ADVANTECH
WISE-PaaS
AIoT Solutions & Marketplace

WISE-DeviceOn

IoT Device Operation Management

User Manual

English v-1.0.14

Revision History

Date	Version	Author	Reviewer	Description
2019-12-16	1.0.5	Sephiroth.Wang		First version released Section 1, 2, 3, 4.4, 6, 7
2019-12-23	1.0.5	Alex.Shao	Sephiroth.Wang	Section 4.1, 4.2, 4.3, 4.5
2019-12-27	1.0.5	Terry.Lu	Sephiroth.Wang	Section 5.1
2020-01-08	1.0.5	Iris.Pan	Sephiroth.Wang	Section 5.2
2020-01-10	1.0.5	Wunhuei.Liou	Sephiroth.Wang	Section 5.3
2020-02-03	1.0.6	Sephiroth.Wang		Section 6.7, 6.8, 6.9
2020-02-10	1.0.6	MingWei.Tasi	Sephiroth.Wang	Section 4.6
2020-02-12	1.0.6	Sephiroth.Wang		Section 6.10
2020-03-11	1.0.6	Scott.Chang	Sephiroth.Wang	Section 6.3
2020-03-25	1.0.7	Sephiroth.Wang		Add Section 6.8, and move 6.8-> 6.9, 6.9->6.10 Add Section 7.2
2020-04-01	1.0.8	Sephiroth.Wang		Change wallpaper on first page Add new log mechanism on Section 6.6 Add Section 4.7 for device provisioning.
2020-04-06	1.0.9	Alex.Shao	Sephiroth.Wang	Section 2.1.4
2020-04-10	1.0.10	Daniel.Hung	Sephiroth.Wang	Add plugin development on RISC (Section 5.1.4)
2020-04-21	1.0.11	Listzt.Kao	Sephiroth.Wang	Add plugin development on Android (Section 5.1.5) Modify WISE-DeviceOn -> WISE-DeviceOn
2020-09-02	1.0.12	Sephiroth.Wang		Review all document and update new user interface for v-4.2.3
2020-09-09	1.0.13	Sephiroth.Wang		Add Telegram, Teams, Slack Notification Service (4.3.5 ~ 4.3.7)
2020-09-22	1.0.14	Sephiroth.Wang		Add Linux Agent Onboarding and Re-adjust catalog.

Table of Contents

1. Introduction

A surge in market demand for Industrial IoT products has rapidly increased the number of connected devices that are currently deployed and managed across different locations. It is essential to effectively manage, monitor, and control thousands of connected devices while ensuring uninterrupted service. Devices must work properly and securely after they have been deployed - without requiring frequent visits from service technicians. Customers require secure access to their devices in order to detect, troubleshoot, and undertake time-critical actions.

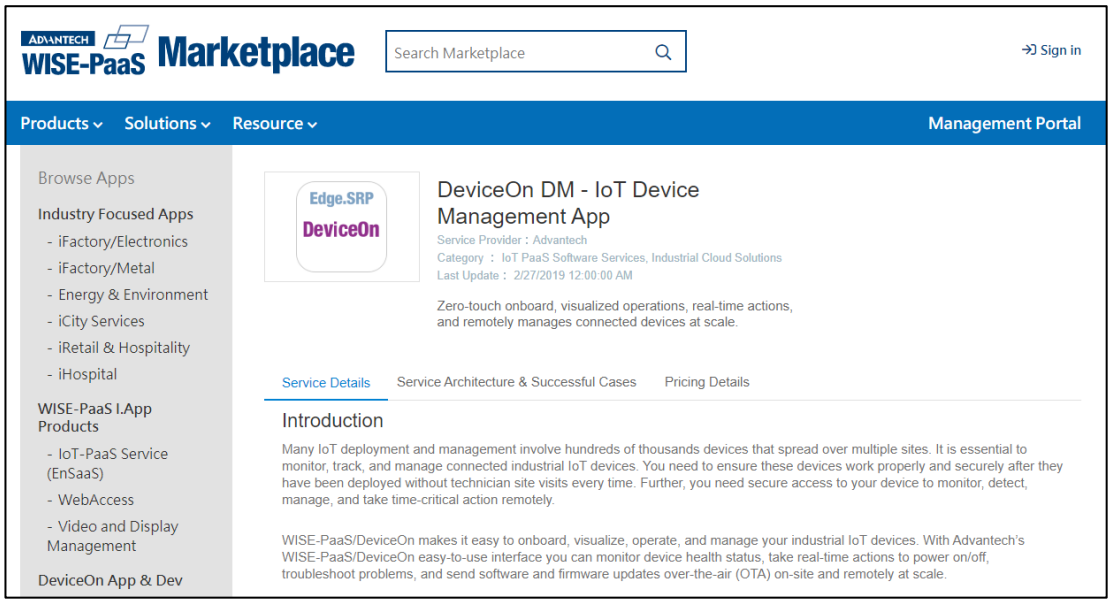
With Advantech's WISE-DeviceOn, users can swiftly utilize onboard devices, efficiently monitor device health status, and securely send software and firmware updates over-the-air (OTA) on-site and remotely at scale.

Advantech's brand-new designed IoT device operations and management App solution gives users a transformational plug-and-play experience. Beginning with onboarding devices, **WISE-DeviceOn**'s zero-touch IoT tech seamlessly registers Advantech hardware systems with identity security and field site settings. A fast and simple setup helps provide instant intelligent edge onboarding, data acquisition, and status visualization at the device operations center. Power on/off,

troubleshooting, and mission-critical actions are available at the tap of a button for quick and easy access. OTA software updates itself securely by sending software patch, firmware, software, and configuration updates through batch provisioning. The App is designed to ensure maximum efficiency in IoT device operations and management.

Power up your IoT devices with this hardware and software integrated solution. Get the most out of the WISE-DeviceOn's features with predictive device maintenance like IPC HDD lifecycle prediction, analytics-based dashboard and automated event alerts. In bringing artificial intelligence to your IoT needs, Advantech delivers improved risk management, faster daily operations, and better device performance while improving business value and intelligence through the extraction of big data.

WISE-DeviceOn is compatible with all Advantech hardware systems and works on popular platforms and services like the WISE-PaaS public/private cloud, Microsoft Azure, VM on-premise, and Kubernetes. **Get your WISE-DeviceOn version on the WISE-PaaS Marketplace** and kick-start your new and improved device operations and management experience.



1.1 Feature Highlights

- **Device Connectivity & Monitoring**

  With more and more IoT devices in the field and the need for remote management and monitoring of those devices, the most important thing is how to achieve secure and fast onboarding to WISE-DeviceOn. There are two mechanisms provided, one is **Zero-touch**, where the user does not need to configure any of their devices. Just power-on the devices and they will connect to DeviceOn automatically. However, there is the limitation that the device's network must have the ability to directly connect to the public cloud. The second mechanism is called "**One-time configuration, automated onboarding**". Based on this mechanism, the user only sets up one device to connect to the cloud and uses this device to search and bring others to the cloud. Furthermore, this scenario supports public/private provisioning if there is no public cloud connection due to environmental limitations.

  DeviceOn supports general real-time monitoring of device health that includes hard disk, CPU, memory, network load and provides various alerting mechanisms. Additional proprietary sensors such as CO2, battery monitoring or various proprietary protocols can be supported through design-in services.

- **Bulk Management & Maintenance**

  For management and real-time control of a group of devices, DeviceOn offers a default overview with one-click actions, such as "One-Click Power On", "One-Click Protection", "One-Click Recovery", "One-Click Turn off backlight" and so on. Operators do not need to spend lots

of effort to setup devices one by one, but can simply “One-Click” maintain their field devices. The following actions are supported by DeviceOn:

- Power Saving
  - Power On/Off, Reboot
  - *Backlight On/Off
- Security
  - Protection On/Off
  - System Backup/Recovery
  - **USB Lock/Unlock
    Block USB drives and removable disks (*Not supported on “Administrator” user)*
  - **Keyboard Lock/Unlock
    Block function key, such as “ALT”, “CTRL”, and windows key.
  - **Touch Gesture Lock/Unlock (*supported with capacitive touch panel only)*
  - **Touch Lock/Unlock
- System
  - Screenshot
  - Audio Mute/Unmute
  - *Watchdog Enable/Disable (*Default reset time is 60s)*
    Reboots the system if it becomes unresponsive, to avoid hanging at “BSoD” (Blue Screen of Death) or similar situations
  - **Notification Block/Unblock
    Disable windows notification from applications and other sources
  - **UWF Enable/Disable
    Helps to protect your drives by intercepting and redirecting any writes to the drive (app installation, settings changes, saved data) to a virtual overlay

Above actions prefixed with ‘*’ require the respective Advantech SUSI Driver and actions prefixed with ‘**’ require following operating systems:

- **Windows 10 Enterprise LTSC 2019 (LTSC)**
- **Windows 10 Enterprise 2016 LTSB (LTSB)**

- **Device Remote Control**
  - **Device Diagnostics**

    Provides remote control mechanism, such as KVM (Remote Keyboard-Video-Mouse) for real-time remote desktop access to the devices. The screenshot functionality allows to capture the device’s current screen output for potential troubleshooting. Another feature is access to Windows or Linux shells, for example in order to quickly retrieve

network status via ipconfig/ifconfig, netstat to dump socket/TCP/UDP information, without having to use the full graphical user interface.

- **OTA (Over the Air)**

  OTA supports an open framework, which can easily integrate 3rd party storage, such as FTP and cloud solutions (Azure Blob, AWS S3, AliYun, Openstack Swift). It does not only support remote update and deployment, but supports automatic update from server side as well as scheduled updates that get triggered from the agent side. Scheduling helps to avoid peak network traffic times and allows implementation of download and deployment schemes that reduce potential impact to a minimum.

  The framework supports upgrade package backups as well as rollback to the previous version when required.

  Scripting support (shell/batch) allows to implement flexible update mechanisms.

- **Power Management**

  Sets the power on/off schedule for remotely located devices; the schedule can be set on a daily, weekly, monthly, or yearly basis. Supports Agent mode enable powering on across networks.

- **Protection Management**

  DeviceOn system protection is powered by McAfee, providing white list protection against unauthorized application execution, and also sending warnings of any unauthorized activities.

- **Backup & Recovery**

  DeviceOn system recovery is powered by Acronis, providing hot backup and scheduled backup, and also one-click recovery.

- **Simplified Operation & Support**

  In general, the utmost goal of system integrators or IoT device operation managers is meeting service level KPIs without having to spend huge efforts or daily maintenance. Once hardware fails, it results in a serious increase in operation cost. DeviceOn provides rule-based management and implements HDD failure prediction. If a managed device shows any anomaly on a specific component or sensor, DeviceOn can send alert messages through **email** or **SMS,**

or can optionally integrate with social media services such as **LINE**, **WeChat**. The DeviceOn overview shows overall status, upcoming schedule, top 5 potential risk devices as well as device location at a glance.

There is a summary for these feature highlights on different operation system and hardware requirement.

	DeviceOn Feature Highlight	Windows 7, 8, 10	Windows 10 LTSC, LTSB	Ubuntu 16.04 x64	Linux on RISC (Yocto)	Android on RISC
Standard Offering	Role-Based Access Control	●	●	●	●	●
	Device Zero-touch Onboarding	●	●	●	●	●
	Device & Device Group Management	●	●	●	●	●
	Device Threshold Detection (Rule-based Engine)	●	●	●	●	●
	Notification & Alert Service (Mail, SMS, LINE, WeChat, WhatsApp, Telegram, Teams, Slack)	●	●	●	●	●
	Device Real-time & Historical Data Monitoring	●	●	●	●	●
	OTA, Software, Firmware Provisioning	●	●	●	●	●
	Power Control, Terminal, Screenshot, Remote Desktop	●	●	●	◐	◐
	Backup/Recovery, Protection	●	●	●		
	Device Data with Zero-Downtime	●	●	●	●	●
	Operation Management (Batch Control & Statistical Analysis)	●	●	●	●	●
	Audio Volume Control	●	●			
	1-Click to Data Visualization	●	●	●	●	●
	Statics System Report	●	●	●	●	●
	Intel AMT Remote Control and Management	●	●	●		
	Device Map (Open street, Google, Baidu)	●	●	●	●	●

Advantech Hardware Support	Hardware Watchdog Monitoring	◉	●	◉		
	Hardware GPIO Control & Customized	●	●	●	●	●
	Brightness & Backlight Control	●	●	●	●	●
	Hardware Sensor Monitoring	●	●	●	◐	◐
	BIOS Update	●	●	●		
	Advantech Industrial SQ Flash/RAM Remote Management & Monitoring	●	●			
Windows 10 Lockdown Features	USB Drive Block		●			
	Keyboard Lock & Filter		●			
	Touch Screen & Gesture Lock		●			
	Windows Notification Block		●			
	UWF Protection		●			

1.2 DeviceOn Server Versions

DeviceOn is based on a microservice design, each component is stateless and supports multiple instances for scale up. This results in heavily simplified deployment to WISE-PaaS (Cloud Foundry), Azure PaaS, standalone virtual machines or Kubernetes. Both public cloud and private cloud (on-premise) deployments are supported. This chapter provides an introduction and provides a summary of requirements for those scenarios. The container version of DeviceOn starts from version number **v-1.1.x** (WISE-PaaS/Azure Kubernetes), while the standalone version starts from **v-4.2.x.** The standalone version comprises of IoTHub, database (PostgreSQL and MongoDB), Dashboard (Grafana), Webservices (Tomcat) and DeviceOn core applications.

1.2.1 Standalone, VM (Cloud)

The standalone version provides all packages of the DeviceOn software in one installer package, including RabbitMQ as a message broker, MongoDB, PostgreSQL as databases, Grafana for visualization, Tomcat for web services, and a watchdog service that protects DeviceOn core components from crashing or becoming unresponsive.

This section specifies the minimum hardware requirements for DeviceOn Cloud (Standalone) and the operating systems on which DeviceOn is supported. In general, the better the hardware configuration

of your computer, the better your experience with DeviceOn will be. To achieve a more satisfying experience with DeviceOn, particularly in terms of the client software, it is highly recommended that your system be substantially better than the minimum requirements specified in the following sections. This is particularly true if running server software locally on the same system as the client software.

Attention to the following areas can make a significant improvement to your overall user experience and enjoyment of the software:

- Memory - the more RAM your computer has, the better.
- CPU speed - the faster, the better.
- Hard Drive - the larger, the better.

General Operation Systems and Recommendations:

- ✓ **Windows Server 2008 R2 64-bit (KB2999226 Required)**
- ✓ **Windows Server 2012 R2 Standard 64-bit (KB2919442, KB2919355, KB2999226 Required)**
- ✓ **Windows Server 2012 R2 Datacenter 64-bit (KB2999226 Required)**
- ✓ **Windows Server 2016/2019 64-bits**

Reserve Port for DeviceOn Server Used

	Name & Description	Inbound Port
1	DeviceOn HTTP, HTTPs Web Services	80, 443 [Depends on Installation]
2	DeviceOn Dashboard (Grafana)	3000 [Depends on Installation]
3	Message Broker (RabbitMQ) MQTT, MQTTs	1883, 8883
4	Message Broker (RabbitMQ) AMQP, AMQPs	5671, 5672
5	Message Broker (RabbitMQ) Management Console	15672
6	Repeater for Remote Desktop	5501
7	Websockify for Remote Desktop	6083 ~ 6183
8	Database for MongoDB	27017
9	Database for PostgreSQL	5432

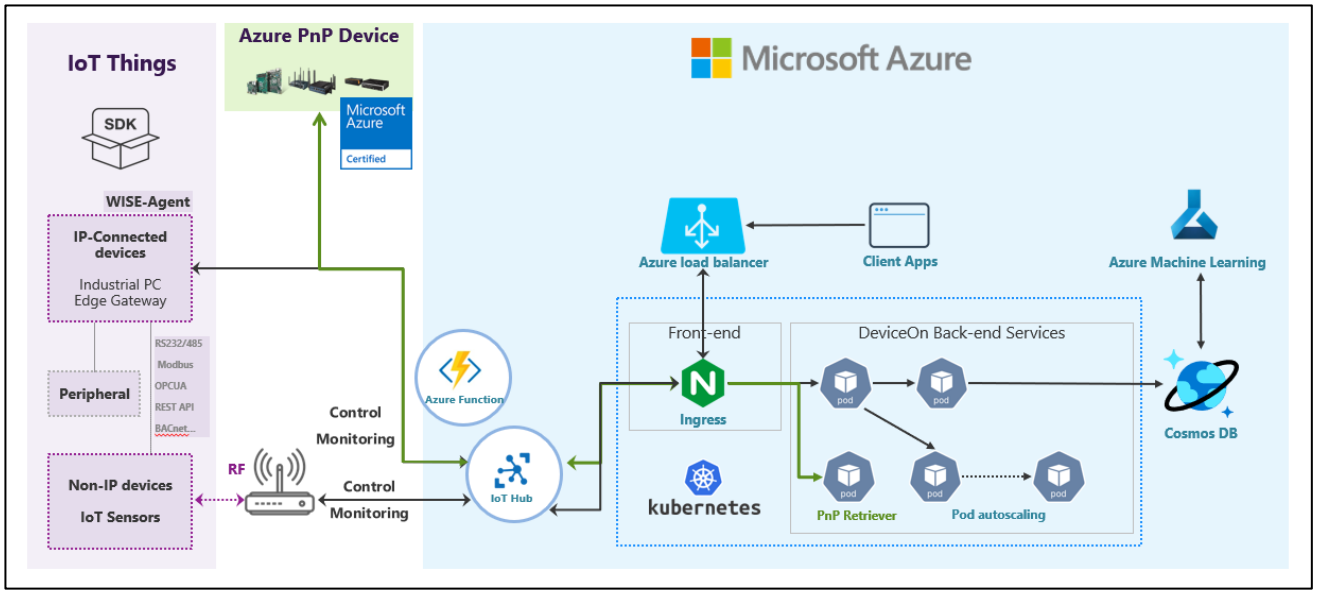
Hardware Minimum Requirements:

- ✓ **Intel® Core™ i5 2.3 GHz CPU and at least 8GB of RAM**
- ✓ **25 GB root partition for the system**
- ✓ **100 GB data storage partition (for documents and indexing)**

1.2.2 Azure PaaS, Kubernetes (Cloud)

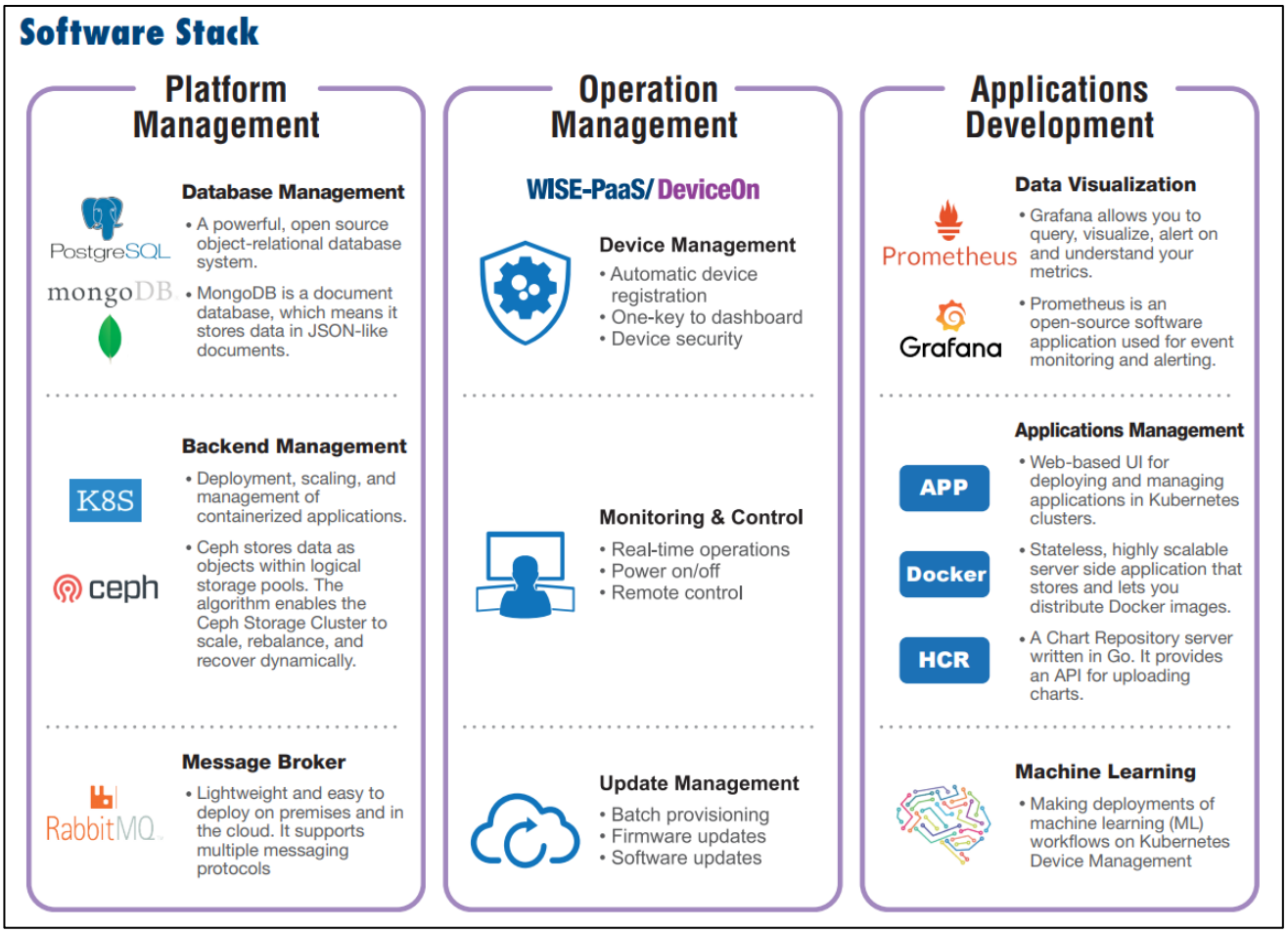
The Azure Kubernetes Service (AKS) makes it easy to deploy a managed Kubernetes cluster to Azure. AKS reduces the complexity and operational overhead of managing Kubernetes by offloading much of that responsibility to Azure. Azure handles critical tasks like health monitoring and maintenance for those Kubernetes services.

Deploying DeviceOn on the Azure Kubernetes Service is easy and with just a few steps, containers or nodes can be scaled up to manage thousands of devices. Moreover, DeviceOn can leverage the Azure IoTHub and Cosmos DB for Azure native security and performance. Since the data is already stored on the Azure cloud, it is much easier to leverage the Azure ecosystem – for example using the provided data for Azure Machine Learning. DeviceOn can be deployed to Azure Kubernetes directly from the WISE-PaaS/Marketplace.



1.2.3 Data Service Server for Private Cloud

For accelerated IoT application deployment, Advantech offers the Data Service Server EIS-S230 as a stable and reliable all-in-one solution for your back-end data service or light private cloud. It is built around an Intel Xeon or Core i7 CPU to offer best in class computing performance for data services. Moreover, EIS-S230 comes preinstalled with Kubernetes to support micro-services, as well as complete back-end software components including RabbitMQ as IoT Hub, MongoDB and PostgreSQL as database, Grafana for data visualization and Prometheus for back-end management. EIS-S230 also provides a dynamic scale out function that allows extension of resources as necessary. It is a perfect tool to create IoT applications more easily and flexibly and to speed up time to market.



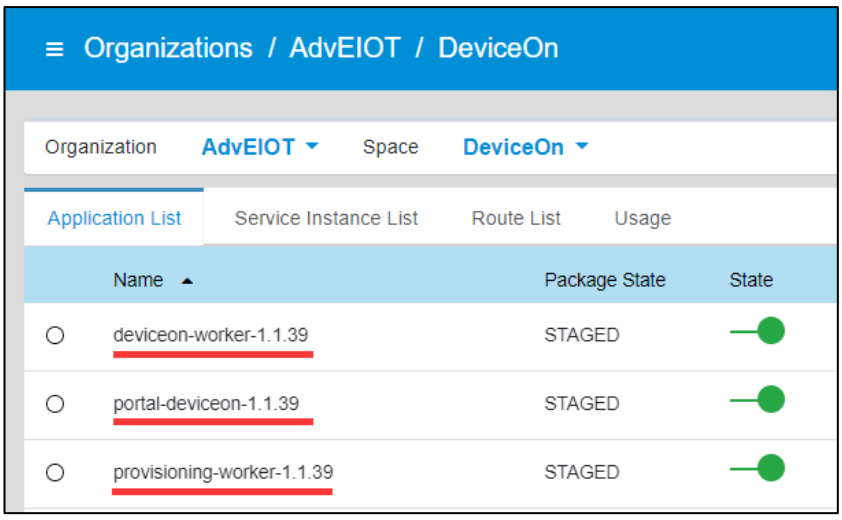
Features:

- Integrated solution (HW+SW bundle) for back-end data service and light private cloud
- Pre-configured system: Intel Xeon platform with 32GB RAM, 512GB mSATA SSD including Ubuntu Linux OS
- Open and flexible infrastructure: Kubernetes support, multiple database options, on-demand microservices
- Integrated IoT Software: Private Cloud Deployment, Platform Management, Application Integration
- Integrated Applications: WISE-DeviceOn, Grafana, Prometheus, Kubeapps, Kubernetes Dashboard
- Sustainable Management: Condition Monitor, Load Balance, Advanced Recovery
- **WISE-DeviceOn inside for feature-rich IoT Device Management**

1.2.4 WISE-PaaS/EnSaaS (Cloud)

The WISE-PaaS/EnSaaS version consists of three containers as listed below. In this scenario DeviceOn requires 1408 MB of RAM at least.

Application Name	Version	Memory Used	Purpose
deviceon-worker-1.1.x	v-1.1.x	384MB	Worker that processes device messages, status, notification, scheduling etc.
portal-deviceon-1.1.x	v-1.1.x	768MB	Provides the DeviceOn web interface for remote control and monitoring.
provisioning-worker-1.1.x	v-1.1.x	256MB	Worker that provisions devices with configuration, software, firmware etc.



1.3 DeviceOn Agent Versions

Advantech provides a device client that is used to communicate and exchange information between IoT (Internet of Things) devices and the DeviceOn cloud services, called **WISE-Agent**. WISE-Agent provides a rich set of user-friendly features that are intelligent, standardized and scalable.

- Standardization

  The communication protocol between client and cloud is based on the industry standard MQTT protocol. The IoT sensor data format is following the IPSO Alliance definition, implemented in JSON.

- Portability

  The whole framework is written in C language and follows the ANSI C Standard. C compilers are widely available for most platforms and allow easy porting to different architectures or operating systems.

- Scalability

  The WISE-Agent has a modular design and provides a plugin concept that allows flexible addition of new data sources or extra functionality.

1.3.1 WISE-Agent (Client)

WISE-Agent is support on different platforms running Windows 7 (or newer) or Ubuntu 16.04 x64 (or newer). Please contact us for others architectures (e.g. RISC) or operating systems (e.g. Yocto based Linux/Android).

General Operation Systems and Recommendations:

- ✓ **Windows 7 SP1/8/10 32-bit/64-bit**
- ✓ **Ubuntu 16.04, 18.04, 20.04 x64**
- ✓ **CentOS 7.6, 8.2 x64**
- ✓ **Other Linux flavours (e.g. Yocto) on x86 or RISC (on a per project basis)**
- ✓ **Android on RISC (on a per project basis)**

Assigned Ports for Device Communication

	Name & Description	Outbound Port
1	MQTT, MQTTs Message Client	1883, 8883
2	Remote Desktop VNC Client	5501

Hardware Minimum Requirements:

- ✓ **Intel® Celeron™ 1.10 GHz CPU and at least 2GB of RAM**
- ✓ **500 MB root partition for the system**
- ✓ **Advantech HW with respective SUSI driver 3.02/4.0 support is required for the HWM (Hardware Monitoring Management) feature to be available**

1.4 Security

1.4.1 Role-Based Access Control (RBAC)

DeviceOn supports three different user roles - “Root” (perpetual version only), “System Admin” and “Device Admin”. There is only one single “Root” account per system, which has the highest permission level and can create “System Admin” or “Device Admin” accounts. The intermediate user level “System Admin” can be used to create “Device Admin” accounts. “Device Admin” accounts have the lowest permission level. Please refer to Section 7.1 for details on access permission levels.

1.4.2 SSL Encryption

- **HTTPS on DeviceOn Web Server**

  The principal motivations for HTTPS are authentication of the accessed website, protection of the privacy and integrity of the exchanged data while in transit. It protects against man-in-the-middle attacks. The bidirectional encryption of communications between a client and server protects against eavesdropping and tampering of the communication.

- **SSL Connection on Database (PostgreSQL, MongoDB)**

  PostgreSQL and MongoDB have native support for using SSL connections to encrypt client/server communications for increased security.

- **Create Security Credentials on Database**

  Databases are by default protected by secure credentials and require explicit authentication for connections. This avoids accidentally deploying platforms with unprotected access.

- **Device Connectivity via MQTT SSL**

  RabbitMQ supports multiple protocols including MQTT, which the most popular IoT (Internet of Things) protocol. By default, SSL is used to encrypt all MQTT traffic for device connectivity.

- **Enforce Password Policies**

  While DeviceOn allows you to set some of your own passwords, please make sure those meet the minimum complexity requirements established by your specific organization.

1.4.3 Security Scan

The DeviceOn server pass through below famous vulnerability tools to ensure security for your AIoT solutions. Furthermore, all the testing including anti-malware (**Trend Micro** and **Kaspersky**)

- **Web Application Assessment Report (Micro Focus)**

  WebInspect is an automated dynamic testing tool that mimics real-world hacking techniques and attacks, and provides comprehensive dynamic analysis of complex web applications and services.

- **OpenVAS (Open Vulnerability Assessment System)**

  OpenVAS is a full-featured vulnerability scanner. Its capabilities include unauthenticated testing, authenticated testing, various high level and low-level Internet and industrial protocols, performance tuning for large-scale scans and a powerful internal programming language to implement any type of vulnerability test.

  The scanner is accompanied by a vulnerability tests feed with a long history and daily updates. This Greenbone Community Feed includes more than 50,000 vulnerability tests.

- **Nessus**

  Nessus is the de-facto industry standard vulnerability assessment solution for security practitioners. The latest intelligence, rapid updates, an easy-to-use interface.

  ✓ Covers an industry-leading 47,000+ vulnerabilities
  ✓ Unlimited scans at no extra cost
  ✓ Compliant with PCI, HIPPA, GLBA, CIS, NIST, and more

- **OWASP ZAP**

  The OWASP Zed Attack Proxy (ZAP) is one of the world's most popular free security tools and is actively maintained by hundreds of international volunteers*. It can help you automatically find security vulnerabilities in your web applications while you are developing and testing your applications. It's also a great tool for experienced pen testers to use for manual security testing.

- **Skipfish**

  Skipfish is an active web application security reconnaissance tool. It prepares an interactive sitemap for the targeted site by carrying out a recursive crawl and dictionary-based probes. The resulting map is then annotated with the output from a number of active (but hopefully non-disruptive) security checks. The final report generated by the tool is meant to serve as a foundation for professional web application security assessments.

  Key features:

  ✓ High speed: pure C code, highly optimized HTTP handling, minimal CPU footprint – easily achieving 2000 requests per second with responsive targets.
  ✓ Ease of use: heuristics to support a variety of quirky web frameworks and mixed-technology sites, with automatic learning capabilities, on-the-fly wordlist creation, and form auto completion.
  ✓ Cutting-edge security logic: high quality, low false positive, differential security checks, capable of spotting a range of subtle flaws, including blind injection vectors.

- **Nikto**

  Nikto is an Open Source (GPL) web server scanner which performs comprehensive tests against web servers for multiple items, including over 6700 potentially dangerous files/programs, checks for outdated versions of over 1250 servers, and version specific problems on over 270 servers. It also checks for server configuration items such as the presence of multiple index files, HTTP server options, and will attempt to identify installed web servers and software. Scan items and plugins are frequently updated and can be automatically updated.

- **W3af**

w3af is a **Web Application Attack and Audit Framework**. The project's goal is to create a framework to help you secure your web applications by finding and exploiting all web application vulnerabilities.

- **Arachni**

  Arachni is a fully featured web security scanning tool, it is based on ruby framework. It is an open source, modular and high performance tool. It comes with both command line interface as well as web based gui interface, it is highly versatile tool for security scanning purpose. It supports almost all of the popular web application such as HTML5, Java Script and AJAX etc, additionally it is enables with multi user-multi platform collaboration. It allows you to generate reports in desired format (.txt, XML, HTML).

