Red Hat Research Days
presents

Cloud Auto-scaling Mechanism Under DDoS Attacks: Yo-Yo Attack and Tandem Attack

Speaker

Anat Bremler-Barr

Tel Aviv University

Speaker

Michael Czeizler

Reichman University

Wednesday | Mar 15th

5:00PM - 6:30PM IST

Conversation Leader

Jeremy Eder

Red Hat

Cloud Auto-Scaling Mechanisms Under DDoS Attacks: Yo-Yo Attack and Tandem Attack

Anat Bremler-Barr

The Blavatnik School of Computer Science
The Raymond and Beverly Sackler Faculty of Exact Sciences
Tel Aviv University

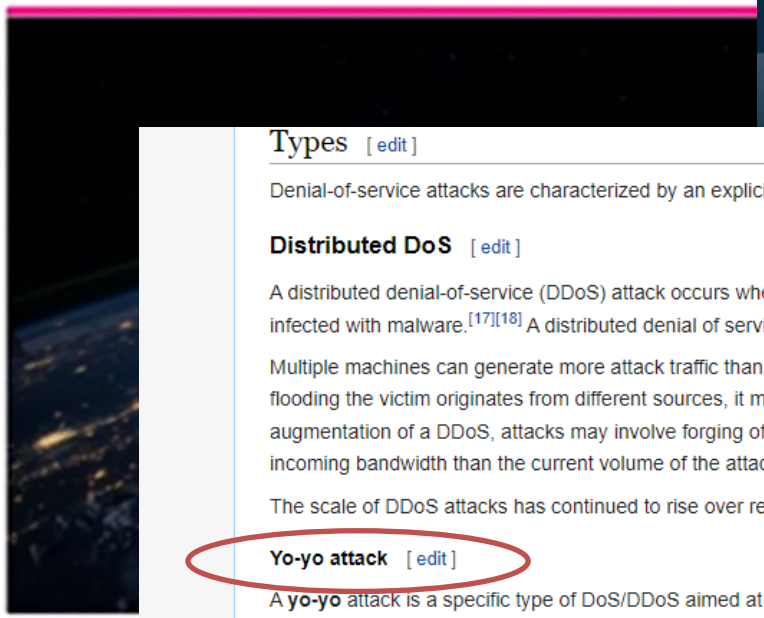
Michael Czeizler

Reichman University
Efi Arazi School of Computer Science

‘Yo-Yo’ DDoS Cyber Attacks; What they Are and How You Can Beat Them

‘Yo-Yo’ DDoS cyberattacks might sound like a bad video game or science fiction movie, but in reality, they’re a type of innovative distributed denial-of-service (DDoS) attack.

Guest Contributor / 1 May 2022 · 4 Min Read



Typically, DDoS (Distributed Denial of Service) attacks use massive traffic such
HTTP, DNS, TCP, and other methods to allow attackers to disrupt even the mo
well-defended networks or servers. But Yo-Yo DDoS is an entirely different ani

radware

APPLICATION PROTECTION DDOS PROTECTION PUBLIC CLOUD P

SECURITY / DDOSPEDIA

Yo-yo attack

Reblaze Solutions Verticals Pricing Why Reblaze Resources Partners About

Main WAF Bot Detection DDoS Protection API Security Cloud Security DevSe

Types [edit]

Denial-of-service attacks are characterized by an explicit attempt by attackers to prevent legitimate use of a service. There are two general forms of DoS attacks: those that crash services and those that flood services. The most serious attacks are distribut

Distributed DoS [edit]

A distributed denial-of-service (DDoS) attack occurs when multiple systems flood the bandwidth or resources of a targeted system, usually one or more web servers.[16] A DDoS attack uses more than one unique IP address or machines, often from thousan infected with malware.[17][18] A distributed denial of service attack typically involves more than around 3–5 nodes on different networks; fewer nodes may qualify as a DoS attack but is not a DDoS attack.[19][20]

Multiple machines can generate more attack traffic than one machine, multiple attack machines are harder to turn off than one attack machine, and the behavior of each attack machine can be stealthier, making it harder to track and shut down. Since the in flooding the victim originates from different sources, it may be impossible to stop the attack simply by using ingress filtering. It also makes it difficult to distinguish legitimate user traffic from attack traffic when spread across multiple points of origin. As an alt augmentation of a DDoS, attacks may involve forging of IP sender addresses (IP address spoofing) further complicating identifying and defeating the attack. These attacker advantages cause challenges for defense mechanisms. For example, merely purch incoming bandwidth than the current volume of the attack might not help, because the attacker might be able to simply add more attack machines.

The scale of DDoS attacks has continued to rise over recent years, by 2016 exceeding a terabit per second.[21][22] Some common examples of DDoS attacks are UDP flooding, SYN flooding and DNS amplification.[23][24]

Yo-yo attack [edit]

A **yo-yo** attack is a specific type of DoS/DDoS aimed at cloud-hosted applications which use autoscaling.[25][26][27] The attacker generates a flood of traffic until a cloud-hosted service scales outwards to handle the increase of traffic, then halts the attack, le with over-provisioned resources. When the victim scales back down, the attack resumes, causing resources to scale back up again. This can result in a reduced quality of service during the periods of scaling up and down and a financial drain on resources of over-provisioning while operating with a lower cost for an attacker compared to a normal DDoS attack, as it only needs to be generating traffic for a portion of the attack period.

Yaniv Yagolnitzer | June 18, 2020

Cloud technologies make it easier to mitigate most forms of DDoS attacks. But threat actors are adapting, and there

Papers

- **Anat Bremler-Barr, Mor Sides, Elisha Rosensweig , Yo-Yo Attack - Vulnerability in auto-scaling mechanism (brief announcement), 2016**
- **Anat Bremler-Barr, Mor Sides, Eli Brosh, DDoS Attack on Cloud Auto-scaling Mechanisms, INFOCOM, 2017**
- **Anat Bremler-Barr, Ronen Ben David, Kubernetes Autoscaling: YoYo Attack Vulnerability and Mitigation CLOSER 2021**
- **Anat Bremler-Barr, Michael Czeizler, Tandem attack: DDoS attack on microservice auto-scaling mechanisms (brief announcement), INFOCOM 2023**

For more information on our research
http://www.deepness-lab.org

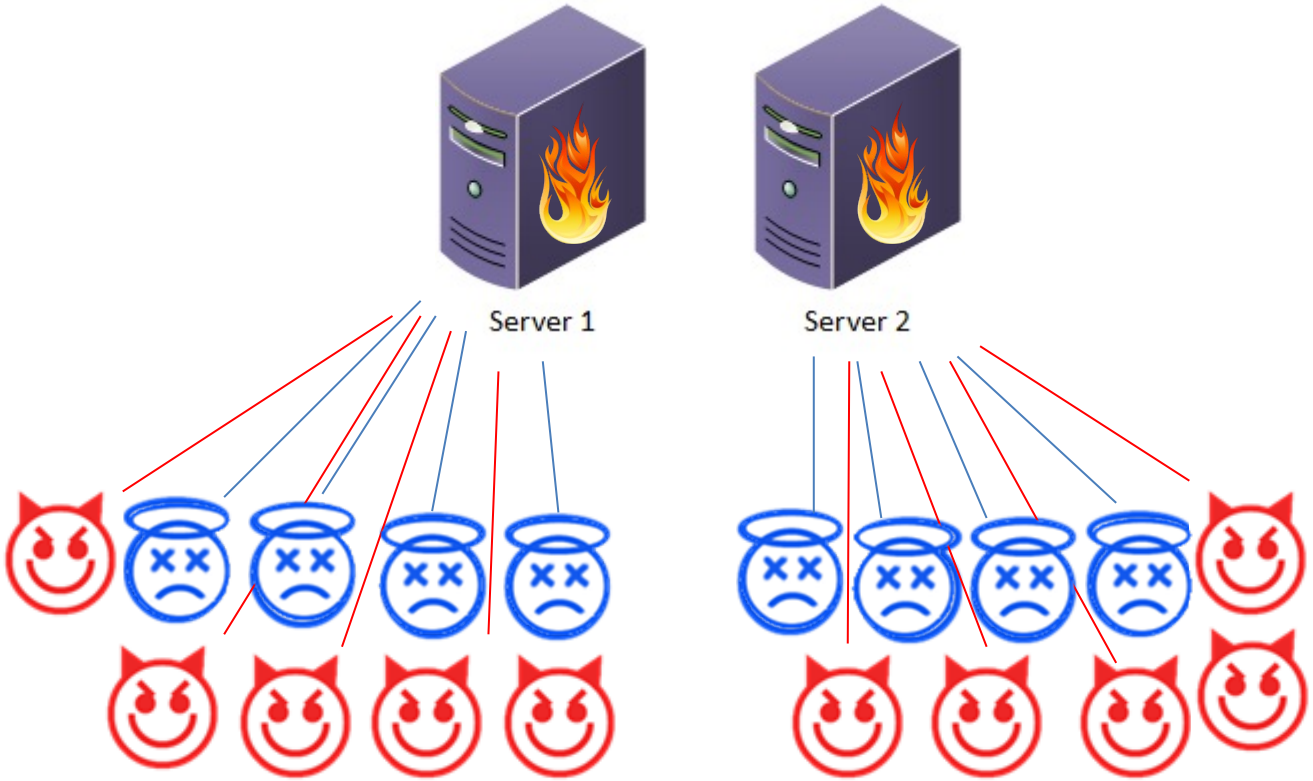
Agenda

- Pre-Cloud: DDoS Attacks
- Cloud: DDoS Attacks
  - Yo-Yo Attack
    - VM (IaS)
    - Kubernetes
    - Detection & Mitigation
- Tandem Effect: Microservice Auto-Scaling Mechanisms
  - Tandem Attack
- Leech Attack
- Conclusions & Future work

Pre-Cloud
Distributed Denial of Service (DDoS) Attacks

Pre-Cloud: Application-Level DDoS

- DDoS at the **application level** creates an overload of requests → performance degradation



Attacker Motivation

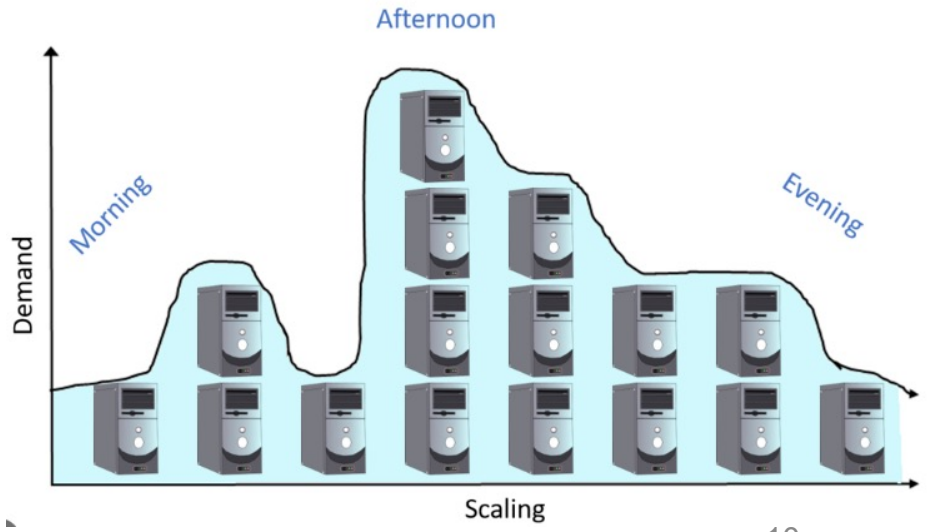
1. Personal enjoyment, Intellectual challenge
2. Financial gain – extortion
3. Business warfare – run the competitors out of business
4. Political hacktivism
5. Cyberwarfare

