Establishing PDCA Cycles for Agile Network Management in SDN/NFV Infrastructure

Kotaro Shibata† Hiroki Nakayama‡
Tsunemasa Hayashi‡ Shingo Ata†

†Osaka City University ‡BOSCO Technologies Inc.

Problem statement

1. Recently, numerous studies on software defined networking (SDN) have been performed.
2. SDN enables logical centralized management of the network system.
3. However, SDN does not support management plane involving operators who maintain policies.
4. So, we need a general framework including management plane on SDN.

Motivation of our policy framework

- Manage both network and functional resources adaptively, quickly and timely for policy adaptation
- Control traffic according to policy
- Realize on-demand policy adaptation
- Mediate multiple different policies which occur conflict

To manage policy

- We maintain a policy with traffic dynamics.
  - We must reconfigure system quickly to adapt a policy.
- We change a policy according to the real scenario.
  - Policy is empirically defined from past experiences.

Considering the above, we conduct PDCA cycle for the policy management.

- PDCA (Plan-Do-Check-Act) cycle is well known as an iterative management method for the control and continuous improvement of processes.

To realize PDCA cycle

1. We provide planes according to PDCA cycle.
   - "Plan", "Do", "Check", "Act"
2. However, "Act" is an human intervention or an alert from "Check".
3. So, we don't need a dedicated plane for "Act".

Monitoring Plane	• collect managed resource info • raise an alert
Decision Plane	• make a configuration set to adapt a policy
Control Plane	• send messages to resource managers

To make policy management easily

1. Some policies are too complex to run a PDCA cycle.
2. We consider the policy as composition of policies which run a PDCA cycle more easily.
3. In other words, policy is stackable like slices.
4. So, we call *policy slice* as policy in our framework.

To avoid conflict

1. Between *policy slices*, resource or *policy slice* itself may conflict.
2. To avoid the conflict, we establish global PDCA cycle to manage *policy slices*.
3. Global PDCA cycle reconstructs *policy slice(s)* to run PDCA cycle easily and rapidly.

To manage PDCA cycle

1. By the story to here, we can run a manual PDCA cycle.
2. So, we deploy PDCA controller to drive both cycles of the global PDCA and *policy slices*.
3. PDCA controller has four states associated to PDCA cycle.

Plan	• Prepare a solution *plan* against the alert • Aggregate and filter alerts
Do	• Send a set of control messages according to the *plan*
Check	• Verify the plan is running correctly
Act	• Reconstruct *policy slices* by global PDCA cycle • Install *policy slices* by administrator

Experiment Overview

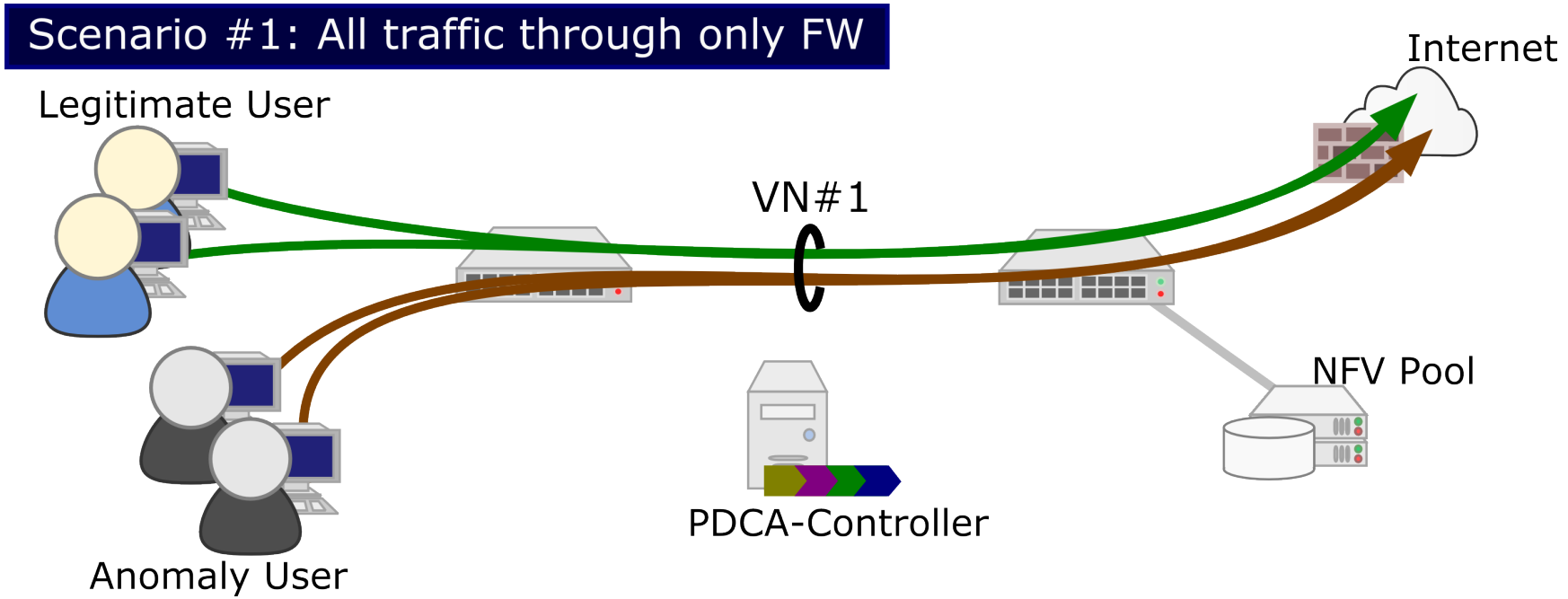
- Purpose
  - To get the proof of concept, we check how our framework works.
- Environment
  - 20 users repeat the upload of the file to the Internet.
  - The size of uploaded file is varied from 10 MB to 100 MB.

