Clustering-Based IaaS Cloud Monitoring

Mahmoud Abdelsalam*, Ram Krishnan† and Ravi Sandhu*
*Department of Computer Science
†Department of Electrical and Computer Engineering

10th IEEE International Conference on Cloud Computing (CLOUD)
June 25-30, 2017

Goals

Develop a security monitoring framework for anomaly detection in cloud IaaS by:

leveraging essential cloud characteristics.

(i.e., Rapid Elasticity -> Auto-Scaling)

Proposed Approach

VMs doing the same function SHOULD behave similarly

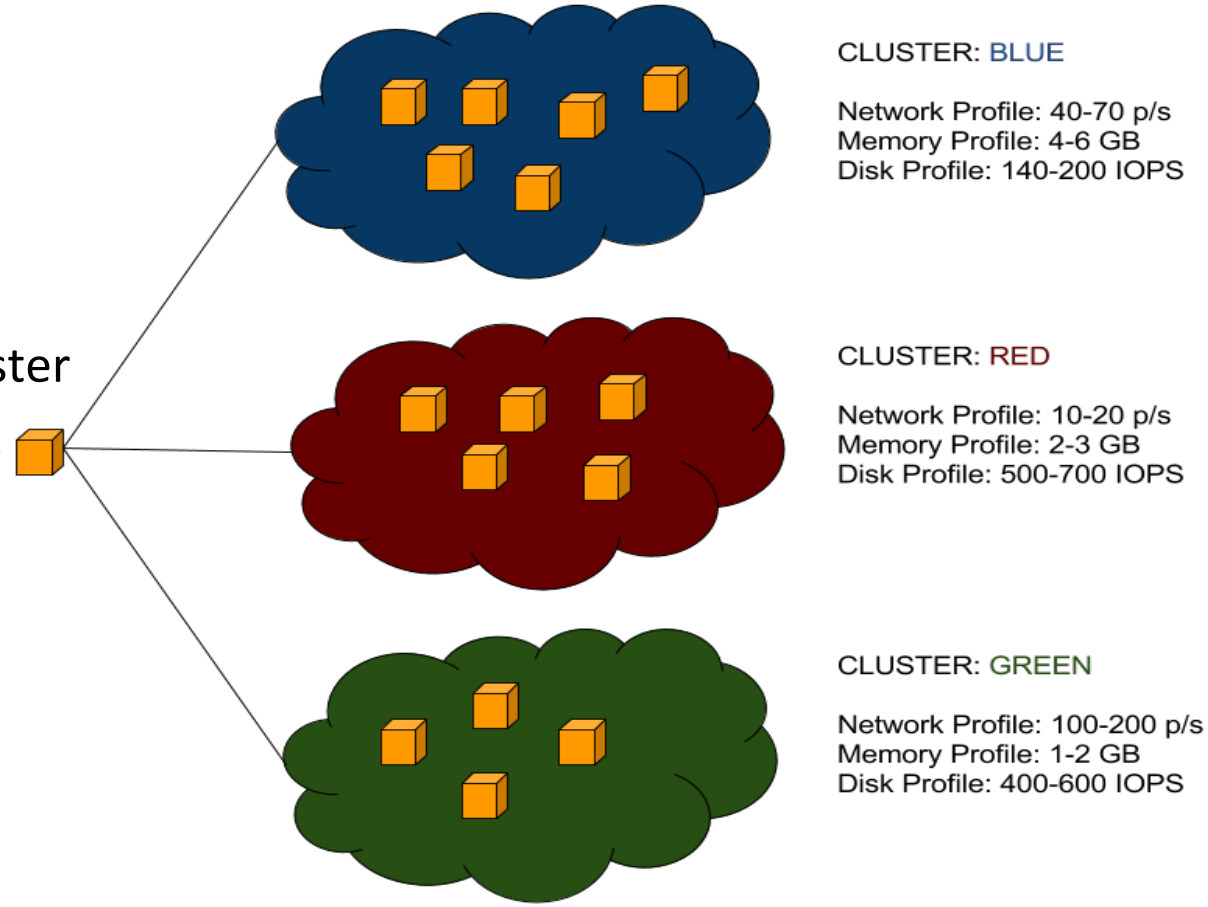
Cluster VMs based on their attributes

New VM gets created?