Riverhead networks

Cloud: DDoS Attacks

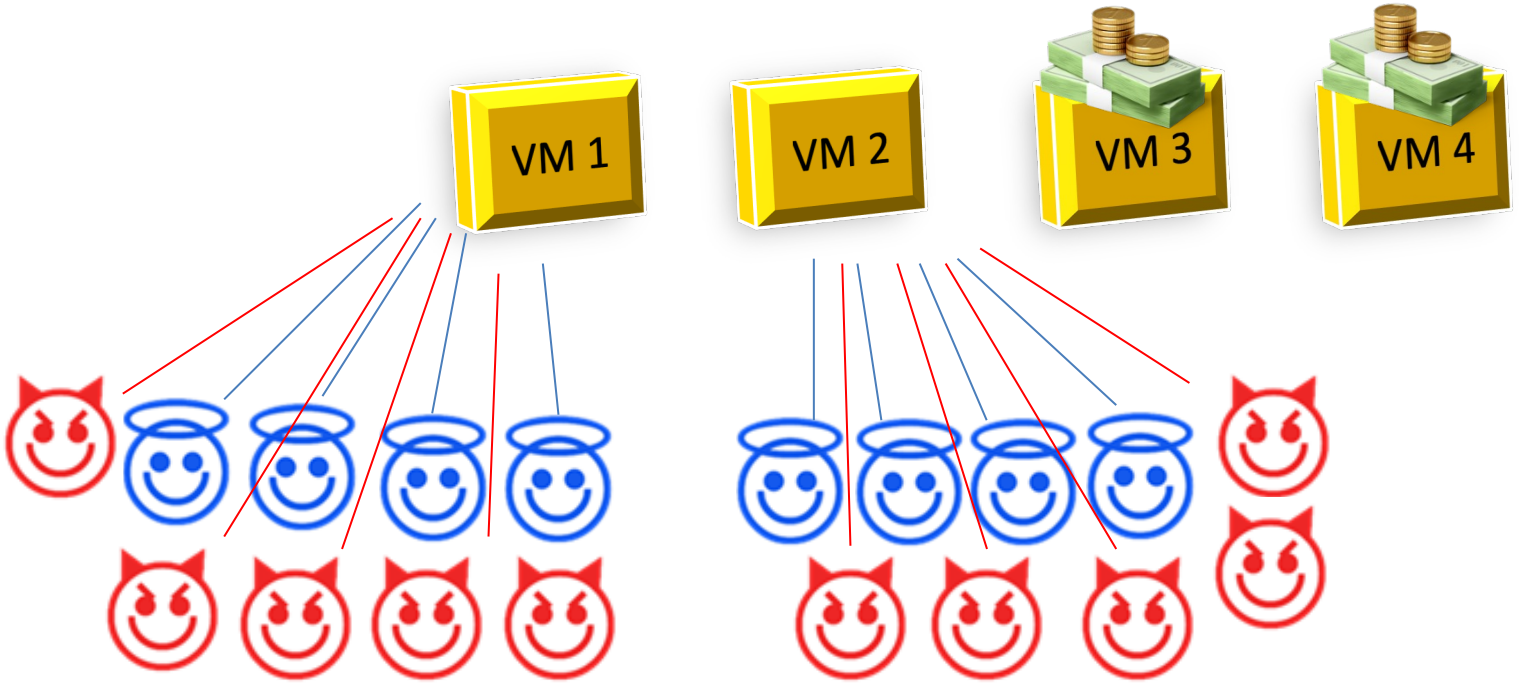
Auto-Scaling Mechanism

- Cloud: Outsourcing many of the maintenance and infrastructure pains
- Autoscaling: The ability to add machines to cope with the overload
- Admin defines auto-scaling rules by metric's threshold and duration to track (**scale-interval)**
  - If VM's CPU utilization is above 80% for **2 minutes** then perform a scale-up → add 3 machines

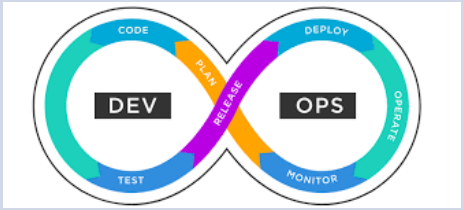


Cloud: Application-Level DDoS

- Auto-scaling: AWS best practices for DDoS resiliency
- No performance damage → Economic damage
  - Economic Denial of Sustainability attack (EDoS) == Denial of Wallet (DoW)



Performance Damage vs Economic Damage

	Performance Damage	Economic Damage
Department		
Publicity of the attack	Bad PR	Under the radar
Damage	• Denial of service (failures), service degradation (latency) • Translate to revenue loss • Reputation damage	• Direct expenses 

- Need finance observability!
  - Cloud provides information at the level of cloud service
  - Customer is responsible for translating it to the application level

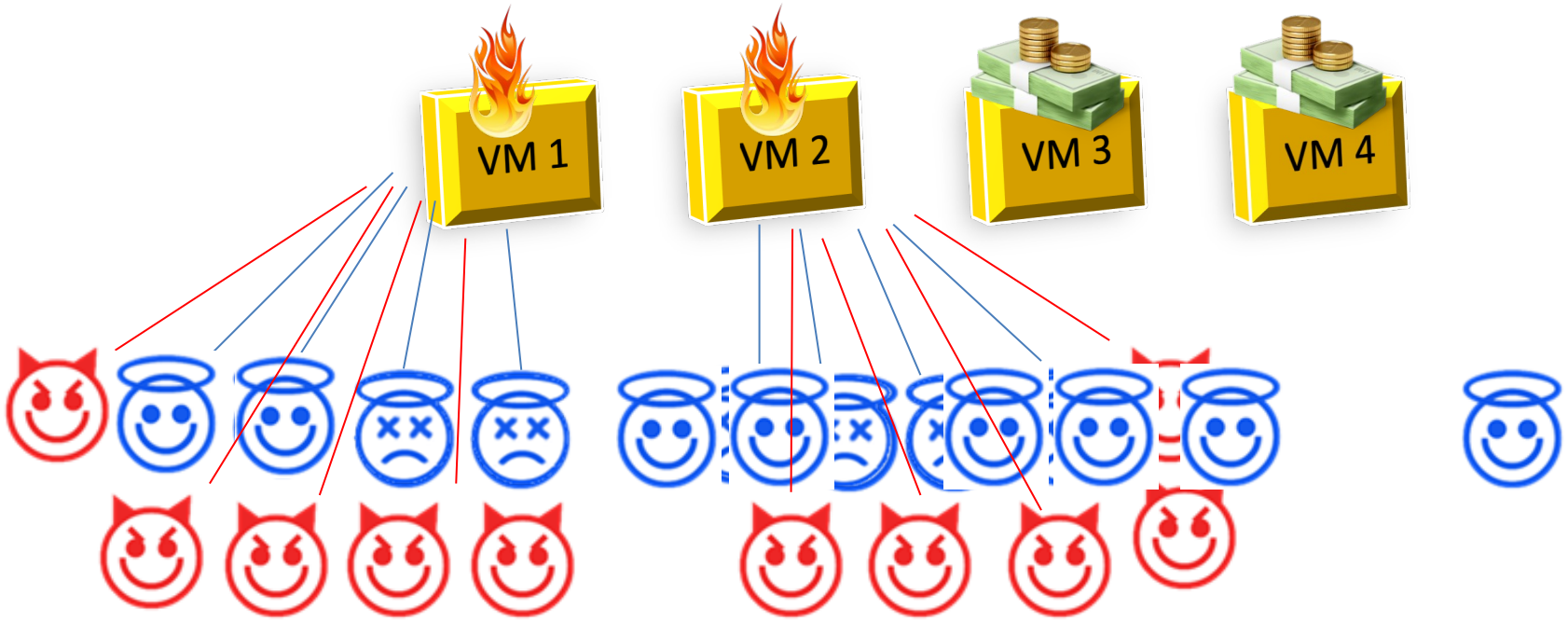
Cloud and DDoS: The Big Picture

- Cloud helps mitigate **network level DDoS** – due to the large pipes and anti-spoofing mechanisms built into the cloud infrastructure
- Here, we focus on the **application-level DDoS**
- Overload the application: API, search pages, login pages, and so on
  - Responsibility of the cloud customer and not the cloud provider
  - No remedy from the large pipes of the cloud or CDN

Yo-Yo Attack

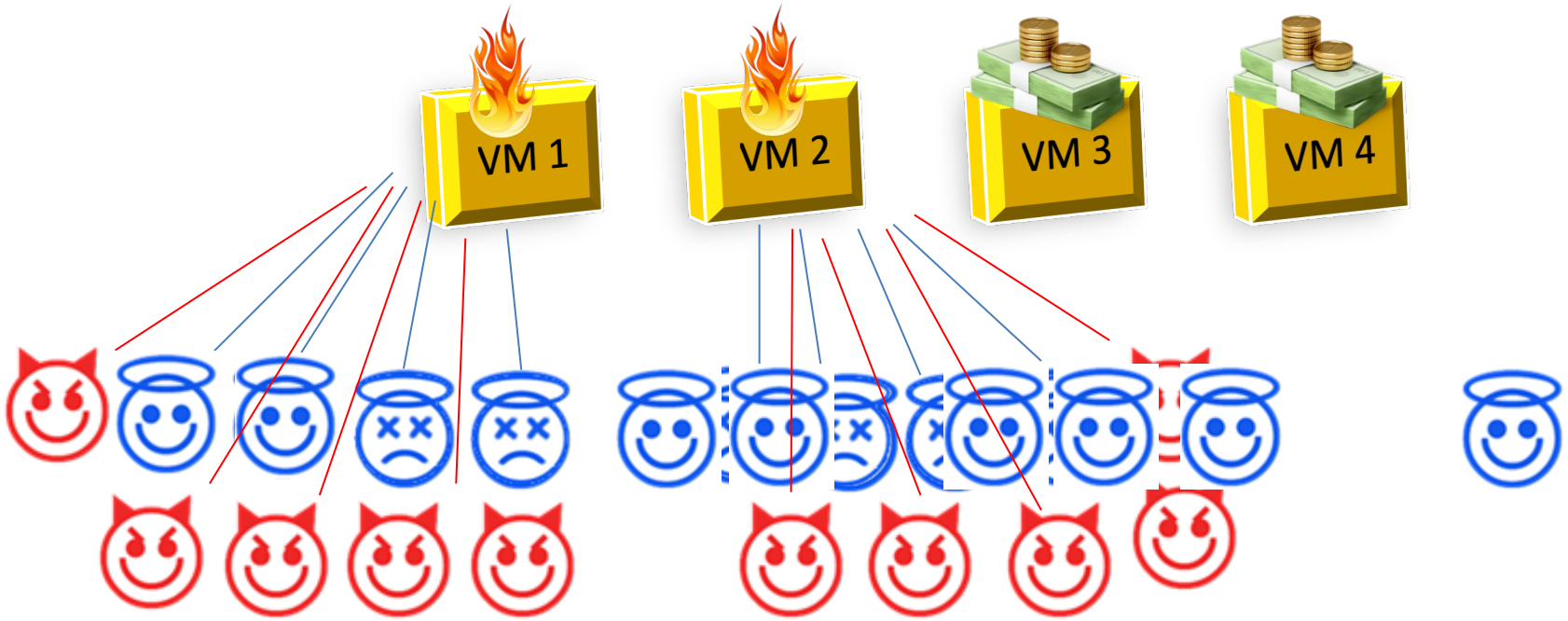
Yo-Yo Attack: Economic + Performance Damage

- Attacker can carry out an attack on the auto-scaling mechanism
  - **Yo-Yo attack:** Specially crafted waves of DDoS
    - Nowadays it is very common to be attacked **by waves of DDoS**