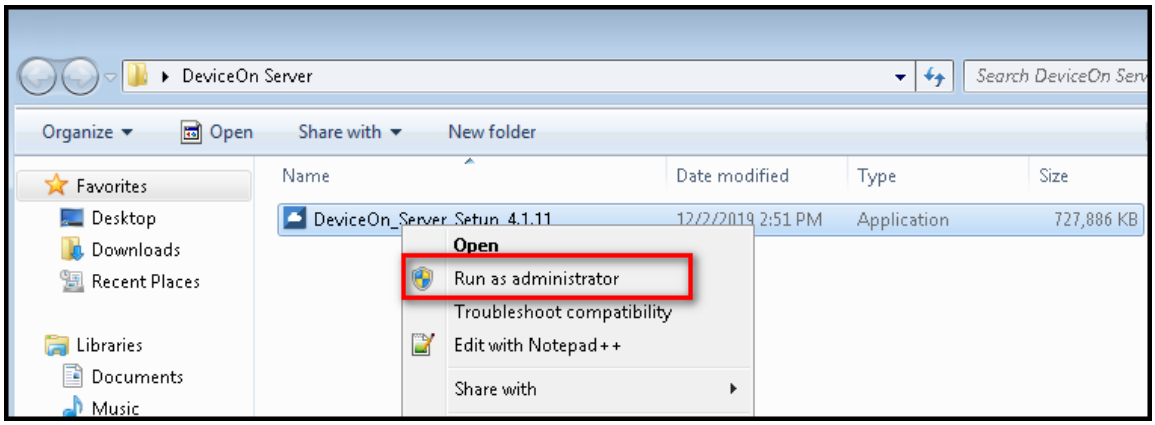
2. Getting Started

2.1 DeviceOn Cloud Installation

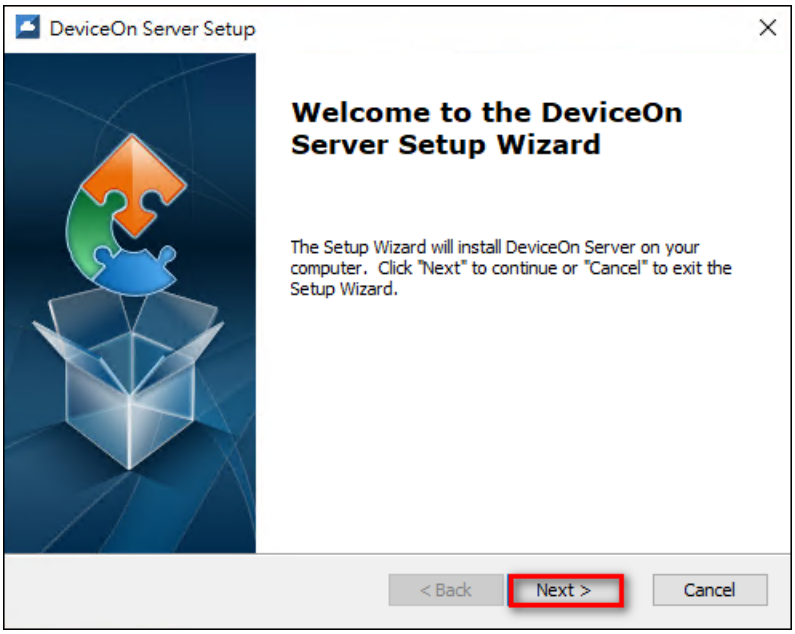
2.1.1 Setup Standalone Version (On-premise)

Step 1: Install the DeviceOn package on your system

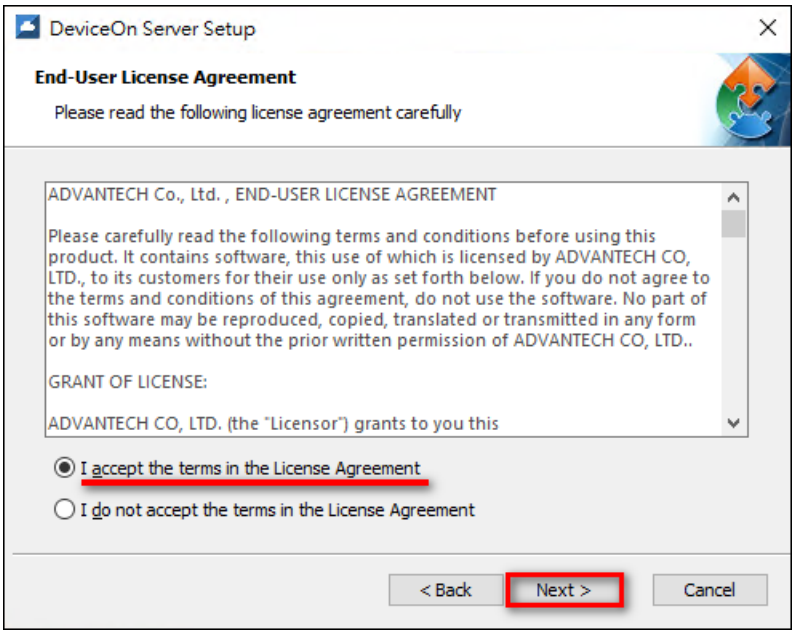
Copy the installation file (**DeviceOn_Server_Setup_4.2.x.exe**) to your target system and run it as administrator.



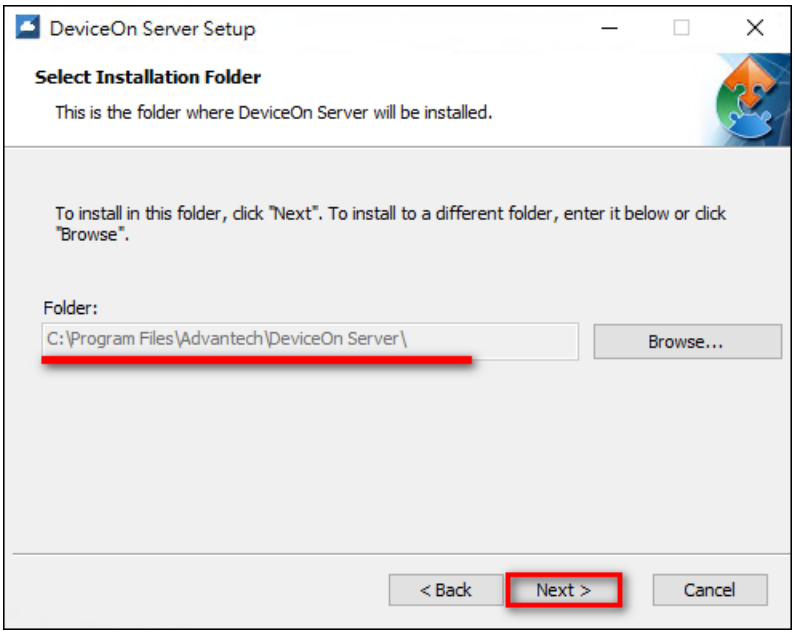
Click "**Next**" to start the installation process.



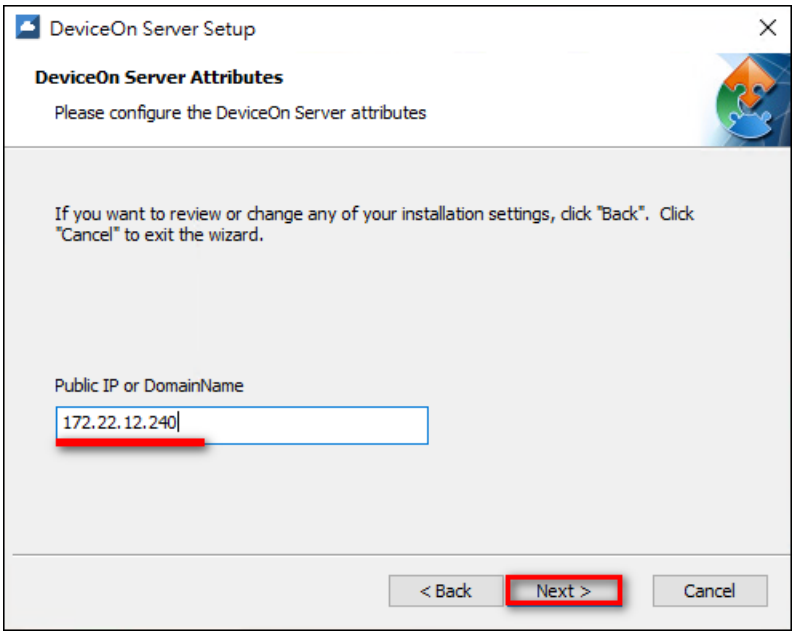
Select "**I Accept the terms in the License Agreement**" and click "**Next**"



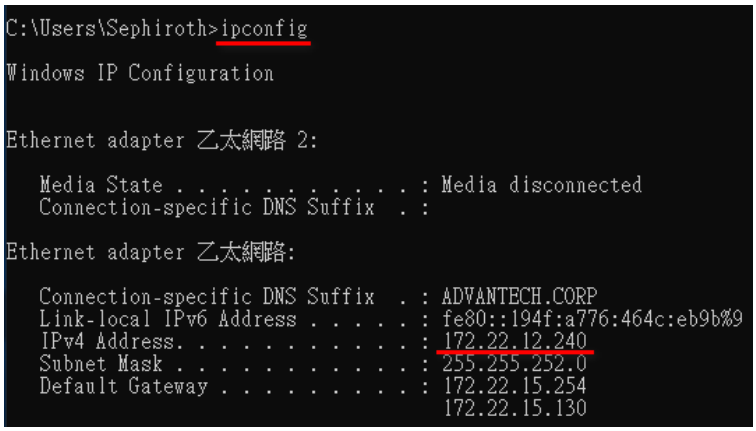
Select the "**Installation Folder**" for DeviceOn Server and click "**Next**"



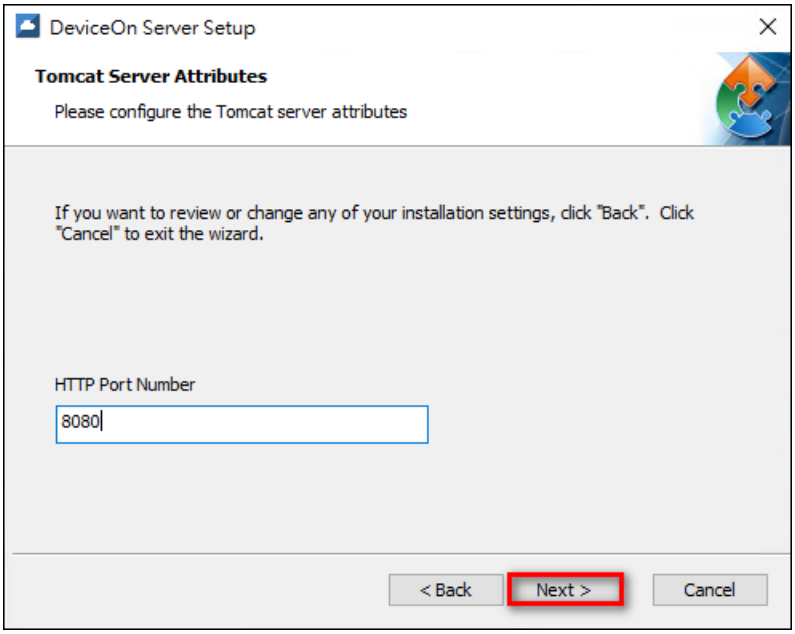
Enter "**Public IP**" or "**Domain Name**" for this physical/virtual machine and click "**Next**". This information is required for "Edge Device" connectivity, please make sure your device is reachable under this IP or Domain Name.



Note: You can start a Windows command prompt and type "ipconfig" to retrieve your IP address(es) on this physical/virtual machine.



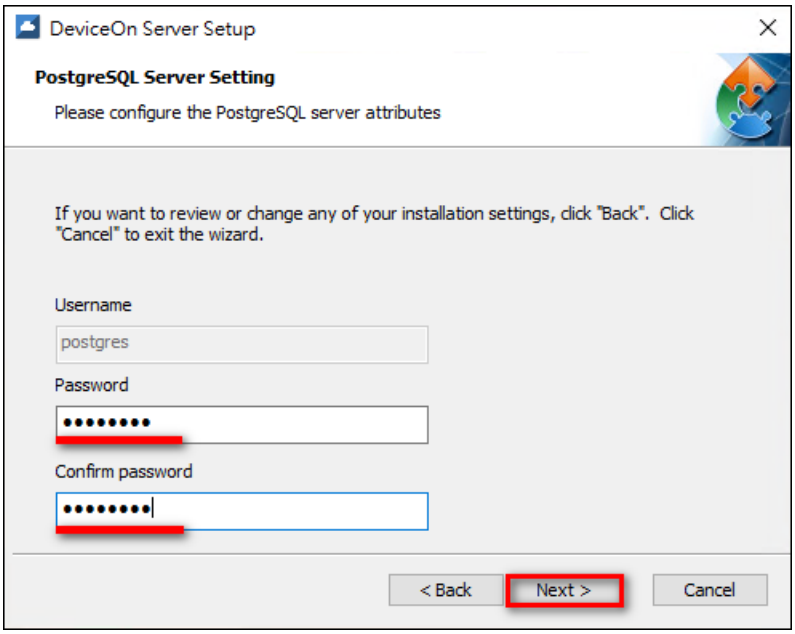
You will need to configure the HTTP port number that is used for web browser-based access the DeviceOn management portal. The default port is 8080, but you can select any other port as long as it does not conflict with any other application or service. Click "**Next**".



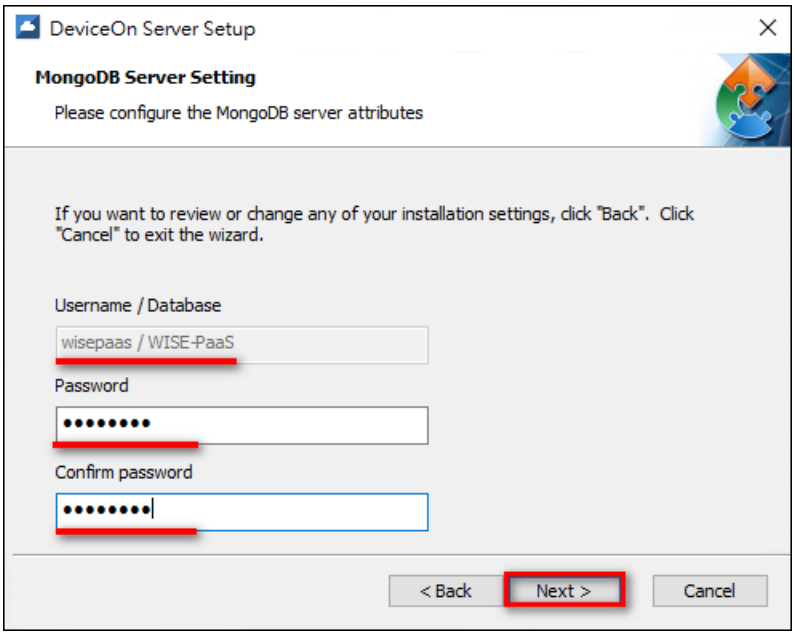
Configure the password of the relational database (PostgreSQL) that DeviceOn uses to manage account, device, permission, and relation data. The default account name is "**postgres**" and the password should follow below guideline.

Strong Password Rules:

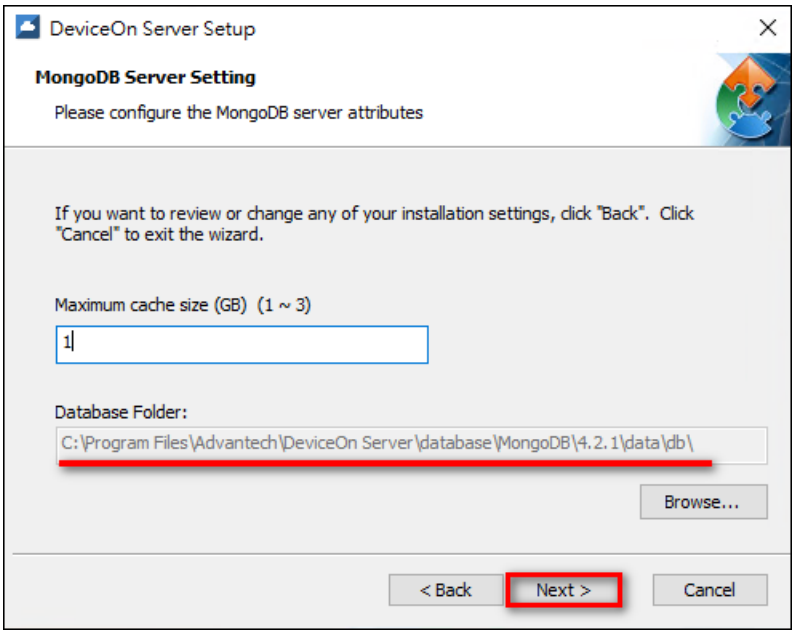
Minimum eight characters, at least one number, one lowercase letter, one uppercase letter, and one special character (Blank character, Backslash(\), Double quotes(") are prohibited)



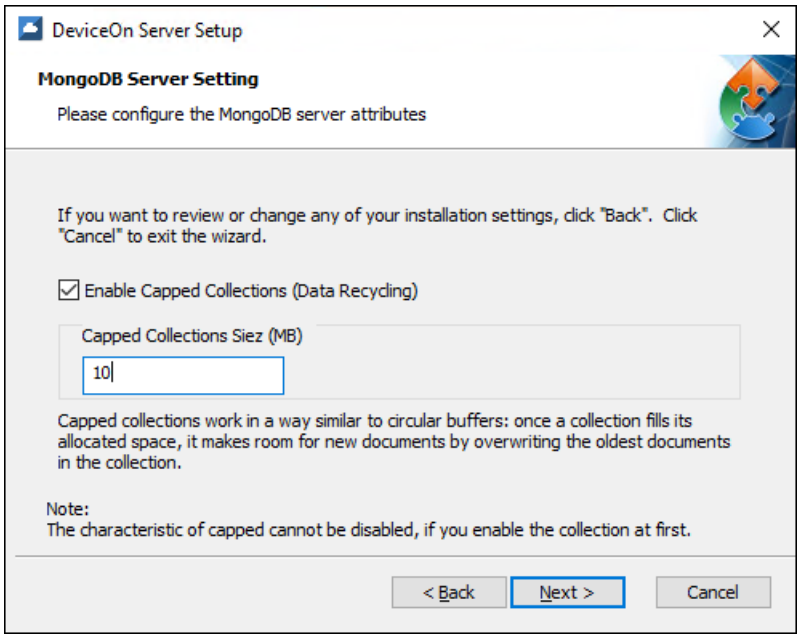
Configure the password of the NoSQL database (MongoDB) that stores device sensor data. The default account and database is “wisepaas/WISE-PaaS”. This password should also follow strong password rules as outlined above.



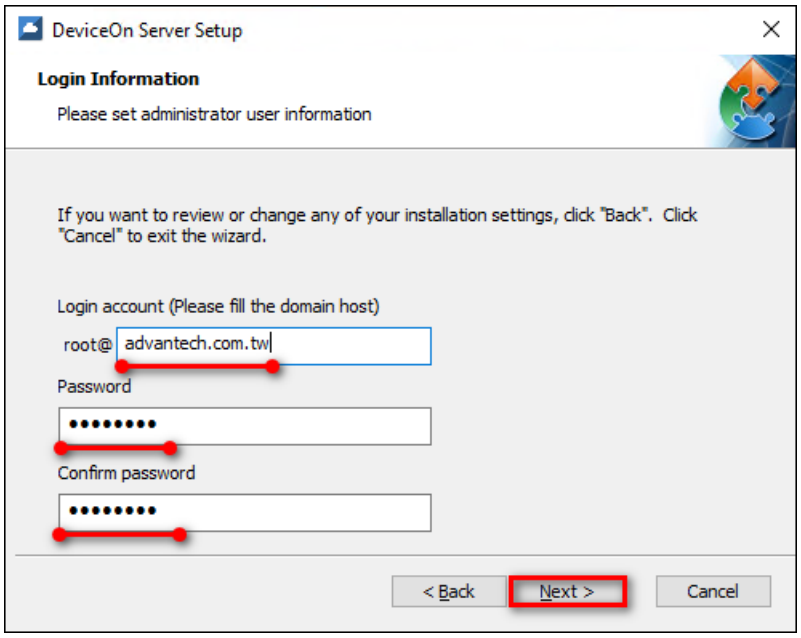
Select the database installation path and cache size of MongoDB and click “**Next**”. A larger cache size will result in better performance. For more information on this parameter, please referend to the official documentation.



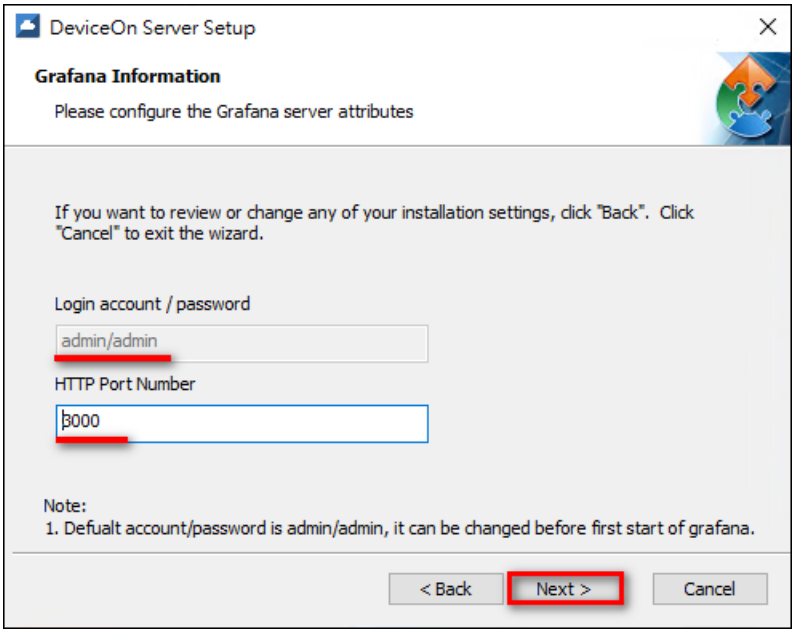
In order to avoid the hard disk space being used up by device data, the MongoDB provide a capped mechanisms, that similar to circular buffers: once a collection fills its allocated space, it makes room for new documents by overwriting the oldest documents in the collection. **Please note, the characteristic of capped cannot be disable, if you enable the collection at first.**



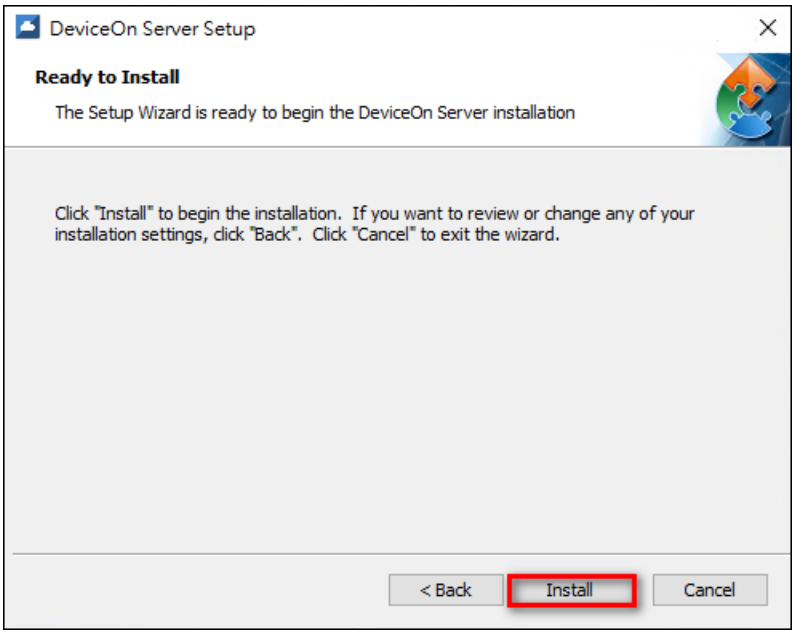
Configure the password and suffix domain of the root account (dummy name “root@”) and click “Next”. This root account has the highest permission level and is used to log in to the DeviceOn web service and create other user accounts.



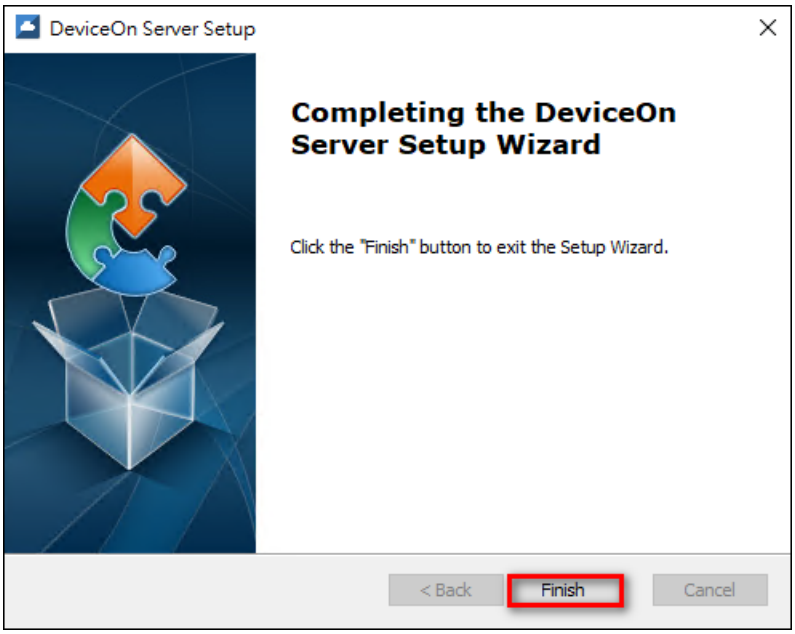
Set up the HTTP service port for Grafana dashboard. The default user name and password is admin/admin. You will be able to modify this at the first login.



Click "**Install**" to begin the installation.



Click “**Finish**” to exit the program.

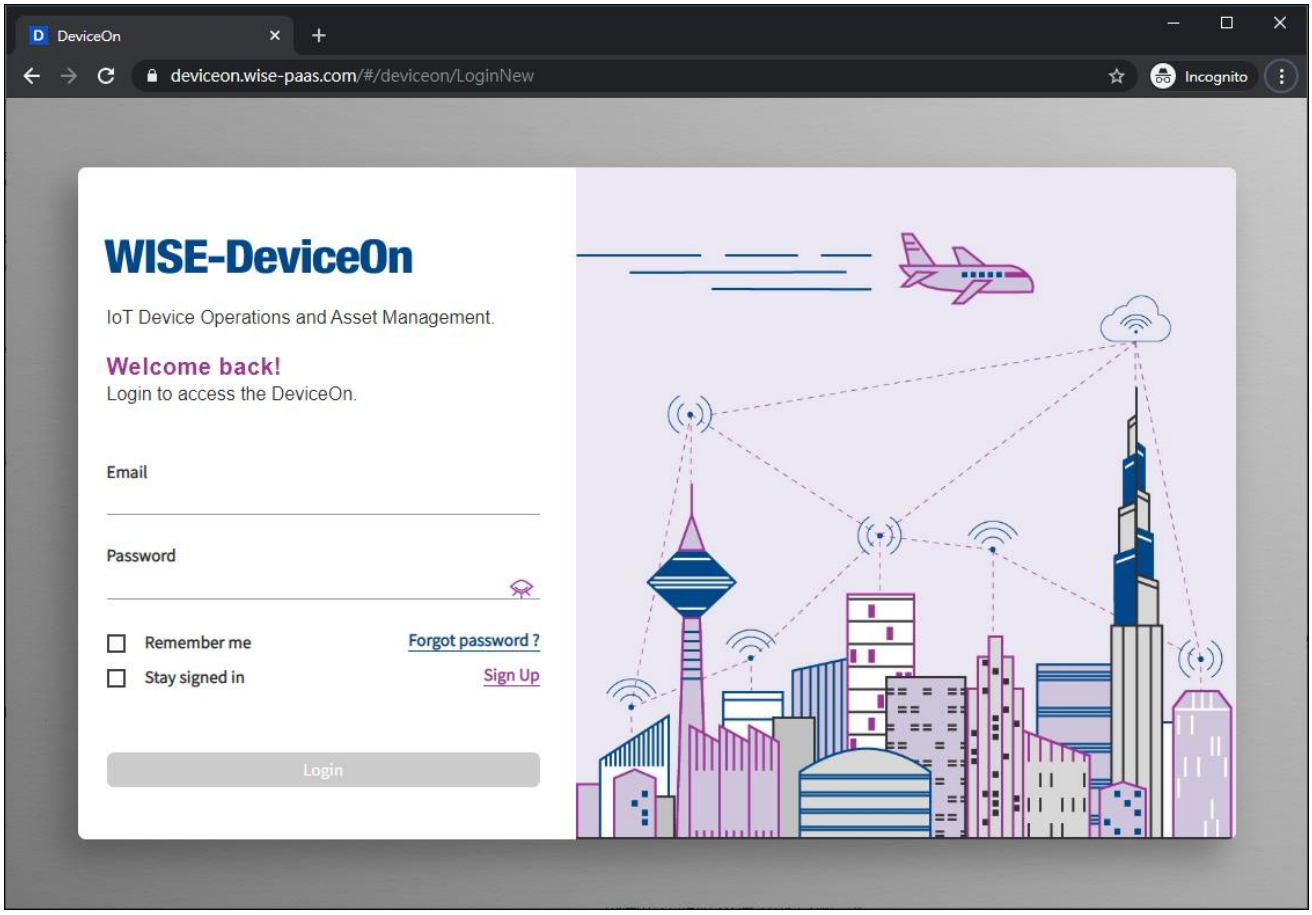


Step 2: Launch DeviceOn Web Service Shortcut on Desktop

Two shortcuts will be generated on the desktop - one is for the DeviceOn web portal and the other one is for the Grafana dashboard.

Click the “**DeviceOn Server**” shortcut in order to launch a browser and to start device operation and

management. It is recommended to use **Chrome** for the best user experience.



2.1.2 Setup Standalone Version for Ubuntu Linux (On-premise)

If you are interested in DeviceOn and used to Linux platform, On-Premise, we also provide an installer for Ubuntu Linux (one of the most popular Linux distribution). This section will guide you how to install DeviceOn on Ubuntu Linux.

Note here that:

- The DeviceOn Ubuntu Linux installer is named something like "**DeviceOn_Server_Ubuntu 18.04_x64_4.2.x.run**". To acquire the installer and ensure having the latest version, please contact us.
- If you are running the installer with an account other than "root", you should use "**sudo**" command to obtain higher privileges, or the installation may fail at any step.

Step 1: Open a terminal

The installer runs in CLI (Command Line Interface) mode. As such, open a terminal preferable for you.

Step 2: Copy the installer to target host

Use the way you like to copy the installer to the target host.

Step 3: Set the installer as executable

In the terminal, run "**chmod 0755 DeviceOn_Server_Ubuntu 18.04_x64_4.2.x.run**" so that the installer as an executable file under Ubuntu Linux.

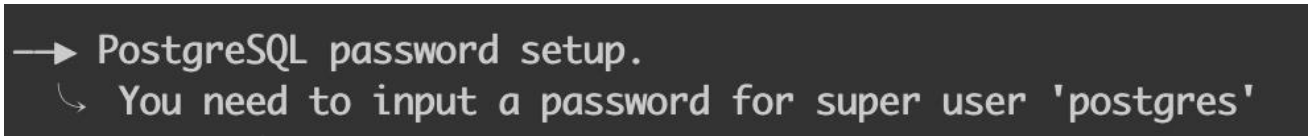
Step 4: Running the installer

Change your working directory to where the installer is and run "**./ DeviceOn_Server_Ubuntu 18.04_x64_4.2.x.run** ". You may need to run "**sudo ./ DeviceOn_Server_Ubuntu 18.04_x64_4.2.x.run** " to acquire higher privileges if you were logged in as a normal user.

Step 5: Answering some questions

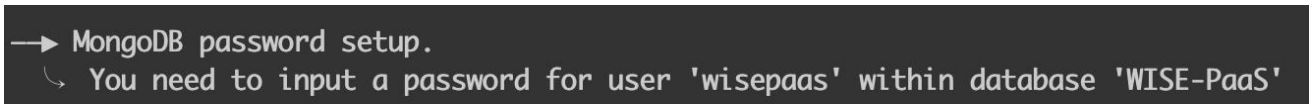
Throughout installation process, it's necessary to answer some questions to complete the installation:

A. The password of user "**postgres**" to login PostgreSQL database.



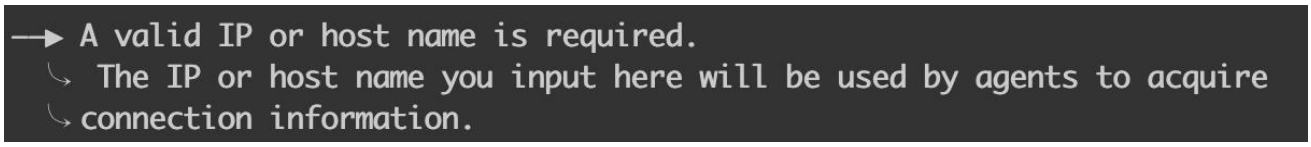
When you run into this step the question shows like above. Just input the password you would like to use to login PostgreSQL database for "**postgres**" account.

B. The password of user "**wisepaas**" to login MongoDB database.



When you run into this step the question shows like above. Just input the password you would like to use to login MongoDB database for "**wisepaas**" account.

C. The valid IP or host name of the target host.



When you run into this step the question shows like above. Just input the IP address of the target host. A hostname (even a FQDN) is also acceptable if you are sure that agents can connect to via the name you provide.

D. If turn MongoDB capped functionality on or not.

```
⟶ Turn 'capped' on or not.
  ↳ MongoDB has a feature named 'capped'. It will recycle disk size for those
  ↳collections turn this functionality on.
```

When you run into this step the question shows like above. Just input "**yes**" or "**no**" to enable or disable "capped" functionality. If you answer "yes", a subsequent question followed to ask you "how much capped size, in MB, to be used? ". Just input the size, in MB, you want to use in "capped" functionality in MongoDB database.

Capped collections are fixed-size collections that support high-throughput operations that insert and retrieve documents based on insertion order. Capped collections work in a way similar to circular buffers: once a collection fills its allocated space, it makes room for new documents by overwriting the oldest documents in the collection.

E. The password of user "**root@advantech.com.tw**" to login DeviceOn portal, and the rule should follow below guideline.