Users	Upload text file including string of
10 legitimate users	“This is a good content”
10 anomaly users	“This is a bad content“

Scenario #1

1. The experiment starts (at *0* second).
2. All traffic flows are sent out to the Internet via a firewall directly.

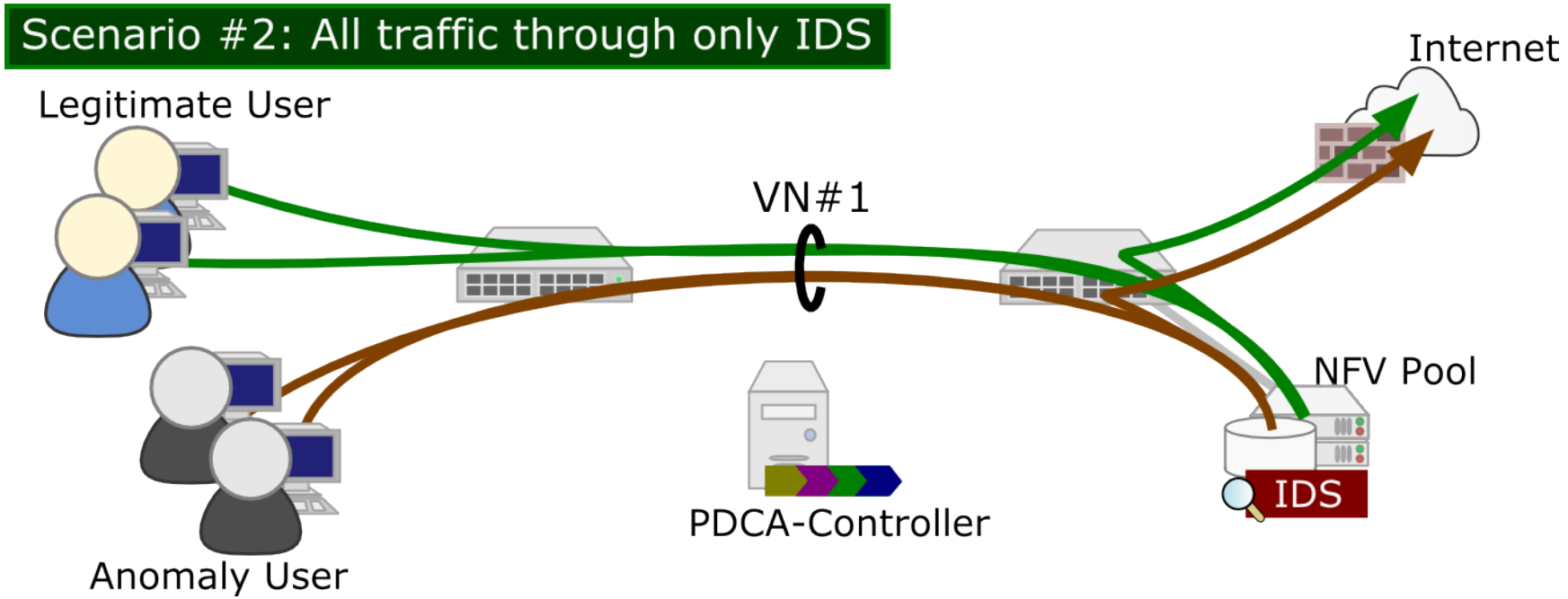
The anomaly traffic spoils reliability of the network.