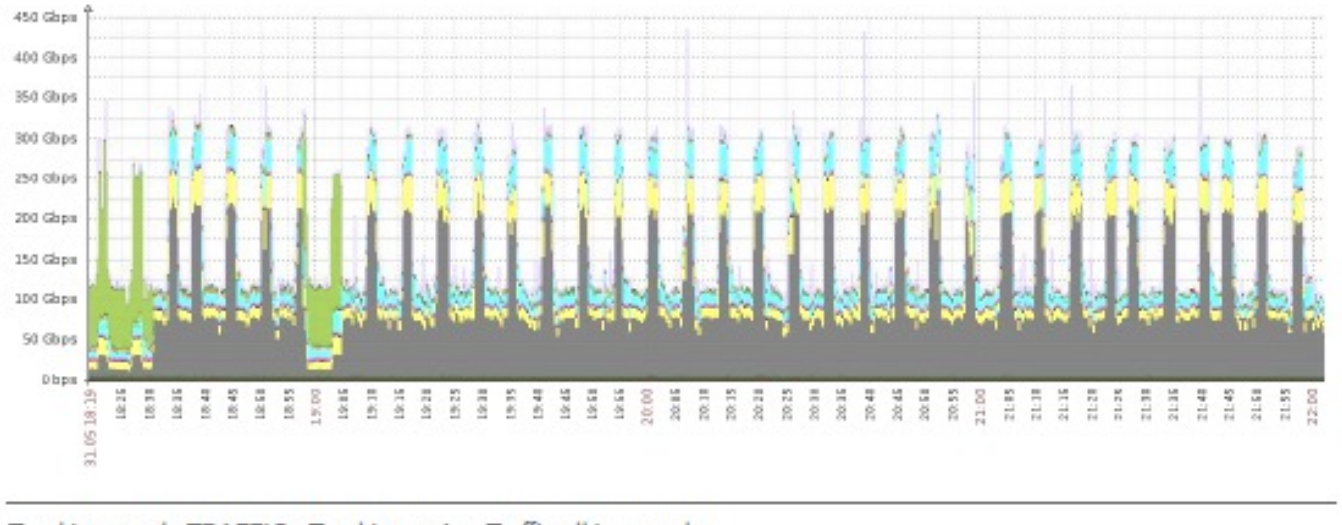
Yo-Yo Attack: Economic + Performance Damage

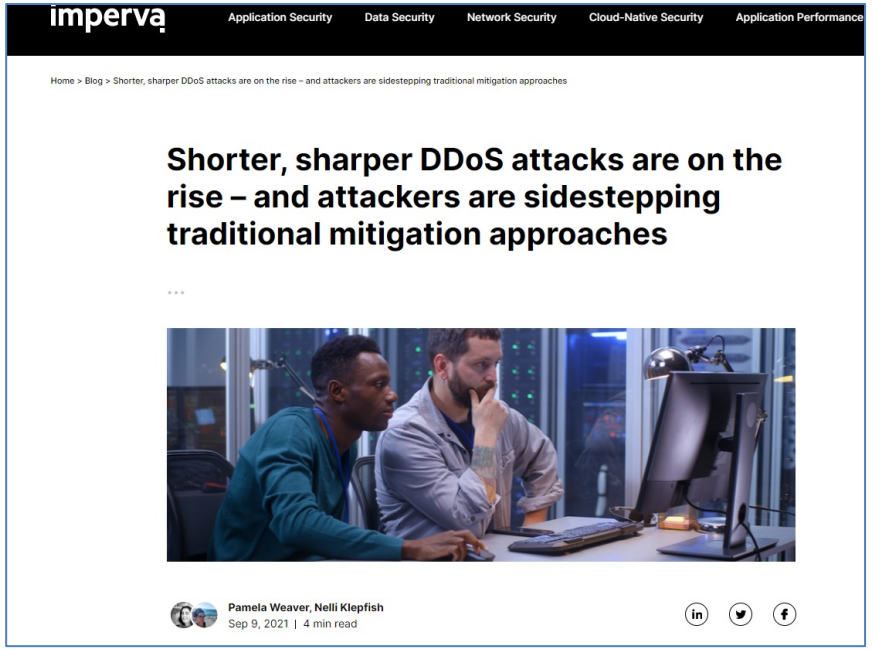
- Attacker can carry out an attack on the auto-scaling mechanism
  - **Yo-Yo attack:** Specially crafted waves of DDoS
    - Nowadays it is very common to be attacked **by waves of DDoS**
  - Economic damage & performance damage
  - Harder to detect & requires fewer resources from the attacker



DDoS Burst Attack

- Today it is common to be attacked by waves of DDoS
  - **Called:** Wave of attack, pulse attack, burst attack
  - Over 50% [Imperva, 2021]
- Attackers use DDoS bursts to take down multiple targets
- Aim to confuse the DDoS scrubbing mechanisms
  - Yo-Yo attack: confuse the auto-scaling mechanisms
- Harder to detect, cost-efficient (from attacker's perspective)



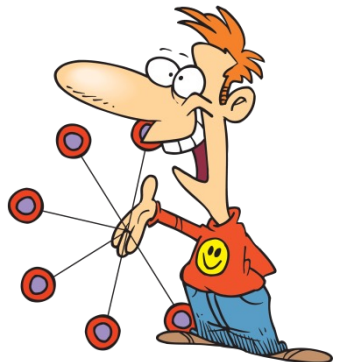


Yo-Yo Attack Details: Scale Interval + Warming Time

- **Scale Interval :** configured by the admin
  - If threshold of a metric exceeds duration of the **scale-interval**, the system will scale
- **Warming Time :** given by the system infrastructure.
  - **Warming time** of a scale-up – the time until the machine is ready to function, few minutes
    - The VM runs with the relevant software and state
  - **Warming time** of a scale-down – the time until the machine is closed and all its resources are released, few minutes
    - Backup, Moving states

Yo-Yo Attack: On/Off-Attack Phases

- The attacker repeatedly oscillates between the two phases:
  - On-attack phase: Sends a burst of traffic → scale-up
    - Several minutes
  - Off-attack phase: Stops sending the excess traffic → scale down
    - Start off-attack phase when the attacker detects the **scale-up** has occurred and ended
  - Repeat when the attacker detects the **scale-down** has occurred and ended.

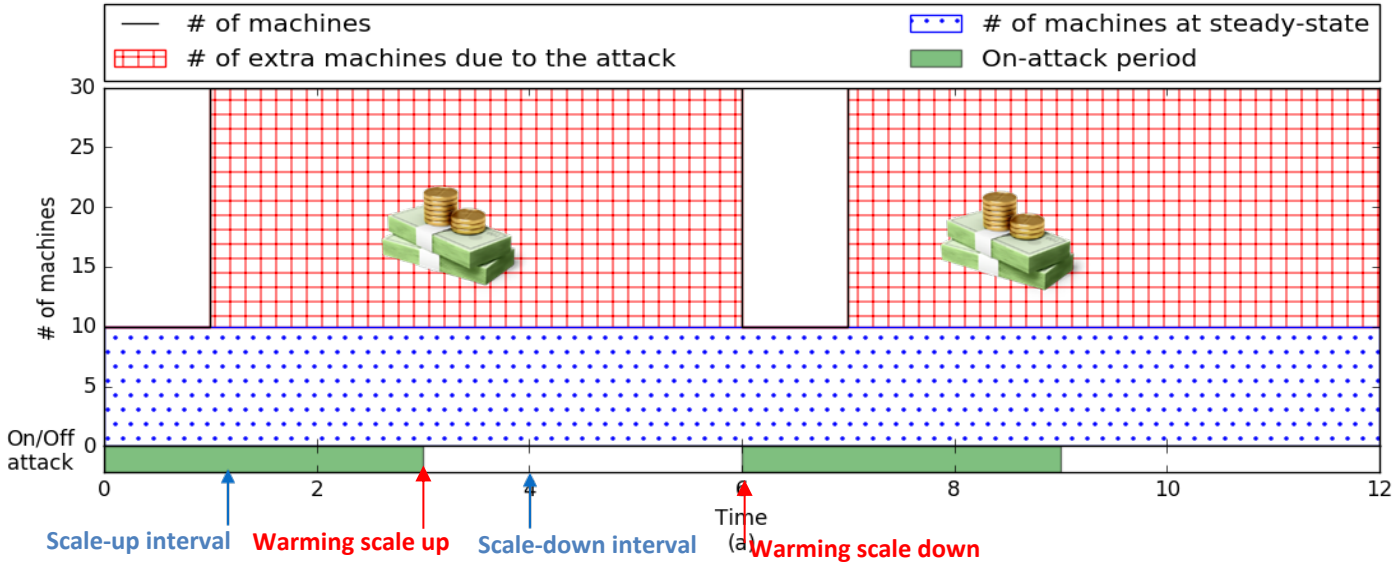


Yo-Yo Attack: Detecting System State

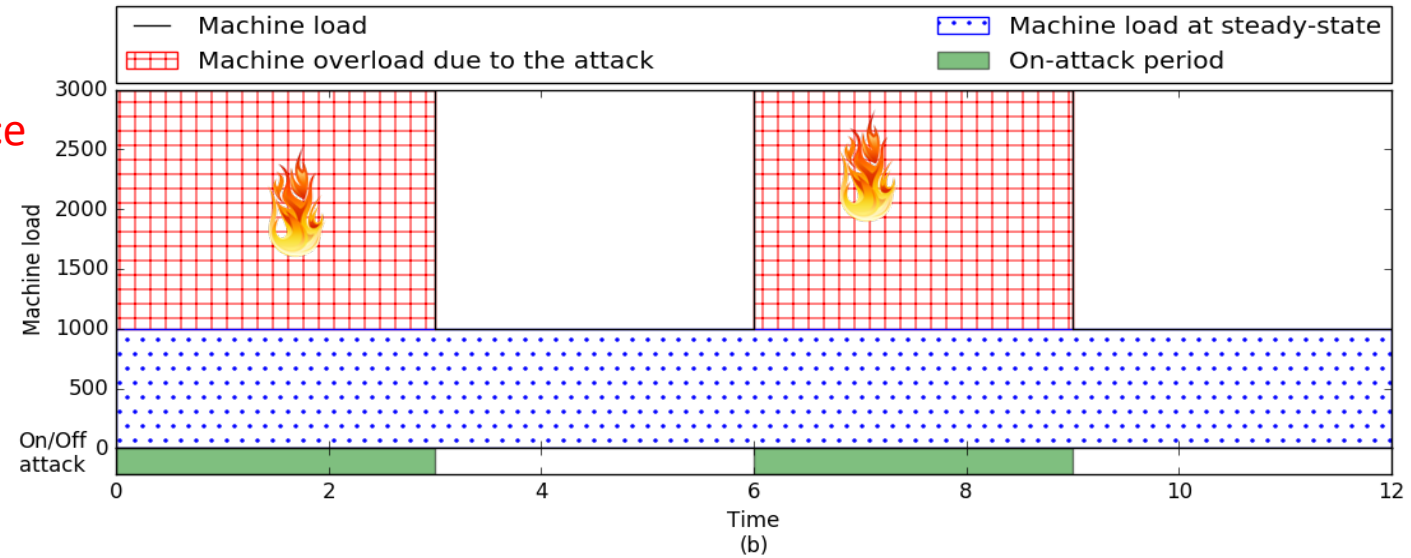
- Attacker: When to oscillate between on-attack and off-attack?
  - Send probe requests and check the response time
  - Rule of thumb:
    - T - threshold for the peace-time RTT
    - > T sec → scale up process has not ended
    - < T sec → scale down process has not ended

Yo-Yo Attack on Adaptive Scaling

Economic damage



Performance damage



Parameter	Value
Requests	10,000 rpm
Machines	10
Scale up/ Scale down Interval	1 minutes
Warming up/ Warming down	2 minutes
Peak extra load	x2

Use case example

Use Case: Analysis

System	Cost of attack	Performance damage 	Economic damage 
Pre-Cloud: DDoS	100% active	x2 extra load	0
Cloud: DDoS with auto-scaling	100% active	0	x2 extra cost of cloud
Yo-Yo attack	50% active	Avg. x1 extra load	Avg. x1.66 cost of cloud