Strong Password Rules:

Minimum eight characters, at least one number, one lowercase letter, one uppercase letter, and one special character (Blank character, Backslash(\), Double quotes(") are prohibited)

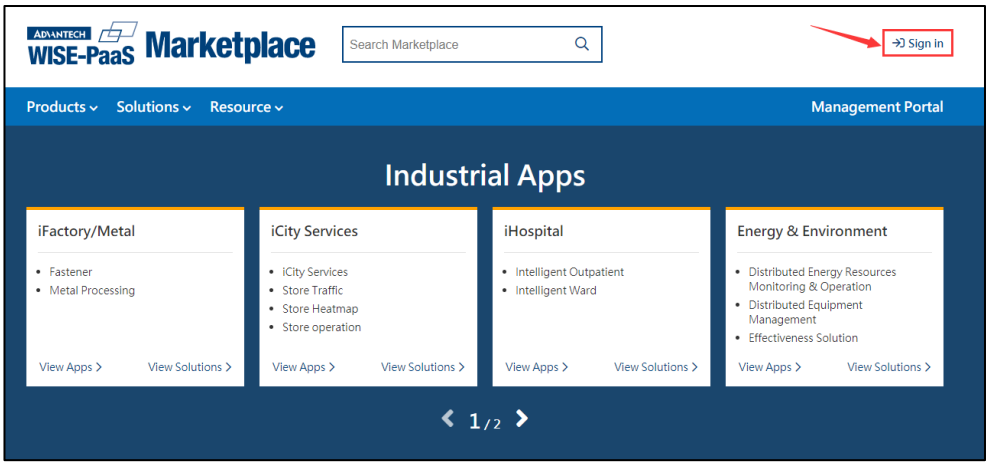
```
⟶ DeviceOn portal password setup.
  ↳ You need to input a password for super user 'root' to login DeviceOn portal
  ↳
  ↳ NOTE THAT A VALID PASSWORD TO LOGIN PORTAL MUST CONTAIN:
  ↳ 1) at least eight characters
  ↳ 2) at least a number
  ↳ 3) at least a lowercase letter
  ↳ 4) at least an uppercase letter
  ↳ 5) at least a special character but ' ', '\', and '"'.
```

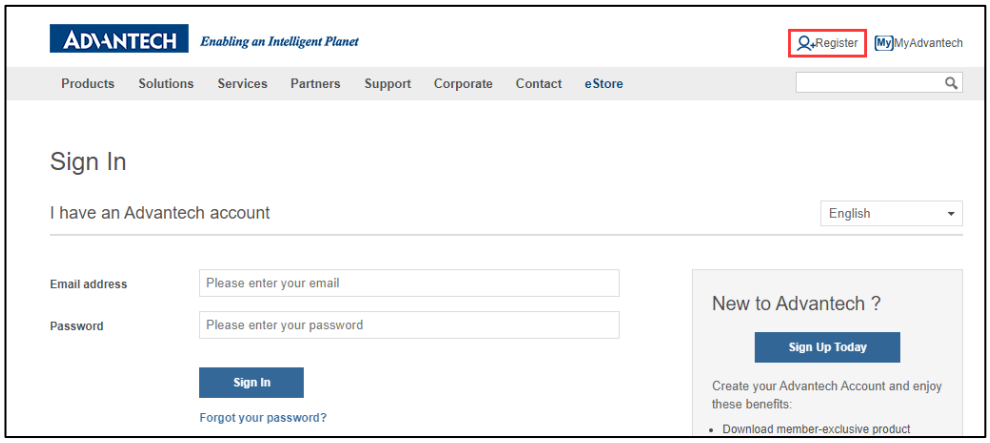
When you run into this step the question shows like above. Just input the password you would like to use to login DeviceOn portal for "**root@advantech.com.tw**" account.

Finally, a workable DeviceOn server should be there the target host. Open a browser and input http://{IP-USED-IN-QUESTION-C}, you should see the DeviceOn login page.

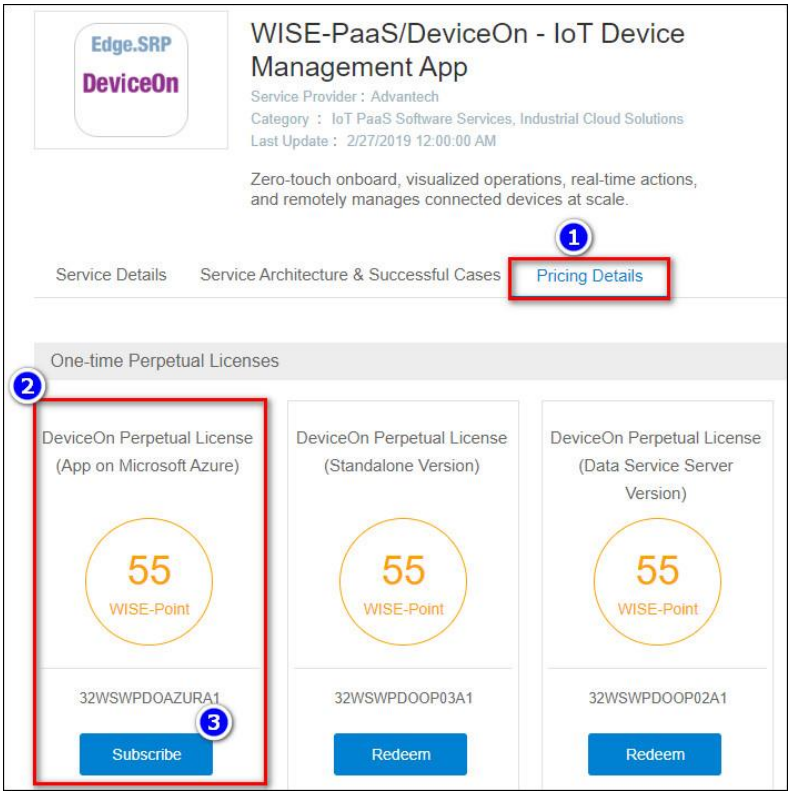
2.1.3 Redeem AKS Version from WISE-PaaS Marketplace

Step 1: Sign in to your MyAdvantech Account on Marketplace or create one

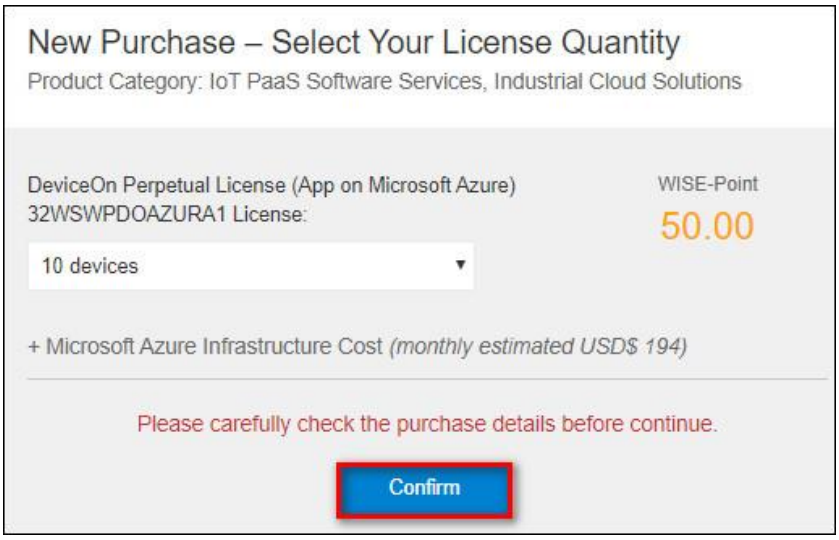




Step 2: Redeem "WISE-DeviceOn (App on Microsoft Azure)"

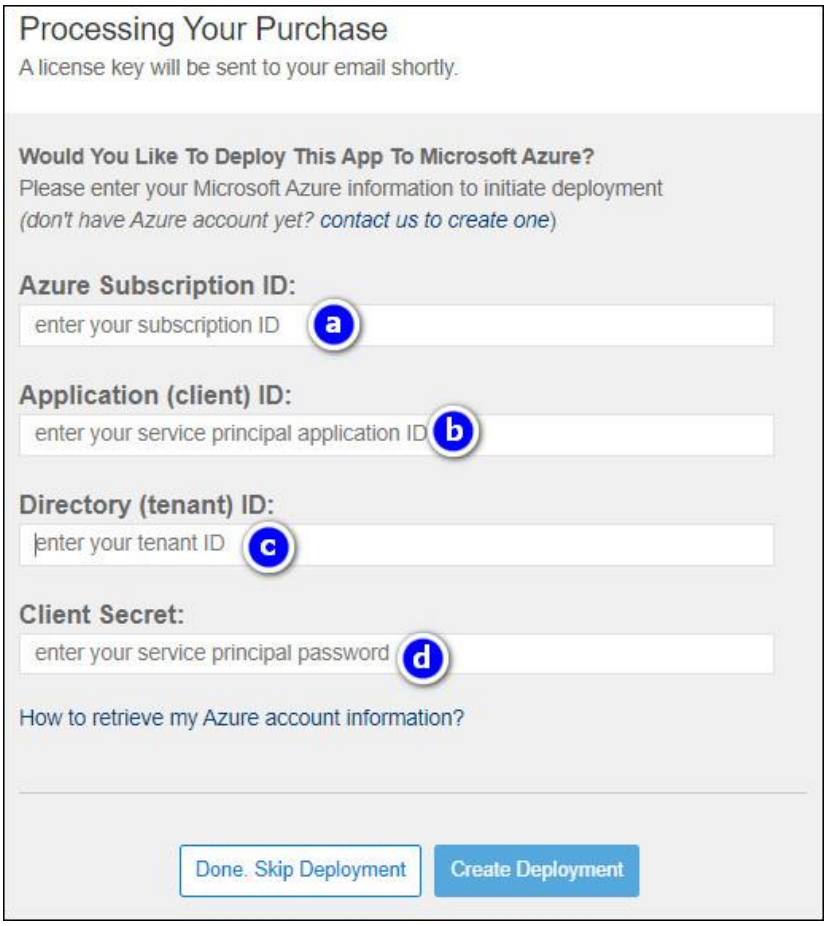


Select your license quantity and click “**Confirm**”.

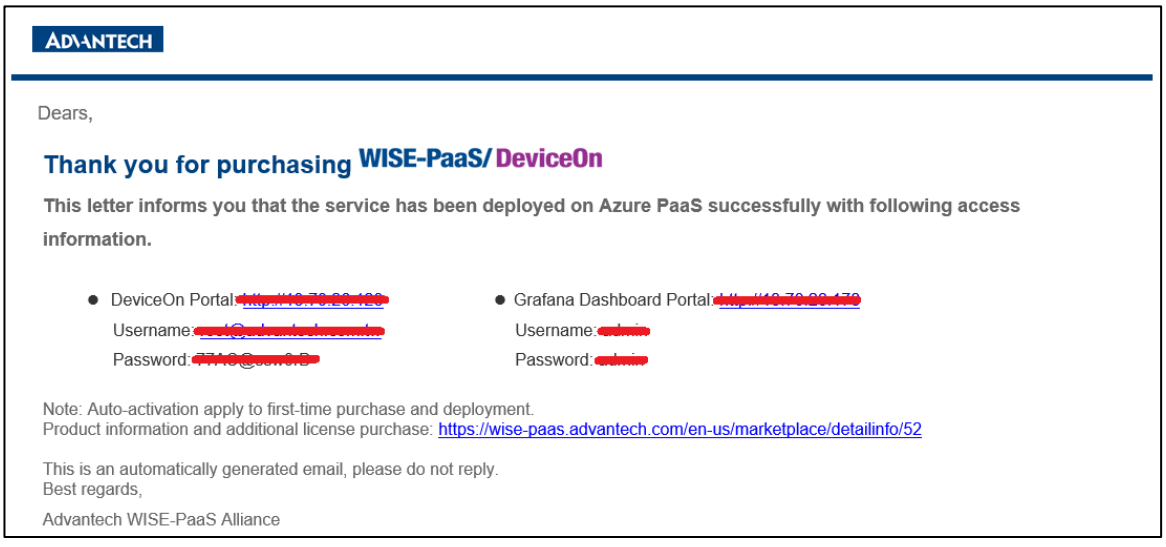


Confirm your license quantity and click “**Confirm Redeem**”.

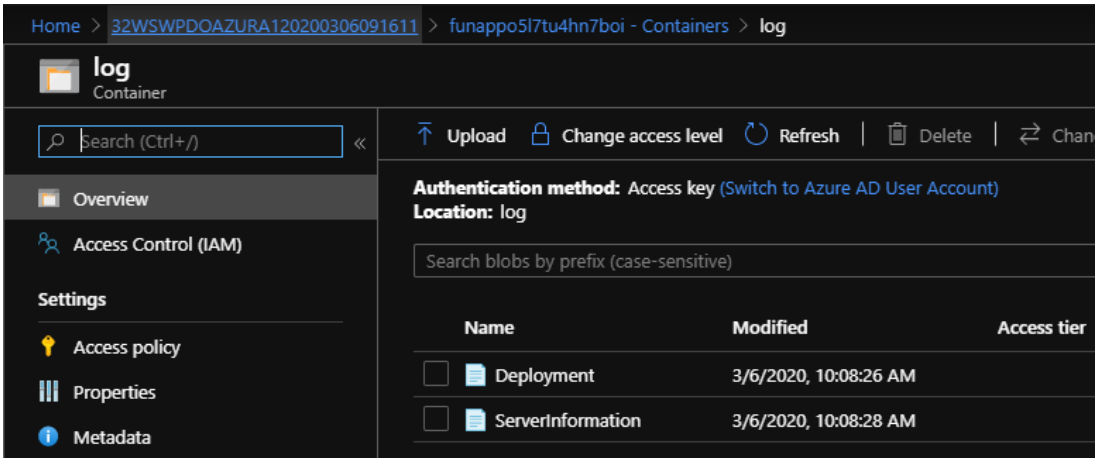
To deploy DeviceOn to your Azure subscription, a set of Azure account information is required. You will be prompted to enter required information on the WISE-PaaS/Marketplace when you choose “Create Deployment”. WISE-PaaS Marketplace will use the provided information to automatically deploy DeviceOn to your Azure subscription. There are two methods to retrieve those parameters for your Azure subscription, please reference Section 7.2.



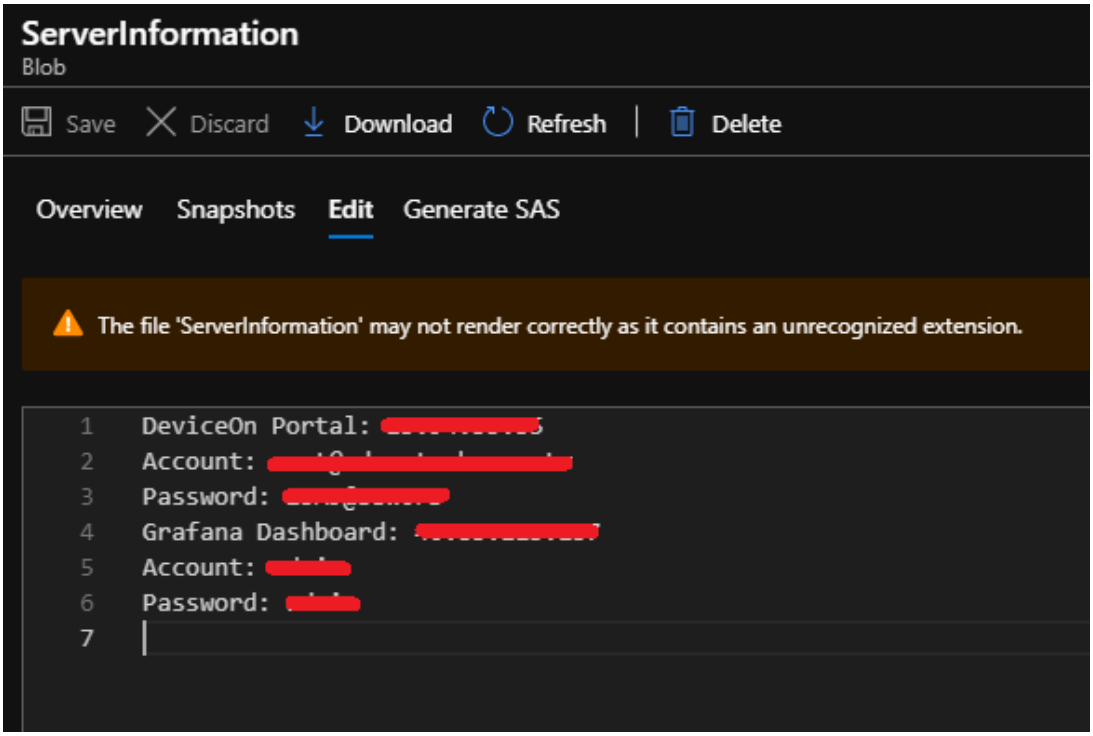
After deployment, you will receive a mail to get server information, including account, password and URL.



To prevent your mail blocked, we write the server information in Azure blob simultaneously.



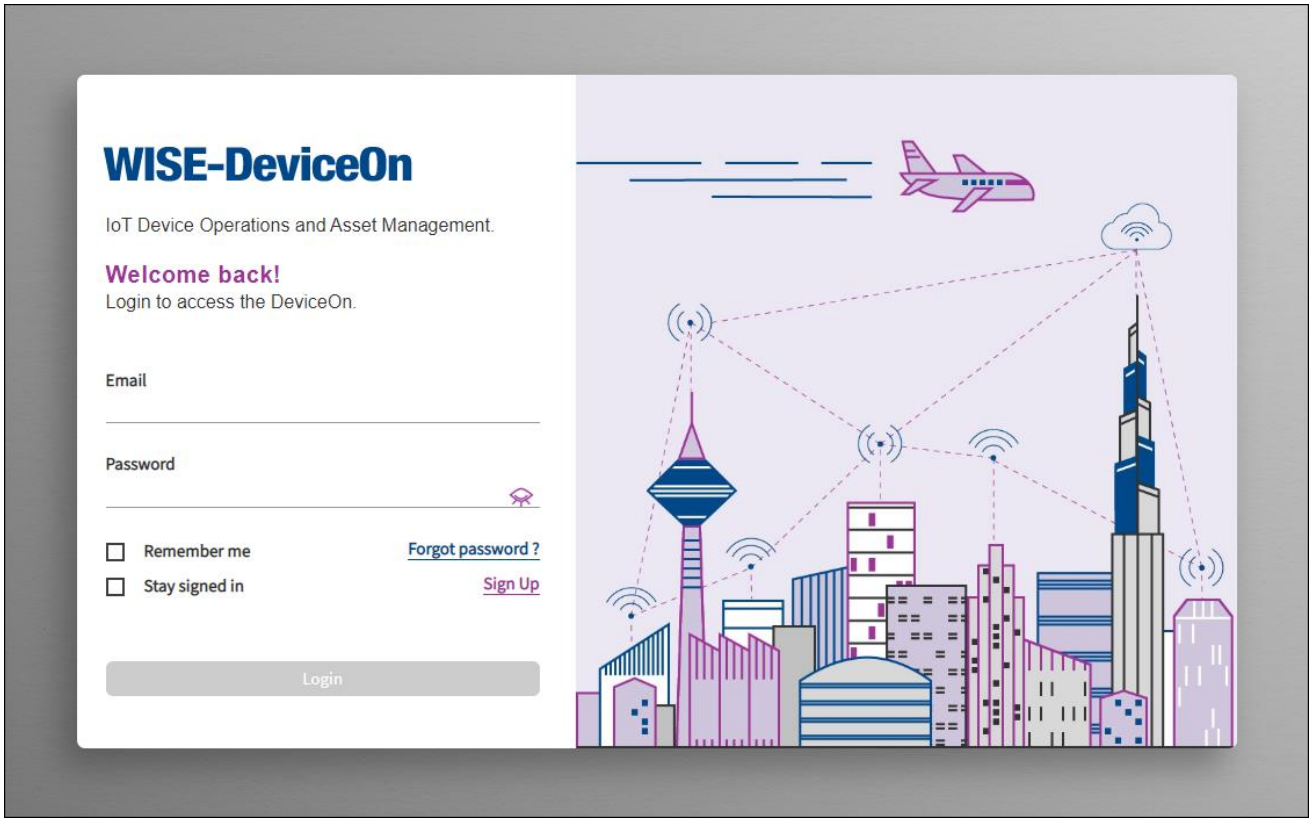
The credential and access information also on the “ServerInformation”.



2.2 DeviceOn Client Installation

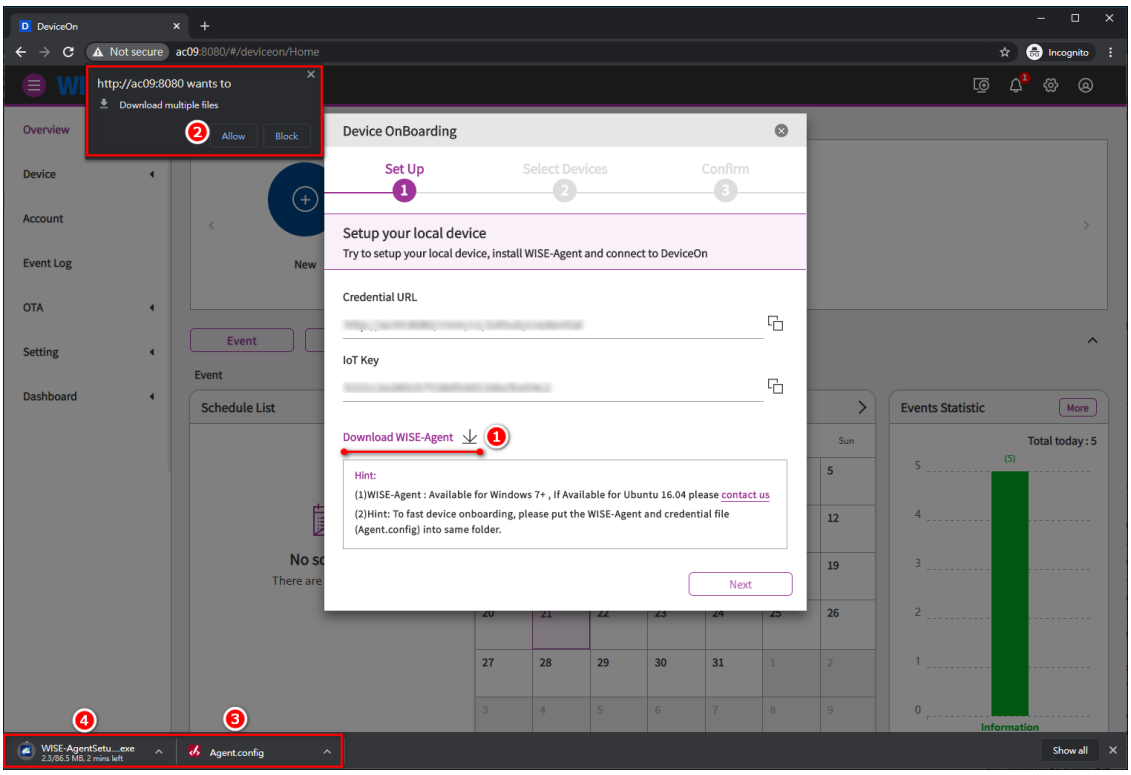
2.2.1 Setup Device Onboarding (Windows)

Step 1: Log in to the DeviceOn Cloud Service with Your Account and Password

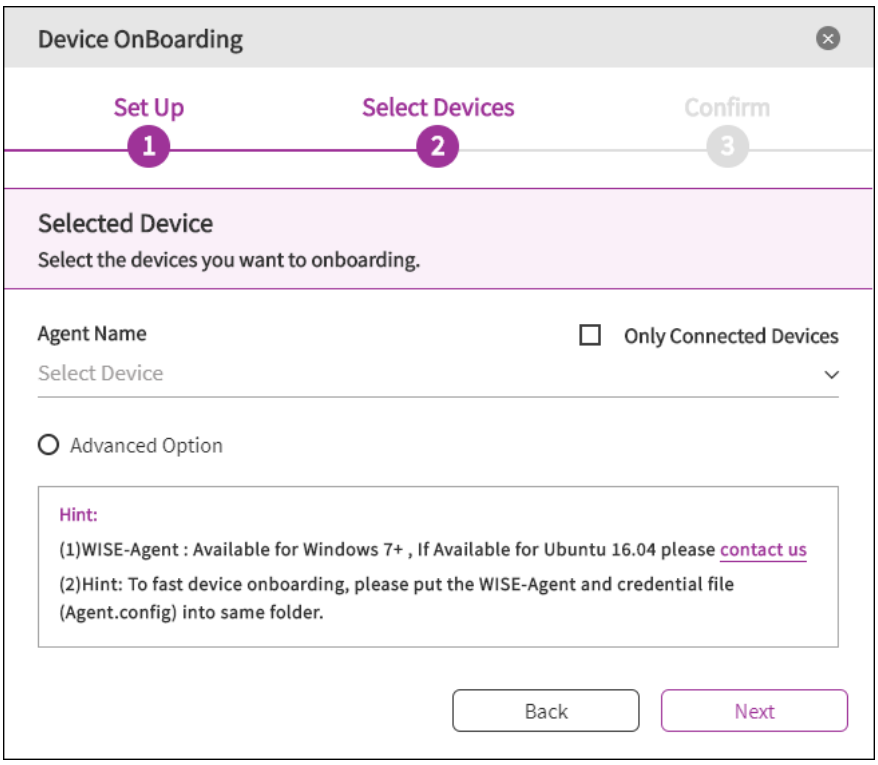


Step 2: Download WISE-Agent and Connection Configuration (Agent.config)

At the first login, the "Device Onboarding" dialog will pop up automatically. Please click "**Download**" to get the latest version of **WISE-AgentSetup.exe** and the respective connection configuration. (**Agent.config**)

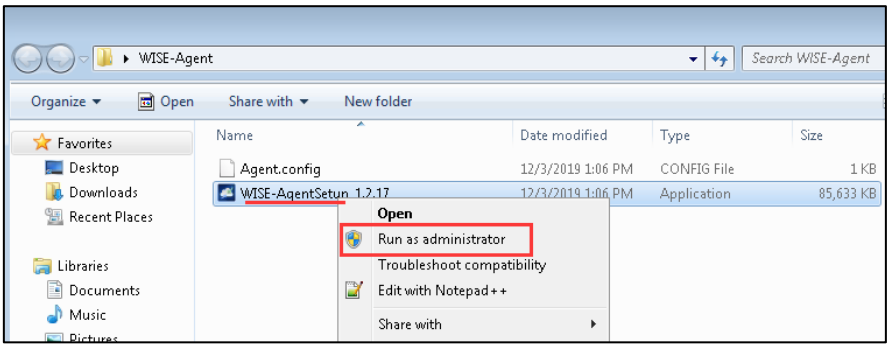


Click “**Next**” to wait for connecting devices.

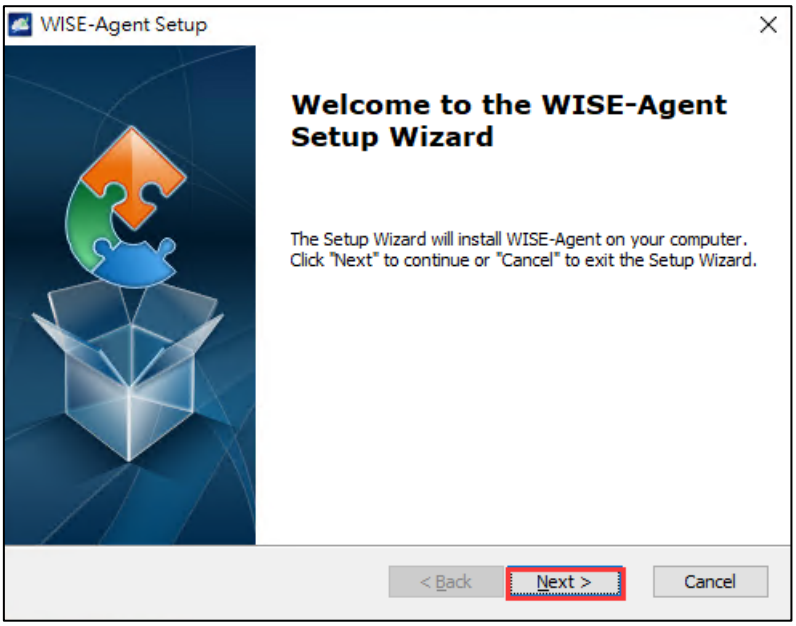


Step 3: Set up Your Local Device

Copy those two files (**WISE-AgentSetup_1.x.x.exe** and **Agent.config**) to the target device and launch “**WISE-AgentSetup_1.x.x.exe**” as administrator.



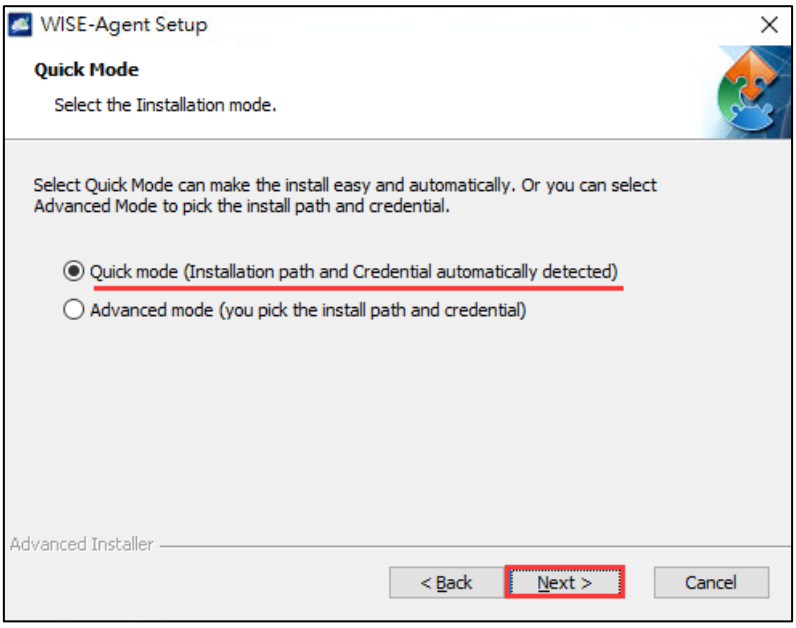
Click “**Next”** to set up the WISE-Agent program.



Select "**I Accept the terms in the License Agreement**" and click "**Next**"

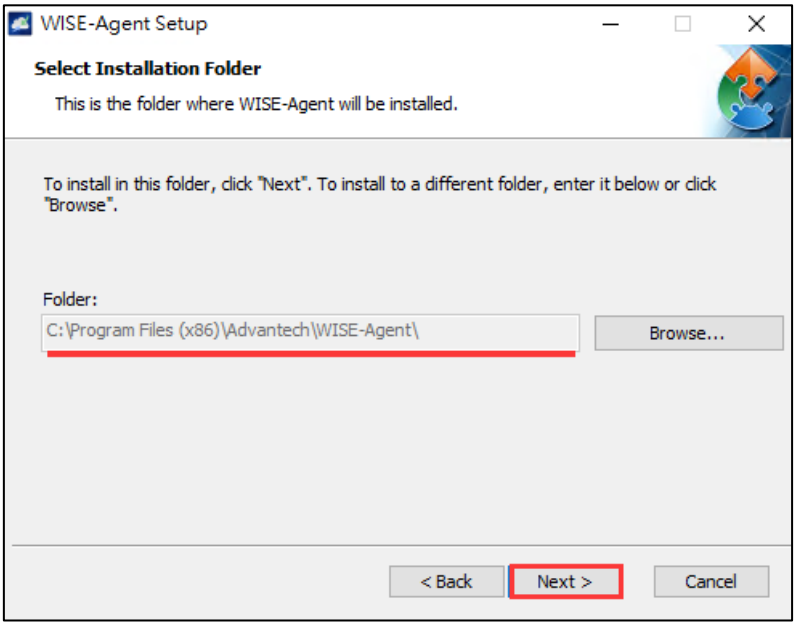
When the "**WISE-AgentSetup_1.x.x.exe**" program detects a cloud connection configuration file (**Agent.config**) in the same folder, "**Quick Mode**" as shown in this dialog will be available. For "**Quick Mode**", the installation path is fixed to "C:\Program Files (x86)\Advantech\WISE-Agent". If you would like to adjust the installation location, please select "**Advanced Mode**".

Quick Mode:



Advanced Mode:

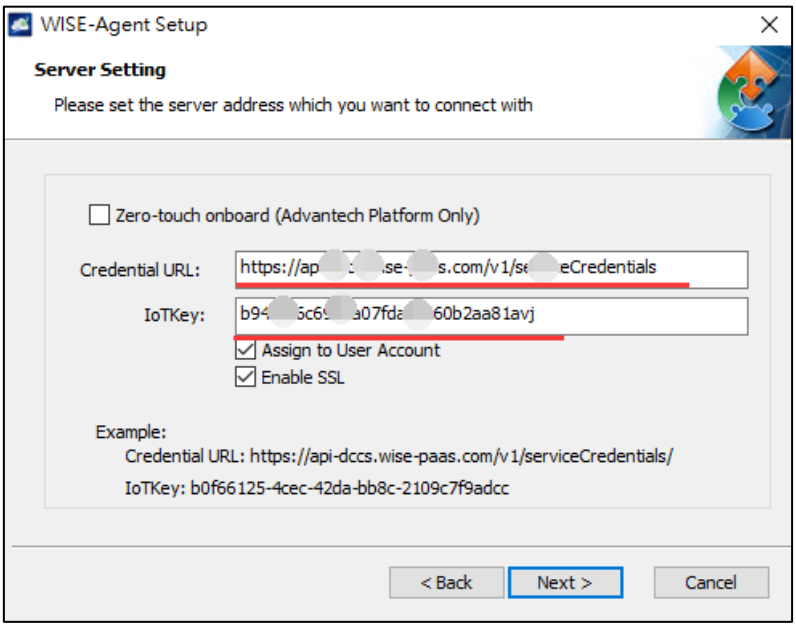
Select the Installation folder for WISE-Agent



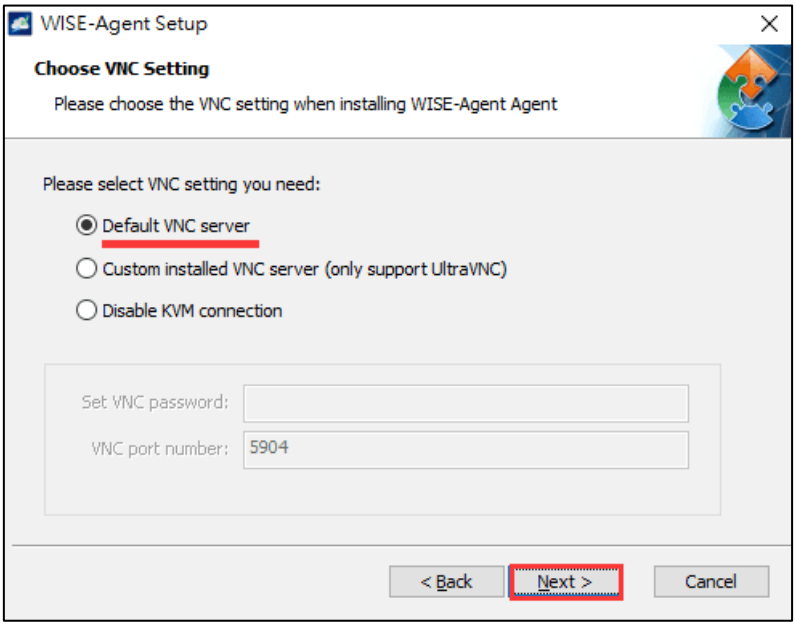
Set up the cloud connection configuration (**Credential URL** & **IoTKey**). This information can be retrieved from the DeviceOn web portal as shown in Step2, and click "**Next**".

