

# Getting Started With GRANULATE

Real-Time Continuous Optimization.  
Improve Performance. Reduce Compute Costs.

Reduce  
Costs By

60%

Reduce  
Response  
Time By

40%

Increase  
Throughput

5X

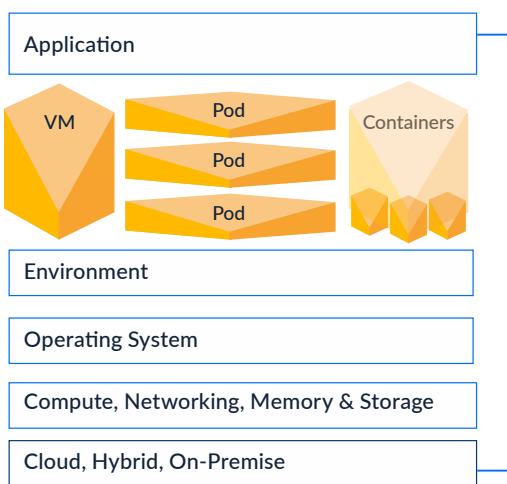
Code  
Changes

0

## Overview

Granulate employs real-time continuous performance optimization to allow organizations to handle compute workloads with 60% fewer servers while improving performance by 40%, with no code changes required.

## Technology Stack



## Real Time Continuous Optimization





### Simple Installation

Install the Granulate GAgent with simple command line.

The rest is completely automated — zero manual configuration.



### Better Performance

Optimize the infrastructure for the application - Reduce response time and improve throughput on every machine



### Reduced Costs

Deliver better performance with smaller cluster size, less compute resources and lower 3rd party costs

## Granulate Agent - The GAgent

Granulate's agent is the foundation of the real-time continuous optimization solutions.

Granulate's agent automatically learns the application's specific resource usage patterns and data flow to identify contended resources, bottlenecks and prioritization opportunities in each specific server. The agent then tailors OS-level scheduling and prioritization resource management decisions to improve the infrastructure's application specific performance and enable significant cost reduction.

The agent is comprised a kernel module that enables to monitor, analyze and optimize contended resources managed by the OS - CPU scheduling, I/O, networking, memory, disk and critical sections.

## GAgent Data Collected

The GAgent monitors and collects real-time granular measurement of the utilization and performance of each machine. The agent consolidates the information and uses its AI algorithms to optimize the infrastructure to the specific application needs at any given moment.

The GAgent minimizes the footprint in both memory and CPU by taking advantage of the concept of time-localization, learning using a sliding window of only a few seconds.

No PII Data or user data is collected by the agent.

### CPU

- Thread scheduling per process
- Thread scheduling per core
- Critical sections lock contentions
- Load averages

### I/O

- Access patterns per process (read/write, random/sequential)

### Memory

- Allocation and release patterns per process
- Access patterns per process (read/write, random/sequential)
- TLB Misses

### Network

- Per socket metadata - Window Size, receive vs send ratio, congestion control, RTT

## GAgent Overhead

GAgent code modules are optimized to efficiently limit the overhead on each machine resource consumption. The agents code modules are optimized to use memory and to free resources when they're no longer needed to burden application execution as little as possible.

An example of the Granulate agent resource consumptions is below, tests were made on AWS EC2 machine C5.xLarge

- CPU: ~ 0.1 % of CPU on average
- Memory: ~ 20MB or RAM (RSS memory)
- Network Bandwidth: ~ 2KB/s ▼ | 20 KB/s ▲