Scenario #2

1. The administrator installs a new *policy slice* to forward traffic flows to the IDS (at *15* second).

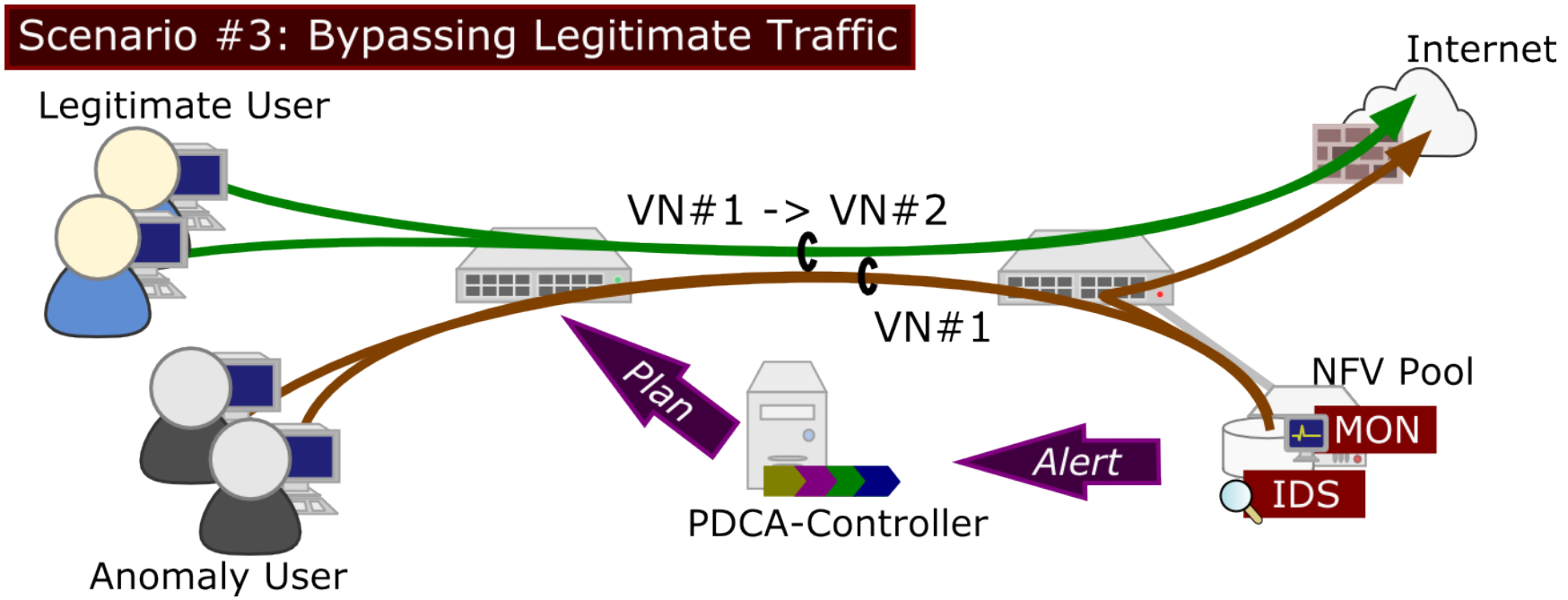
However, the throughput reduces dramatically by a computational overhead of IDS.