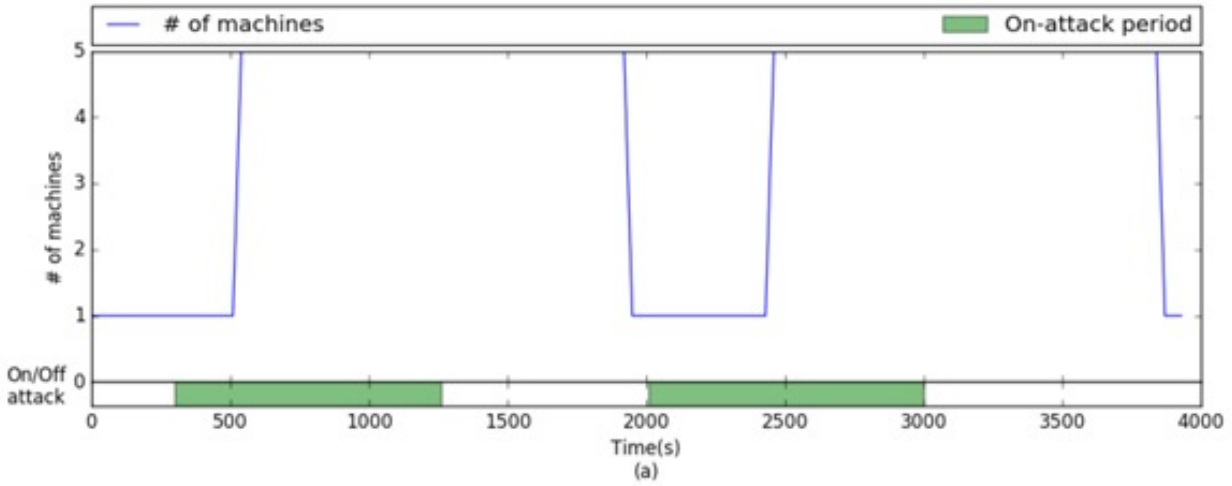
With extra peak load of x2

Outcomes:

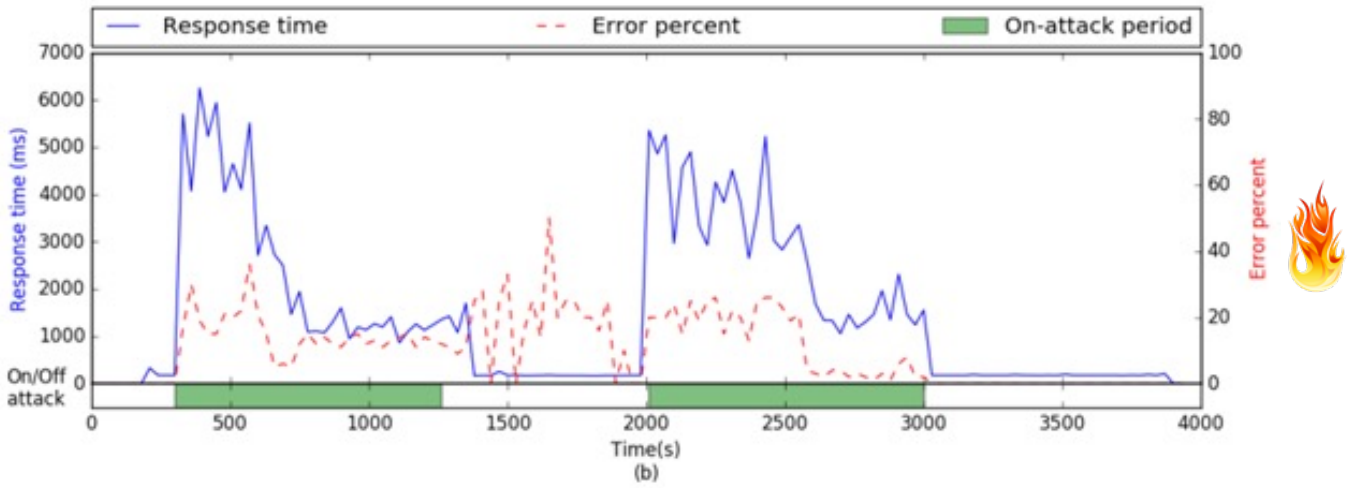
- Performance damage and economic damage
- Lower cost for the attacker, harder to detect

Experimental Results on AWS: Adaptive Auto-Scaling

Economic Damage



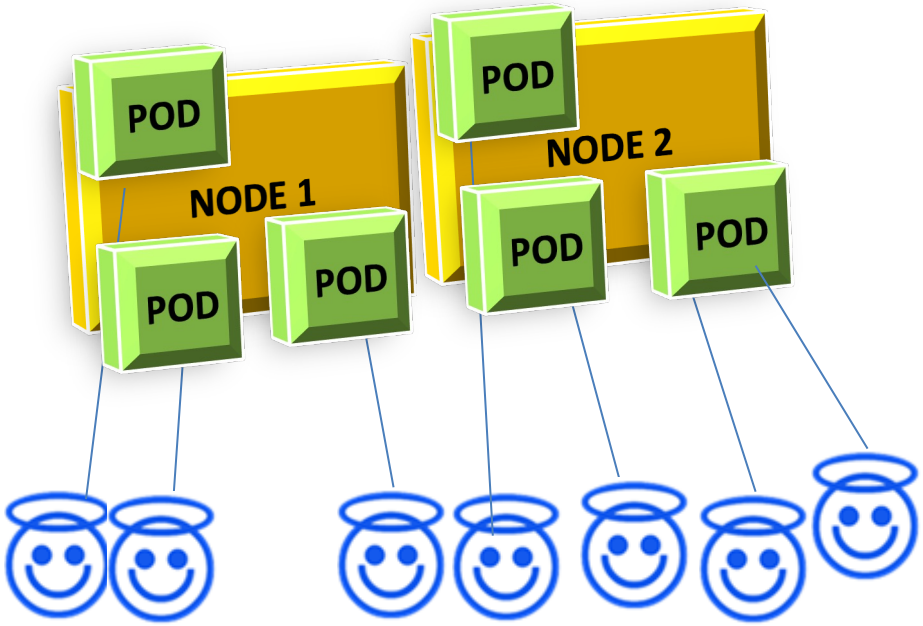
Performance Damage



Yo-Yo Attack on Kubernetes

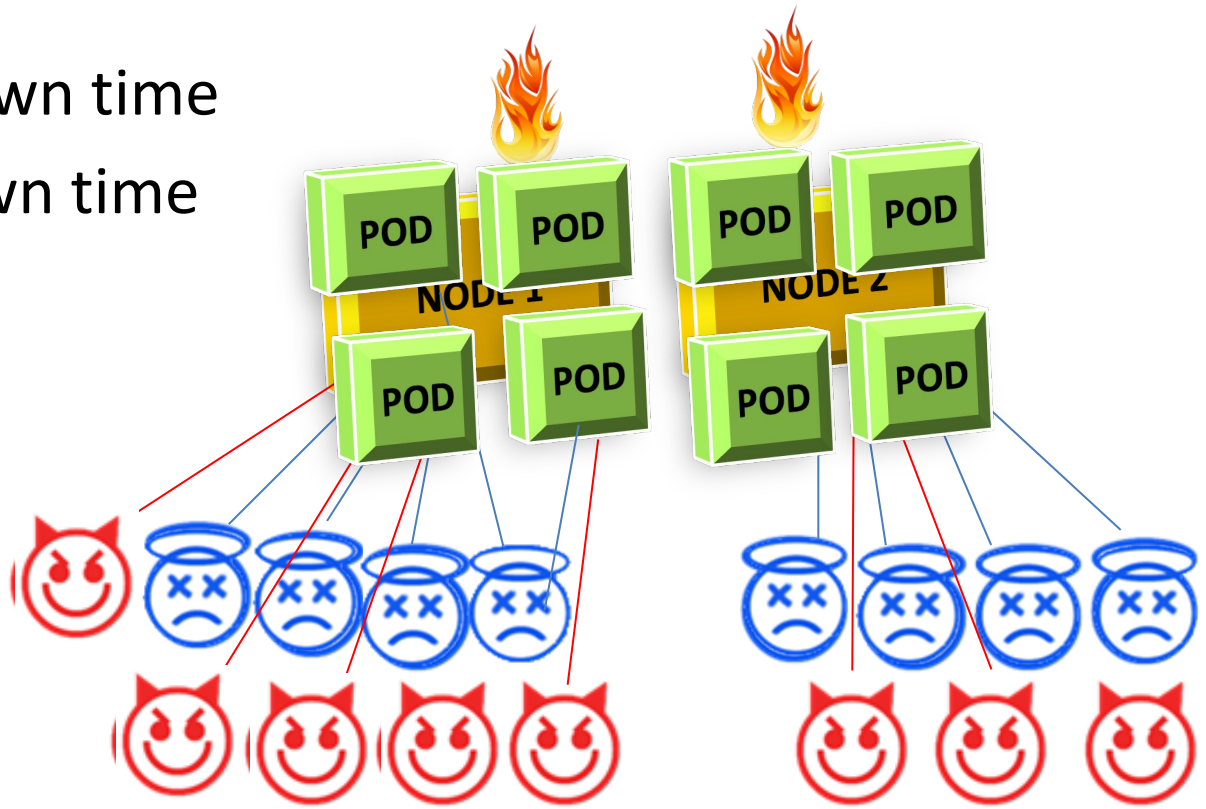
Kubernetes - Modern Cloud Platform

- Cluster of **nodes** (Node = VM)
- **Pod** is a basic compute unit (run one or more containers)
- Several pods run in a single node
- Auto-scaling for pods and nodes



Kubernetes Auto-scaling

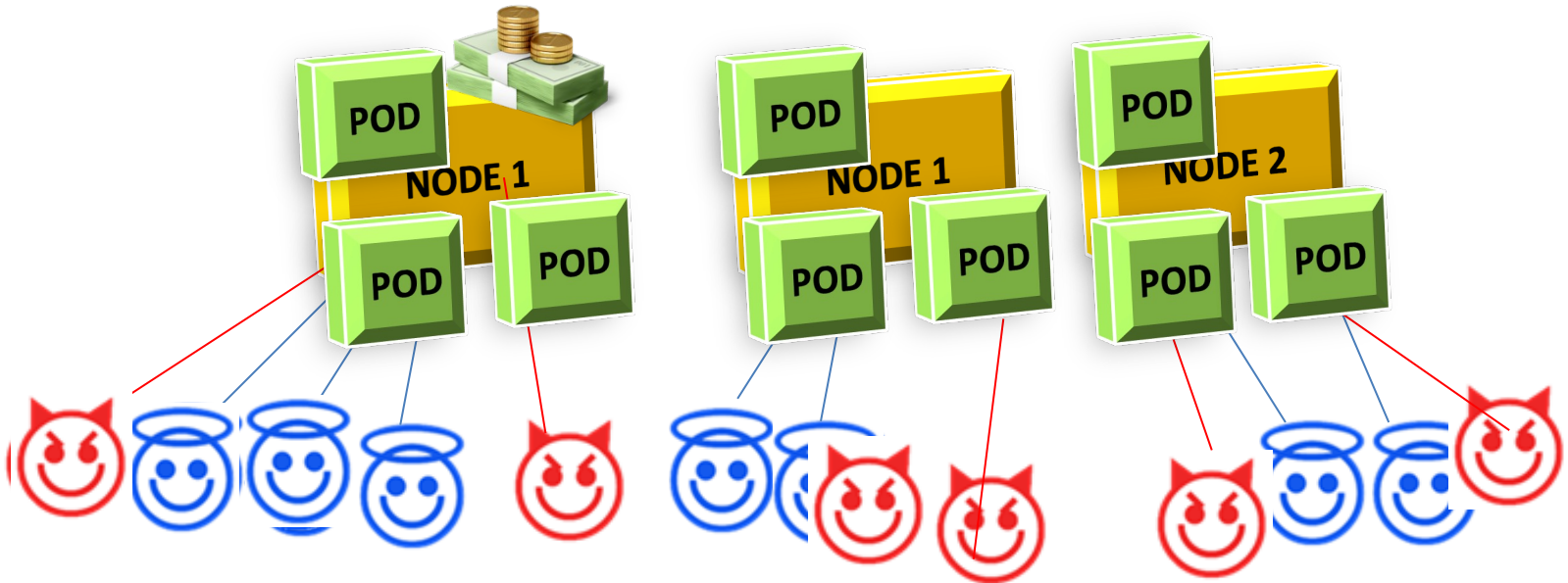
- Admin configures auto-scaling rules
  - Admin defines auto-scaling rules by metric's threshold and duration to track (scale-interval)
  - Action: Pods scale-up or scale-down according to: Target # Pods = $\lceil \frac{\sum_{1 \le i \le P} U_i}{U_{target}} \rceil$
  - Scale **nodes** to fit **pods** target number
- **Pods** : Short scale-up and scale-down time
- **Nodes**: Long scale-up and scale-down time