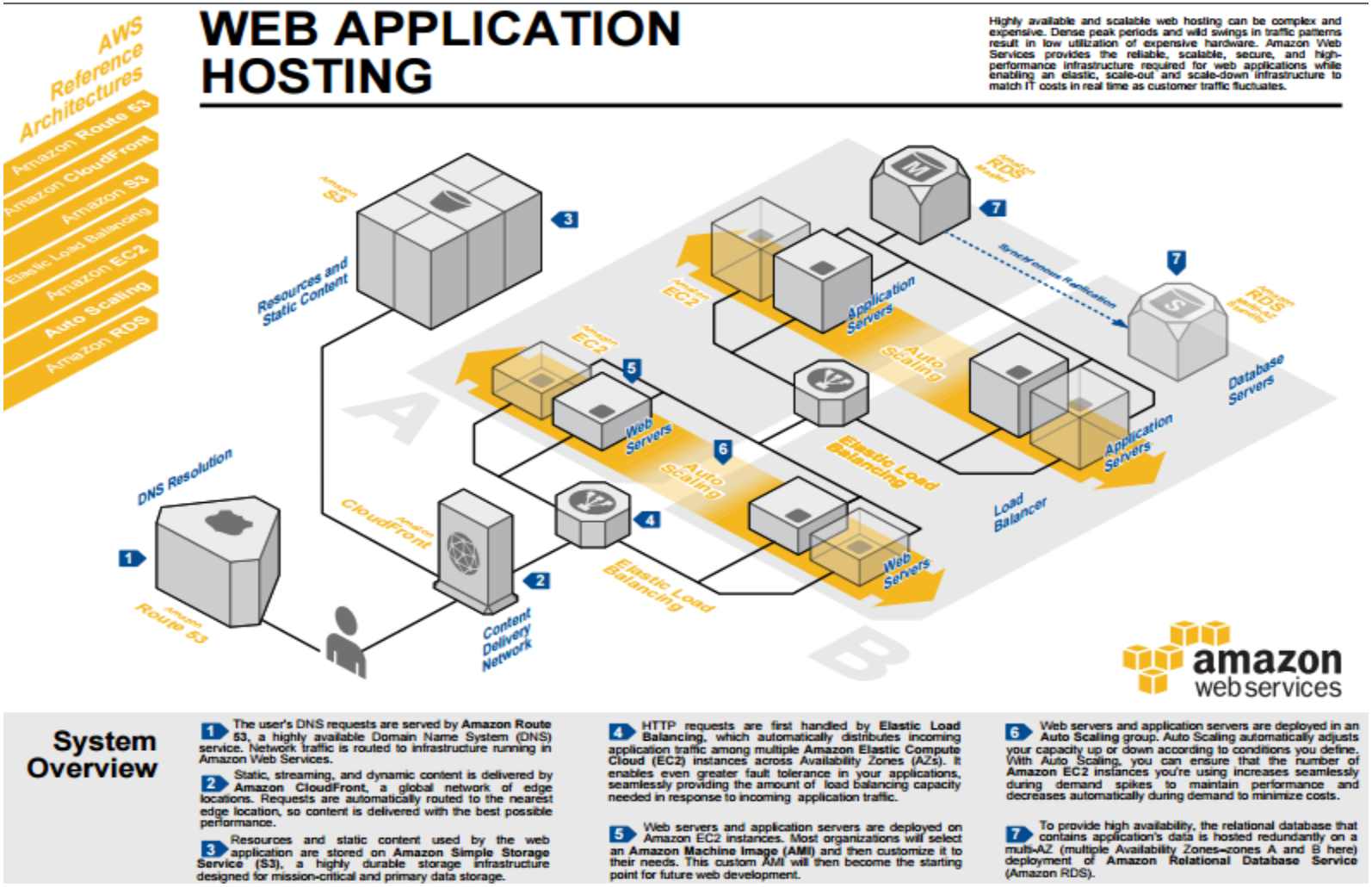
"fit" VM with existing cluster

If successful: good

If not: report anomaly



3-tier example



System Overview

1. The user's DNS requests are served by **Amazon Route 53**, a highly available Domain Name System (DNS) service. Network traffic is routed to infrastructure running in Amazon Web Services.
2. Static, streaming, and dynamic content is delivered by **Amazon CloudFront**, a global network of edge locations. Requests are automatically routed to the nearest edge location, so content is delivered with the best possible performance.
3. Resources and static content used by the web application are stored on **Amazon Simple Storage Service (S3)**, a highly durable storage infrastructure designed for mission-critical and primary data storage.
4. HTTP requests are first handled by **Elastic Load Balancing**, which automatically distributes incoming application traffic among multiple **Amazon Elastic Compute Cloud (EC2)** instances across Availability Zones (AZs). It enables even greater fault tolerance in your applications, seamlessly providing the amount of load balancing capacity needed in response to incoming application traffic.
5. Web servers and application servers are deployed on Amazon EC2 instances. Most organizations will select an **Amazon Machine Image (AMI)** and then customize it to their needs. This custom AMI will then become the starting point for future web development.
6. Web servers and application servers are deployed in an **Auto Scaling** group. Auto Scaling automatically adjusts your capacity up or down according to conditions you define. With Auto Scaling, you can ensure that the number of Amazon EC2 instances you're using increases seamlessly during demand spikes to maintain performance and decreases automatically during demand to minimize costs.