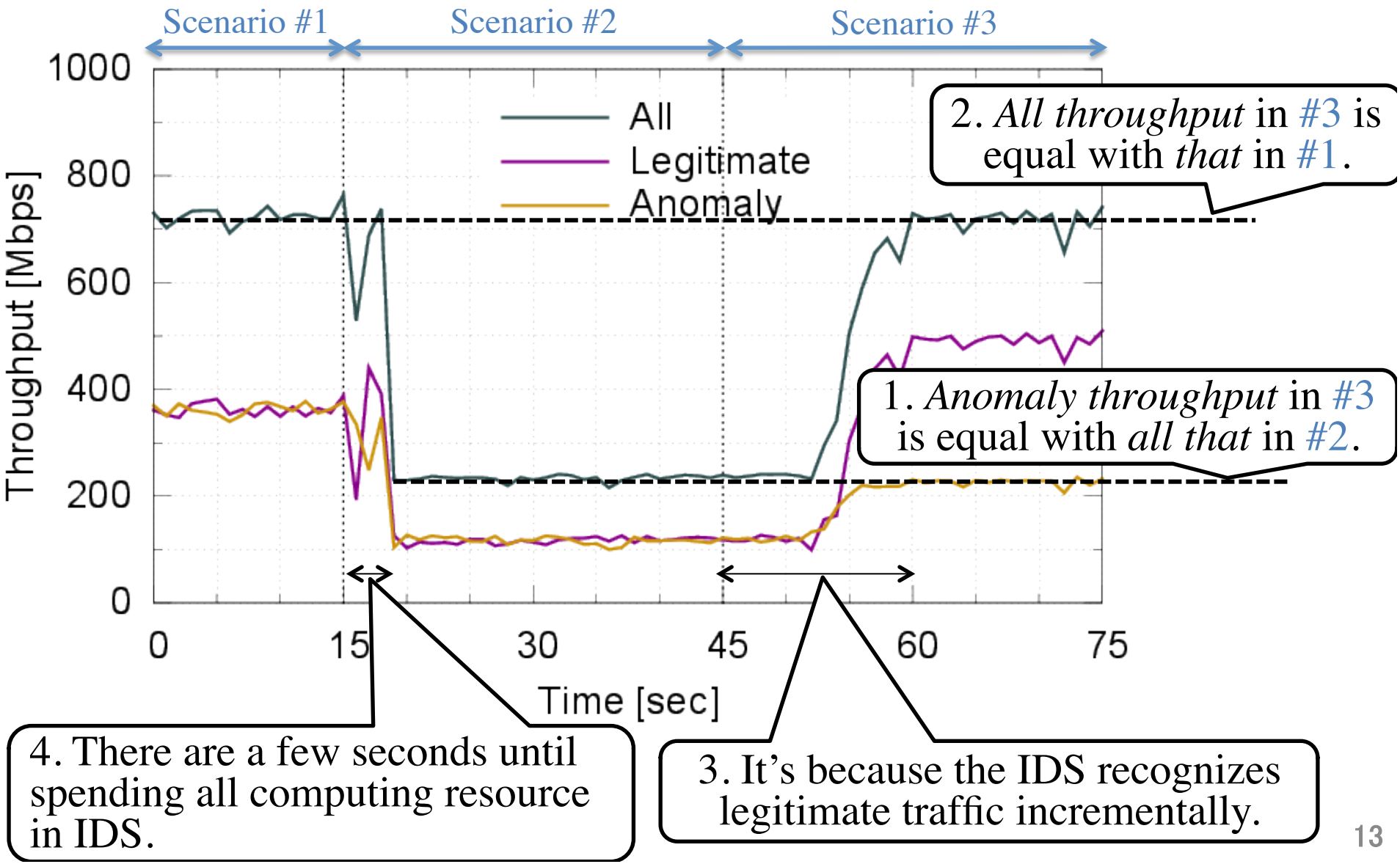
Scenario #3

1. The administrator installs a new *policy slice* that the legitimate traffic flows connect to the firewall directly (at *45* second).