Kubernetes Auto-scaling

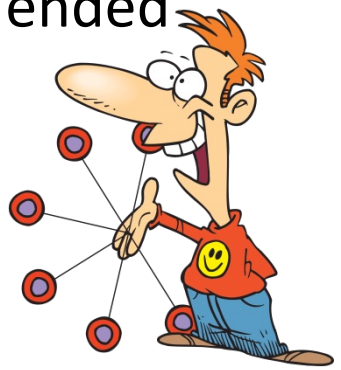
- Cost model: According only to the number of **nodes**
- Usually configured with spare capacity for scaling pods in nodes
- Scaling pods in existing nodes:

  Almost no performance damage + **no** economic damage
- Scaling nodes: Performance damage + economic damage

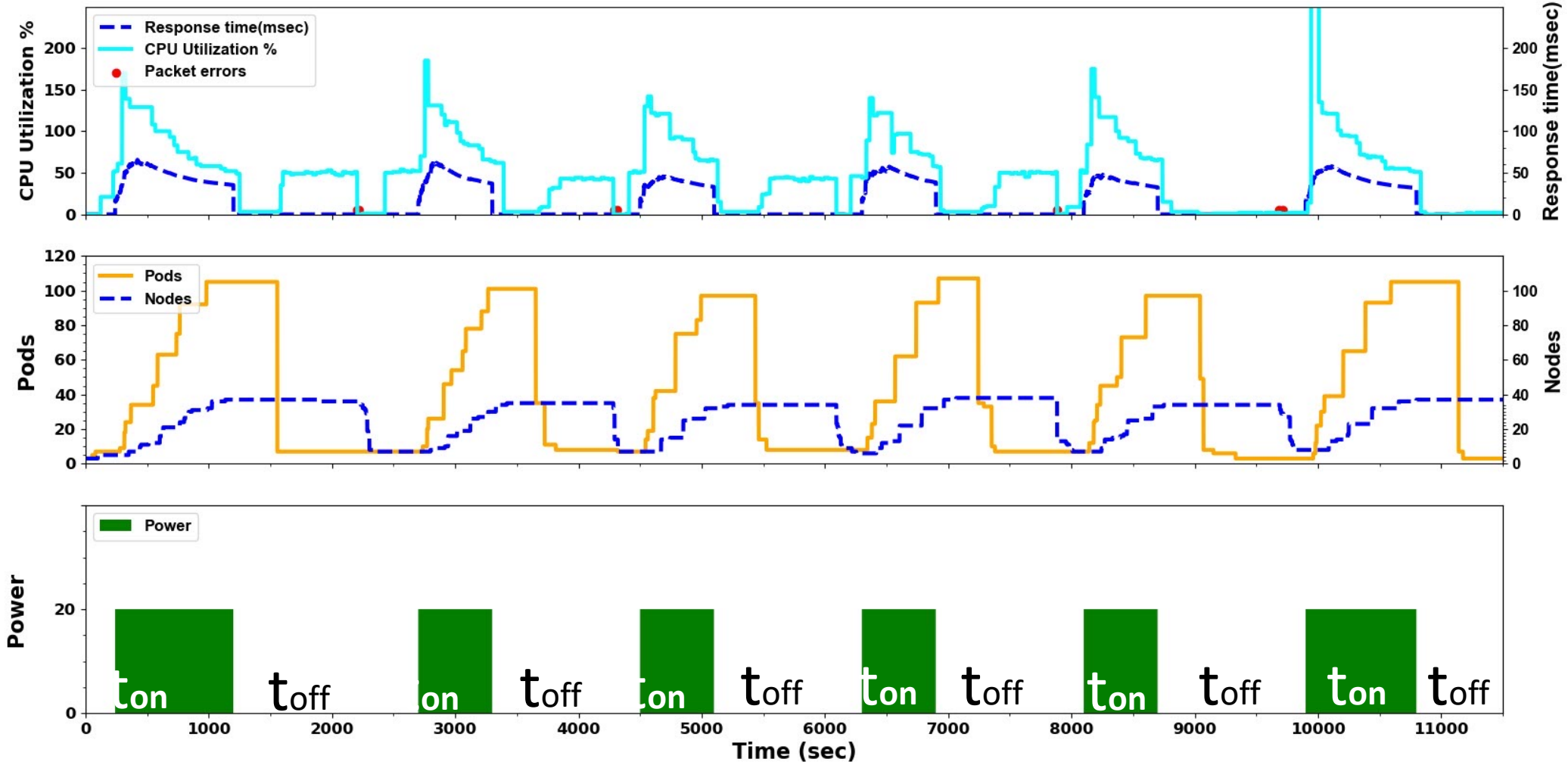


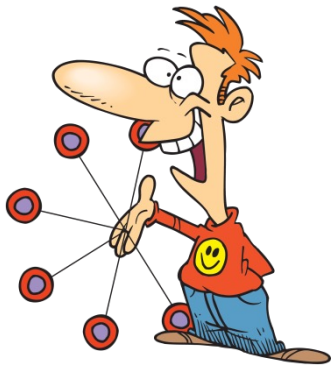
Yo-Yo Attack on Kubernetes

- The attacker repeatedly oscillates between the two phases:
  - On-attack phase: sends a burst of traffic → scale-up
    - Increase the number of pods
    - Increase nodes to fit required pods
  - Off-attack phase: stops sending the excess traffic → scale down
    - Start off-attack phase when the attacker detects the scale-up has ended
  - Repeat when the attacker detects the **scale-down** has occurred and ended



Yo-Yo Experiment on Kubernetes





With extra peak load of x20

Use Case Analysis: Experimental Results

System	Cost of attack	Performance damage (avg response time vs steady state) 	Economic damage 
Flat DDoS attack on Kubernetes	100% active	0	x7 of steady state
Yo-Yo attack on VMs	30% active	+66%	x5 of steady state
Yo-Yo attack on Kubernetes	30% active	+15%	x5 of steady state

With extra peak load of x20

Outcomes: (on GCP)

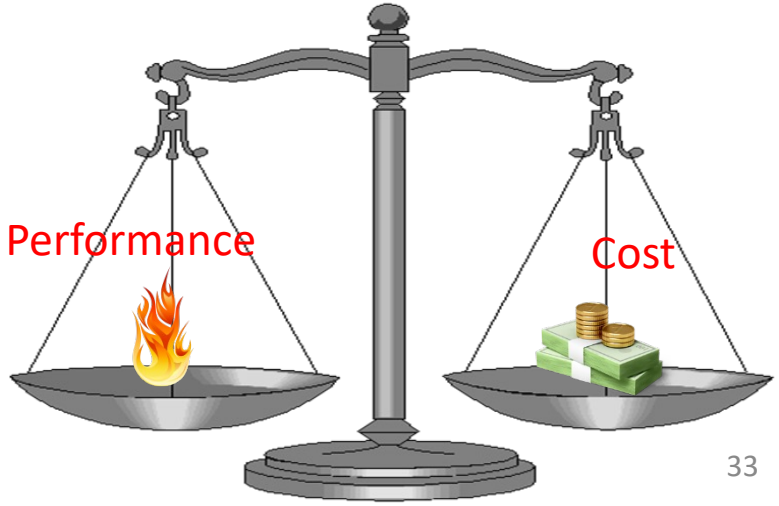
- In Kubernetes: Flat DDoS attack ~40% more economic damage but in Yo-Yo the attacker is active only 30% of the time
- Yo-Yo on Kubernetes vs VM: The same economic damage but reduced performance damage

Detecting and Mitigating Yo-Yo Attack

Solutions

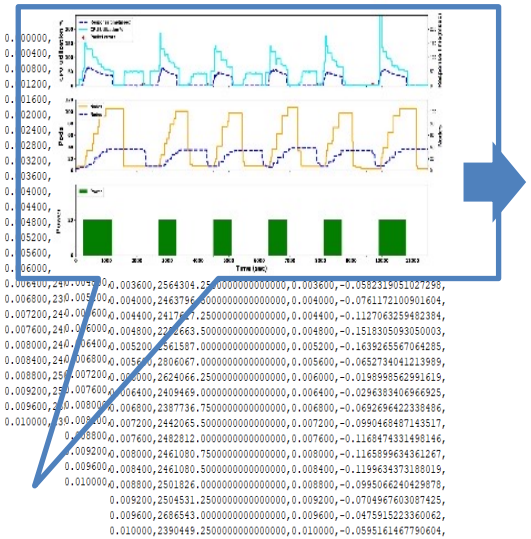
- Auto scaling is not a remedy for application DDoS
  - Addresses peak hours problem
- **Remedy:** Main question - What is more important to the cloud customer?
  - Performance: reserve pool, scale up early – scale down slowly
  - Cost: resource limitation
- **Mitigation:** Need for DDoS scrubber that copes with Yo-Yo attack
    - Machine learning detection
    - Finding attack “signature” if it exists*
    - Mitigation: filter attacked traffic

* Yehuda Afek, Anat Bremler-Barr, Shir Landau Feibish,
Zero-Day Signature Extraction for High Volume Attacks,
ACM/IEEE Transactions on Networking, 2019

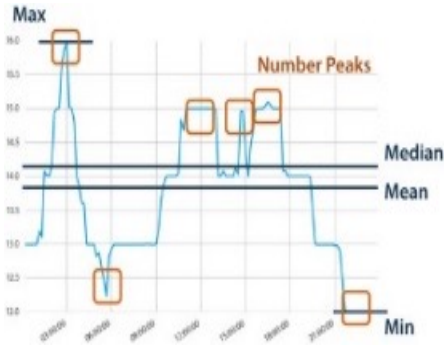


Detecting Yo-Yo Attack with XGBoost

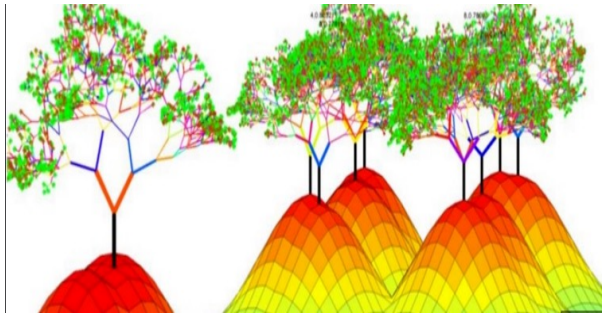
- Anomaly detection based on cluster's time series
  - Features: Response time, CPU Utilization , #Pods, #Nodes (*Max , Std, Mean, Median and Min)*
  - Train machine learning XGBoost model on **normal** and **attack** traffic:
    - *XGBoost is a* decision tree *model for* sparse data and limited datasets