7. To provide high availability, the relational database that contains application's data is hosted redundantly on a multi-AZ (multiple Availability Zones–zones A and B here) deployment of **Amazon Relational Database Service** (Amazon RDS).

Reference: https://aws.amazon.com/architecture/

Sequential K-means


```
make initial guesses for means (centroids) m_1, m_2, ..., m_k
set the counters n_1, n_2, ..., n_k to zero
until interrupted
  get the next sample x
  if m_i is closest to x
    increment ni
    replace m_i with m_i + (1/n_i) * (x - m_i)
  end_if
end_until
```

Modified Sequential K-means

- For the first *m* minutes, VM *v* data samples is compared and counted to all clusters.
- VM *v* is assigned to the cluster with the maximum number of *v*'s data samples.
- After that, *v*'s data samples is compared only to its assigned cluster to check for anomalies as well as updating the cluster's information.
- Stabilizing time is introduced to avoid false alarms.

Black-box Features

Metric	Description	Unit
CPU utilization	Average CPU utilization	%
Memory usage	Volume of RAM used by the VM from the amount of its allocated memory	MB
Memory resident	Volume of RAM used by the VM on the physical machine	MB
Disk read requests	Rate of disk read requests	rate/s
Disk write requests	Rate of disk write requests	rate/s
Disk read bytes	Rate of disk read bytes	rate/s
Disk write bytes	Rate of disk write bytes	rate/s
Network outgoing bytes	Rate of network outgoing bytes	rate/s
Network incoming bytes	Rate of network incoming bytes	rate/s