## Getting Started

### Every Environment

Can be installed on many different platforms

### Every Infrastructure Type

Supports public, private, multi, or hybrid-cloud

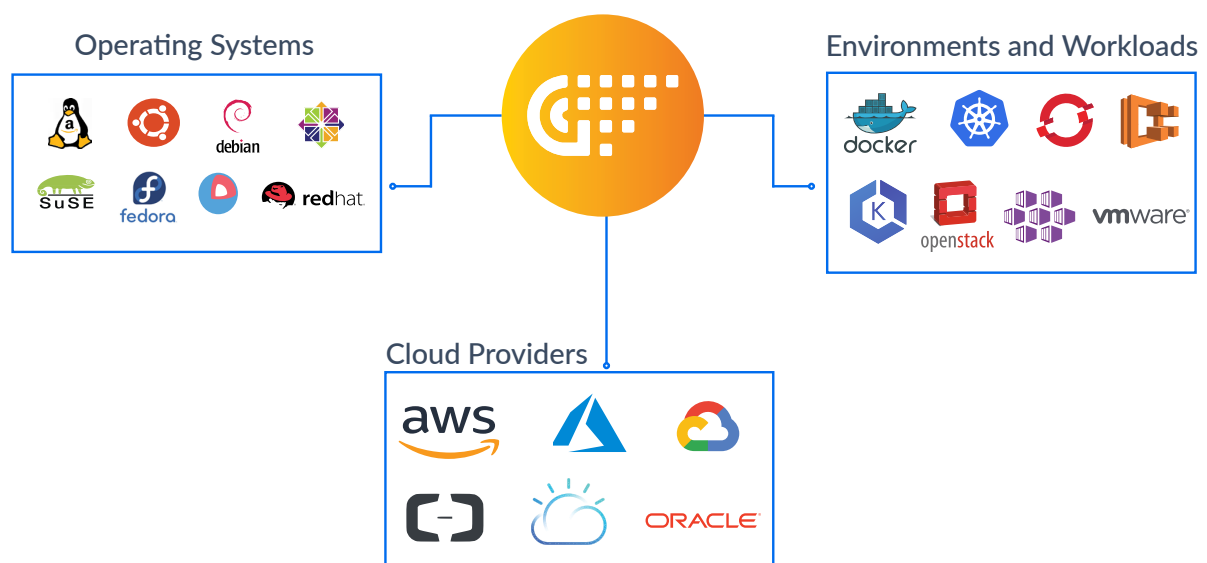
### Easy Installation

Single command line installation

### Quick Value

Cost savings and performance improvements in days

## Supported Infrastructures



# Installation

The agent can be installed on each platform either directly or using an infrastructure provisioning and automation solutions in a frictionless installation process without any code changes required.

## Deployment Models


### Granulate SaaS

If your network allows SSL communication with the Granulate cluster, you're ready to install GAgent and start experiencing performance improvements. The agent will autonomously optimize your infrastructure and will communicate with Granulate cluster for metrics and configuration updates. Granulate SaaS supports out-of-the-box secure installation leveraging VPC Peering and VNet Peering.


### Granulate On-Premise

If your organization's policies prohibit cloud-based solutions, you can still benefit from all Granulate performance improvement features while keeping your data on-premises. This is possible thanks to Granulate on-premise deployment model offered. Granulate on-premise deployment supports both an installation with secure communication with Granulate's cloud infrastructure and a completely independent on-premise deployment.


## Frictionless Installation With 0 Code Changes




```
name 'deploy_granulate_agent'
description 'Role that deploys Granulate components to servers'
default_attributes {
  'granulate' => {
    'api_key' => '<YOUR_API_KEY>',
    'application_key' => '<YOUR_APP_KEY>',
    'agent' => true,
  }
}
run_list node
recipe[granulate::granulate-agent]...
```




```
- name: Install Granulate Agent on servers in AWS
hosts: tag_granulate_yes
become: yes
user: centos
roles:
  - { role: Granulate.granulate, become: yes }
vars:
  granulate_config:
    tags:
      - env:dev
    logs_enabled: true
    process_config:
      ...
```




```
Resources:
  AgentInstaller:
    Type: 'Granulate::Installer::Agent'
    Properties:
      Hostname: <Host Name>
      TenantId: <Tenant ID>
      ApiToken: <API Token>
  EC2Instance:
    Type: 'AWS::EC2::Instance'
    Properties:
      ...
```




```
resource "aws_instance" {
  ami           = "${var.ami}"
  instance_type = "m5.2xlarge"
  provisioner "local-exec" {
    command = <EOH>
    curl -o jq https://s3.amazonaws.com/<Bucket URL>/cloud/8753_jqEDH }}....
```



```
define granulate_agent[tenantid]
  local[present] { absent } $ensure = 'present'
  string $integration_name = 'user'
  string $version = 'user'
  {
    include granulate_agent
    if $ensure == 'present' {
      exec { [ 'install', $integration_name, $tenantid ] :> integration install
        command => 'granulate-agent --tenant-id $tenantid --integration $integration_name'
        unless => [ 'granulate-agent --tenant-id $tenantid --integration $integration_name' ]
        require => [ 'granulate-agent --tenant-id $tenantid --integration $integration_name' ]
      }
    }
  }
end
```



```
curl -s https://s3.amazonaws.com/<download bucket> | sudo \
CLIENT=<Client Hash> \
SERVICE=<Service Name> \
bash
```



```
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: granulate-agent
  namespace: kube-system
spec:
  selector:
    matchLabels:
      app: granulate-agent
  template:
    ...
```

## Learning Process

Following the installation the agent automatically analyzes the running environment and identifies application and resources bottlenecks. The agents passively monitors and learns usage and access patterns of the machines resources - CPU, Memory, I/O, Network.

Taking advantage of time - localization and minimizing the footprint in both memory and CPU, Granulate's agents learn in a sliding window of only a few seconds and begin to optimize accordingly. The learning process is ongoing as traffic changes throughout the day, as deployments of new versions occur and as cluster sizes change sometimes with different machine types.

## Performance Optimization

AI based dynamic, low level adaptations to improve performance and reduce the amount of required machines.

By learning the application's specific resource usage patterns and data flow and analyzing CPU scheduling order, oversubscribed locks, memory, network and disk access patterns it is possible to identify contended resources, bottlenecks and prioritization opportunities and solve them using scheduling and prioritization algorithms.