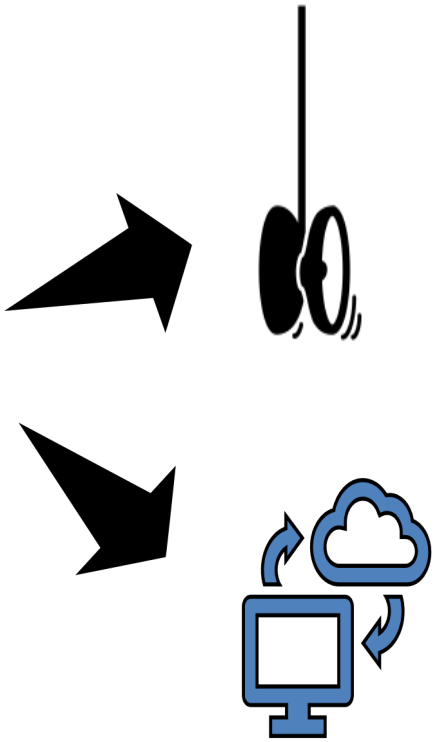
Cluster raw data



Pre-process & feature extraction



XGBoost ML model



Classification

Accuracy & Performance Comparison

	Precision	Recall	F1	Accuracy	Training time
XGBoost classifier	1.0	0.89	0.94	0.94	0.33 sec
1DCNN-LSTM Classifier	0.85	1.0	0.93	0.93	7200sec
Random Forest	0.83	0.82	0.82	0.82	0.15 sec
Logistic Regression	0.78	0.78	0.78	0.78	0.5 sec

$$\text{Recall} = \frac{TP}{TP+FN}$$

$$\text{Precision} = \frac{TP}{TP+FP}$$

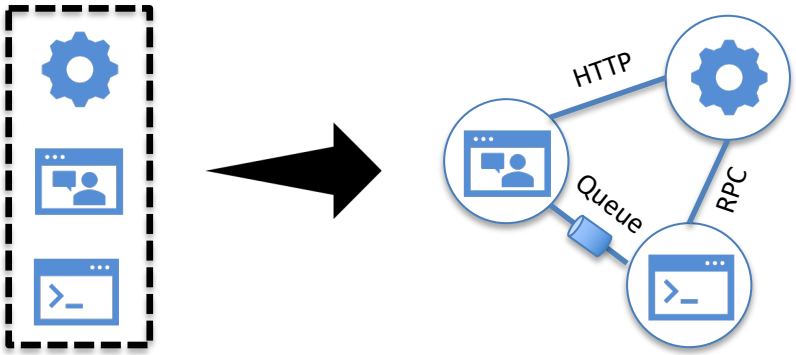
$$\text{F1} = 2 * \frac{Precision * Recall}{Precision + Recall}$$

$$\text{Accuracy} = \frac{TP+TN}{All}$$

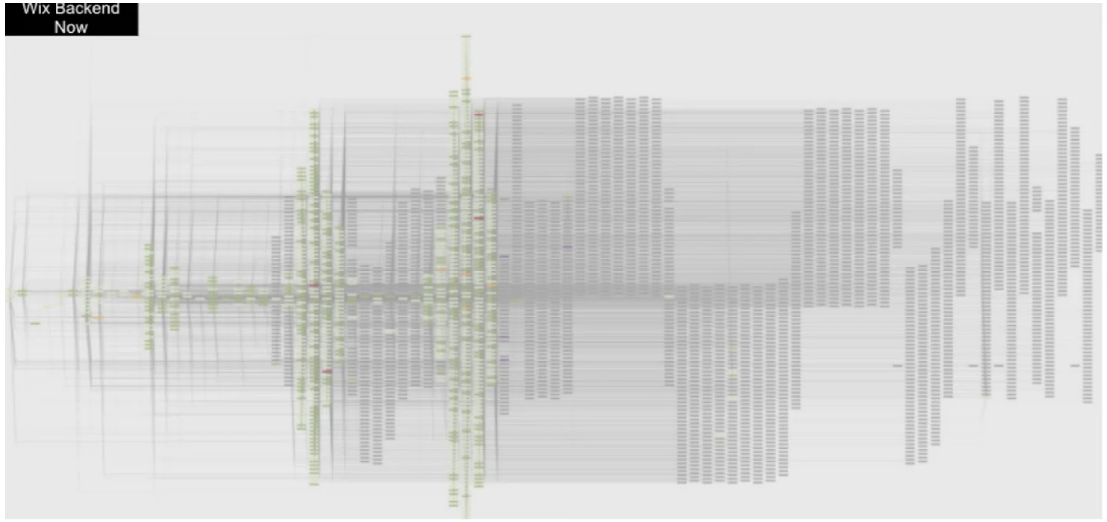
Tandem Effect: Colliding Auto-scaling Mechanisms of Micro-Services

Micro-Services Architecture

Monoliths to Microservices



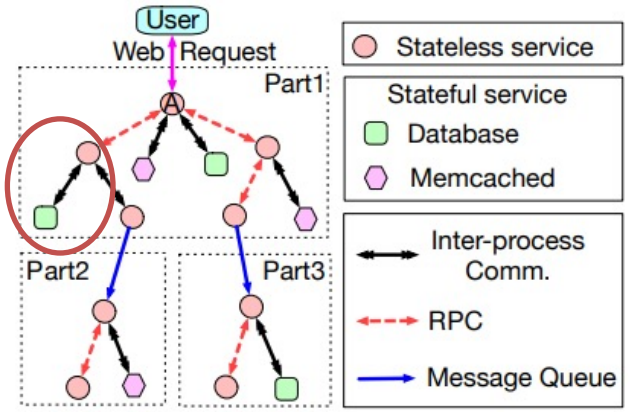
Wix production env. 2022



Our Research (IEEE Infocom May 2023):

Exploit the Tandem behavior of micro-services with separate auto-scaling properties to create DDoS & EDoS

The Rolling Tandem of Micro-services



(a) Components of call graphs

Often Self-Inflicted DDoS

Can drive: 20 km/h | 5 km/h | 5 km/h | 15 km/h

The speed is dictated by the slower participants

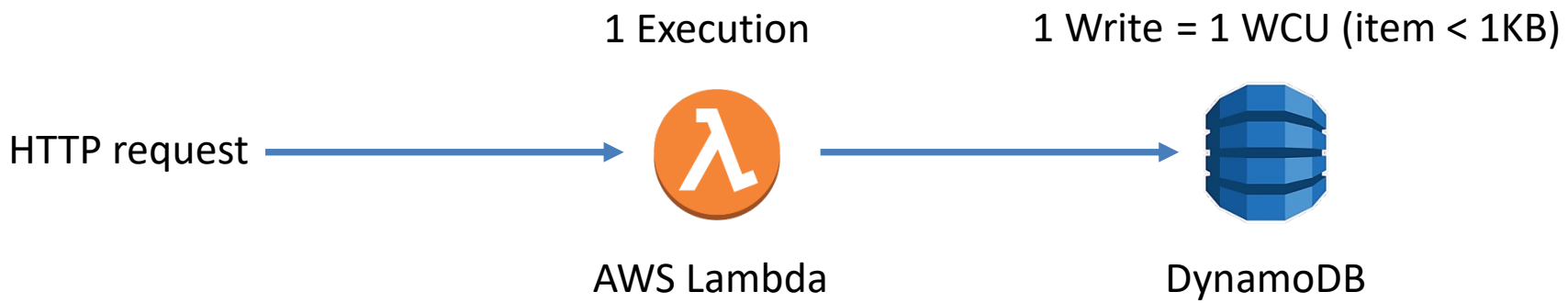
Tandem attack: DDoS Attack on Microservice Auto-scaling Mechanisms

Serverless Services

- Underlying servers & HW are not visible to the user
- Pay-as-you-go (expensive at high scale)
- What we show based on interaction of two serverless services:
  - **Different scaling properties** of services can be exploited to create economic and performance damage
  - **Serverless does not liberate from DDoS mitigation**

    Despite the notion of “everything handled by the cloud provider” same problem applies (and can be more severe)

Different Scaling & Billing Properties

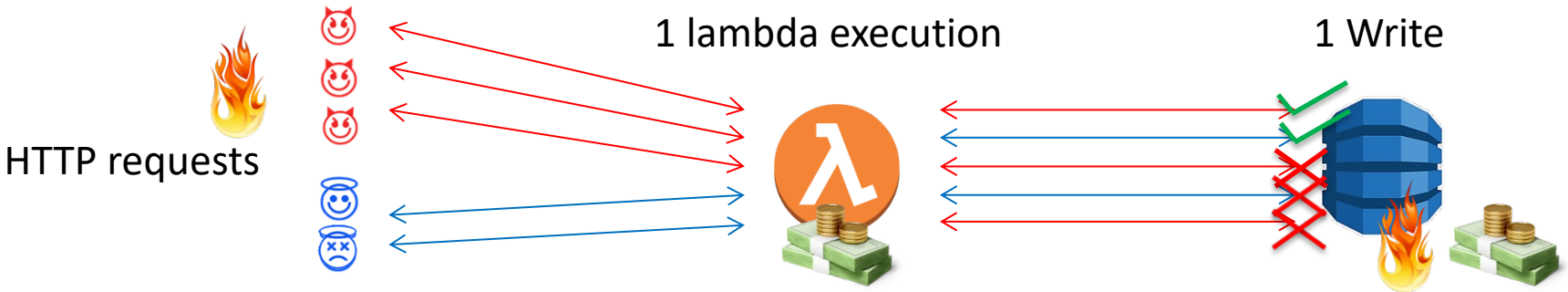


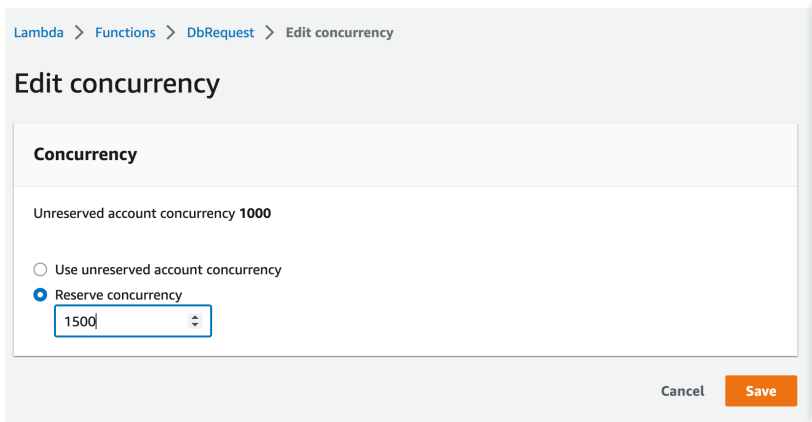
Fast Scaling

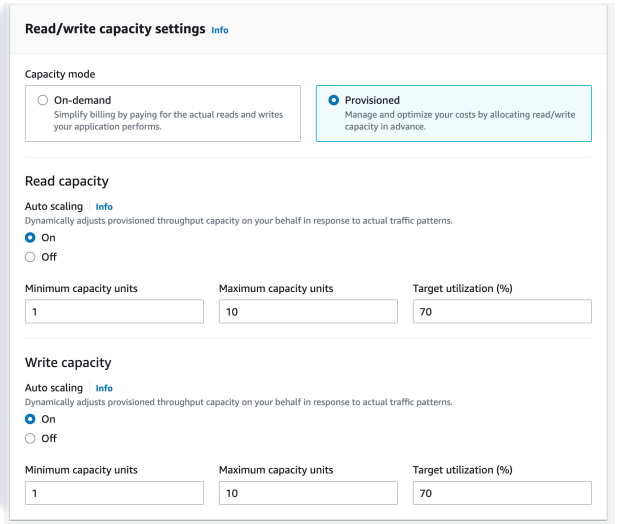
Slow Scaling

	AWS Lambda	DynamoDB	
		On-demand	Provisioned
Cost	• Request → function → cost • Execution time • HW, network and others.	Per successful completion (Pay-per-use)	Allocated resources
Scale Up	No upper bound; 3K for burst followed by 500 per minute	~15min	~7min
Scale Down	→ 0	Never	~22min/60min