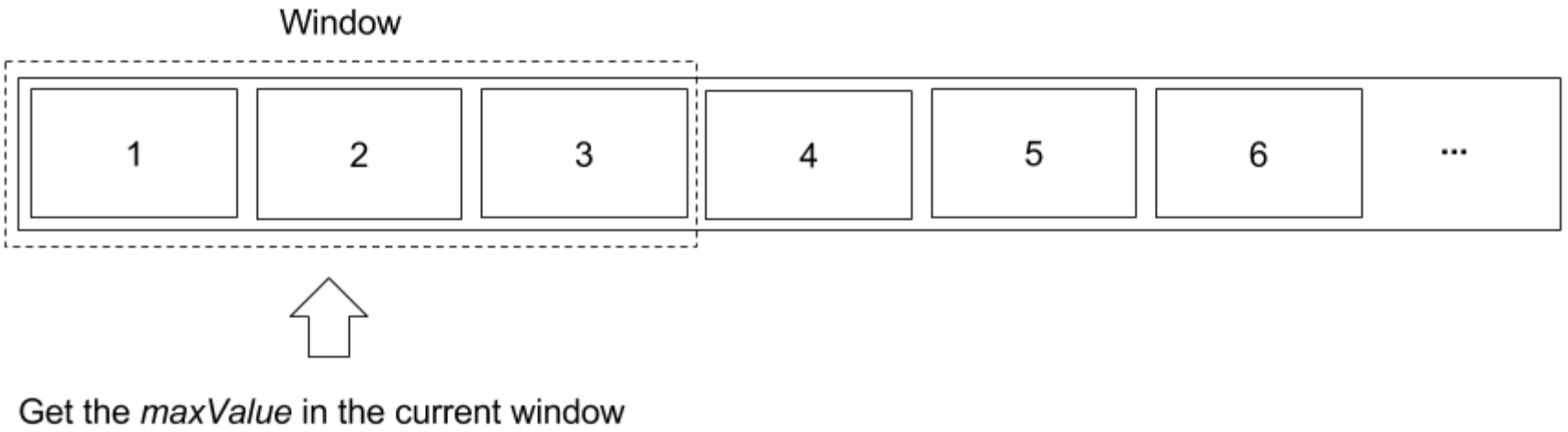
Features Normalization

- Clustering algorithms can be very sensitive to data scales (more weight goes to features with higher values)
- Data samples are not of the same scale.
- Min-Max normalization is a technique where you can fit the data with a pre-defined boundary.

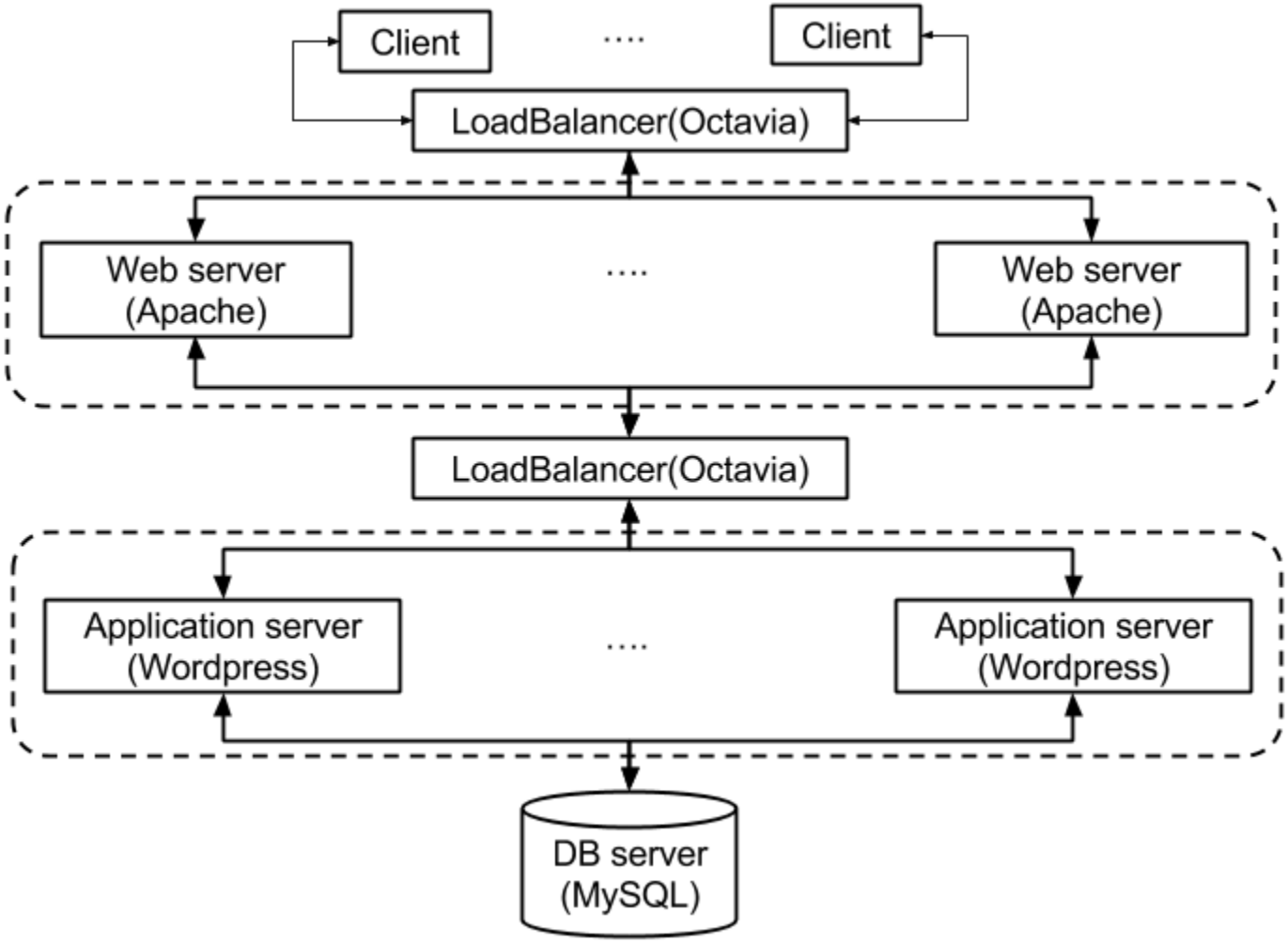
$$A' = \frac{A - minValue_A}{maxValue_A - A}$$

Features Normalization (cont.)