- *"Zero-touch onboarding": Only supported on Advantech platforms with SUSI Driver and pre-configuration on the provisioning server*
- *"Assign to User Account": Each account has its own connection IoTKey. If checked, the device will be assigned to this account automatically.*

- *"Enable SSL": The communication between WISE-Agent and DeviceOn Cloud is MQTT. If checked (default setting), all the messages and content are SSL encrypted (MQTT SSL port: **8883**). Otherwise, port **1883** is used for MQTT without SSL.*

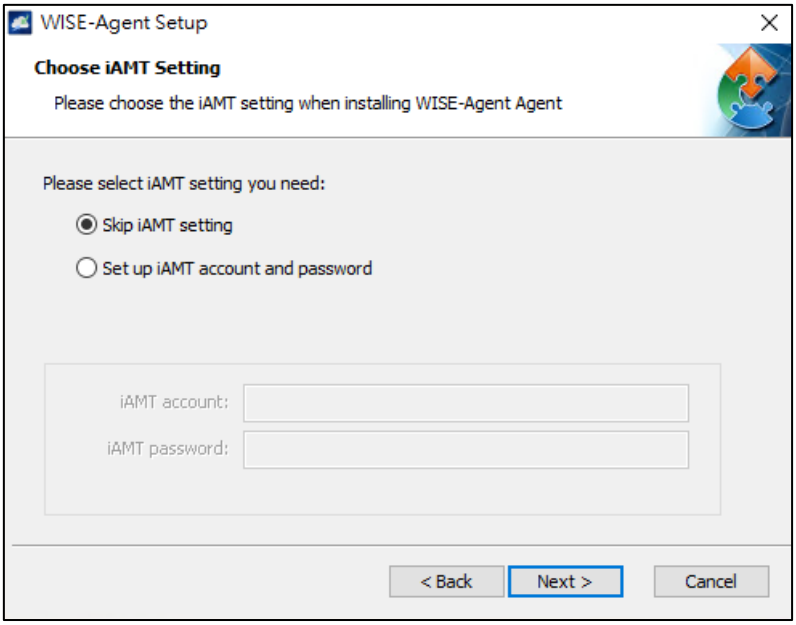


WISE-Agent supports remote desktop through built-in UltraVNC. You can manually specific the location of your own UltraVNC installation if preferred. If you do not want the remote desktop feature to be available, please select "Disable KVM Connection".

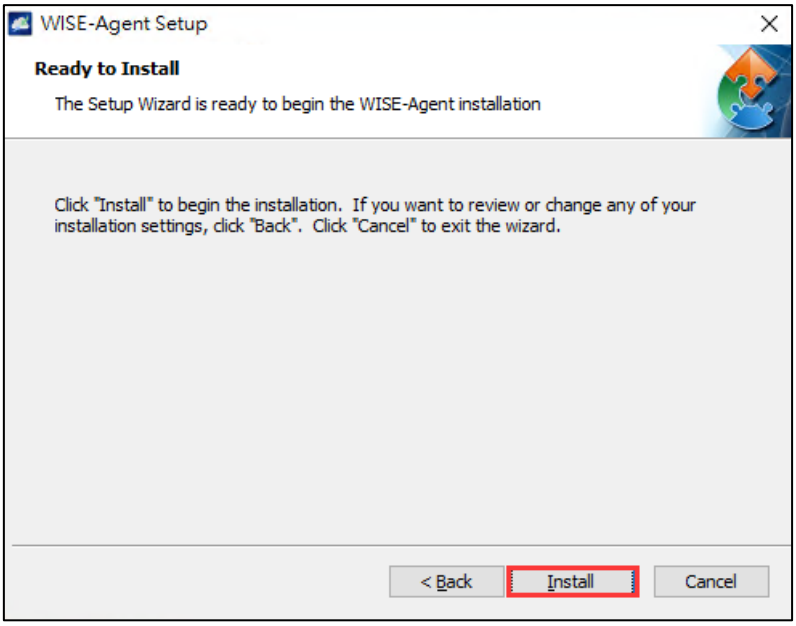


WISE-Agent integrates Intel AMT (Intel Active Management Technology) for remote power management (Power Up, Down, Cycle and Reset) as well as remote desktop access, even in case the operating system has crashed. However, this feature requires hardware support (Intel Core i5, i7) and the target device needs to be on the same local network as the DeviceOn server. Please

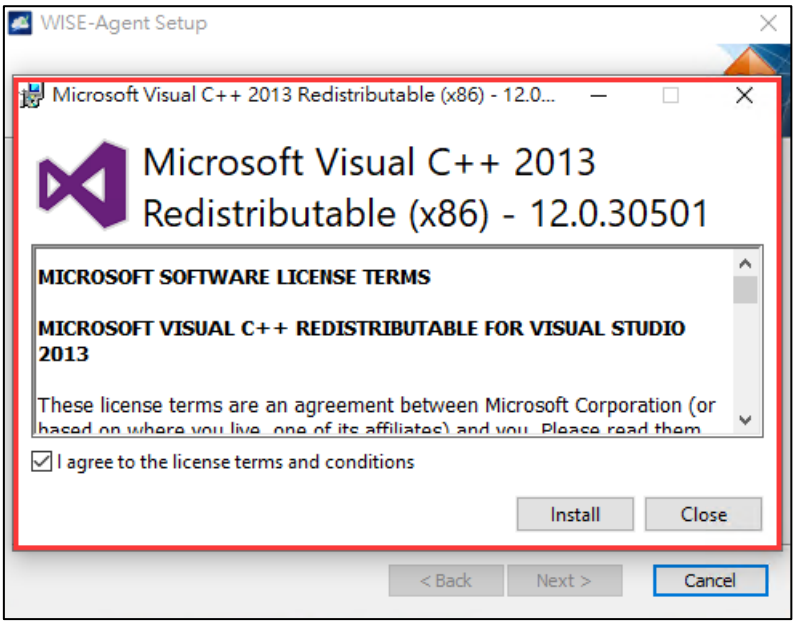
pre-configure iAMT, enable it in the device's BIOS and provide the account and password information in this dialog if you would like to enable iAMT based remote control features.



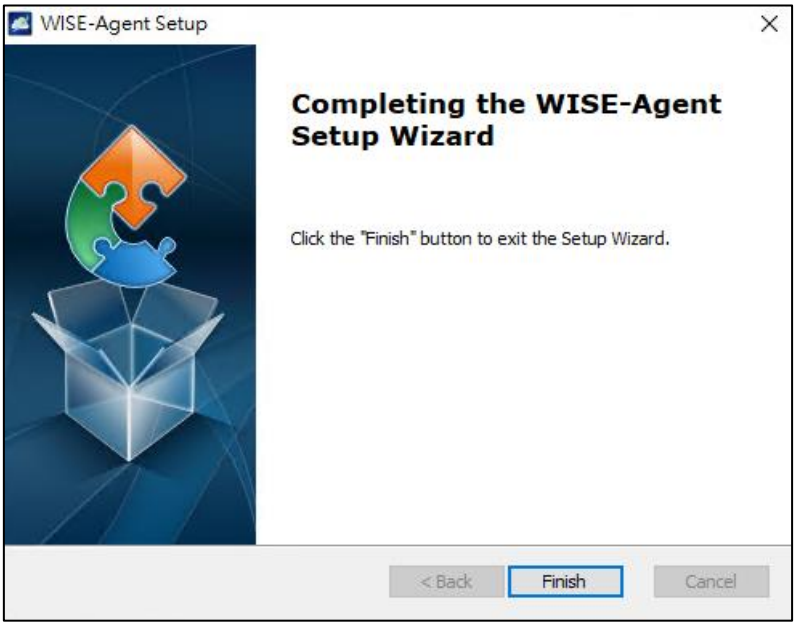
Click "**Install**" to begin the installation.



WISE-Agent requires the Microsoft Visual C++ Redistributable 2008, 2013, 2015 x86 packages, which will be downloaded from the Internet and set up during the installation process. If you are in an environment with limited or no Internet access, please download the "Agent Dependency Package" through an Internet connected device and install this package first.

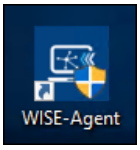


Click “**Finish**” to exit the program.



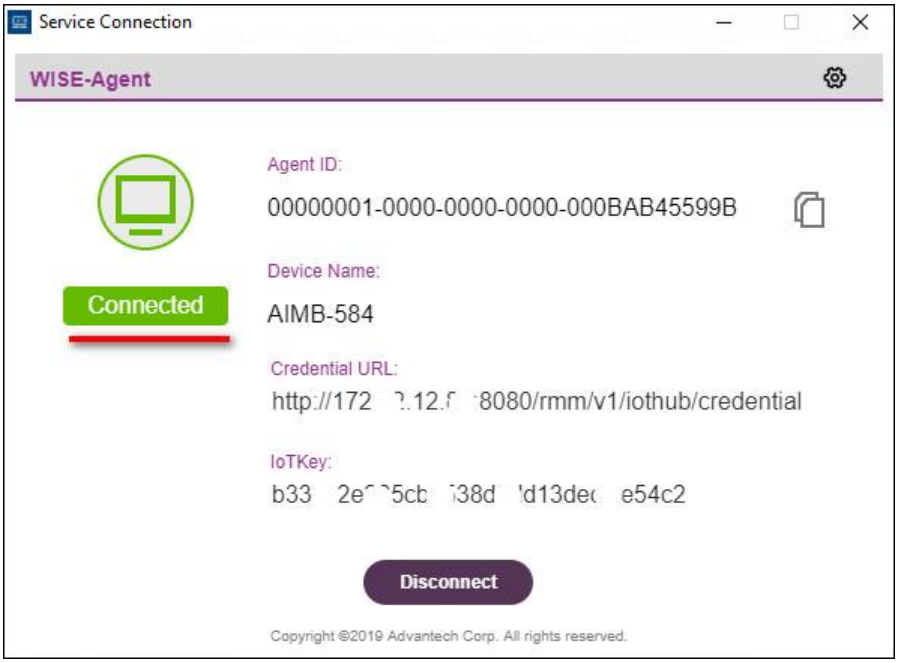
Step 4: Launch the WISE-Agent

Click on the “WISE-Agent” icon on the Windows Desktop to open the WISE-Agent user interface.



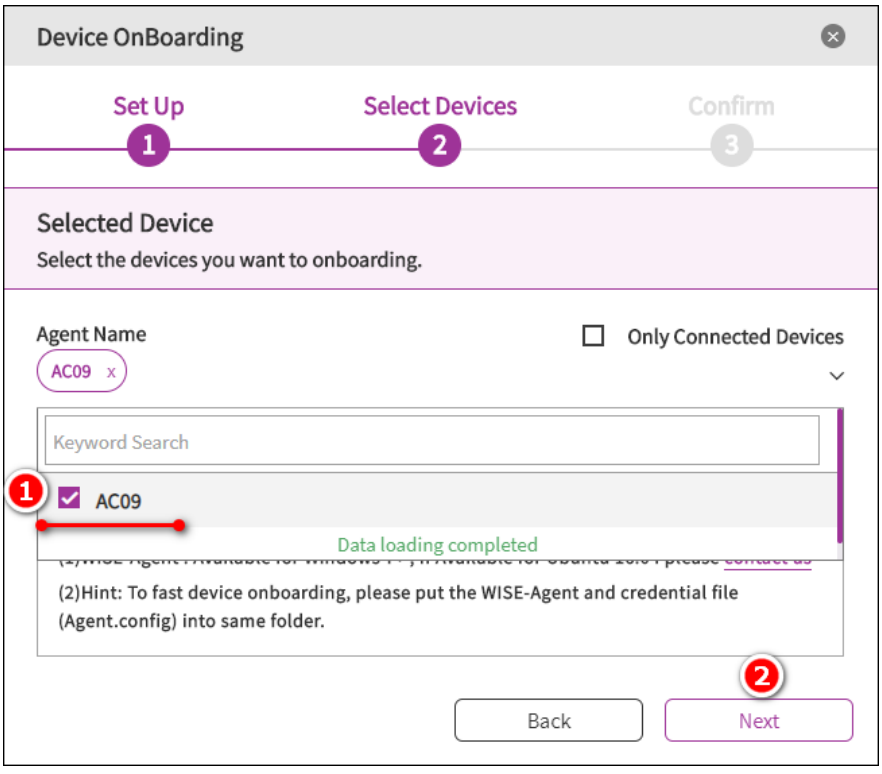
If the status shows “Disconnected”, please make sure your network settings are configured correctly and that you have access to the DeviceOn server-side application, either located in a public cloud

(WISE-PaaS, MS Azure) or on premise (standalone server version) depending on deployment scenario. Then, please click the “Connect” button to try to reconnect.

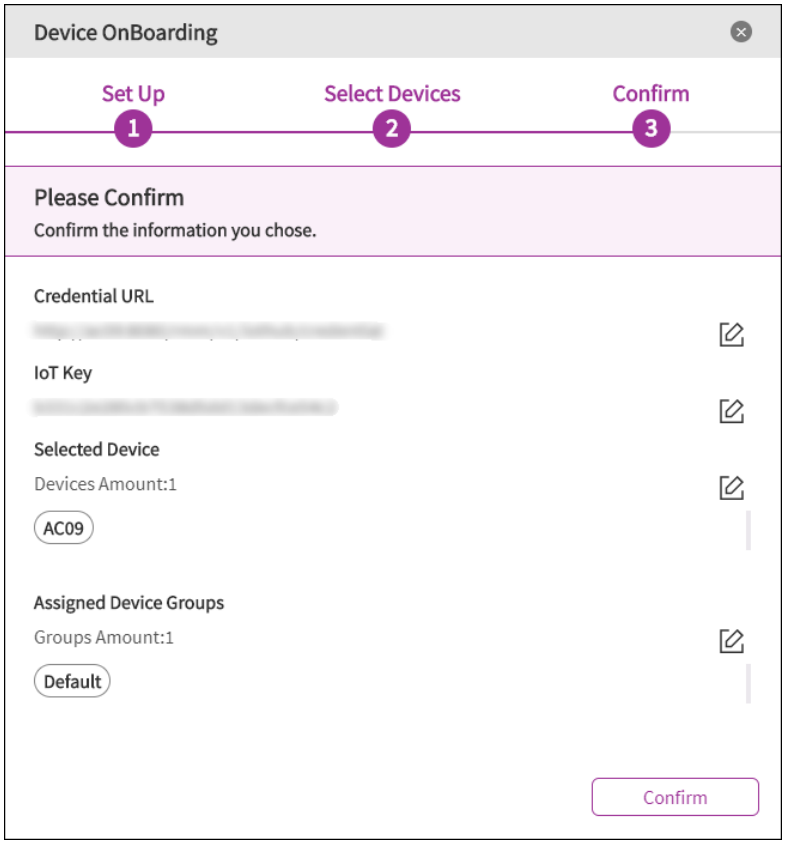


Step 5: Grouping Your Devices

Once the device connects, to device grouping page, where the device group for these devices can be selected.

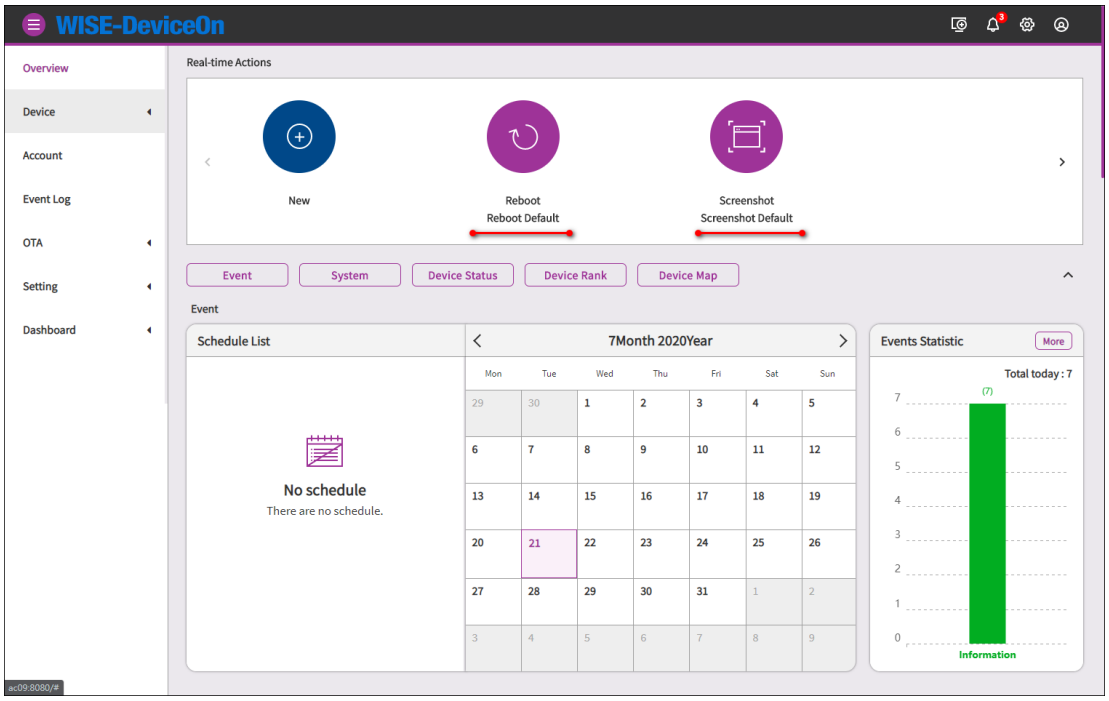


There is a “**Default**” group that can be used, or other groups for this device can be created after checking “Advanced options”. Click “**Confirm**” to start device management.



Step 6: Start Device Management

By default, two “Real-time Actions” are created for a group, one is “**Screenshot**” and the other one is “**Reboot**”. The overview page further shows the online status of registered.



2.2.2 Setup Device Onboarding (Linux)

We also provide an installer for Ubuntu Linux (one of the most popular Linux distribution). This section will guide you how to install WISE-Agent on Ubuntu Linux. Note here that:

The WISE-Agent Ubuntu Linux installer is named something like "**wise-agent-Ubuntu 18.04 x86_64-1.x.x.0.run**". To acquire the installer and ensure having the latest version, please contact us. If you are running the installer with an account other than "root", you should use "**sudo**" command to obtain higher privileges, or the installation may fail at any step.

Step 1: Open a terminal

The installer runs in CLI (Command Line Interface) mode. As such, open a terminal preferable for you.

Step 2: Copy the installer to target host

Use the way you like to copy the installer to the target host.

Step 3: Set the installer as executable

In the terminal, run "**chmod 0755 wise-agent-Ubuntu 18.04 x86_64-1.x.x.0.run**" so that the installer as an executable file under Ubuntu Linux.

```
sephiroth@sephiroth-VirtualBox:~$ chmod 0755 wise-agent-Ubuntu\ 18.04\ x86_64-1.
4.10.0.run
sephiroth@sephiroth-VirtualBox:~$
```

Step 4: Running the installer

Change your working directory to where the installer is and run "**./wise-agent-Ubuntu 18.04 x86_64-1.x.x.0.run** ". You may need to run "**sudo ./wise-agent-Ubuntu 18.04 x86_64-1.x.x.0.run**" to acquire higher privileges if you were logged in as a normal user.

```
sephiroth@sephiroth-VirtualBox:~$ sudo ./wise-agent-Ubuntu\ 18.04\ x86_64-1.4.10
.0.run
[sudo] password for sephiroth:
Verifying archive integrity...  100%   All good.
Uncompressing The Installer for WISE-Agent  100%
Install AgentService.
/tmp/selfgz28285
INFORMATION: Target device (Ubuntu 18.04) matched with (Ubuntu 18.04).
Copy AgentService to /usr/local.
'./AgentService' -> '/usr/local/AgentService'
```

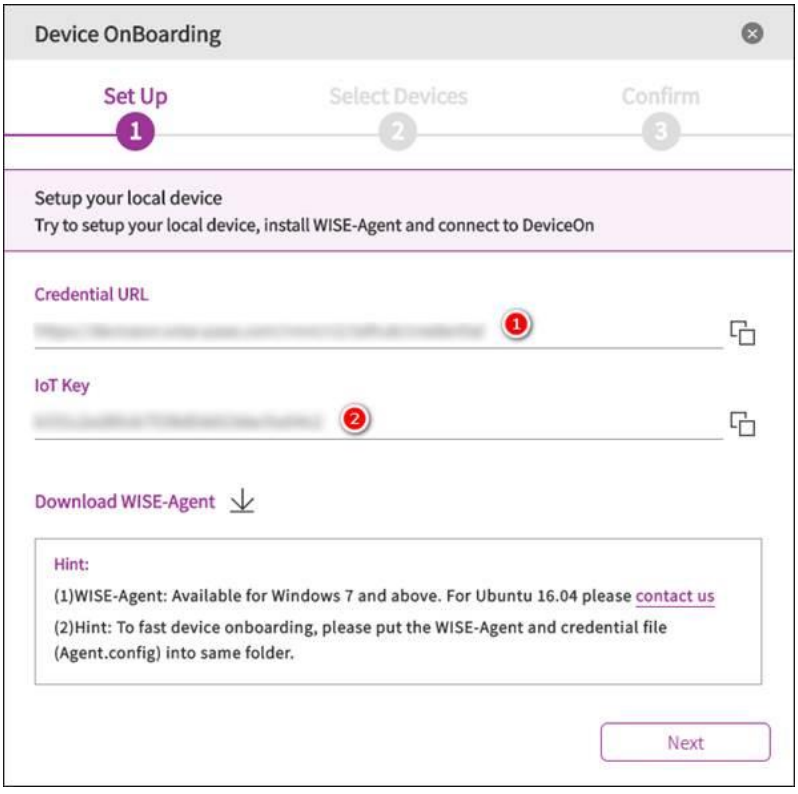
Step 5: Start WISE-Agent and Connect to DeviceOn

Change your directory to **/usr/local/AgentService** and run **sudo ./setup.sh** to answer connection information, such as credential URL, IoTKey, Device Name and etc.

```
sephiroth@sephiroth-VirtualBox:~$ cd /usr/local/AgentService/
sephiroth@sephiroth-VirtualBox:/usr/local/AgentService$ sudo ./setup.sh
========================= AgentService Linux Setup =========================
********************************************************************************
********************
FireWall is disabled
Pid: 28496
find app dir /usr/local/AgentService.
AgentService Path: /usr/local/AgentService
sending request to stop AgentService
********************************************************************************
********************
Do you want to configure WISE-Agent now? [y/n](default: y)y
Zero-touch onboard [y/n](default: n): n
Input Credential API URL(default:https://api-dccs.wise-paas.com/v1/serviceCreden
tials/):
Input IoT Key(default:):
Assign device to User Account [y/n](default: n): y
Enalbe TLS [y/n](default: n): y
Input Device Name[Len:4--35](default:sephiroth-VirtualBox):
Input AMT ID[Len:4--35, or na](default:):
Input AMT password[Len:8--16, or na](default:):
Select KVM Mode[0:default, 1:custom VNC, 2:disable](default:0):
Input VNC Port[1--65535](default :5900):
********************************************************************************
********************
Do you want to start WISE-Agent now? [y/n](default: y)

WISE-Agent Service Starting...
RMM Linux setup successfully!
sephiroth@sephiroth-VirtualBox:/usr/local/AgentService$
```

1. Zero-touch onboard is a zero-configuration and quick connection mode for a special purpose. The default is disabled (n).
2. Enter **Credential URL** and **IoT Key** that information could retrieve from the DeviceOn portal.



3. Assign device to User Account: You can bind the target device into a "Default" group in your account on the portal automatically
4. Enable TLS: Turn ON/OFF the TLS/SSL mode.
5. Input Device Name: Give your device name and show it on the portal.
6. Input AMT ID and password: If your device support Intel AMT, please enter AMT ID and Password to enable these functions.
7. Select KVM Mode [0:**default**, 1:**Custom VNC**, 2:**disable**]: User can use our default VNC to support the Remote Desktop function by entering 0 and give a listen port if you don't want to use the default port. Second, select **Custom Mode**, if they already have a VNC server by entering 1 and provide the listen port and password. To disable the KVM function by entering 2.

When you run into this step the question shows like above, device is connected and under your account.

3. DeviceOn User Interface & Functions

3.1 DeviceOn Server (Standalone)

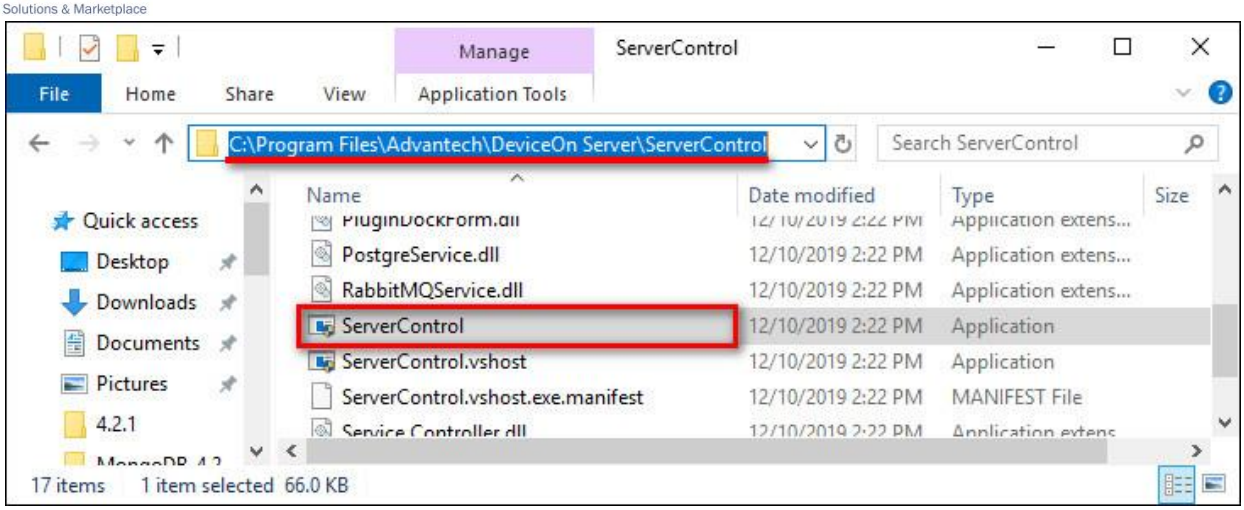
The standalone version provides all packages of the DeviceOn software in one installer package, including RabbitMQ as a message broker, MongoDB, PostgreSQL as databases, Grafana for visualization, Tomcat for web services, and a watchdog service that protects DeviceOn core components from crashing or becoming unresponsive.

The following section (3.1.1) introduces the "Standalone Server Control" tool that allows to monitor and enable/disable DeviceOn core components. The watchdog service is explained in section 3.1.2.

3.1.1 Standalone Server Control

After the DeviceOn standalone version has been installed, a "**Server Control**" icon should show up in the system tray.

If it does not show up for some reason, please go to installation path and launch the program (ServerControl.exe) manually as shown here:



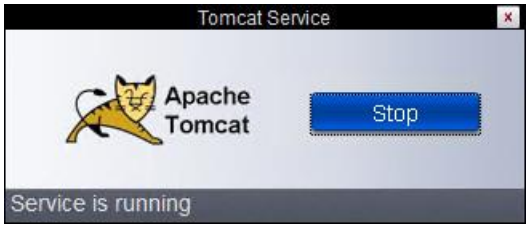
Right click on the tray icon to bring up an overview of each core component status. The green light indicates normal status and a red light means the respective service is stopped.

- Management Service

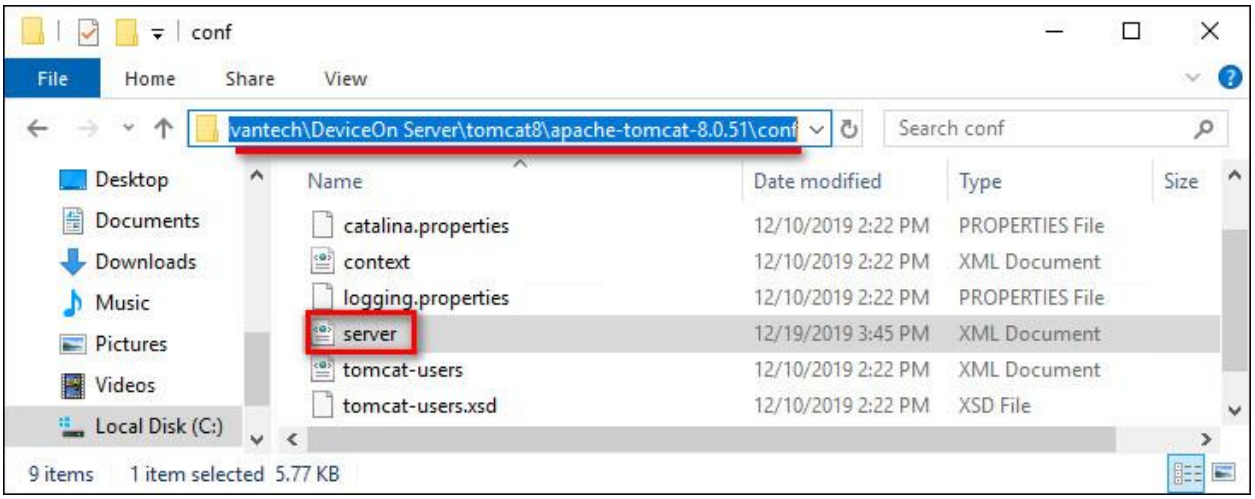
The “Management” service includes the DeviceOn backend core function and consists of two Java processes (DeviceOn and Provisioning Worker) that handle messages and process OTA traffic between the client and server. Click “Stop” to stop the management service.

- Tomcat Service

The DeviceOn web service uses Apache Tomcat to provide the user interface, APIs and WebSockets. Click “**Stop**” to stop the Apache Tomcat service.



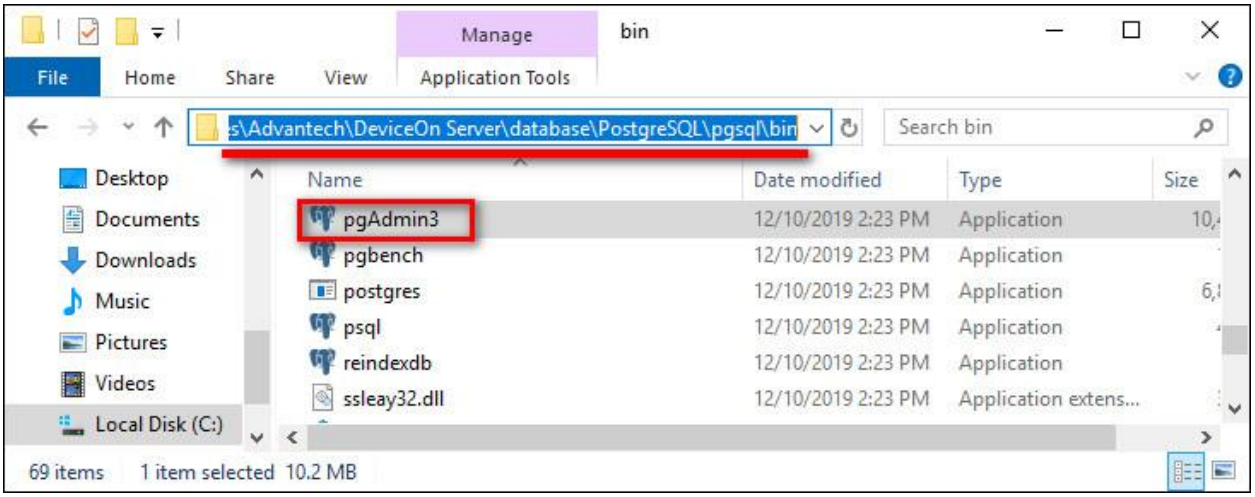
For advanced configuration (SSL, connection pool, etc.), you may modify "**server.xml**" located in the installation folder.



- PostgreSQL Service

The relational database (PostgreSQL) is used to store account, device, map, permission data etc. Click "**Stop**" to stop the PostgreSQL service.

A GUI tool called "**pgAdmin3.exe**" providing access to the PostgreSQL database comes with the PostgreSQL installation and is located in the installation folder as shown below. The default account is "**postgres**" and the password is the one you defined during the installation. We recommend you do not delete/edit any schema, table or data , since DeviceOn might stop to work if data is corrupt or missing.



- MongoDB Service

To process sensor data from client devices, DeviceOn leverages MongoDB to provide better performance and compression rates than relational databases. Click "**Stop**" to stop the MongoDB service.

- RabbitMQ Service

RabbitMQ is one of the most popular open source message brokers, and is used as "IoTHub" to exchange messages between the server and client devices. Click "**Stop**" to stop the RabbitMQ service.

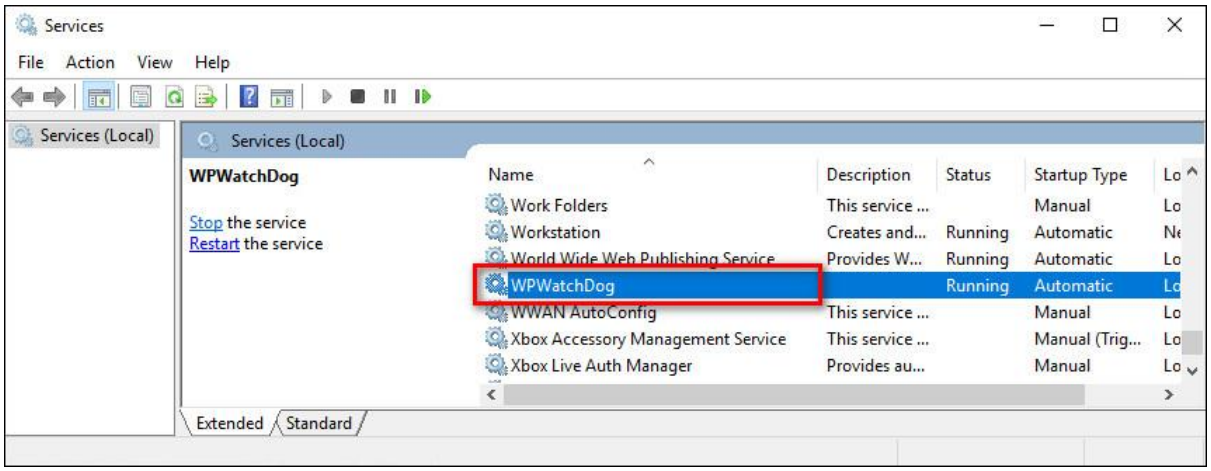
- Grafana Service

Grafana is a popular framework that allows you to query, visualize and alert on data from various data sources. DeviceOn supports a simple JSON API that as can be used as data source in Grafana, effectively making all DeviceOn data available to Grafana. Click "**Stop**" to stop the Grafana service.