Tandem Attack - Basic Example

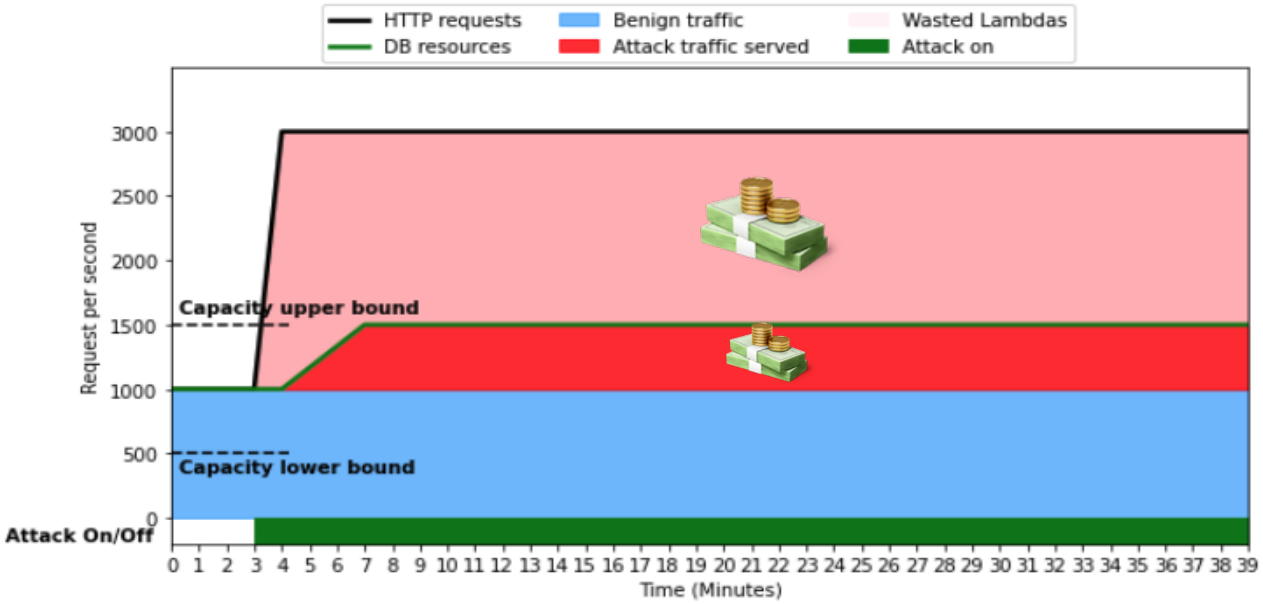






Tandem Attack Model (200% peak load)