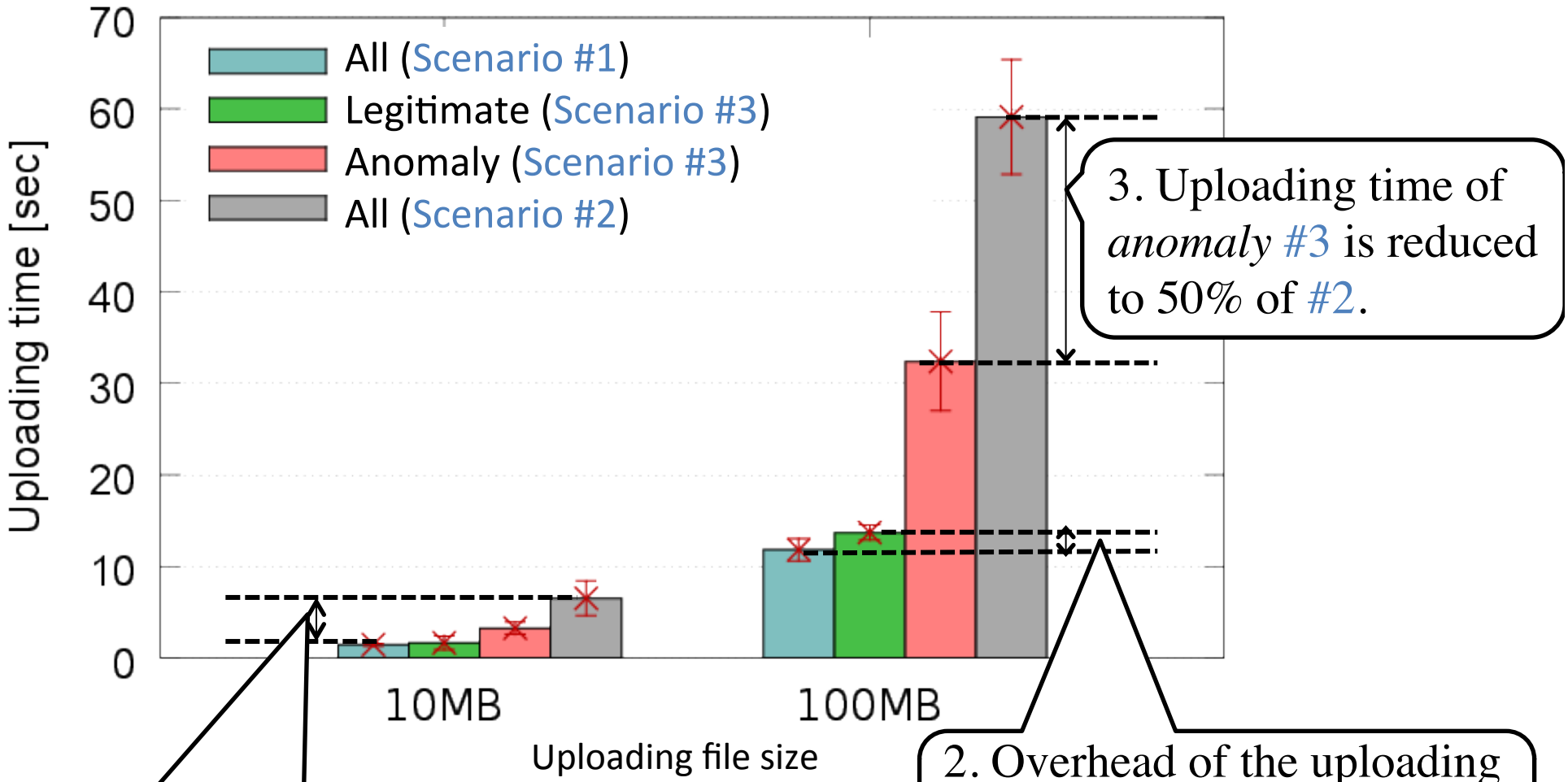
We expect to improve the performance of both total output rate and the transmission delay of the legitimate traffic.



Traffic Variation



Uploading Time



Summary & Future works

- Summary
  - We have proposed a new framework to establish a PDCA cycle for both network and computing resource management adaptively, quickly and timely.
- Future works
  - We need additional experiments in the large network environment for the validation of the scalability.

Thank you for listening

Why PDCA? Why not MAPE-K?

1. In policy management, some problems must resolved by human intervention.
   - Supply shortage-hardware resources
   - Take a legal reaction against an illegal behavior
2. However, there is no human intervention in MAPE-K.
   - MAPE-K came from autonomic computing.
   - PDCA came from quality management.