- How to get maxValue?
  - Pre-defined based on knowledge
  - Get the max value of the data (infinite time series data?)
- Use Min-Max normalization based on a fixed-size sliding window.



Testbed Setup



Traffic Generation

- Simple & realistic traffic generation
  - Poisson
    - Used in many cases due to its simplicity
  - On/Off Pareto
    - Internet traffic is proved to be of self-similar nature

- The simulation parameters are as follows:
  - Generator: On/Off Pareto, Poisson
  - Number of concurrent clients: 50
  - Requests arrival rate/hour: 3600
  - Type of requests: GET and POST(randomly generated)

Experiments Evaluation

- We use four metrics to evaluate the effectiveness and applicability of our approach:

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

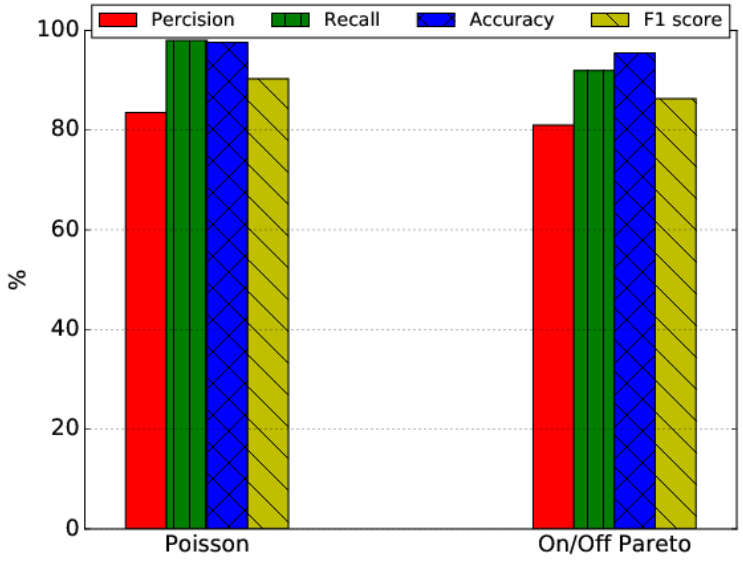
$$Recall = \frac{TP}{TP + FN}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

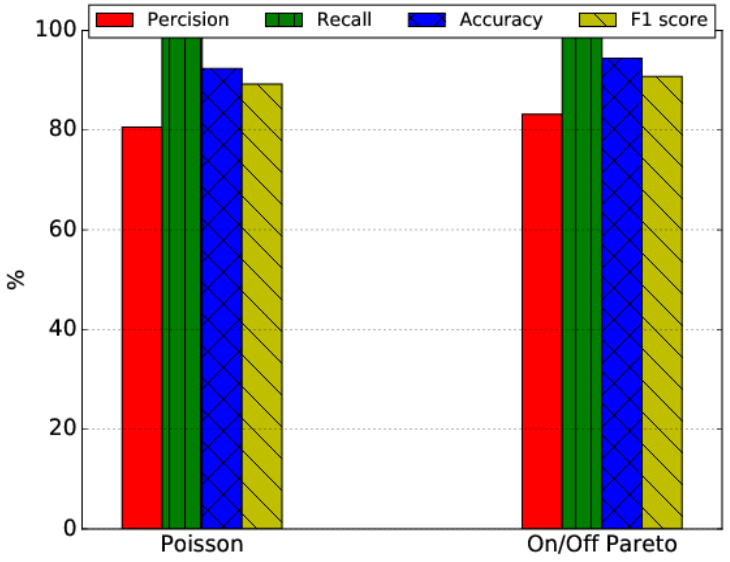
$$Fscore = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Results