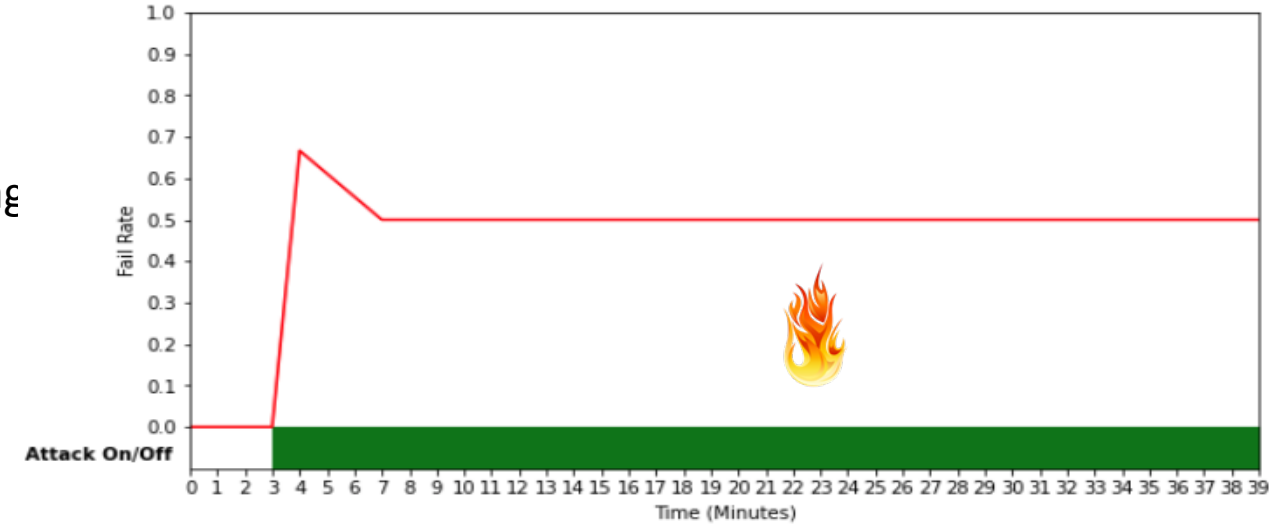
Economic damage



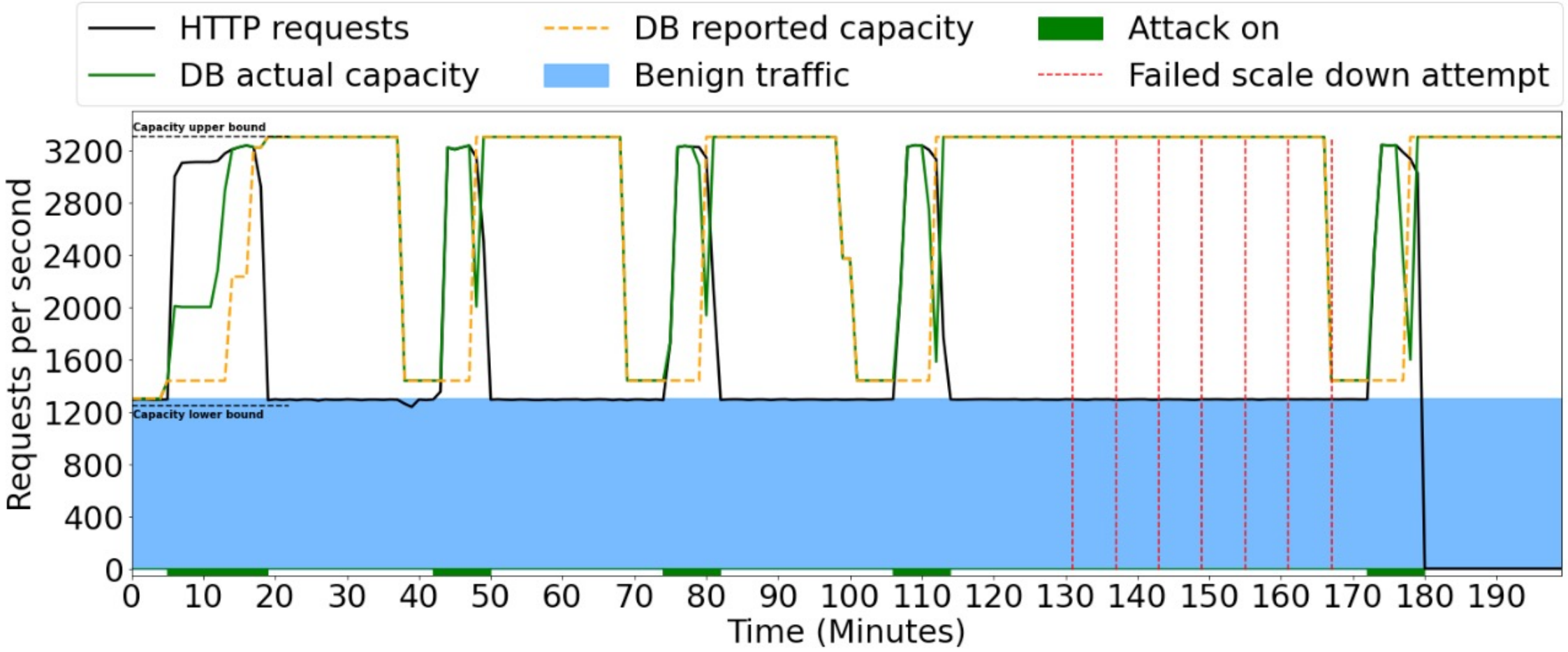
Lambdas that fail to write

DB writes created by attack traffic

Performance damag

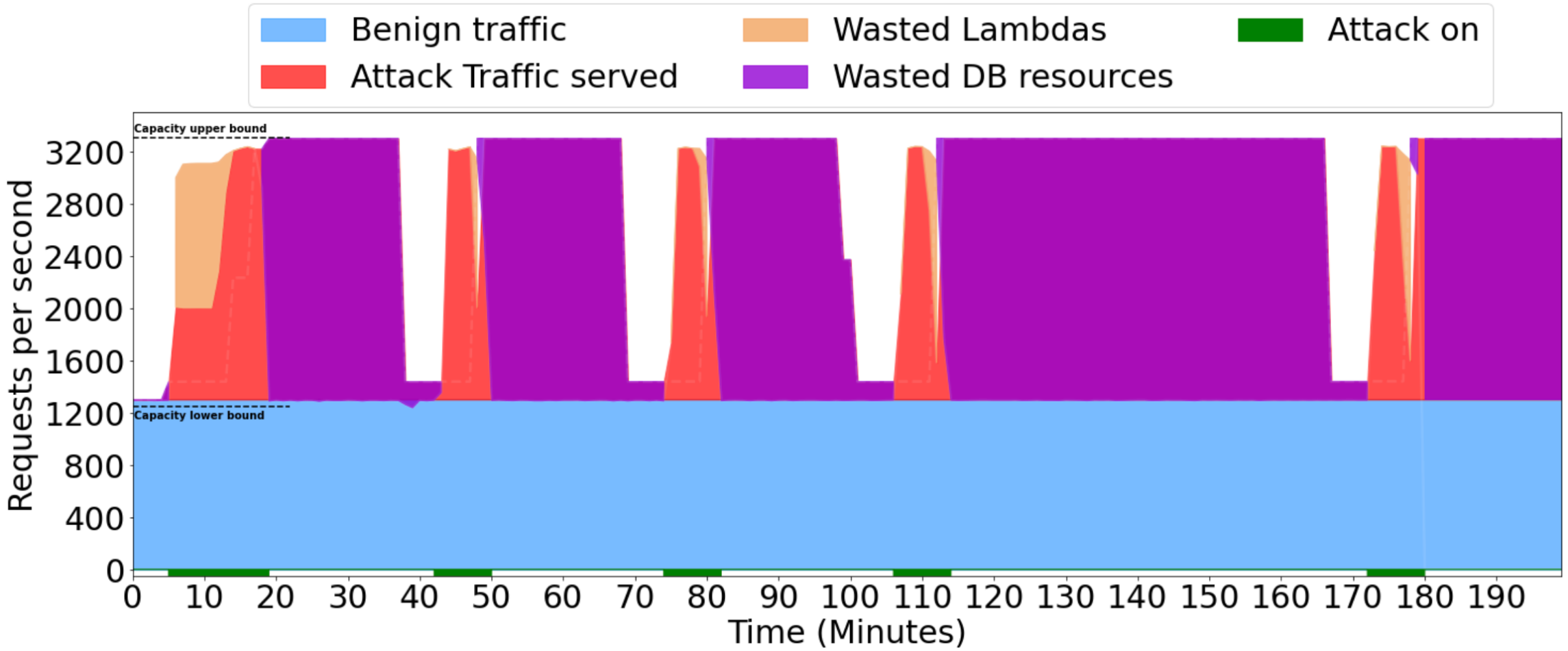


Tandem Attack with Yo-Yo

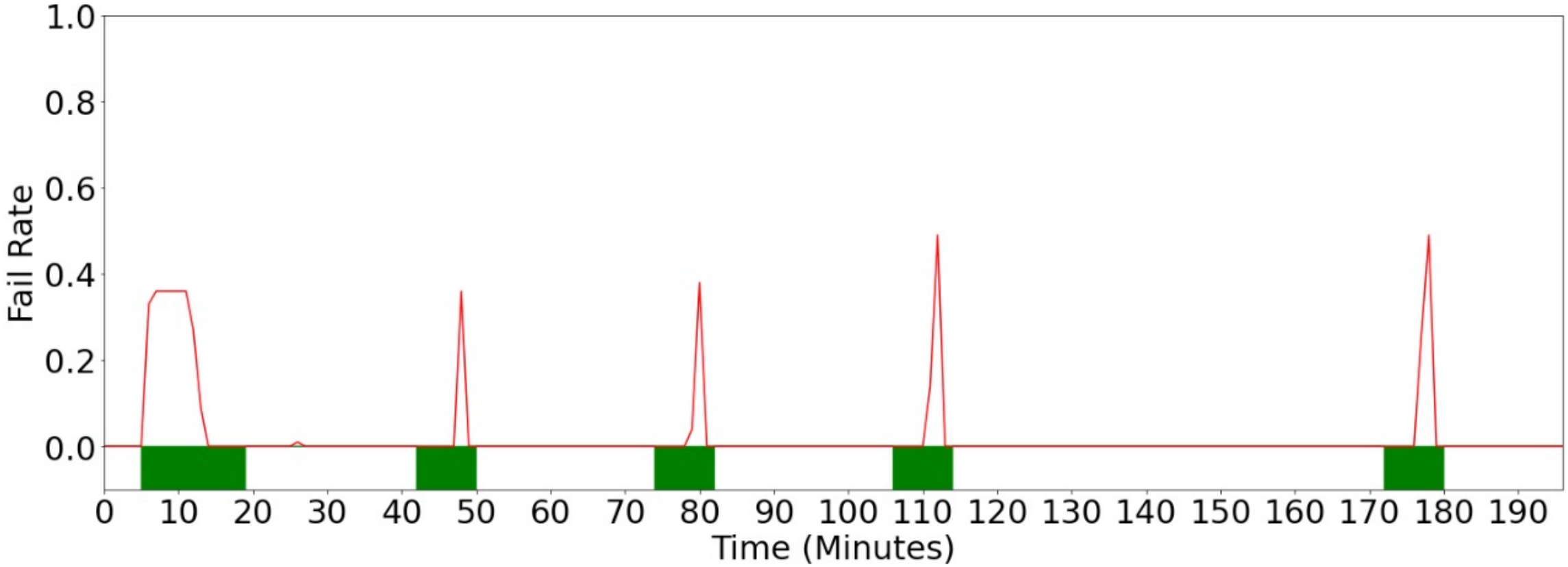


Limit in number of scale-down operations per day

Tandem Attack with Yo-Yo: Economic Damage



Tandem Attack with Yo-Yo: Performance Damage



Use Case Analysis: Experimental Results

System	Cost of attack	Performance Damage (Error Rate)	Economic Damage Lambda	Economic Damage DynamoDB
Flat DDoS (not breaching upper capacity)	100% active	0	x2 of steady state	x2 of steady state
Tandem Attack with Yo-Yo	30% active	+8%	x0.7 of steady state	~x2 of steady state

With extra peak load of x2

Outcomes:

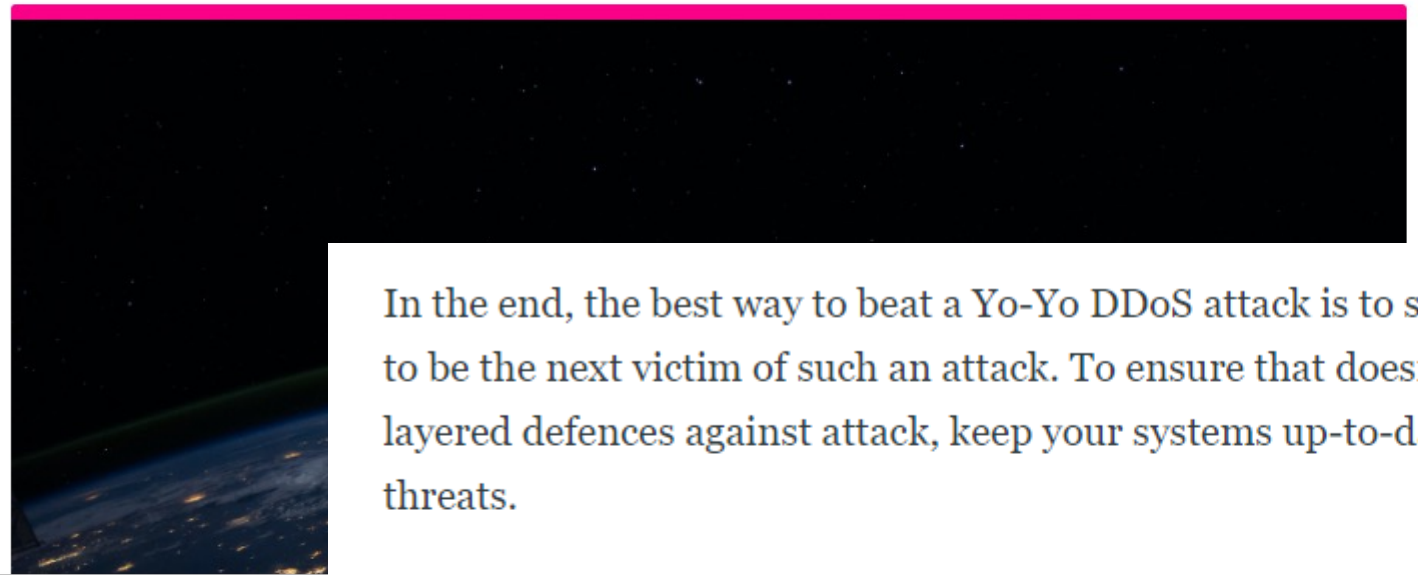
- Most of the economic damage is due to DynamoDB's relatively slow scale-down
- Uncorrelated scale-up of Lambda with DynamoDB create waste of running Lambdas

Startups

‘Yo-Yo’ DDoS Cyber Attacks; What they Are and How You Can Beat Them

‘Yo-Yo’ DDoS cyberattacks might sound like a bad video game or science fiction movie, but in reality, they’re a type of innovative distributed denial-of-service (DDoS) attack.

Guest Contributor / 1 May 2022 • 4 Min Read

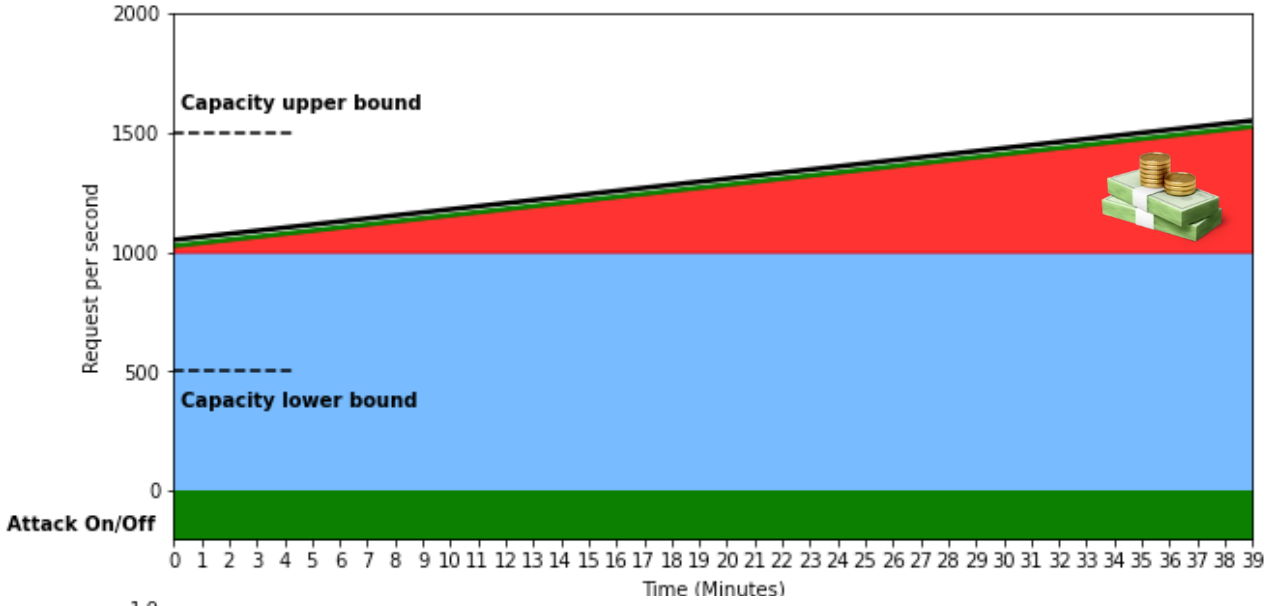


In the end, the best way to beat a Yo-Yo DDoS attack is to stay vigilant. You don’t want to be the next victim of such an attack. To ensure that doesn’t happen, use multiple layered defences against attack, keep your systems up-to-date, and stay on top of threats.

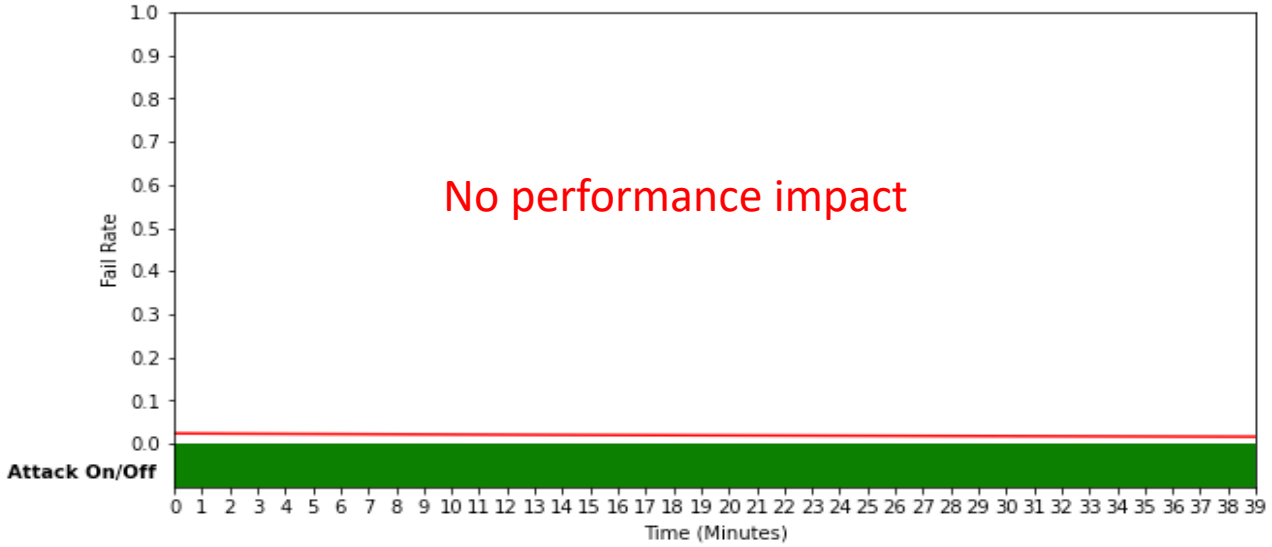
Written by Ido Vapner, CTO and Chief Architect at Kyndryl

Leech Attack: Under the Radar

Economic damage



Performance damage



Possible Mitigation

- Rate limiting - limit traffic as close as possible to the origin:
  - Easy & Effective
  - In our case: limit the Lambda to match the DB scaling capabilities
- Use services that scale quickly – e.g. Lambda
- Decoy by inserting noise to response time (when under attack)
  - We showed that measuring success ratio and RTT are enough for an attacker to create effective attack
  - By randomizing the RTT we take his ability to evaluate the system's state
  - As attacker's using same resources to attack multiple targets might be exaust by the process

Possible Mitigation cont.

- Retry comes with costs
  - Can compensate when services are not synced
  - Significant increase in latency (sometimes not acceptable)
  - If attack is realitvely strong might prolong the effect of the attack
- Developing better service control planes that can backpressure – relevant especially to cloud services
- Validate incoming traffic when possible (even when considering the time penalty)

Conclusions

- Trade-off between cost & performance
  - Over provisioning/reserve pools can compensate for DDOS up to a certain extent but with extra costs
- In large systems the micro-service connectivity and dependency can become complex and hard to analyze
- Serverless is not a solution for Tandem attack

Future Work

- Detection

  Collect real-time data from systems with complex micro-services dependency (eBPF?)

- Solution

  Algorithms that analyze and decide on mitigations in real time by implementing backpressure and sync scaling between services

For more information on our research:

http://www.deepness-lab.org

anatbremlerbarr@gmail.com

michael.czeizler@gmail.com

Questions ?