emptyset), c_{s^*}), \tau \in \Theta\}$, attack cost set $\boldsymbol{\rho}_{s^*}$

Output: Attack strategy $\mathcal{Y}_{s^*}$

$\mathcal{Y}_{s^*} \leftarrow \emptyset$;

initialize the attack strategy

for $\tau \in \Theta$ **do**

$t_e(\tau, \mathcal{Y}_{s^*}) \leftarrow \min_t\{t | Q_{s^*}(\tau, t, \mathcal{Y}_{s^*}) = 0\}$;

expected downlink time

$t_{lb}(\tau, \mathcal{Y}_{s^*}) \leftarrow \max\{t_0, \max_t\{t | Q_{s^*}(t, \mathcal{Y}_{s^*}) = c_{s^*} \text{ and } t_0 \le t < t_e(\tau)\}\}$;

last queue full time without any attack

if *data unit τ is dropped* **then return** $\mathcal{Y}_{s^*}$;

while $t_e(\tau, \mathcal{Y}_{s^*}) - t_e(\tau, \emptyset) \le t^*(\tilde{\tau}) - t_e(\tilde{\tau}, \emptyset)$ **do**

τ needs more attack time slots to delay longer

$\hat{T}_\tau \leftarrow \{t | t_{lb}(\tau, \mathcal{Y}_{s^*}) < t \le t_e(\tau, \mathcal{Y}_{s^*}) \text{ and } t \in \mathcal{A}_{s^*} \text{ and } t \notin \mathcal{Y}_{s^*}\}$;

all the attackable time slots that potentially have attack strength for τ

if $\hat{T}_\tau = \emptyset$ **then return** Attack Fail;

$\hat{t} \leftarrow \arg\min_{t \in \hat{T}_\tau}\{\rho_{s^*}(t)\}$;

find the minimum cost attack strategy for τ

$\mathcal{Y}_{s^*} \leftarrow \mathcal{Y}_{s^*} \cup \{\hat{t}\}$;

add t' to the attack strategy

$t_e(\tau, \mathcal{Y}_{s^*}) \leftarrow \min_t\{t | Q_{s^*}(\tau, t, \mathcal{Y}_{s^*}) = 0\}$;

update the expected downlink time

if $t_e(\tau, \mathcal{Y}_{s^*}) = \infty$ *(τ is dropped)* **then**

return Attack strategy $\mathcal{Y}_{s^*}$;

$t_{lb}(\tau, \mathcal{Y}_{s^*}) \leftarrow \max\{t_0, \max_t\{t | Q_{s^*}(t, \mathcal{Y}_{s^*}) = c_{s^*} \text{ and } t_0 \le t < t_e(\tau, \mathcal{Y}_{s^*})\}\}$;

update last queue full time

return Attack strategy $\mathcal{Y}_{s^*}$;

Data Overflow Attack

Algorithm 2: Data Overflow Attack

Input: Target data Θ, EO constellations state $\{(\mathcal{X}_{s^*}, \mathcal{A}_{s^*}, \mathcal{Q}_{s^*}(t_0, \emptyset), \mathcal{Q}_{s^*}(\tau, t_0, \emptyset), c_{s^*}), \tau \in \Theta\}$

Output: Attack strategy $\mathcal{Y}_{s^*}$

$\mathcal{Y}_{s^*} \leftarrow \emptyset$;
for $\tau \in \Theta$ **do**
  $t_e(\tau, \mathcal{Y}_{s^*}) \leftarrow \min_t\{t | Q_{s^*}(\tau, t, \mathcal{Y}_{s^*}) = 0\}$;
  $t_{lb}(\tau, \emptyset) \leftarrow \max\{t_0, \max_t\{Q_{s^*}(t_0) = c_{s^*}$ and $t_0 \leq t < t_e(\tau, \mathcal{Y}_{s^*})\}\}$;
  Sort set $\{t | t \geq t_e(\tau, \mathcal{Y}_{s^*})$ and $t \in \mathcal{A}_{s^*}\}$ in ascending order as T_n;
  Sort set $\{t | t_0 \leq t < t_e(\tau, \mathcal{Y}_{s^*})$ and $t \in \mathcal{A}_{s^*}\}$ in descending order as T_p;
  $t_n \leftarrow \mathsf{T}_n.pop()$, $t_p \leftarrow \mathsf{T}_p.pop()$;
  while *τ is not be dropped* **do**
    if $t_n \leq t_e(\tau, \mathcal{Y}_{s^*})$ **then**
      $\mathcal{Y}_{s^*} \leftarrow \mathcal{Y}_{s^*} \cup \{t_n\}$;
      $t_e(\tau, \mathcal{Y}_{s^*}) \leftarrow \min_t\{t | Q_{s^*}(\tau, t, \mathcal{Y}_{s^*}) = 0\}$;
      $t_n \leftarrow \mathsf{T}_n.pop()$;
    else if *$T_p \neq \emptyset$ and $t_p > t_{lb}(\tau, \mathcal{Y}_{s^*})$* **then**
      $\mathcal{Y}_{s^*} \leftarrow \mathcal{Y}_{s^*} \cup \{t_p\}$;
      $t_e(\tau, \mathcal{Y}_{s^*}) \leftarrow \min_t\{t | Q_{s^*}(\tau, t, \mathcal{Y}_{s^*}) = 0\}$;
      $t_{lb}(\tau, \mathcal{Y}_{s^*}) \leftarrow \max\{t_0, \max_t\{t | Q_{s^*}(t, \mathcal{Y}_{s^*}) = c_{s^*}$ and $t_0 \leq t < t_e(\tau)\}\}$;
      $t_p \leftarrow \mathsf{T}_p.pop()$;
    else return Attack Fail;
return Attack strategy $\mathcal{Y}_{s^*}$.