Injected Anomalies: cpu, memory and disk intensive



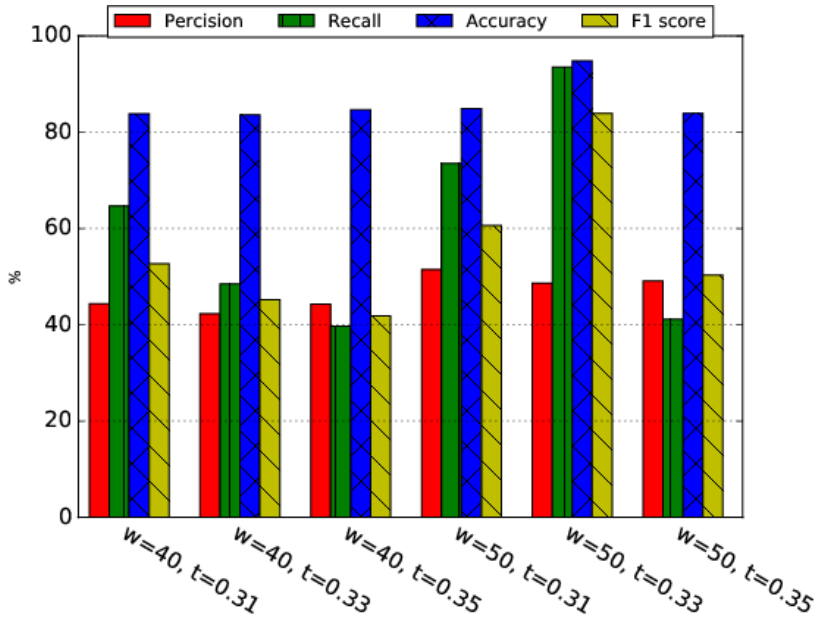
EDoS: One form of EDoS is to create some VMs that remain dormant or idle



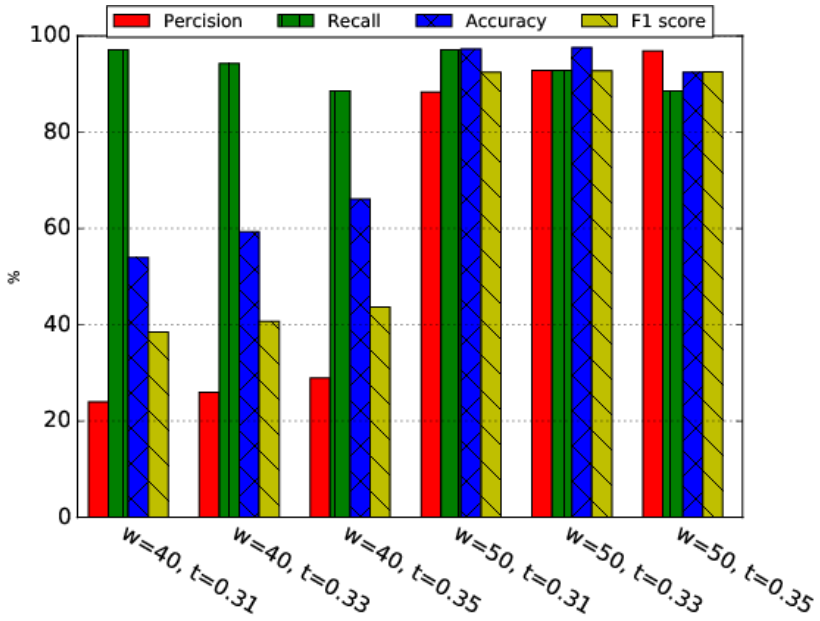
Results (cont.)

- Ransomware is a very critical threat to cloud.
- Netskope's quarterly cloud report states that 43.7% of the cloud malware types detected in cloud apps are common ransomware delivery vehicles.

Ransomware (KillDisk) - Poisson



Ransomware (KillDisk) - On/Off Pareto



Limitations

- Vulnerable to low-profile anomalies and malware.
  - Hard to detect using black-box features.
- Expert is needed in parameter tuning which, in some cases, unavailable/unaffordable for some cloud tenants.
- Attacks gradually change normal behavior of VMs
  - Harder since the change of behavior has to be in all VMs of the same cluster at the same time.

Questions/Comments