3.1.2 Background Watchdog Service

- Watchdog Service

There is a Watchdog service (WP) that monitors the management service (DeviceOn and Provisioning Worker) and ensures all the functions work as expected.

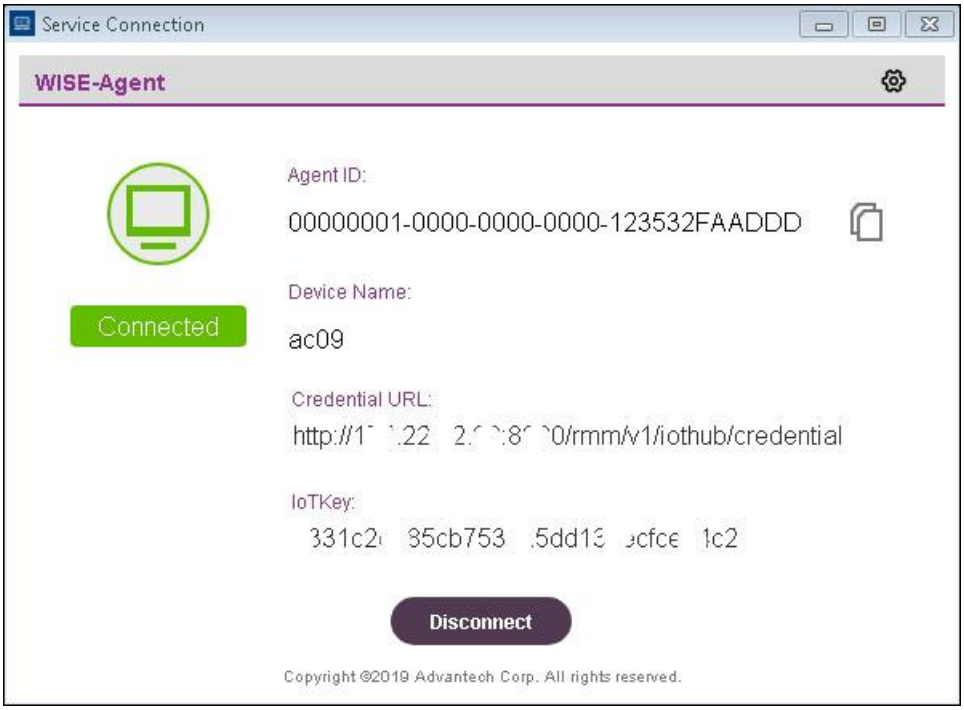


3.2 DeviceOn WISE-Agent

WISE-Agent runs as Windows service, so even without any user logged in, WISE-Agent will establish a connection to the DeviceOn server and the most of the features are supported. Section 3.2.1 explains how to use the WISE-Agent user interface to verify the current connection status and retrieve basic information of the client device. There is another Watchdog service monitoring the WISE-Agent client in order to avoid impact due to crashed or hanging processes.

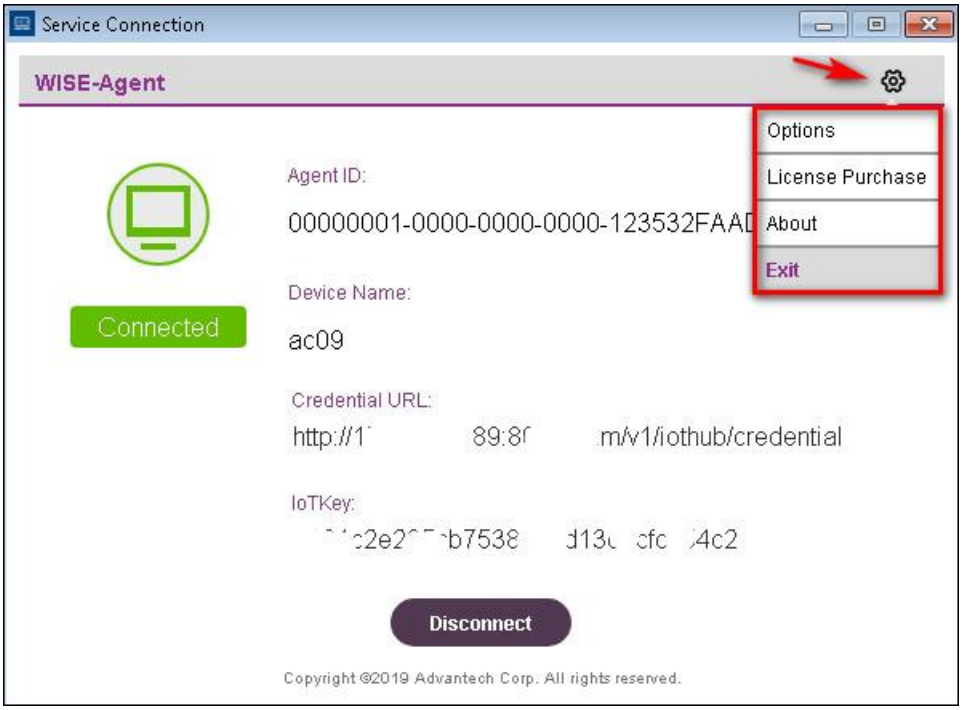
3.2.1 WISE-Agent Connection

If you followed the instructions to set up WISE-Agent and connect to the DeviceOn server/cloud, there should be a WISE-Agent shortcut on your desktop. If not, please refer to Section 2.3 to install WISE-Agent. After launching the WISE-Agent user interface, it will provide an overview of the connection status, device information (AgentID, Device Name) as well as connection credentials (Credential URL, IoTKey).



- **Agent ID**: Device unique ID - the default is 32 characters, prefix (20 characters) and MAC address (12 Characters)
- **Device Name**: Device name as shown on the DeviceOn server
- **Credential URL**: Connection URL, used to authenticate to DeviceOn Server
- **IoTKey**: Connection Key - each DeviceOn client has a unique key that will be used to establish the MQTT session
- **Disconnect**: To stop the device connection and data transmission, you can click “Disconnect” to stop the WISE-Agent service

If you would like to adjust the device name or connection parameters, please click the “Settings” icon on the top right and select “**Options**”.

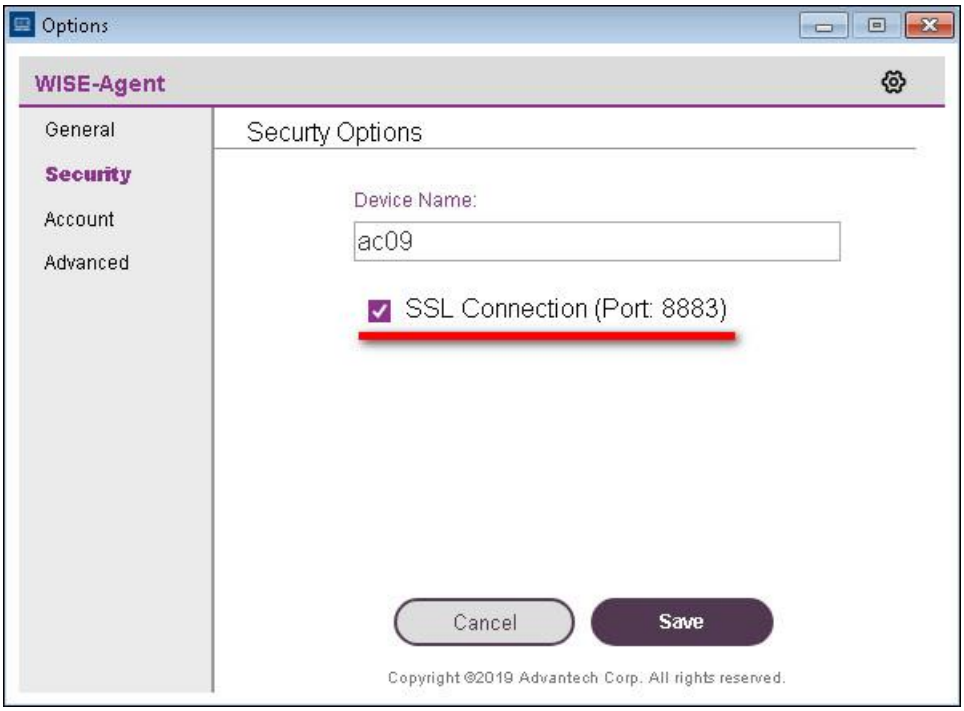


- Option -> **General**

This overview page provides information about “Device Name”, “Operating System” (Windows 7, 8, 10), “MAC Address” of the client, “Memory Capacity” and version of the Advantech SUSI Driver (if applicable). The version of the “Operating System” represents the Windows kernel version. If the client device is an Advantech platform that is supported by SUSI, we recommend to download the latest SUSI driver from the Advantech Support site first. Please click here to obtain the latest driver version.

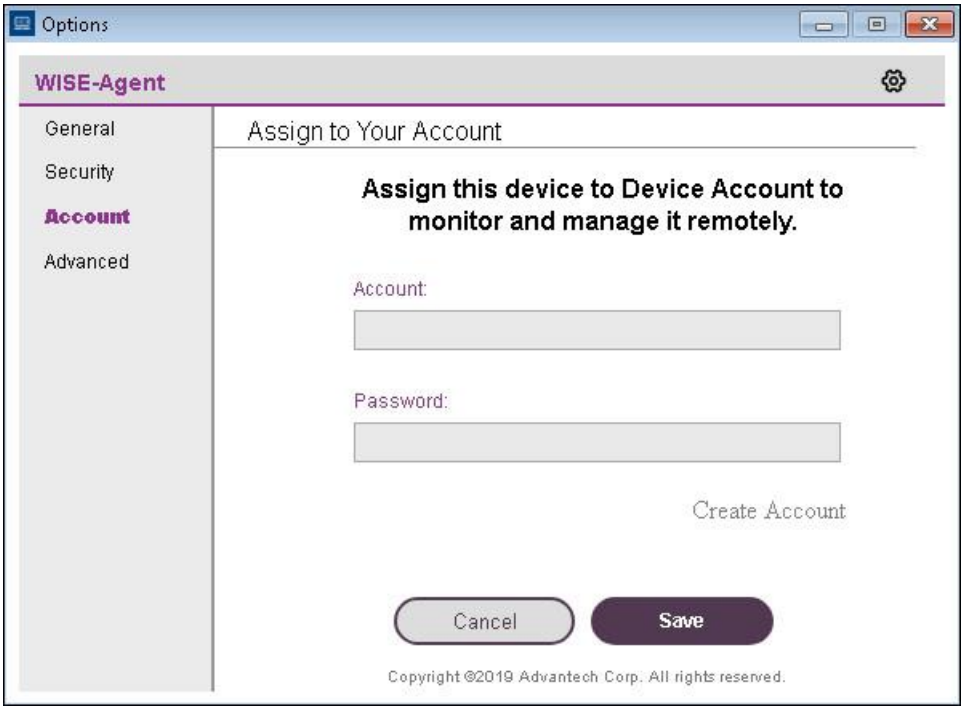
- Option -> **Security**

The communication protocol used for message exchange between the server and client is MQTT, an industry standard lightweight messaging protocol for small sensors and mobile devices. WISE-Agent provides the option to use MQTT with SSL encryption on port 8883, or MQTT without SSL on port 1883.



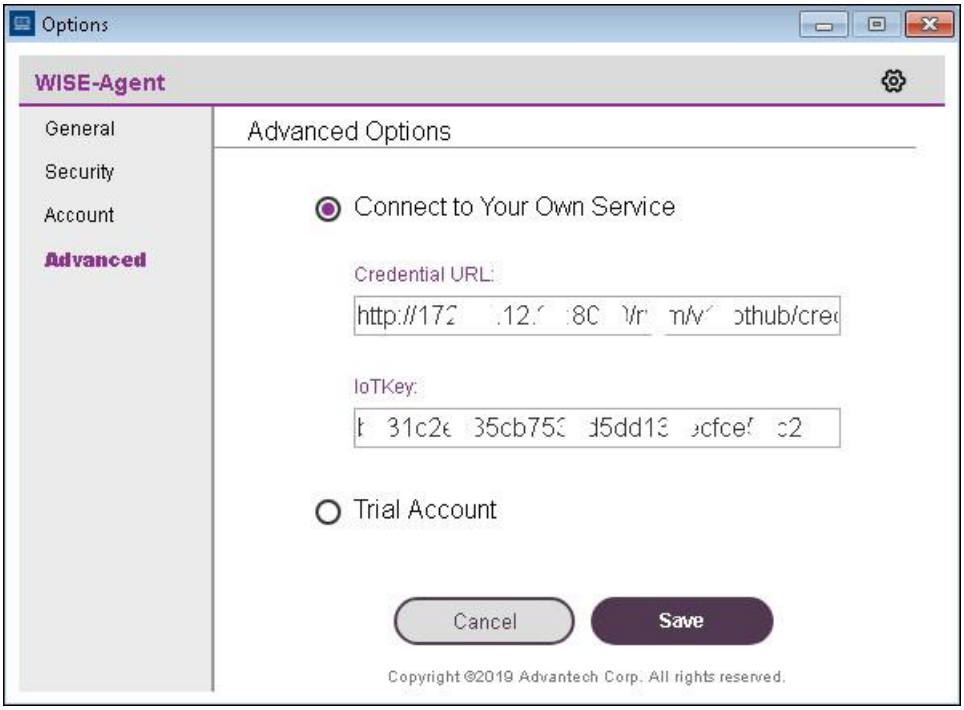
- Option -> **Account**

You can register on the DeviceOn trial site (https://deviceonapp.wise-paas.com) for a six-month trial account and use it with your device. Before you can create a trial account or enter trial account information, please got to the “Advanced” tab and select “Trial Account”.



- Option -> **Advanced**

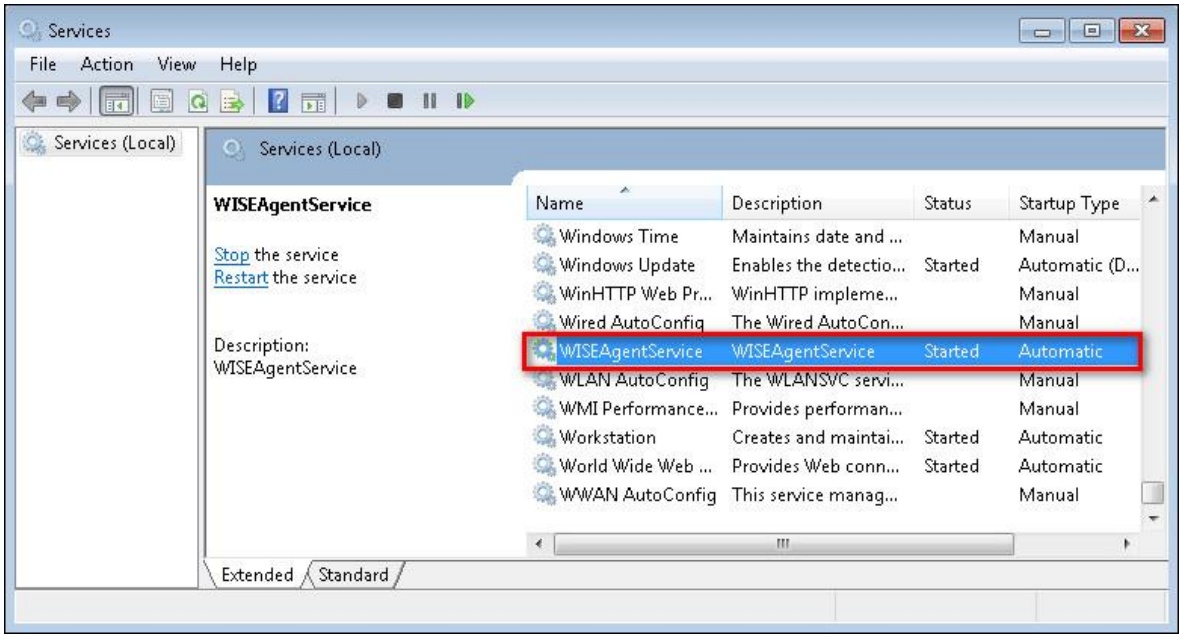
Under the “Advanced” tab, you can select whether to connect to a DeviceOn server/cloud service, or whether to connect to the DeviceOn trial site (https://deviceonapp.wise-paas.com/). In case of trial site, you need to enter account information under the “Account” tab (see previous step) while for a regular DeviceOn server or cloud service, you need to enter the “Credential URL” and “IoT Key” here. Refer to “Step 2” in Section 2.3 on information how to obtain those.



3.2.2 WISE-Agent Services

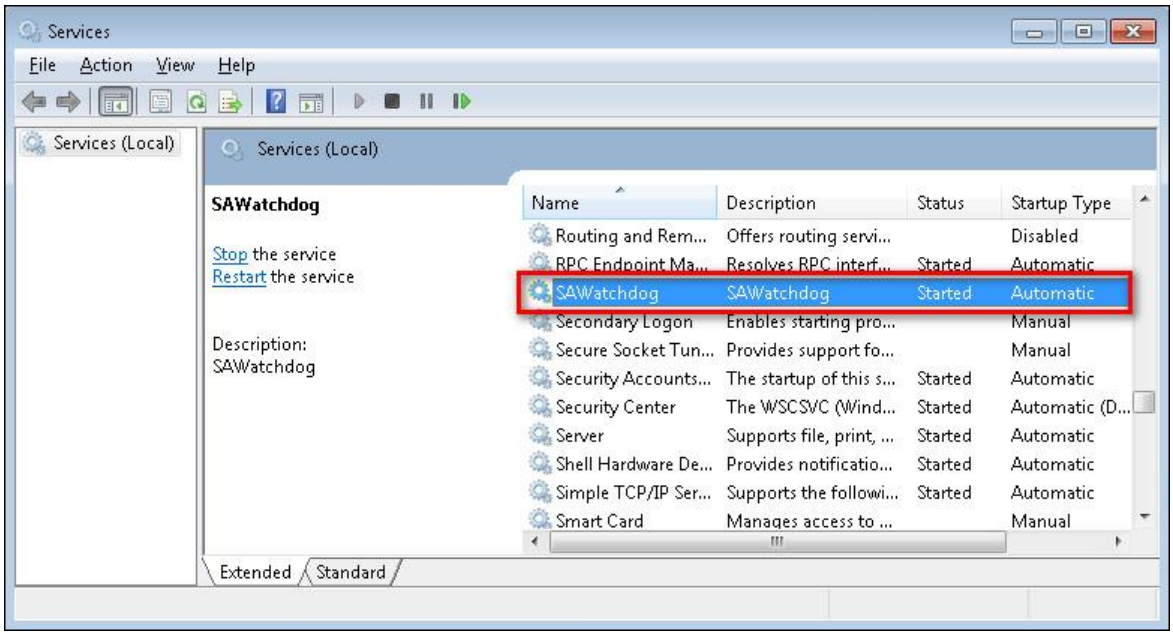
- Main Service

"WISEAgentService" is the main services that connects to the DeviceOn server/cloud service. The service is set to start automatically by default.



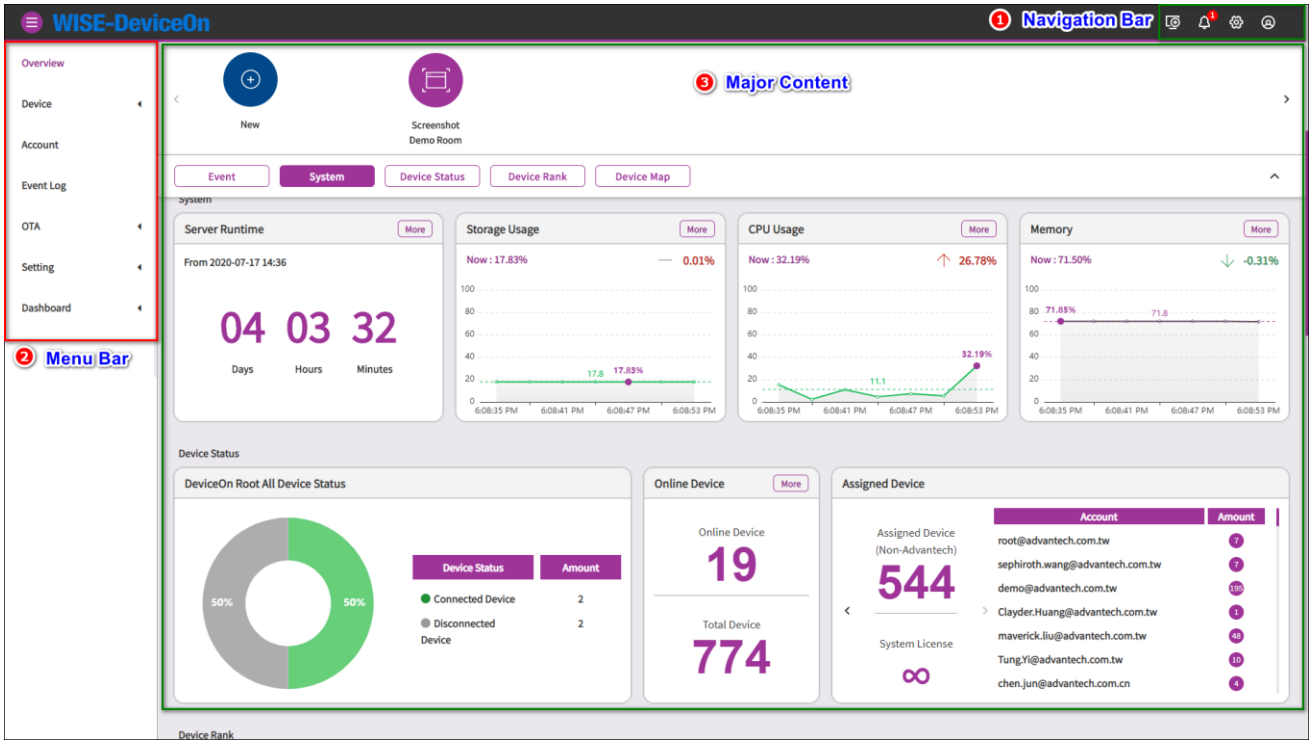
- Watchdog Service

The "SAWatchdog" service is a basic watchdog governing "WISEAgentService" in order to ensure service quality.



3.3 DeviceOn User Interface

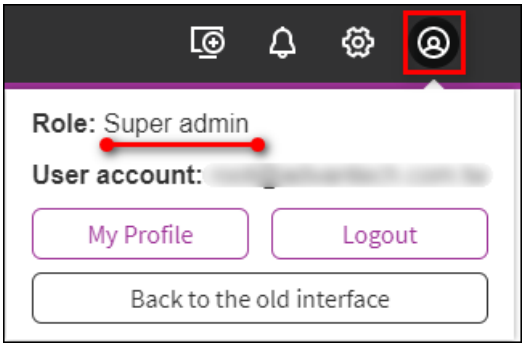
The DeviceOn web interface is based on the VUE framework and leverages the Vuestic Admin template. The user interface is divided into three main parts - the navigation bar at the top, the menu bar at the left and the main content in the center with.



Navigation Bar:

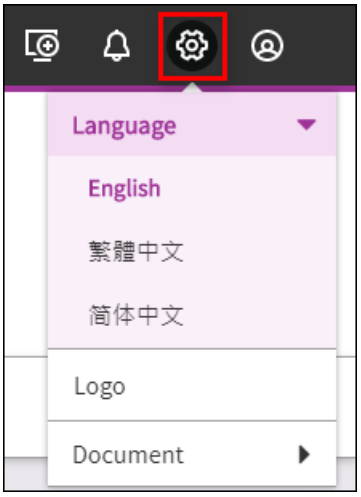
- Account Information

Click the account icon to show the currently logged in account and respective role. For more information, click “My Profile” to open the account page. (Menu Bar -> Account). Click “Logout” to log out from DeviceOn and remove personal information like cookies or tokens.



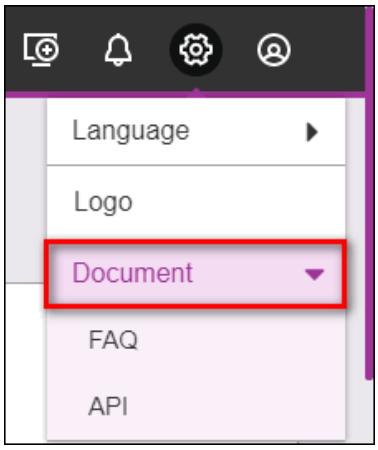
- Languages

DeviceOn supports multiple languages that can be changed by clicking the globe icon in the navigation bar. Currently there are three languages to choose from: English, Traditional Chinese and Simplified Chinese.

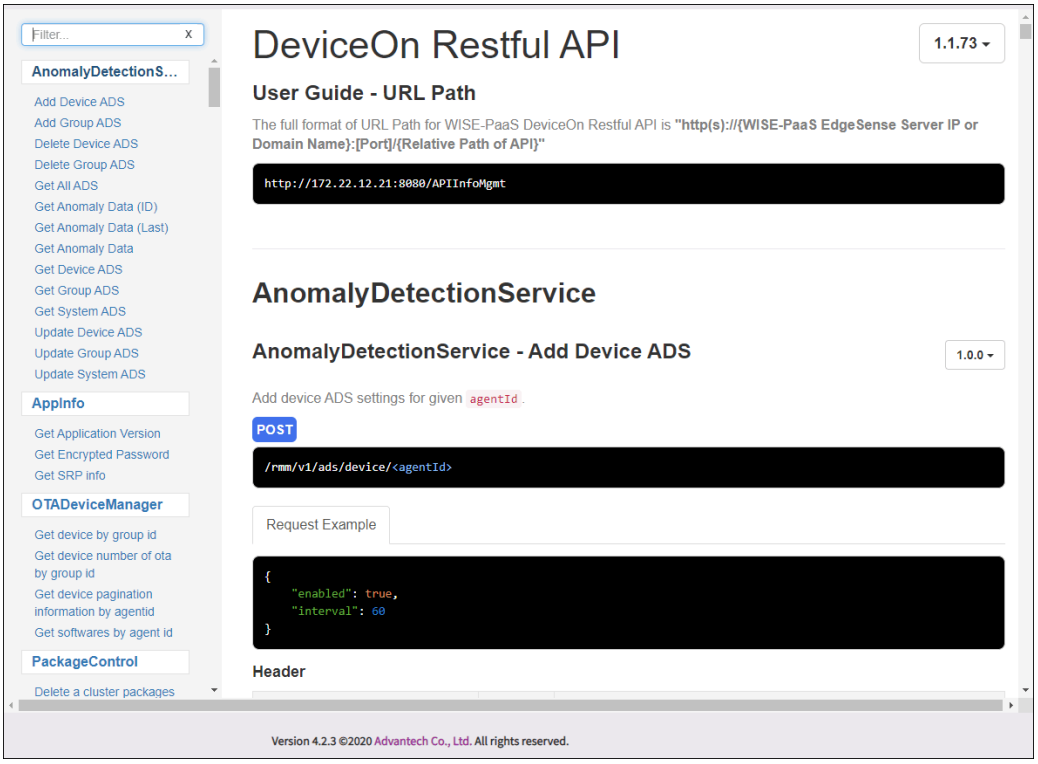


- Document (FAQ & API Reference)

There are two documents on DeviceOn user interface, one is Restful APIs and another is FAQ that including technical and general questions.



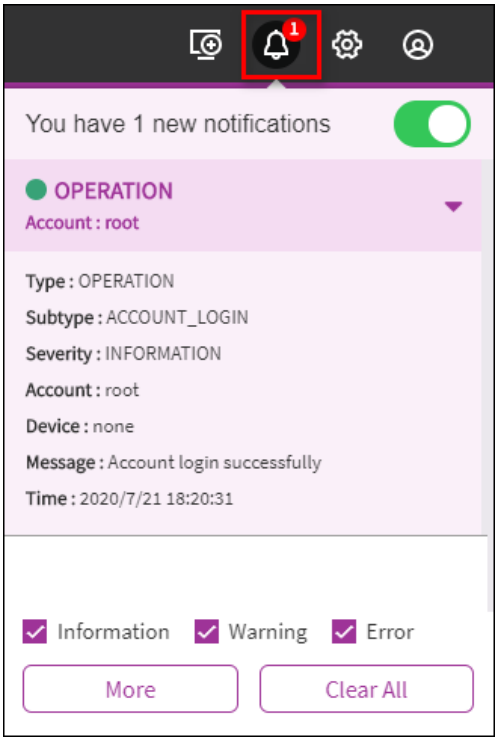
DeviceOn provide hundreds of API for App engineer to build up their AIoT solution, through the APIs to get account, map, device data, and remote diagnostic on devices. The API document is generated by APIDoc, includes API method, request, response, header and testing.



Actually, the developer could design a plugin on WISE-Agent to aggregate edge data (Reference Section 5.1), and get these data via Restful APIs, visualize on Grafana Dashboard (Reference Section 4.4) or develop a UI plugin to customize. (Reference 5.2)

- Notification

If there are any active notifications, the number of event log messages is shown on the notification icon. Click the notification icon to see the event message summary. Three levels of events are supported: "**Information**", "**Warning**" and "**Error**", and the user can select which type of events should be shown on the user interface. For example, clicking the "**Unsubscribe Notification**" would disable any events in the screenshot shown below. Please note that after disabling events, the UI will not refresh automatically but needs to be refreshed manually. Click "**More**" to open the event log page (**Menu Bar** -> **Event Log**)



- Device Onboarding

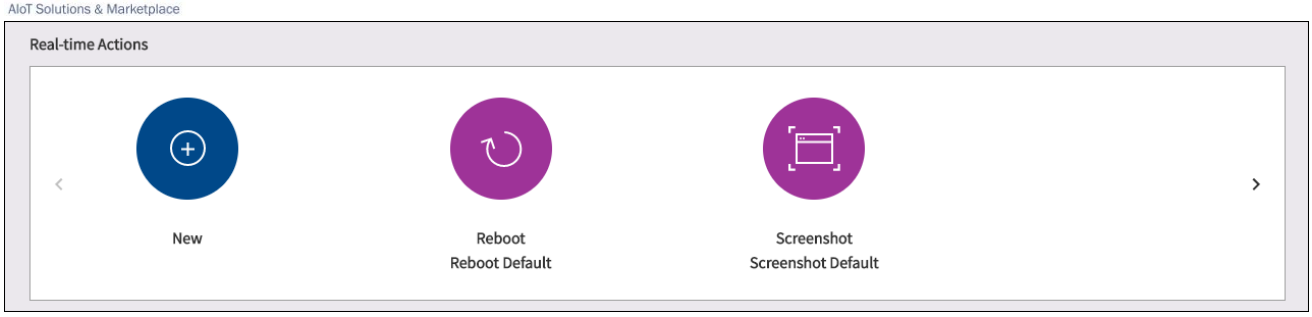
To onboard devices, click the onboarding icon in order to download the WISE-Agent installer and in order to look up the required connection credentials. For more details on onboarding, please refer to Section 2.2.

3.3.1 DeviceOn Overview

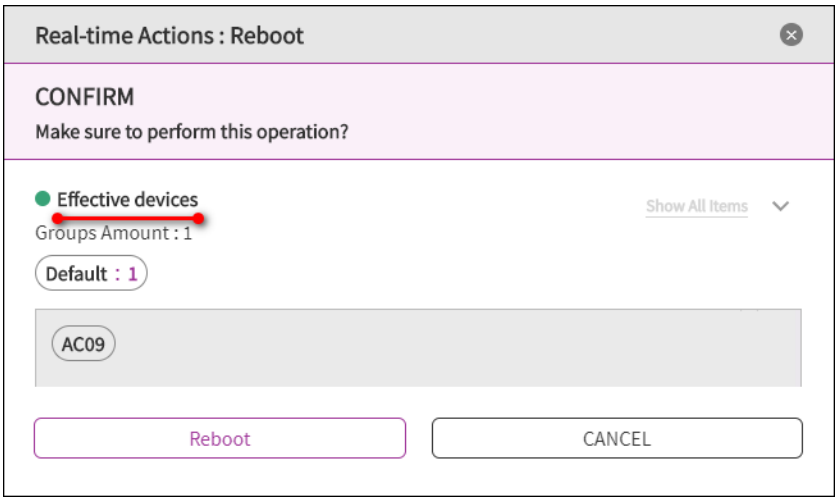
The overview provides quick access to real-time statistics for your managed devices. This information helps to monitor overall status as well as identifying high risk devices. Currently the overview includes Real-time Action, Scheduled Tasks, System Analysis, Device Ranking, and Device Map.

- Real-time Actions

Real-time actions provide one-click access to certain actions defined for specific device groups, providing a shortcut for efficient management. Examples for actions are batch reboot, batch screenshot or batch updates.

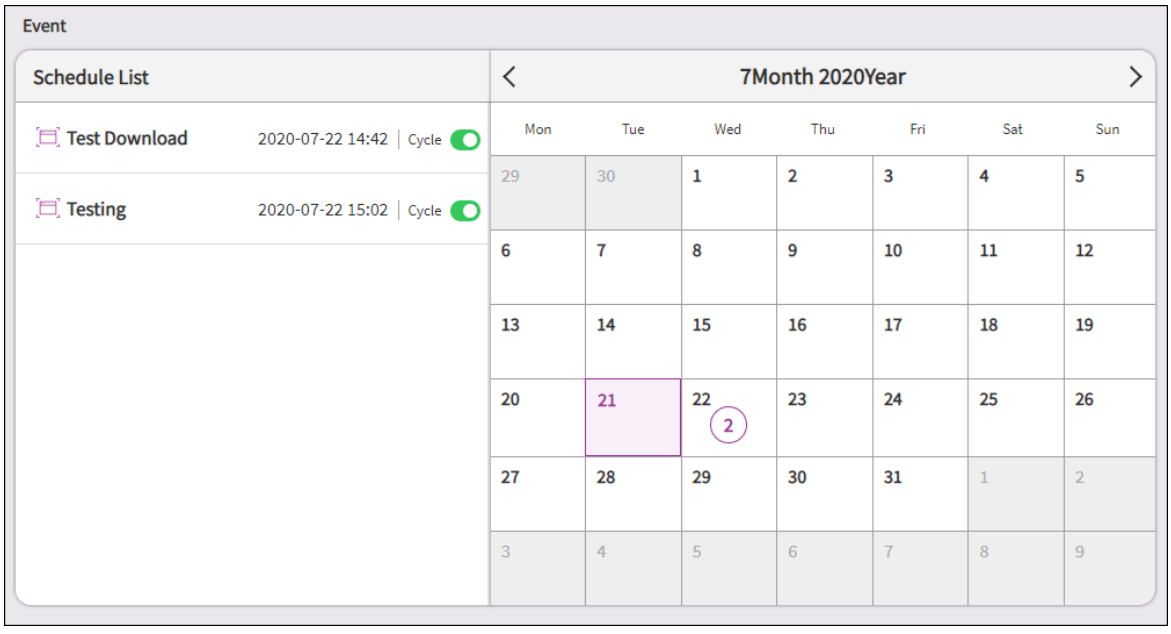


For example, once you click “Reboot”, a confirmation dialog will pop up and will indicate which devices will actually be affected. Click on the device group button to get more details (individual devices names).