Initialize the attack strategy

expected downlink time

last queue full time without any attack

T_n - attackable time slots at and after the initial image downlink time (ascending)

T_p - attackable time slots before the initial image downlink (descending)

attack the time slot t_n in T_n and keep the target data onboard and at the top of the queue

next attackable time slot in T_n cannot contribute to delaying the downlink time of the target data

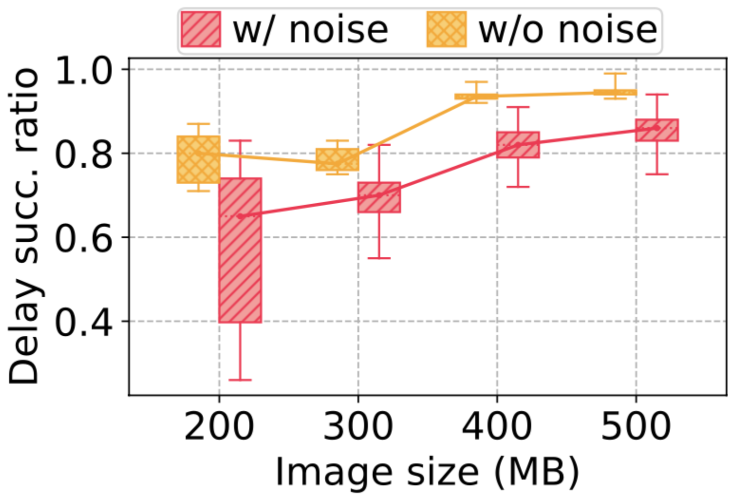
attack the time slot t_p in T_p to delay the downlink time of τ to make sure τ remains onboard

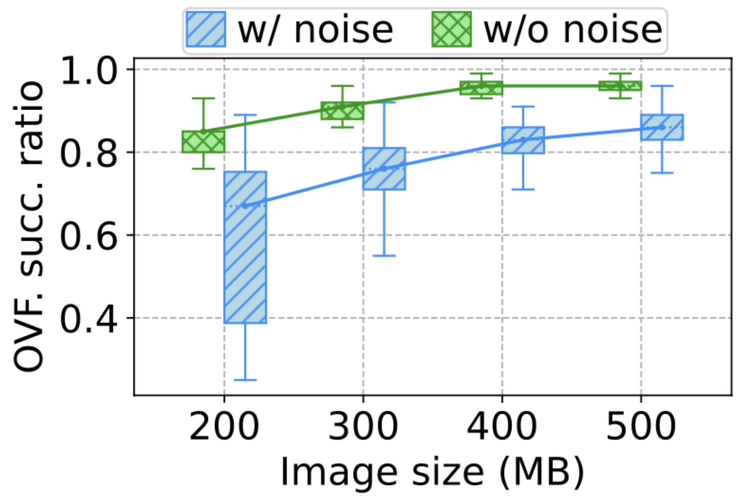
If T_p is empty or new data overflow happens due to the attack on t_p, then the attack fails

Evaluation Settings

- Trace-driven simulation
  - Target satellite and data
    - Real-world metadata of the Planet Labs Dove satellite image data
    - **Target satellite 10 Dove satellites**/118 satellites
    - Target data sampled 1000 images (100 images from each satellite)
    - Image size: 200MB [200MB-500MB], 4 images as target data
    - **Onboard storage: 2000GB**; Initial queue size: 500 images
  - Downlink resource
    - **12 ground stations each with 4 antennas**
    - Average downlink rate: 160 Mbit/s [80Mbit/s - 320Mbit/s]
  - Attacker's resource
    - Orbital information: real-world orbit Two Line Elements (TLEs) information
    - **50 high-priority SkySat satellites [21-50]**
    - Cost budget: 500-4000
  - Gaussian **noise**
    - Image size and data rate [0-0.4 standard deviation] + vary10 image in initial queue
  - 10 seeds