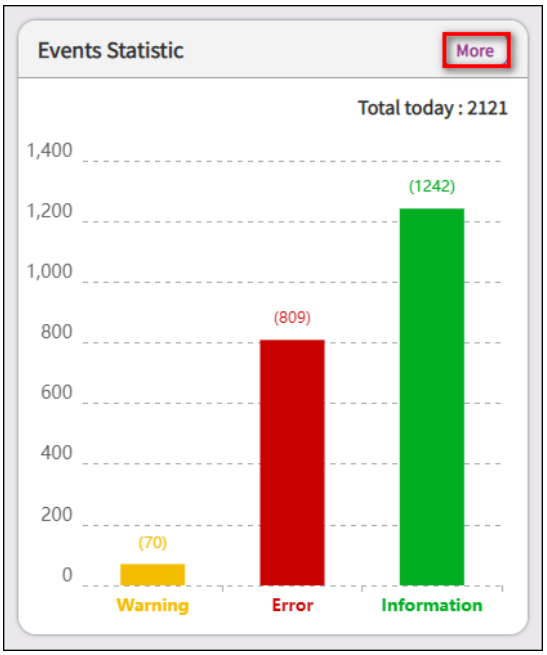
- Scheduled Tasks

In addition to real-time action, actions can be scheduled. An example for this is powering off or rebooting devices at a certain time of day. A calendar view is used to visualize upcoming tasks.

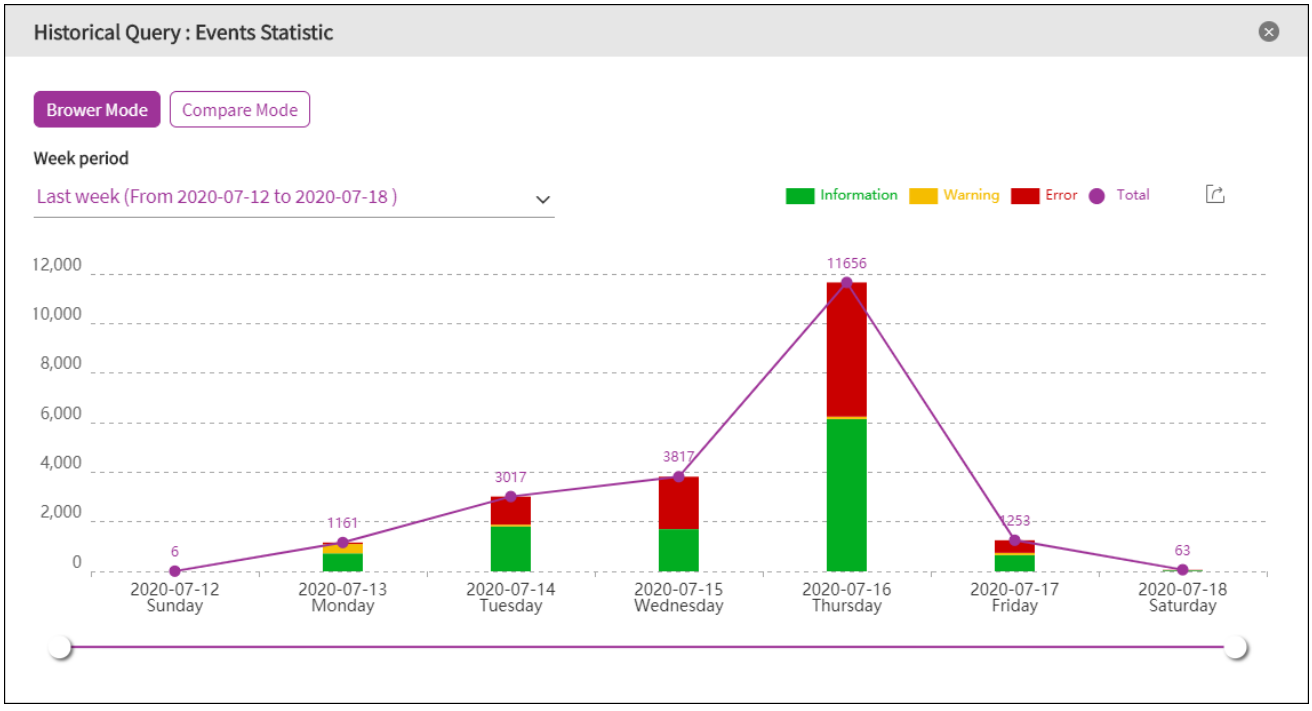


- Event Log Statistic

There are three levels of event log on DeviceOn system, such as **Warning**, **Error**, and **Information**. Gives a summary and statistic result for current day. Click on the bar char to redirect **Event Log** tab to check detail log information.

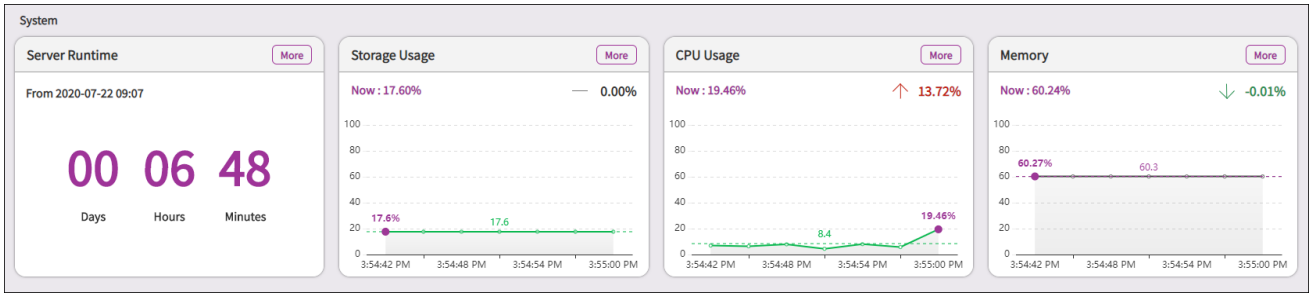


For historical data, please click on "**More**" to compare the daily, weekly result.



- System

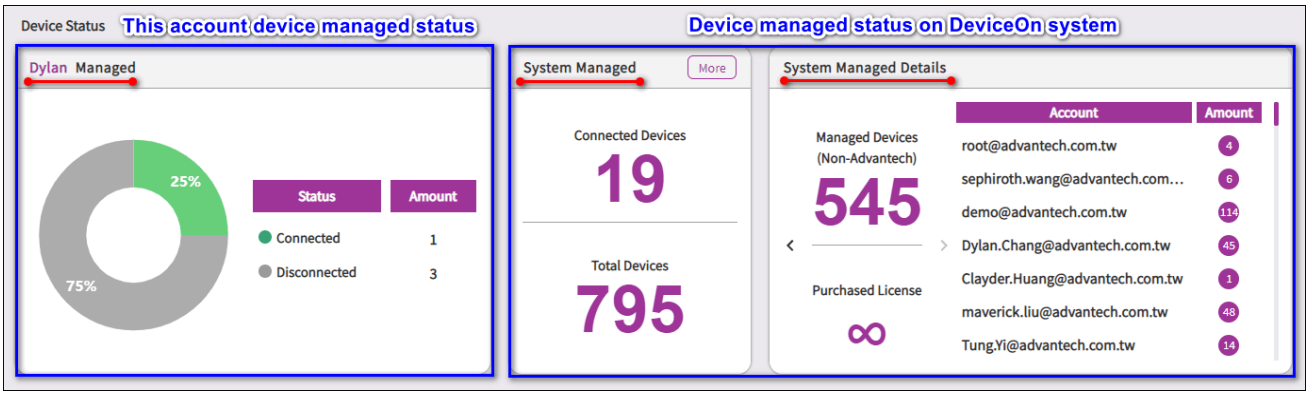
The system is focus on DeviceOn server loading and usage, including storage, CPU and Memory. The administrator could realize the real-time server uptime, downtime through the overview and based on the matrix to scale cloud performance.



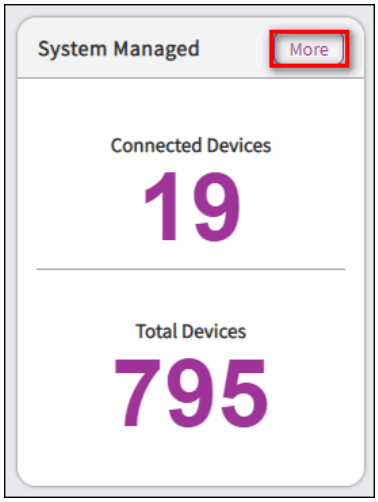
- Devices Status

Shows the total devices of current account and DeviceOn system managed.

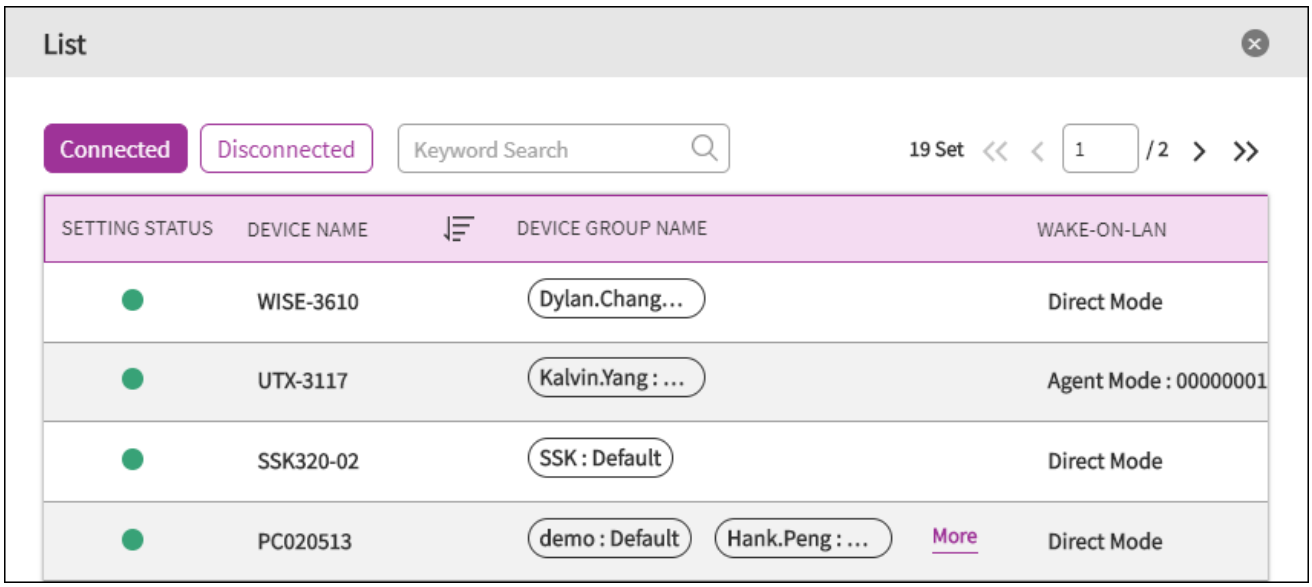
Shows the number of currently online devices as well as total number of managed devices (assigned to account).



Click on the more information on system managed to show the number of currently online devices as well as total number of managed devices.



Clicking this overview will bring up a detailed device list including status as well as group membership information.

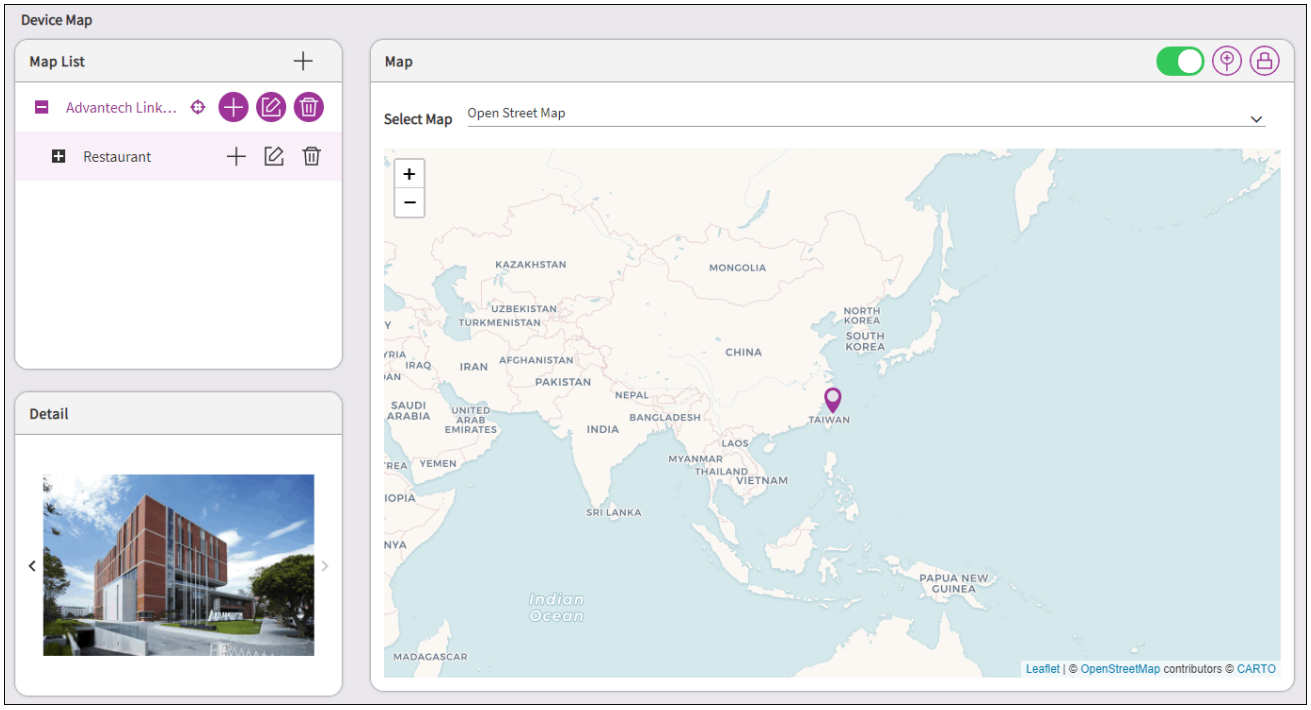


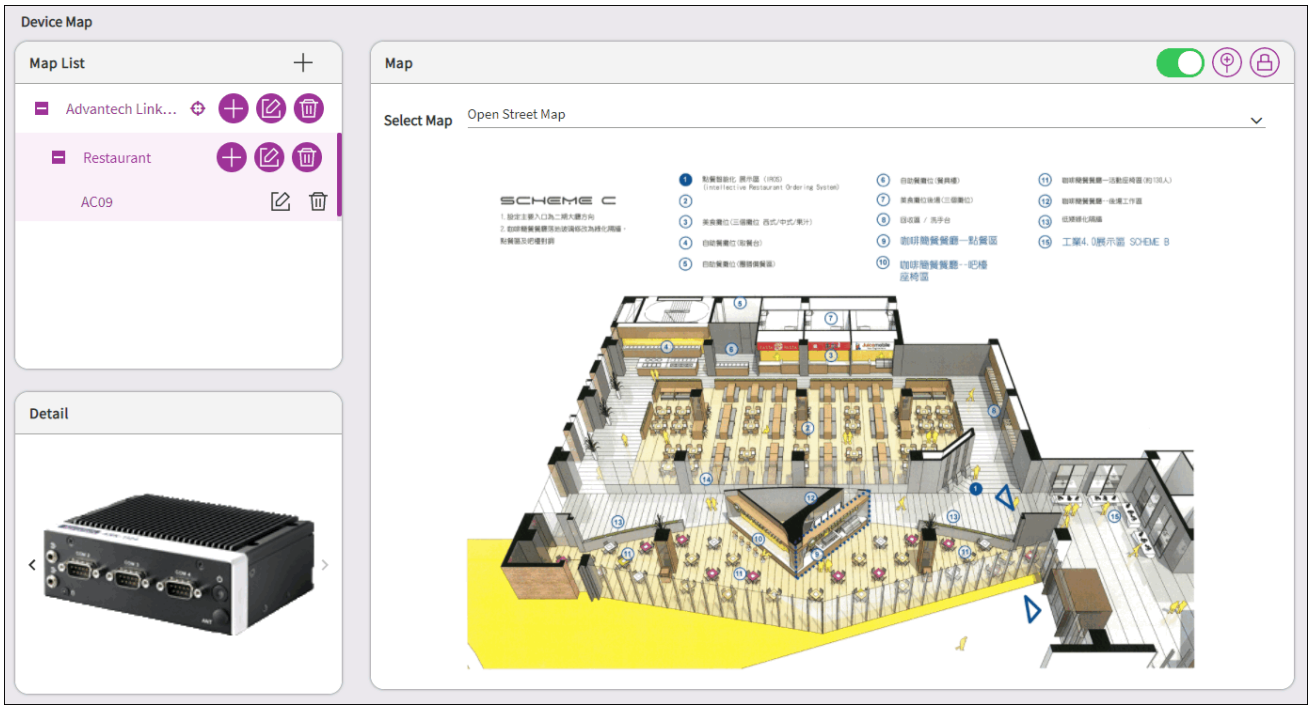
- Top 5 (High-Risk) Statistic

DeviceOn leverages six common sensor types to identify potential high-risk devices. Those sensors are device disconnects, network traffic, disk usage, disk health, CPU usage and memory usage. This “top 5” overview allows to quickly identify potential issues and fix or replace the systems to avoid unexpected downtime.

- Device Map (Open Street/Google/Baidu Map)

DeviceOn offers support for maps (latitude and longitude based position) or floor plans in order to visualize the location of managed devices. User could define their location on the map and place the device to the area (floor plans).

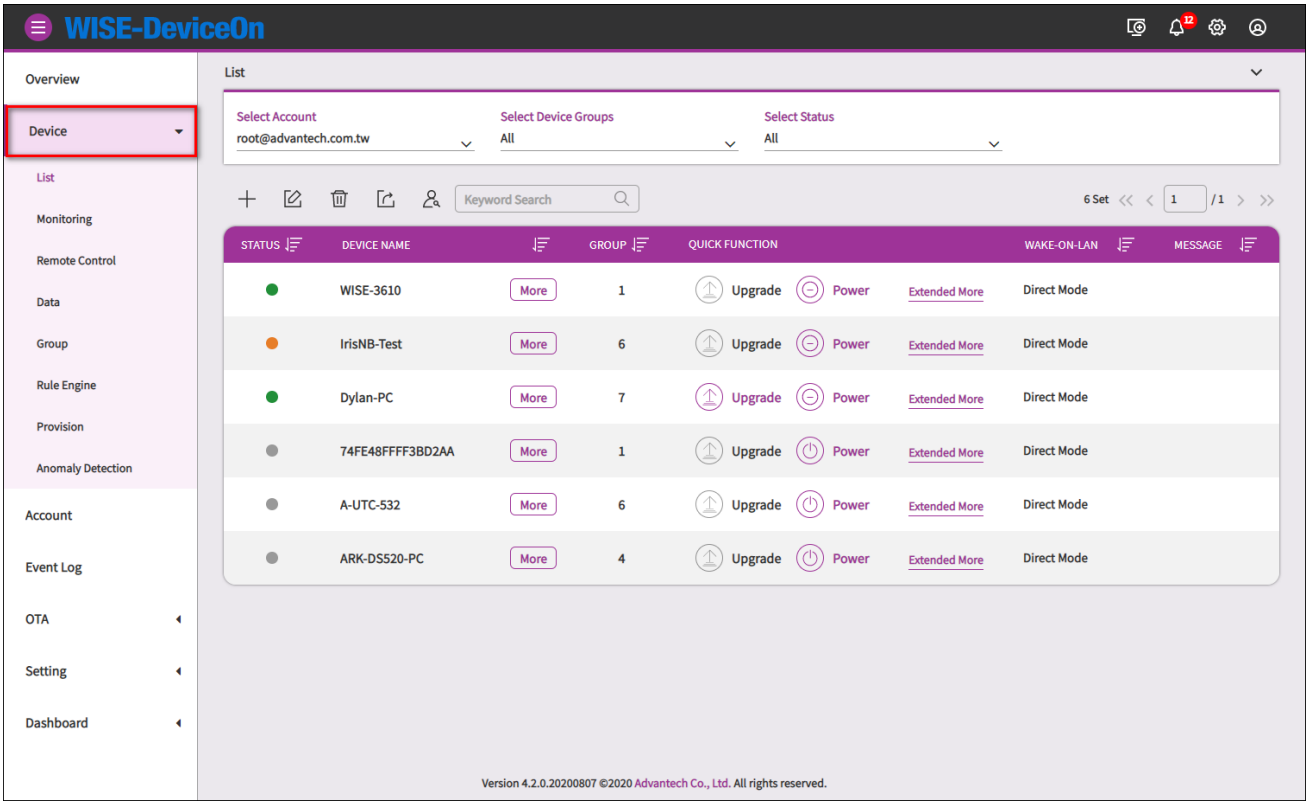




3.3.2 **Device Management**

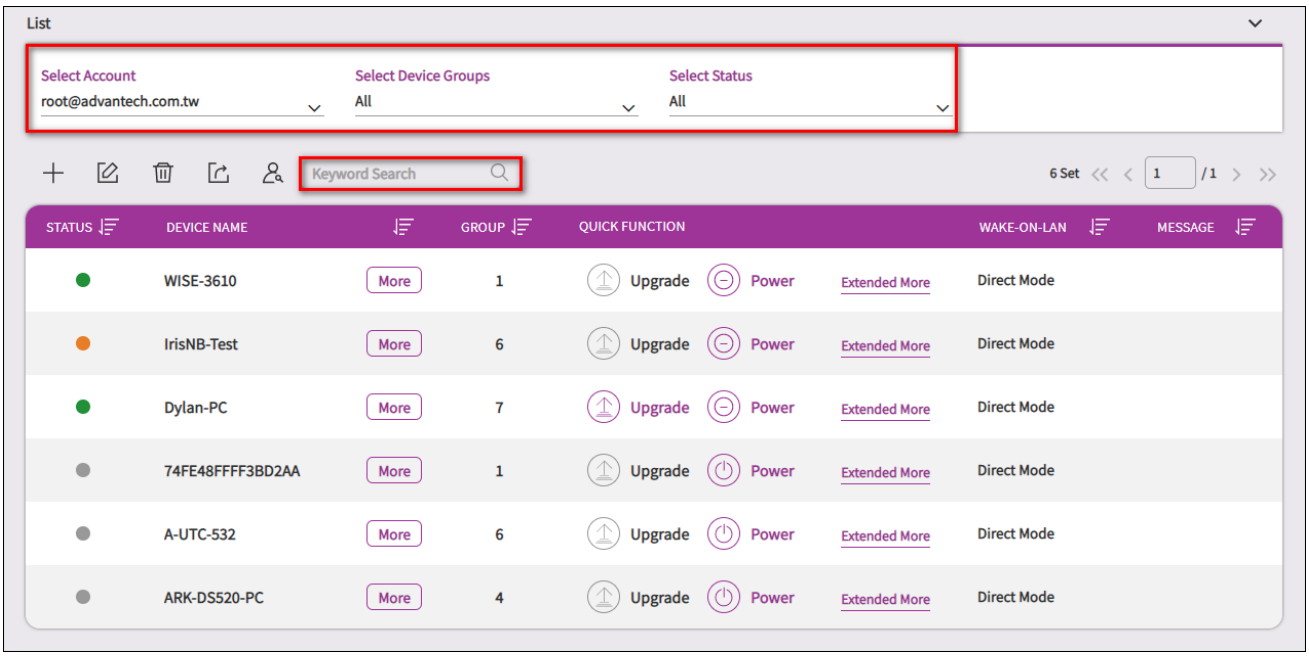
After your device onboarding, you could view, edit device basic information, remote control, and retrieve sensor data on your devices. Eight sub items under Device, Device List contain device name,

upgrade status, power management and etc. Device Monitoring to give device loading at present. To remote diagnostic and debug through Remote Control. Next, the all of plugin sensor data from Device Data. To grouping you device through the Device Group. Rule Engine to set a threshold rule for your devices data in real-time. For advanced configuration, such as WoL, System Backup/Recovery and Protection via Device Provisioning. The last, one of AI solution to detect device screen status on Anomaly Detection.



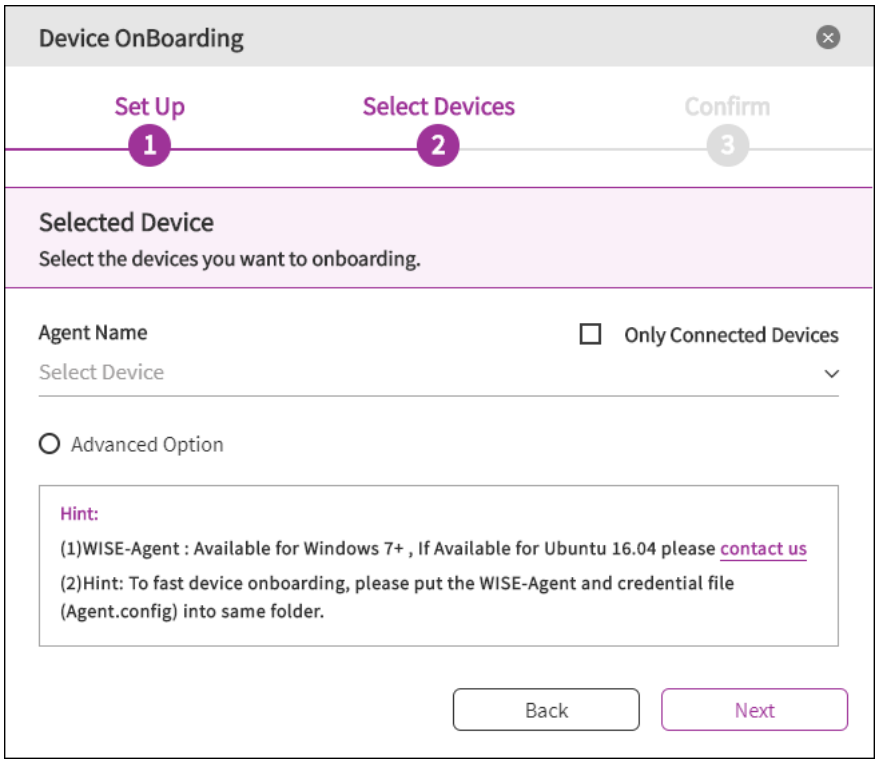
- Device List

The device could be assigned to multiple accounts and device groups; therefore, you could leverage filter to find your device through **Account**, **Device Group, Status** or **Keyword**.

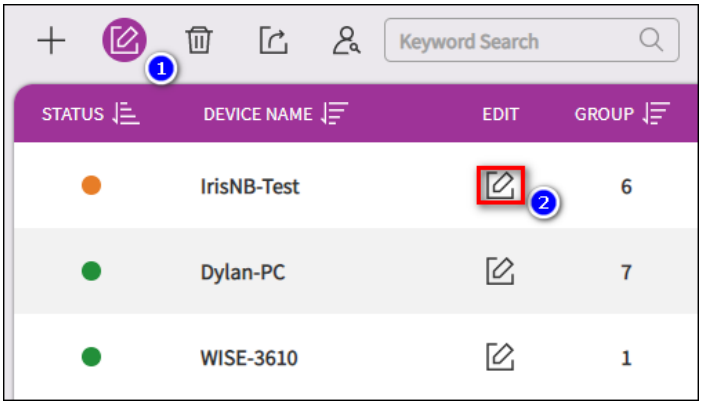


Here is action bar for add, edit, search or export for below table devices.

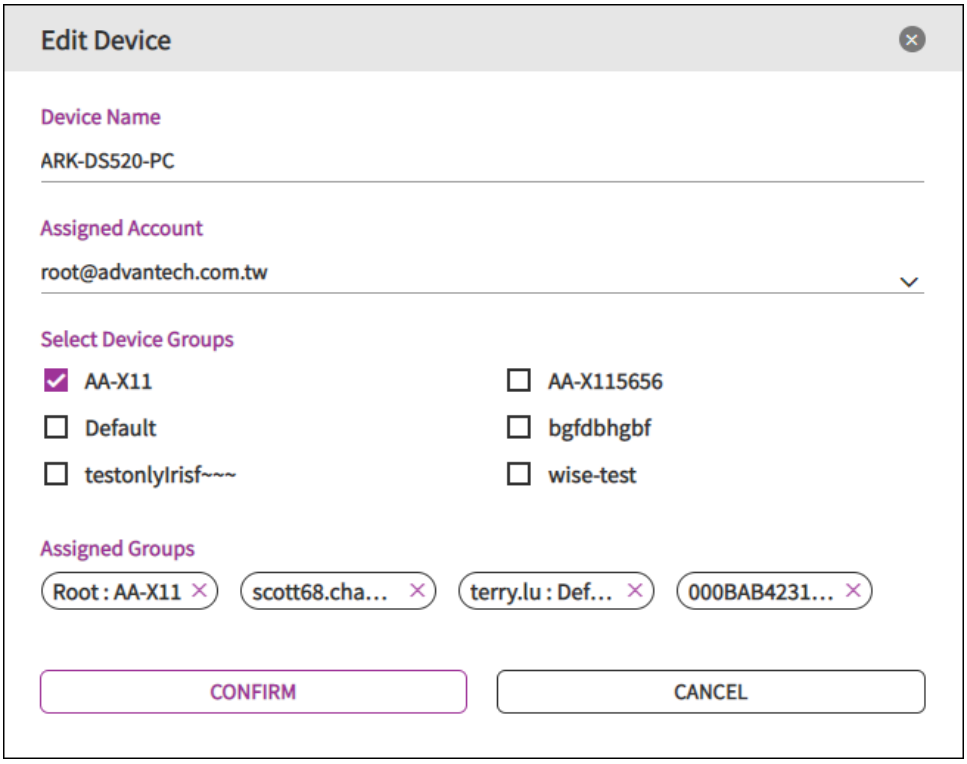
Click the icon to add devices, that's similar to device onboarding, download WISE-Agent, setup to your local device and grouping.



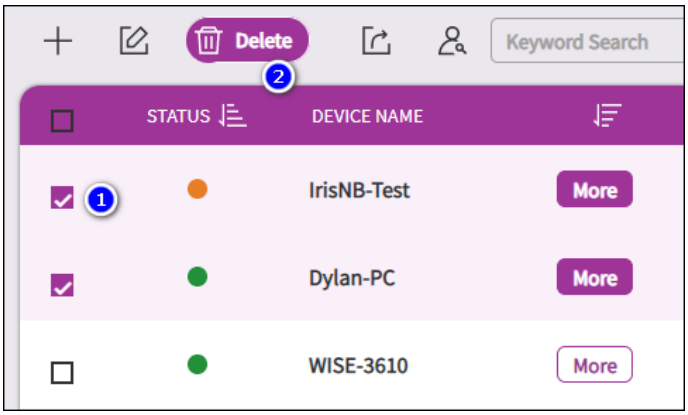
Click the edit icon to display "**Edit**" options on each device list.



You could edit device name, assign to different accounts, device groups in "**Edit Device**"



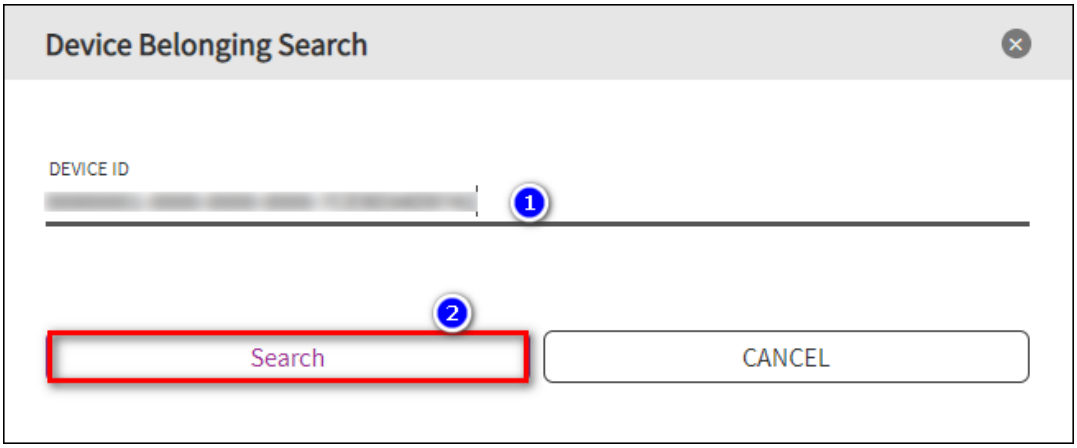
Click the delete icon to display "**Delete**" options on each device list, pick up the checkbox and confirm to delete these devices.



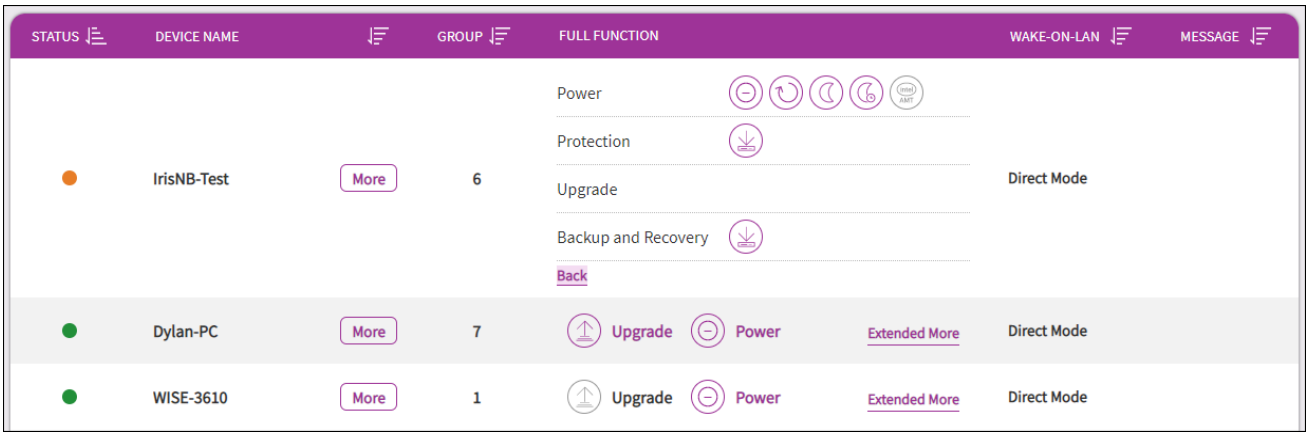
Click on export icon to export devices that in the table as CSV file.

Device Name	Agent ID	WAKE-ON-LAN	Mac	Message	Status
DESKTOP-NRB0J2A	0000[illegible]-00[illegible]-00[illegible]-00[illegible]D30	Not Set	000E[illegible]7E1[illegible]30		Device Online
ac09	000[illegible]-0000-0000-0000-123[illegible]AA[illegible]	Not Set	12[illegible]32[illegible]AD[illegible]D		Device Offline

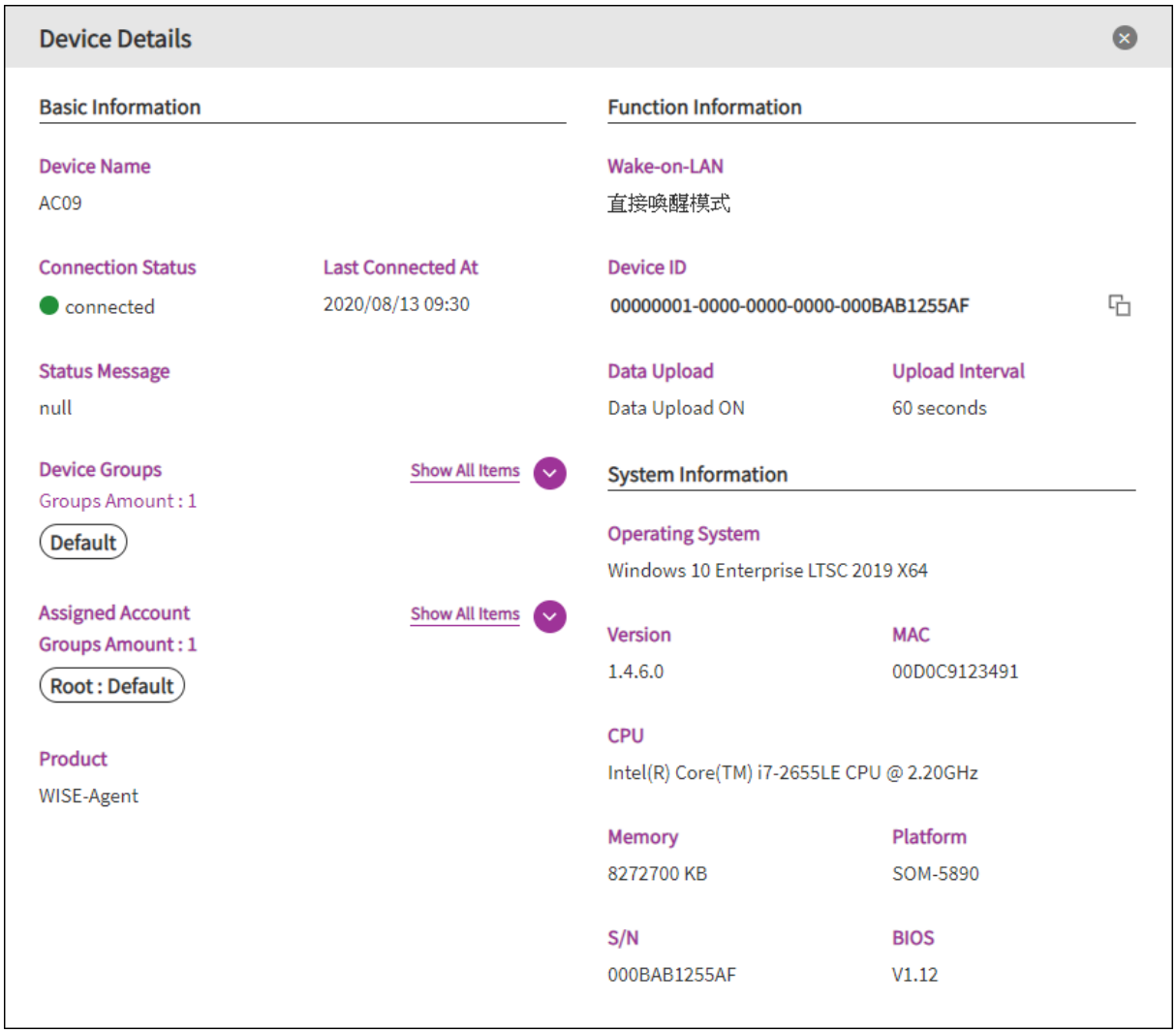
If you would like to know a device be assigned to which account and device group, click search icon to enter Agent ID (from your WISE-Agent UI) to understand.



Actually, you cloud do lots of remote action on the device.

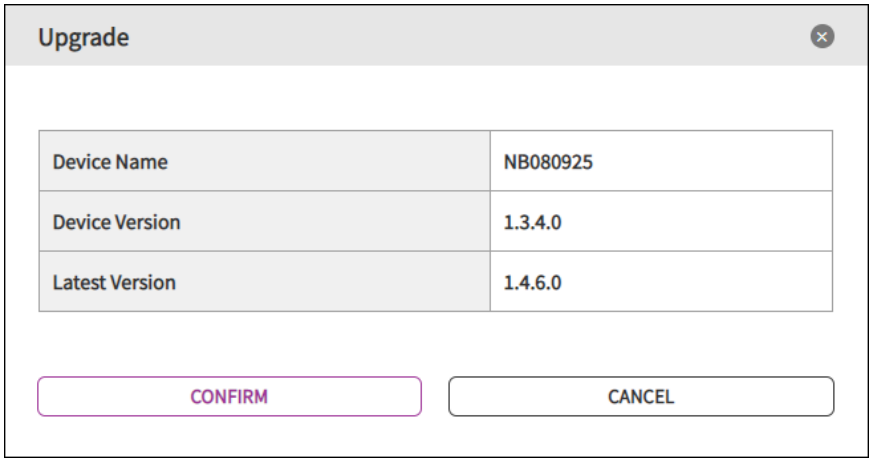


- **Status**: Green light represent device connected, gray for disconnected and orange for device abnormal, due to device over threshold.
- **Device Name**: Device name, click **More** to get more deice information, such as platform, operation system, MAC, memory, etc.



- ✧ **Group**: Number of groups for the devices, for example, the device could belong to multiple group.
- ✧ **Quick Functions**: Including Power Management, System Protection, Upgrade WISE-Agent and System Backup/Recovery.

  The power management supports On/Off, Restart, Sleep and hibernate, the actions depend on your device supported. Second, protection is power-by McAfee white-list protection mechanism to solidify device system. After enable, 3rd execution file, bat, DLL cannot be launch. Please go to **Setting** -> **Provision**-> **Protection** to install first. Next, for the WISE-Agent upgrade, if there is new version released by Advantech, it will check and show the icon automatically.

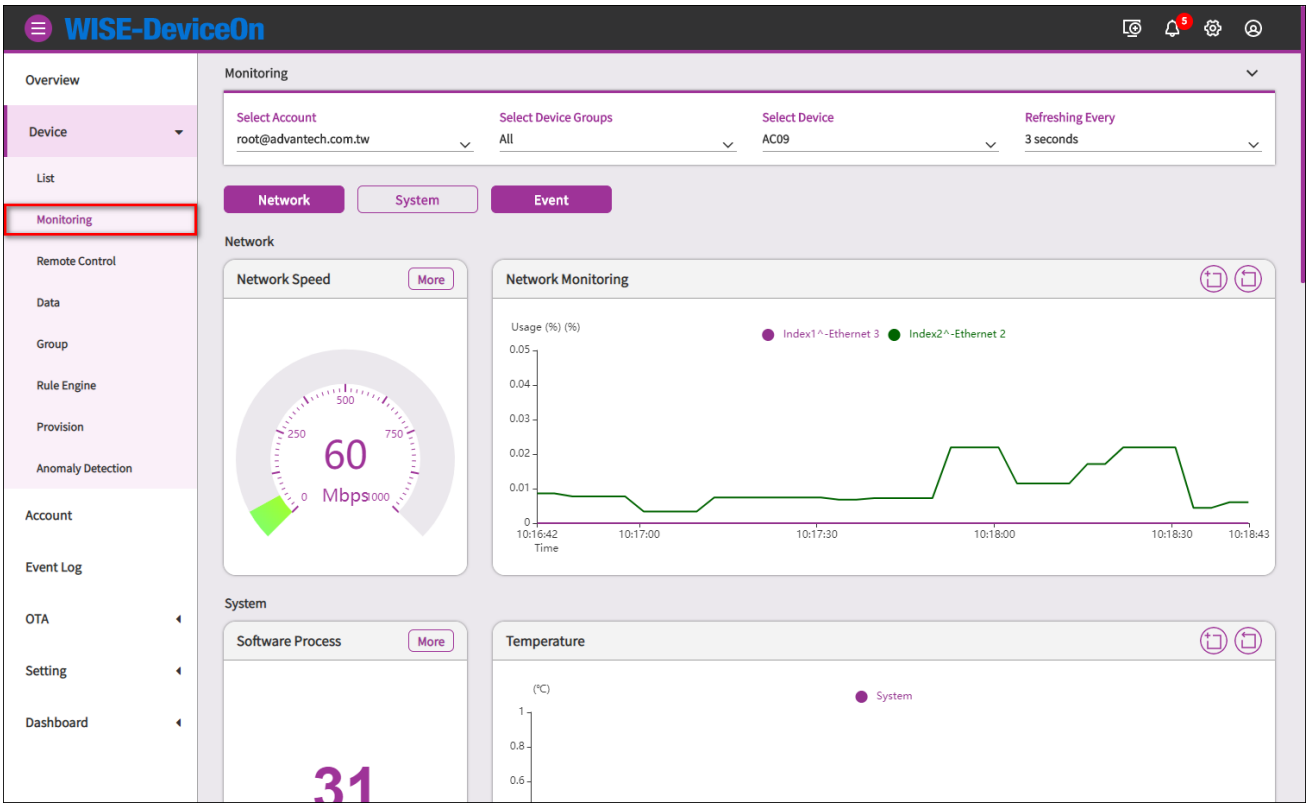


Fourth, System Backup/Recovery is power-by Acronis to backup/recovery device runtime system partition. Please go to **Setting** -> **Provision**-> **Backup/Recovery** to install first.

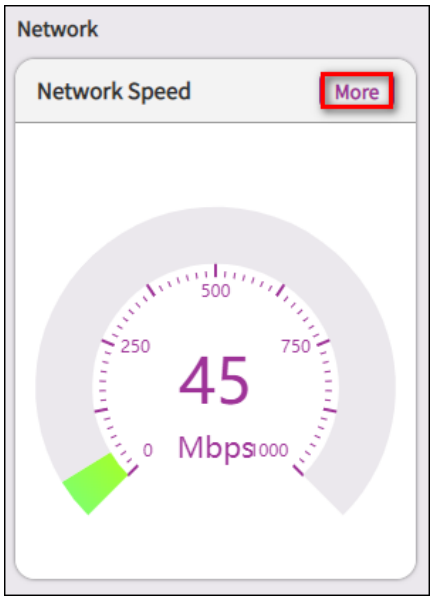
- ✧ **Wake-On-LAN**: Wake-On-LAN mode for device, three mode to power your device up, "**Direct Mode**", "**Agent Mode**" and "**Repeater Mode**". The magic package sent by DeviceOn Server call "Direct Mode", but cannot through different network. Therefore, to overcome this limitation, through another Agent or Router to send, forward magic packet. Please go to **Setting** -> **Provision**-> **Power On** to configure.
- ✧ **Message:** Device current status

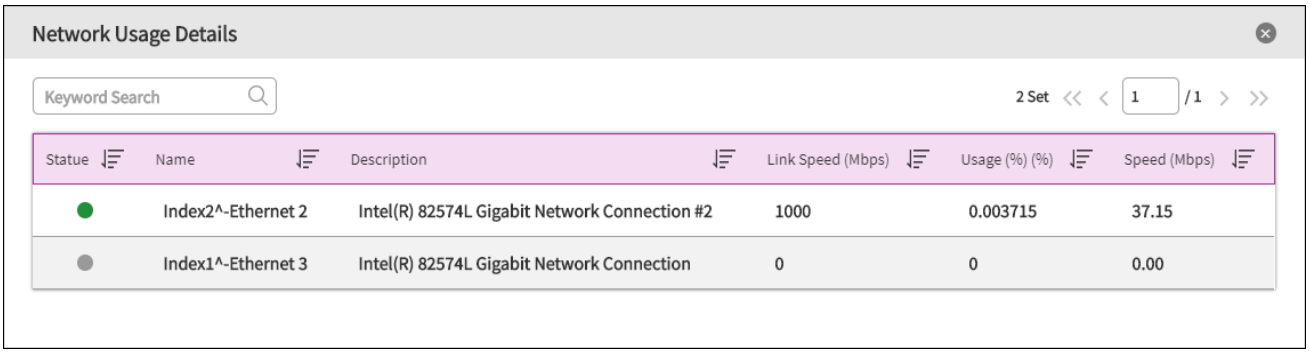
- Device Monitoring

On this page, you could get real-time information about the device that you selected. The information includes general PC status, such as network speed, software process, disk healthy, CPU and memory usage. If the device is Advantech industrial PC and SUSI driver supported, the RPM (Revolution(s) Per Minute) of CPU FAN, system, board level voltage, temperature is displayed on the page.

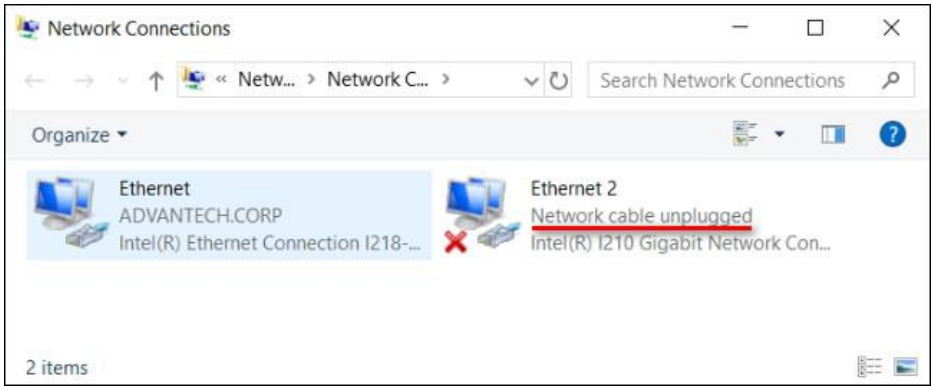


Some of devices support multiple network cards, especial industrial PC. Click on the network button to retrieve others.

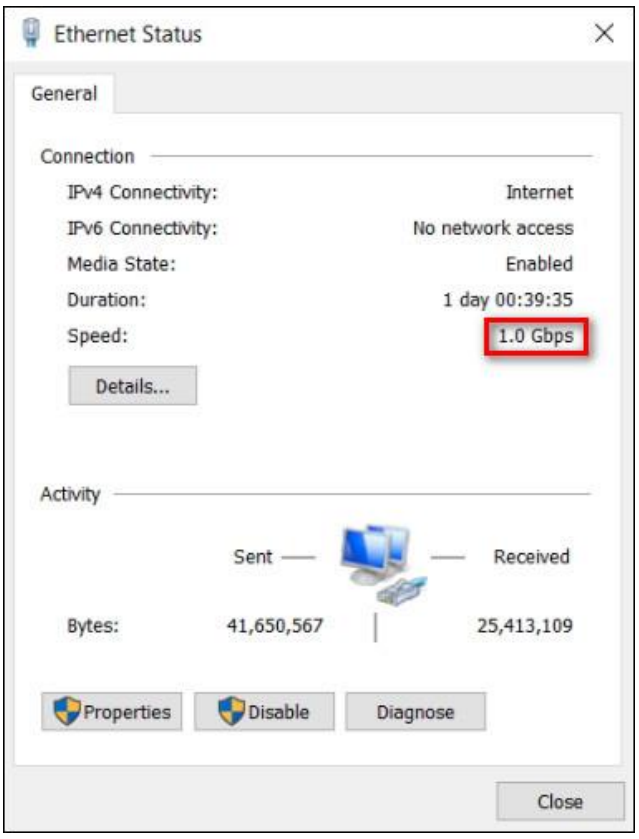




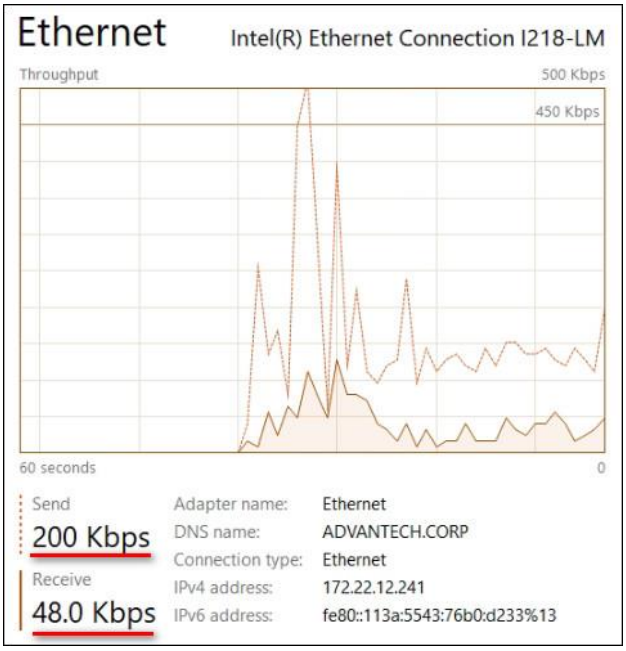
- **Name**: Name of network card
- **Description**: network description
- **State**: Network connected or disconnected, for example, ethernet cable plugin or not.



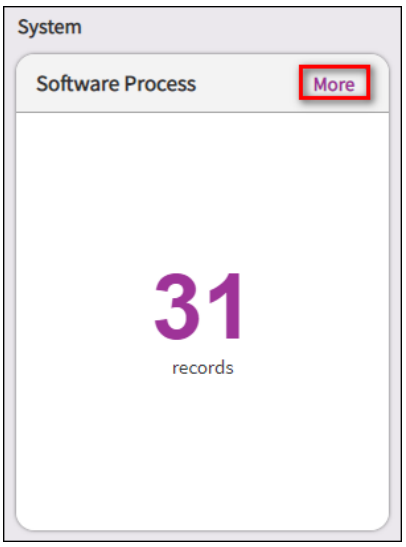
- **Link Speed (MBPS):** Network maximum link speed



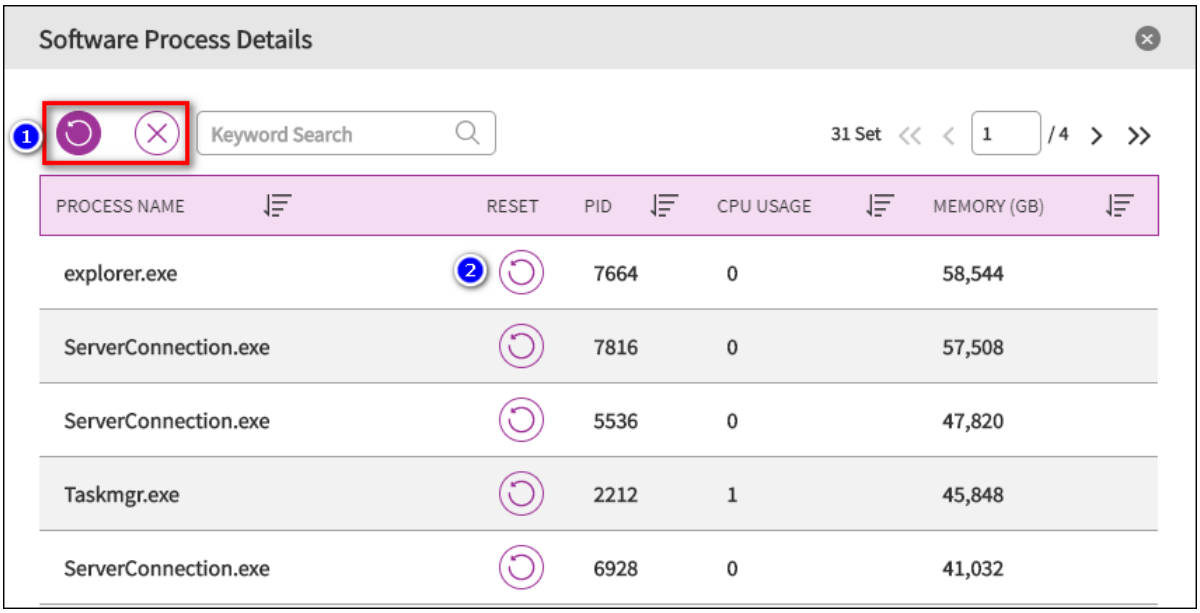
- ✧ **Usage:** Network current usage, **Speed/Link Speed**.
- ✧ **Speed (MBPS):** Send plus receive data rate.



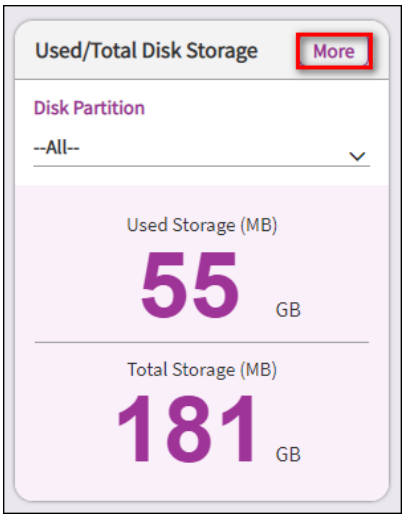
Click on **Software Process** to show **current user process** list, if your device system not login, the result might be zero.

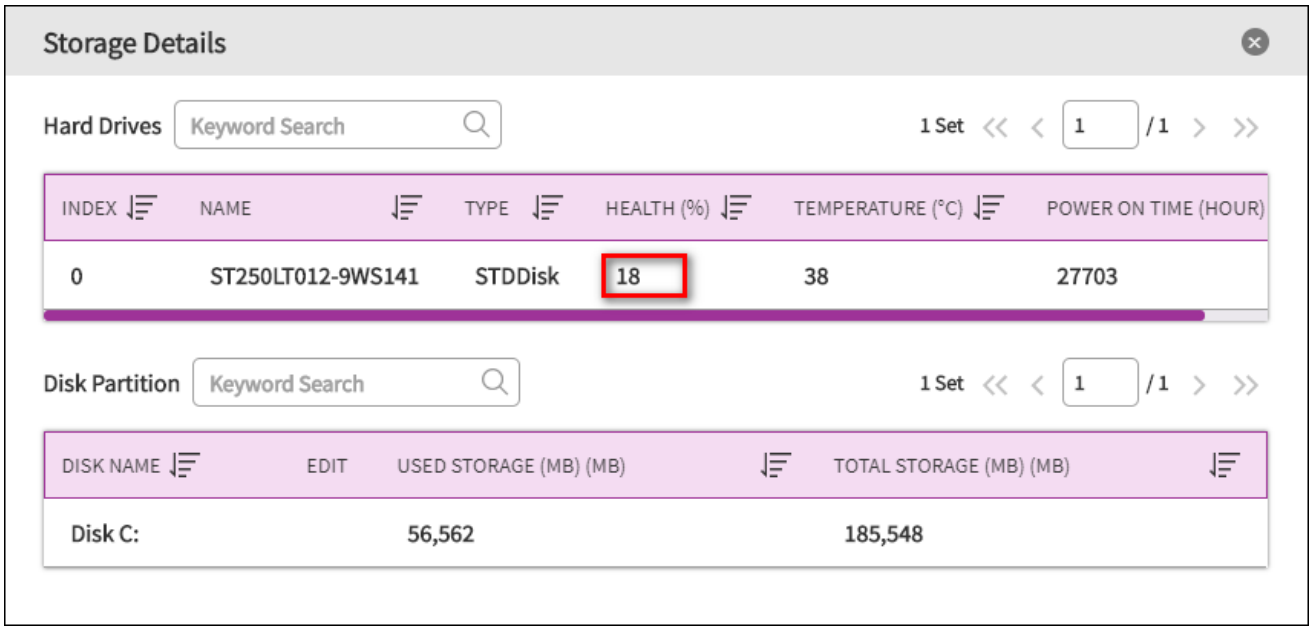


Click on the reset or delete icon, you could restart, terminal your specific process.



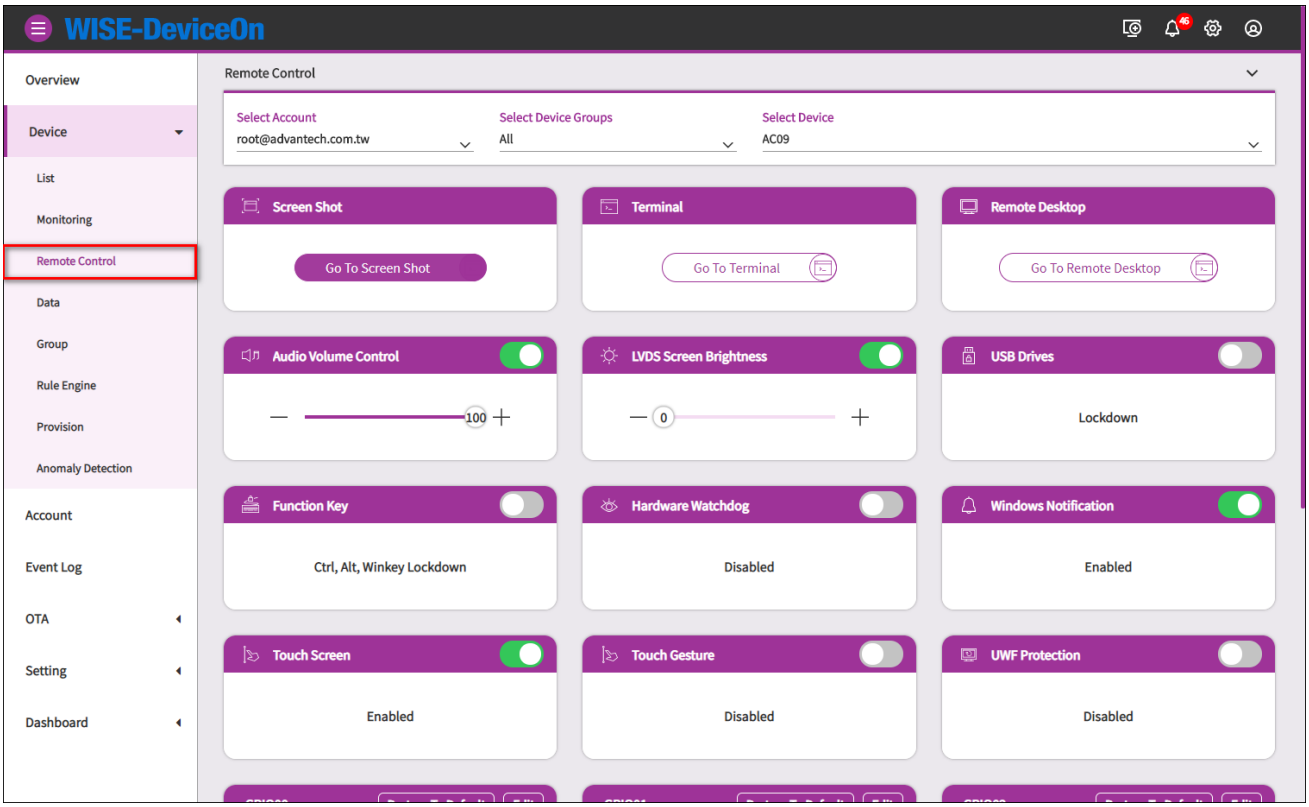
For hard drive status, not only include current **Used Storage**, but **Healthy** and **Power on Time**. The healthy is based on Acronis healthy model, that calculate on edge side, if you are interested, refer to the official page.





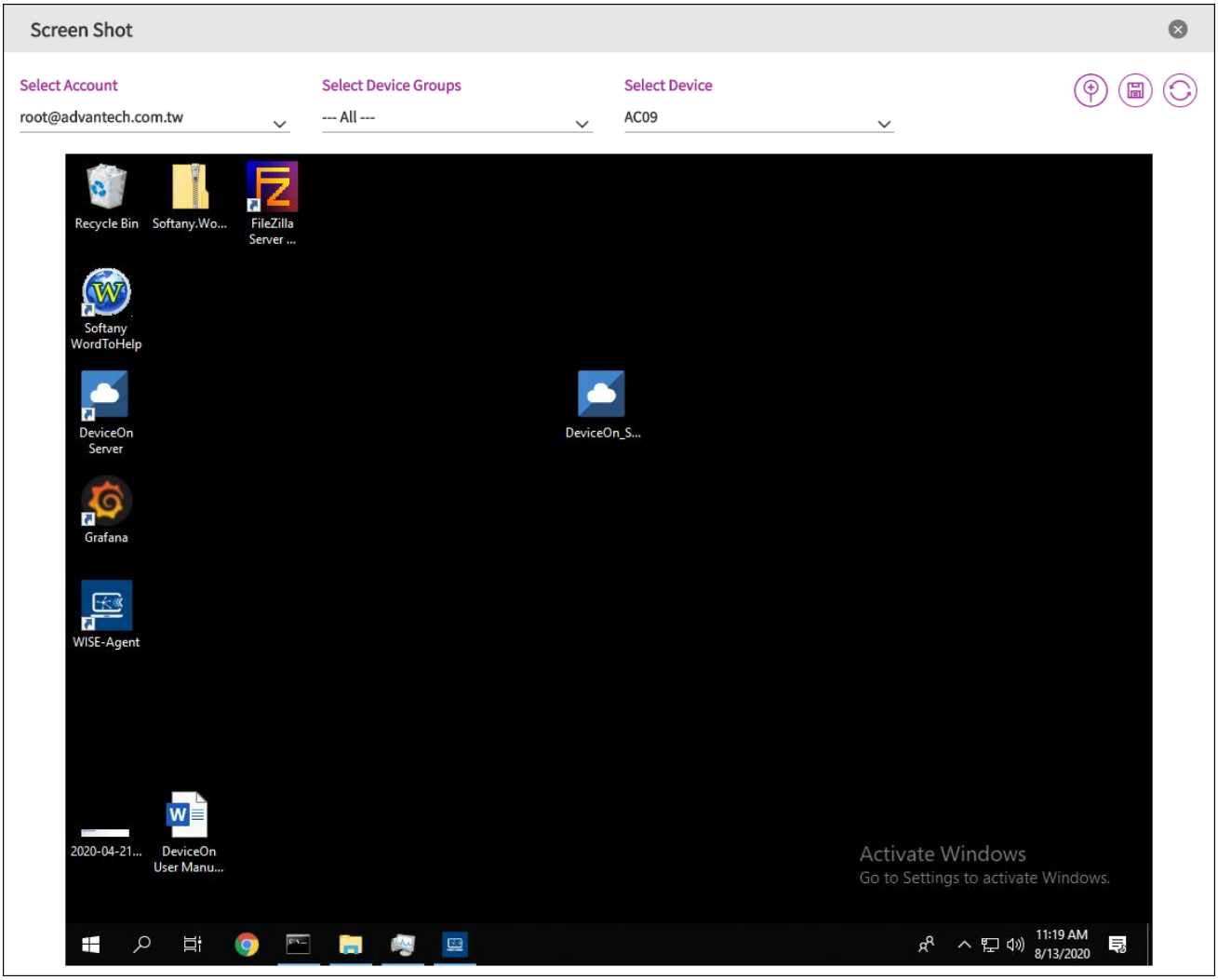
- Remote Control

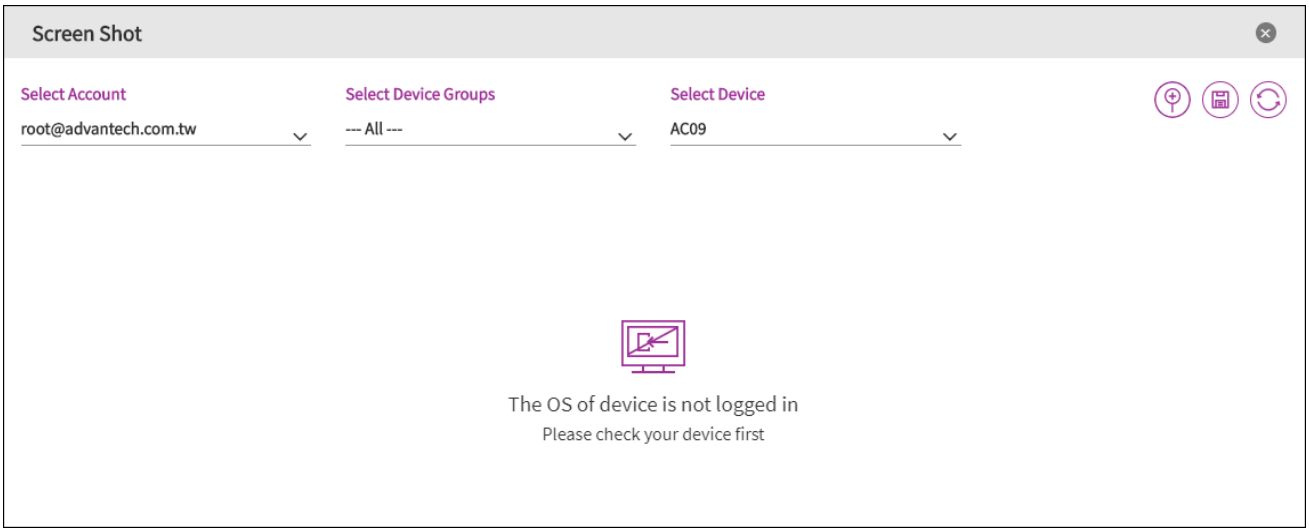
If you need to debug, diagnostic to your devices, actually, do not need go to field side. Through DeviceOn remote control to manage to reduce your operation effort. Basically, there are three functions (**Screenshot**, **Terminal** and **Remote Desktop**) for most devices.