Vary Image Size

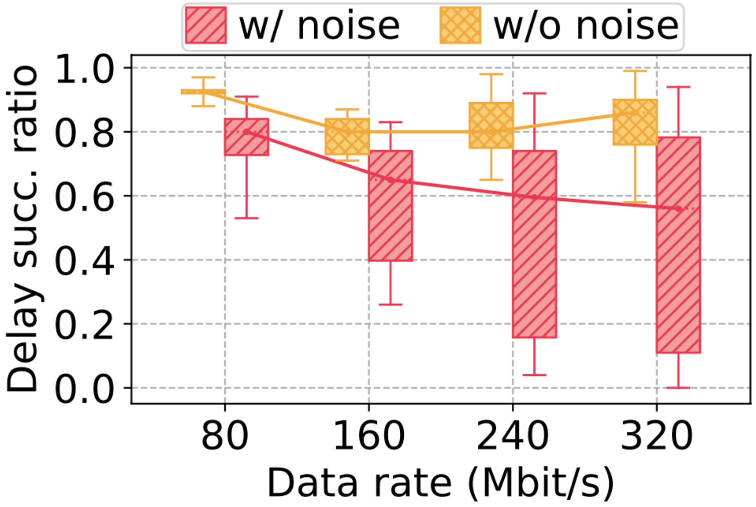


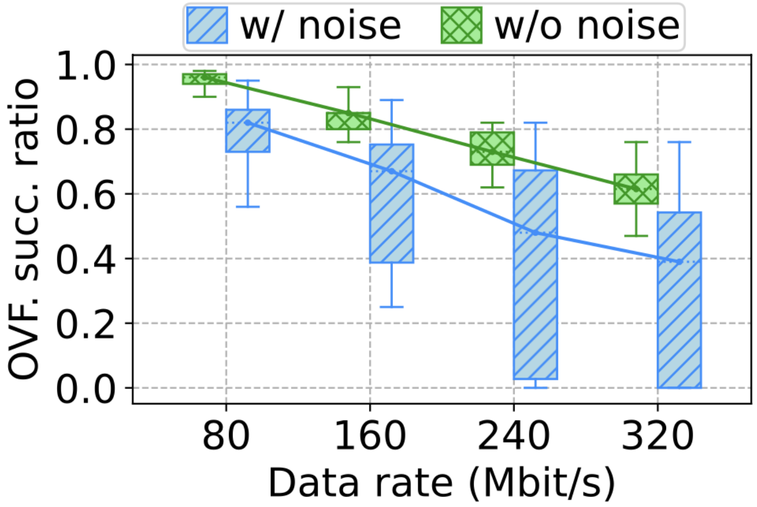


- Attacks with noise had a lower success ratio than those without.

image size ↑
success ratio ↑

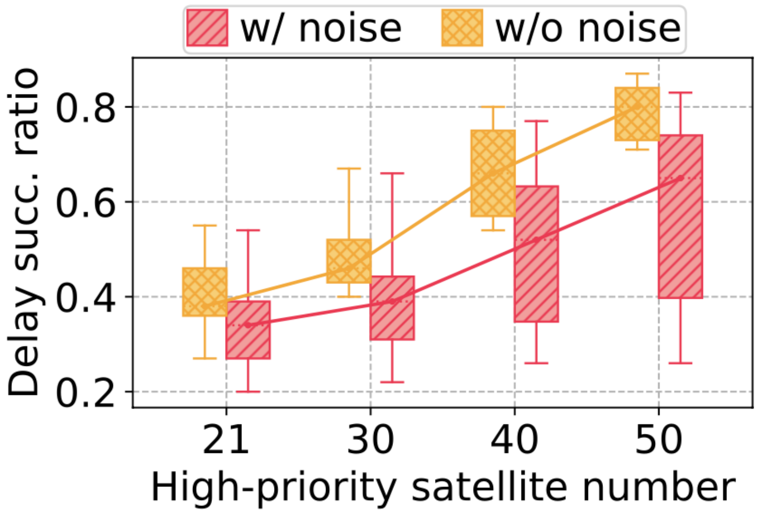
Vary Data Rate

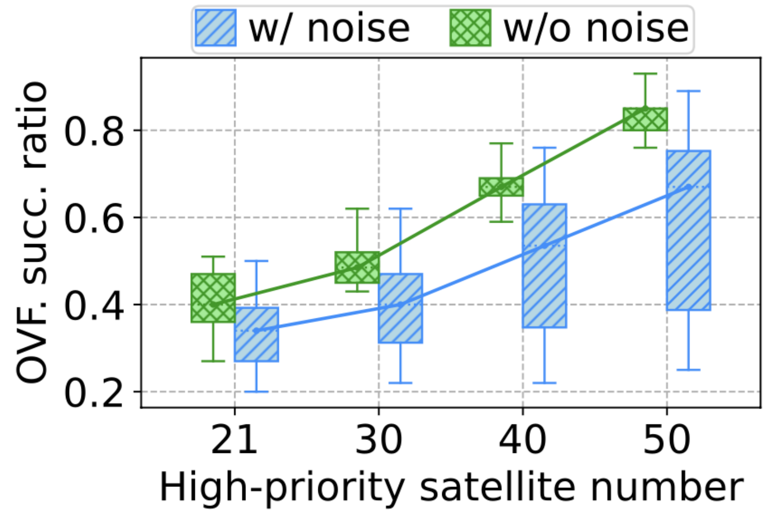




data rate
success ratio

Vary High-priority Satellite Number





high-priority satellite number

success ratio