[Screenshot]

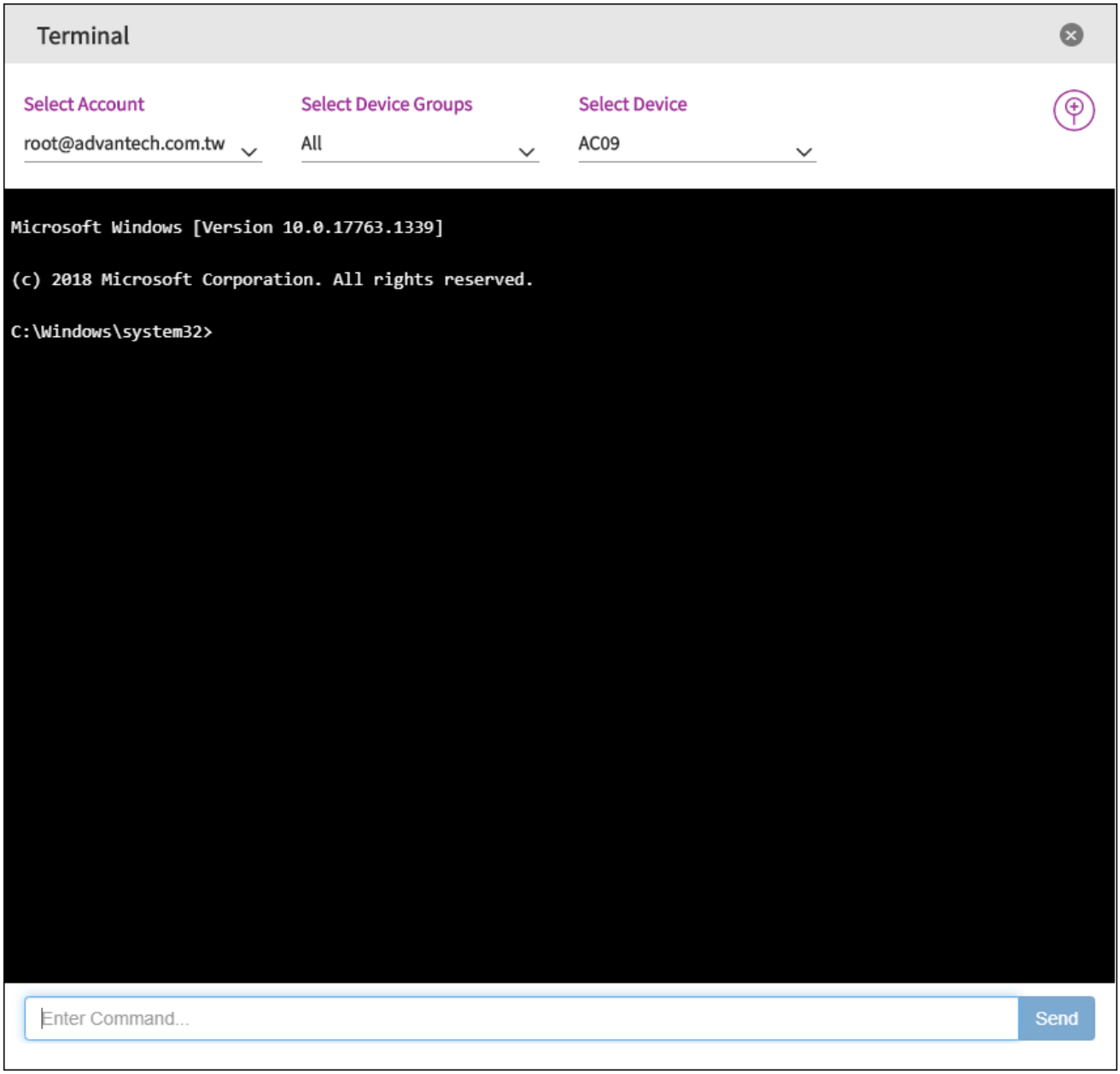
Through the Screenshot to get device real-time screen, there is a limitation, your device **must login to operation system**, otherwise, cannot capture screen and shown "**The OS of device is not logged in**"





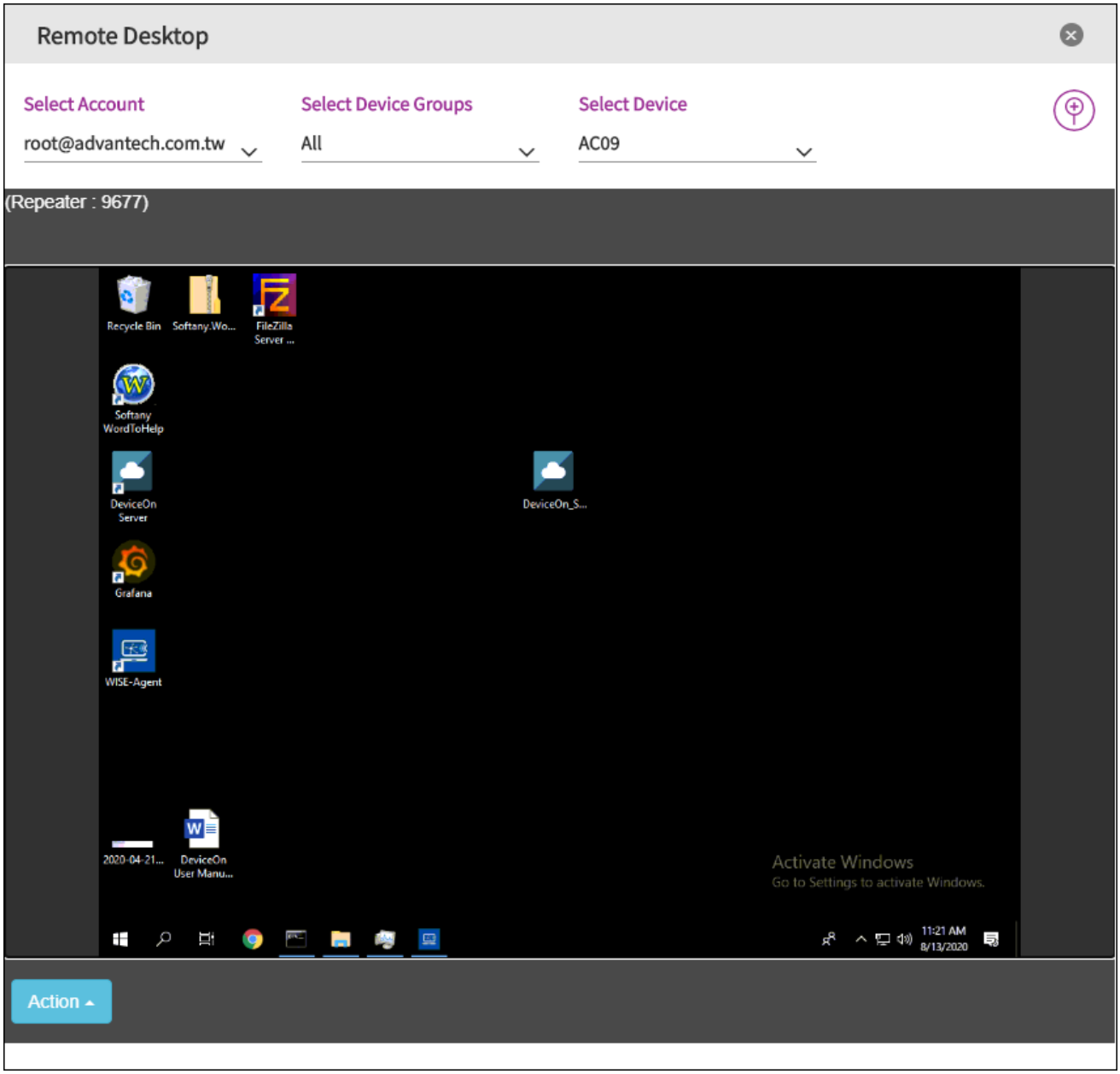
[Terminal]

To terminal support any command to your devices, for instance, realize your device IP, traceroute the network or copy/view file on the device.



[Remote Desktop]

DeviceOn leverage VNC (Virtual Network Computing) technology to achieve remote desktop, to bridge different network between public and private. User do not need to install any program, App on their laptop or mobile devices. Through DeviceOn website to remote desktop to debug and diagnostic.



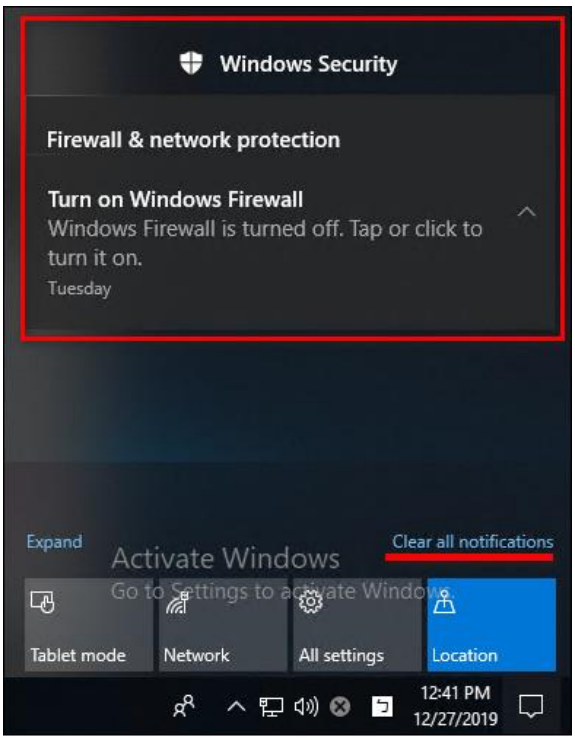
[Advanced Control]

For others features depend on your device operation system and hardware. DeviceOn integrate Windows Lockdown features on LTSC (Long Time Service Channel) and LTSB (Long Time Service Branch) to provide advanced control, such as "**Block USB Drives**", "**Keyboard Filter**", "**Block Windows Notification**", "**Block Touch, Gesture**" and "**UWF (Unified Write Filter)**".

[USB Drive]: Prevent threats from outside **USB drives**, not include keyboard, mouse.

[Function Key]: Disables **Ctrl**, **Alt**, and **WinKey**.

[Windows Notification]: Block application notification.



[Touch Screen]: Disable touch control

[Tough Gesture]: Disable gesture control

[UWF Protection]: To protect your drives by intercepting and redirecting any writes to the drive (app installations, settings changes, saved data) to a virtual overlay. The virtual overlay is a temporary location that is usually cleared during a reboot or when a guest user logs off.

Benefits:

- Provides a clean experience for thin clients and workspaces that have frequent guests, like school, library or hotel computers. Guests can work, change settings, and install software. After the device reboots, the next guest receives a clean experience.
- Increases security and reliability for kiosks, IoT-embedded devices, or other devices where new apps are not expected to be frequently added.
- Can be used to reduce wear on solid-state drives and other write-sensitive media.

For **backlight**, **brightness, GPIO** and **Watchdog** only support on Advantech hardware platform with SUSI driver, please download from Advantech Support site.

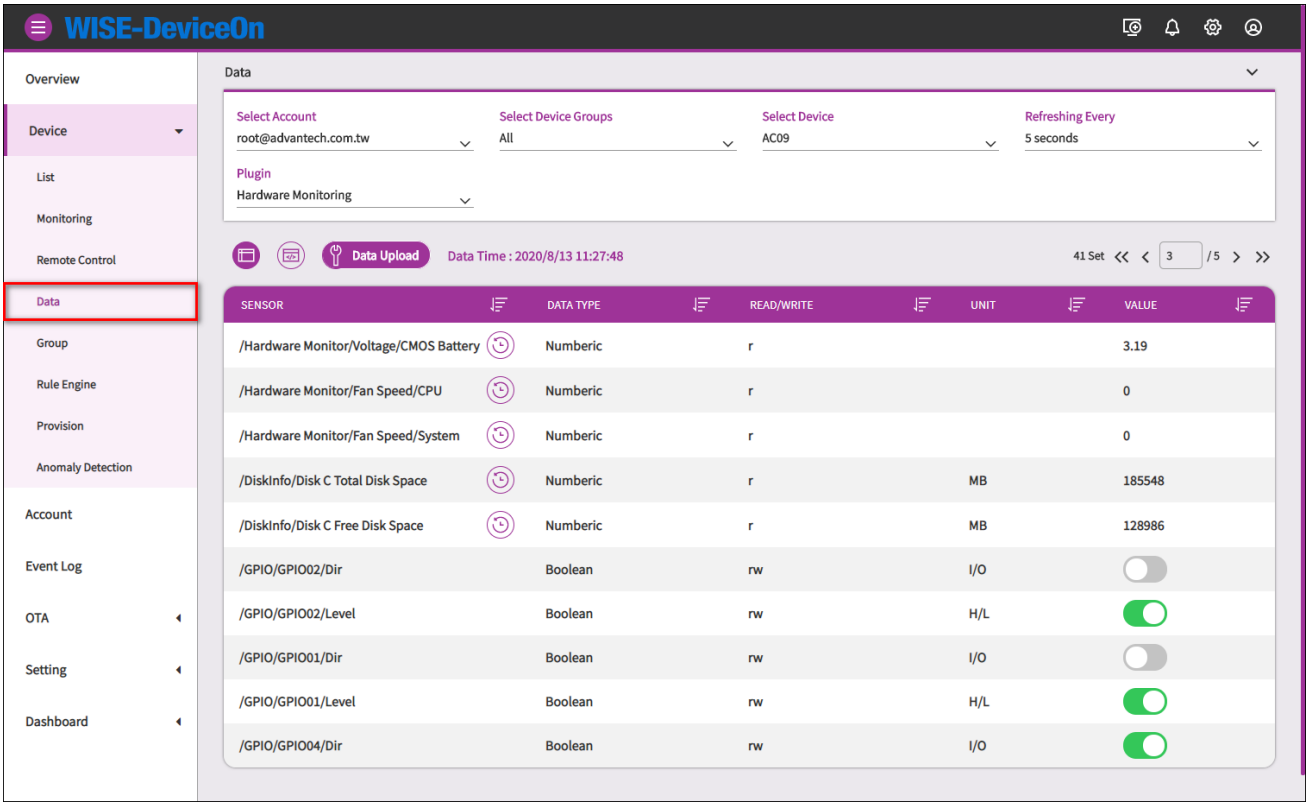
[LVDS, Backlight and Brightness]: Turn on/off LVDS backlight for power saving.

[Watchdog Protection]: Hardware level watchdog to prevent BSoD (Blue Screen of Death) or system hang without any response. If happened, watchdog will restart your device automatically. There is an tool call NotMyFault that you can use to crash, hang, and cause kernel memory leaks on your Windows system.

Benefits: Avoid embarrassing moment, if BSoD on your Signage devices over the airport, department store and public area.

- Device Data

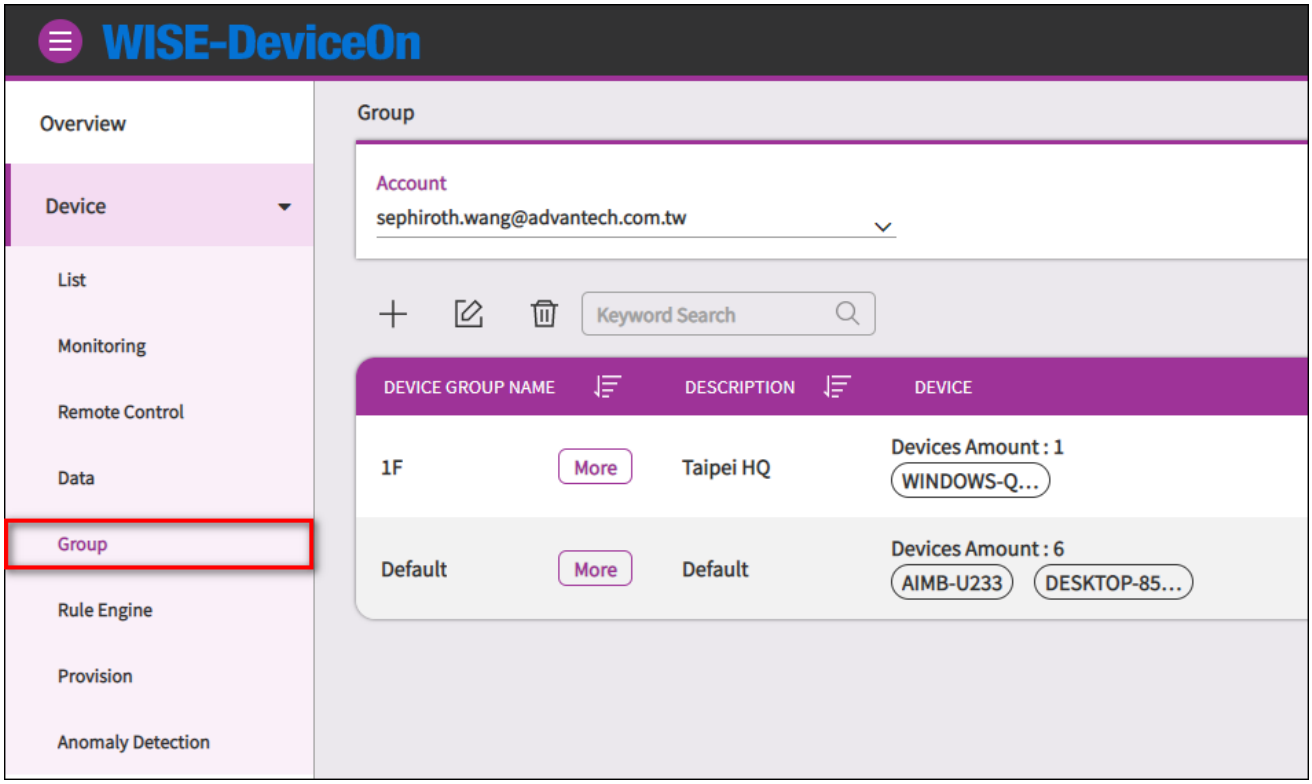
Raw data of each plugin on devices, user could get real-time and historical data on this page. To data analysis and aggregation, user could adjust data report interval or reset to default (60s) for basic sensors.



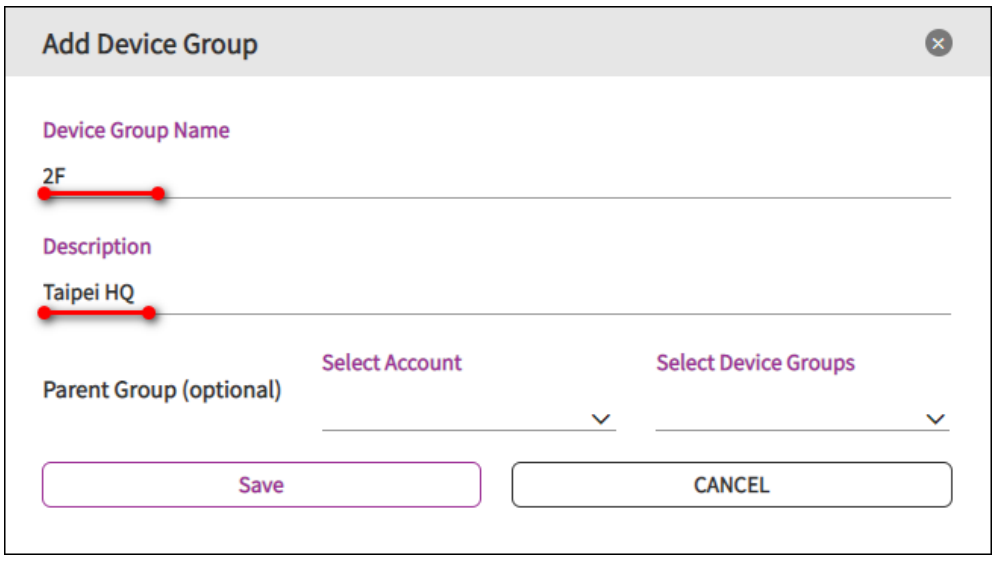
The default display is table mode, you could switch to JSON format through the icon.

- Device Group

Every account could group their device into different groups to manage, for example, device over different floor on the building. User could create 1F, 2F group to easy management.



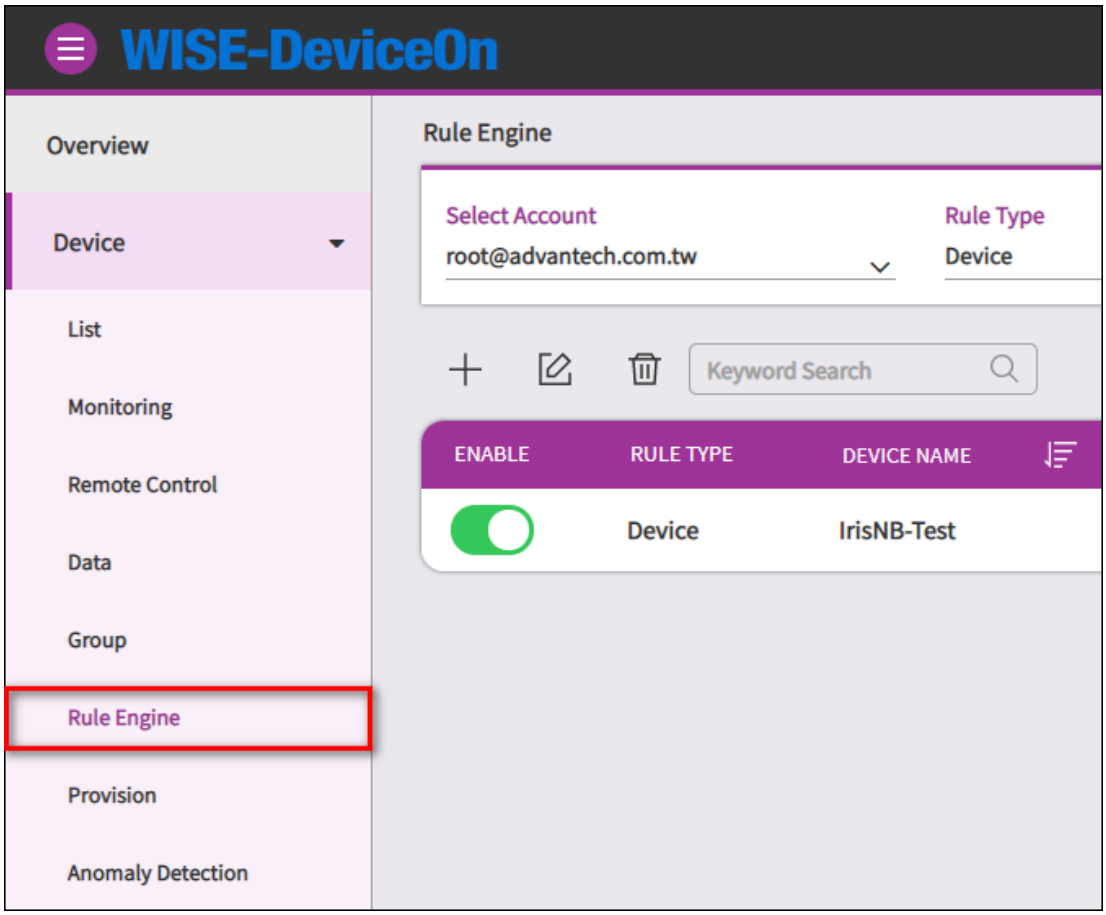
Click on the icon to add "**Device Group**". The option to configure the parent group, that's means share the device group to parent group owner.



Click on the icon to “**Edit**” or “**Delete**” account.

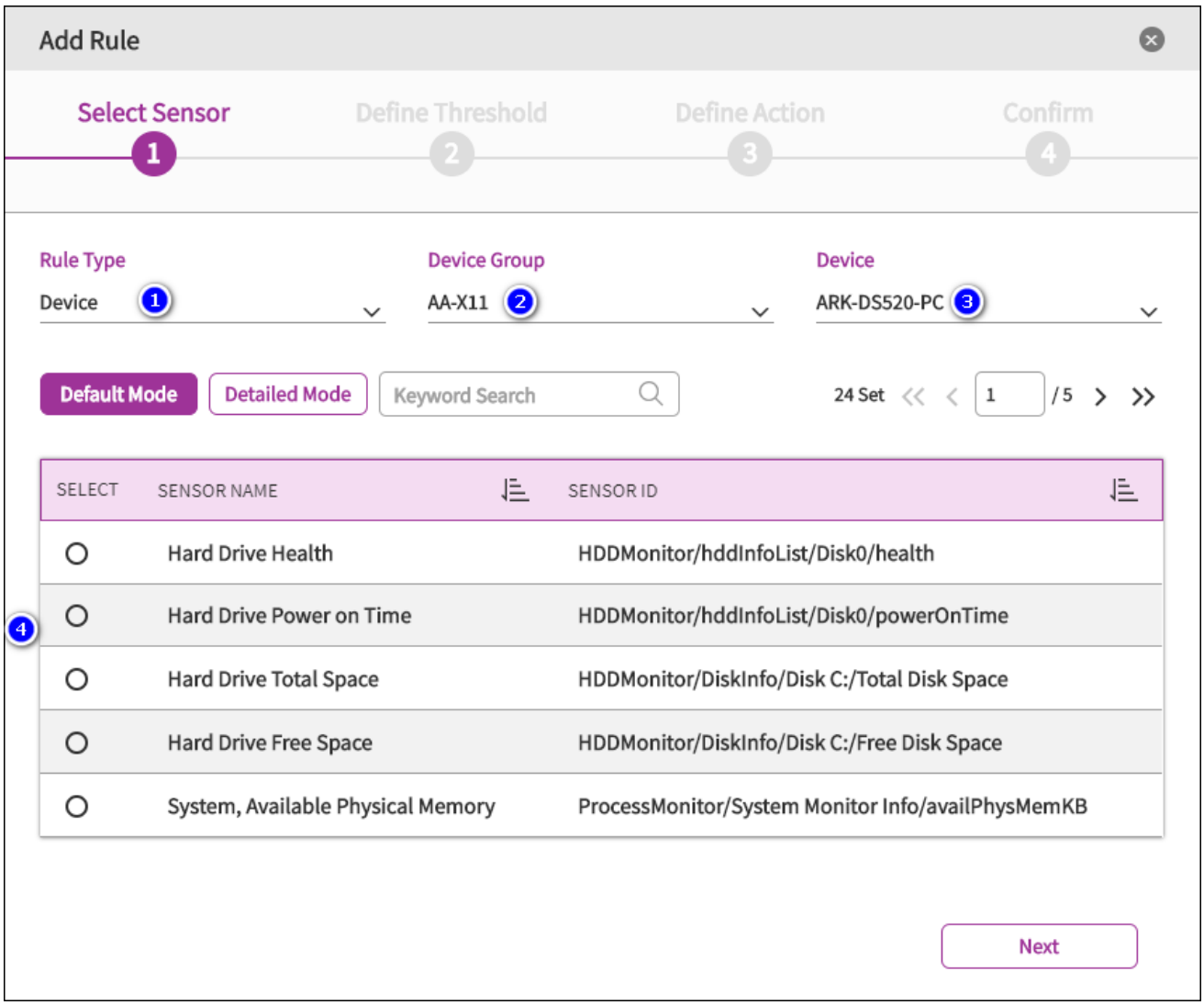
- Rule Engine

DeviceOn provides the rule engine. Users can acquire anomaly situations by means of setting thresholds to those interested devices, and, once one or more thresholds meets, receive alerts via event notification services, another one indispensable feature for users.



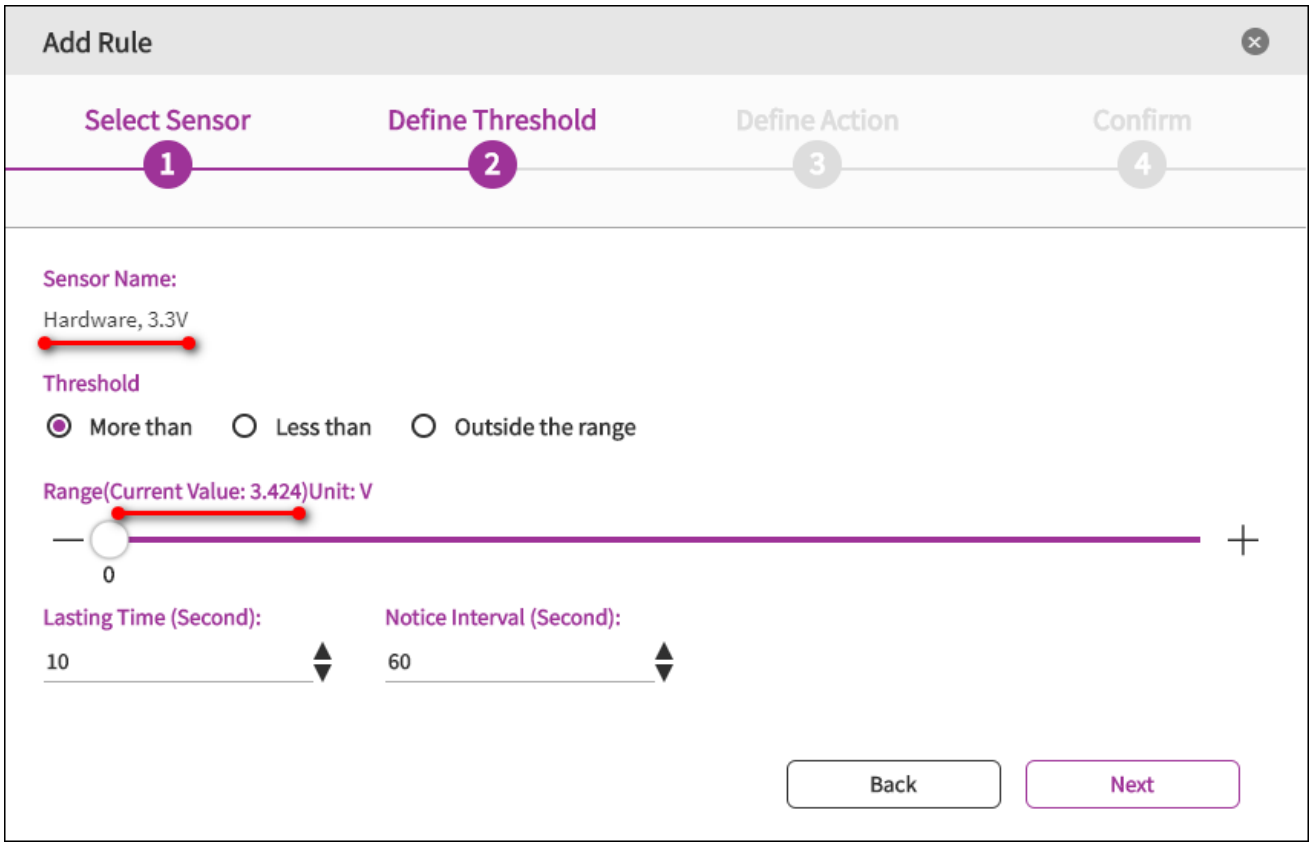
Click on the add icon to create a Rule.

Pick-up the sensor that you want to monitor, the steps are select **Rule Type, Device Group** and **Device**.

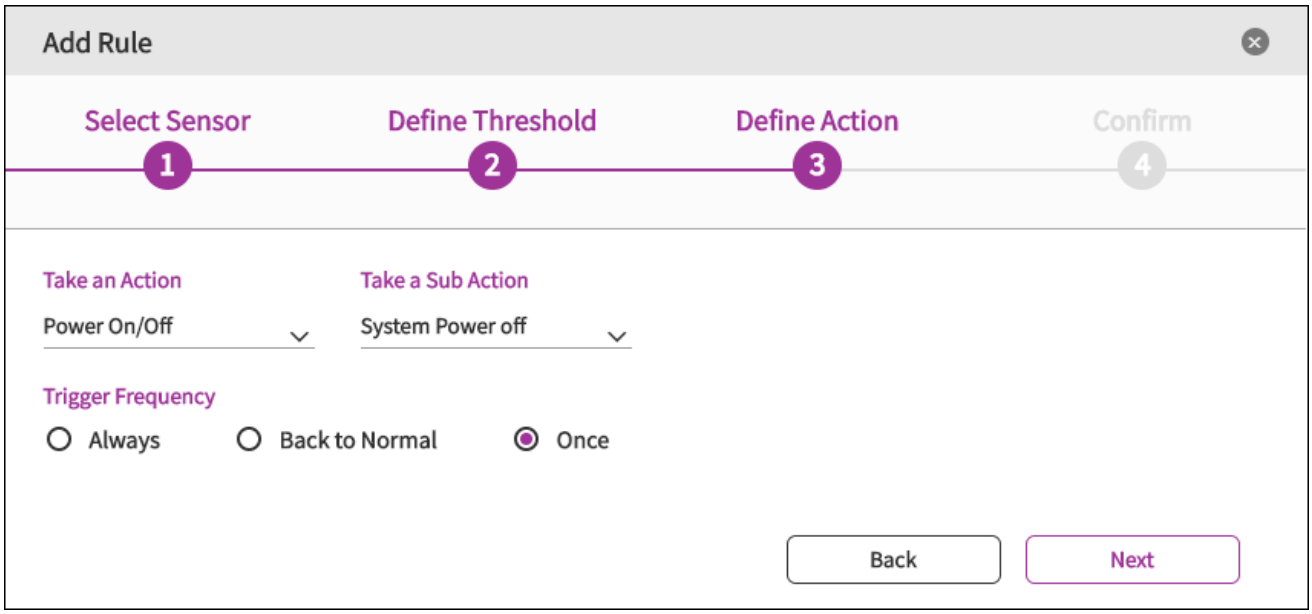


Define the threshold, provide 3 types, **more than**, **less than** and **outside the range**. Also, you could realize current value on the page.

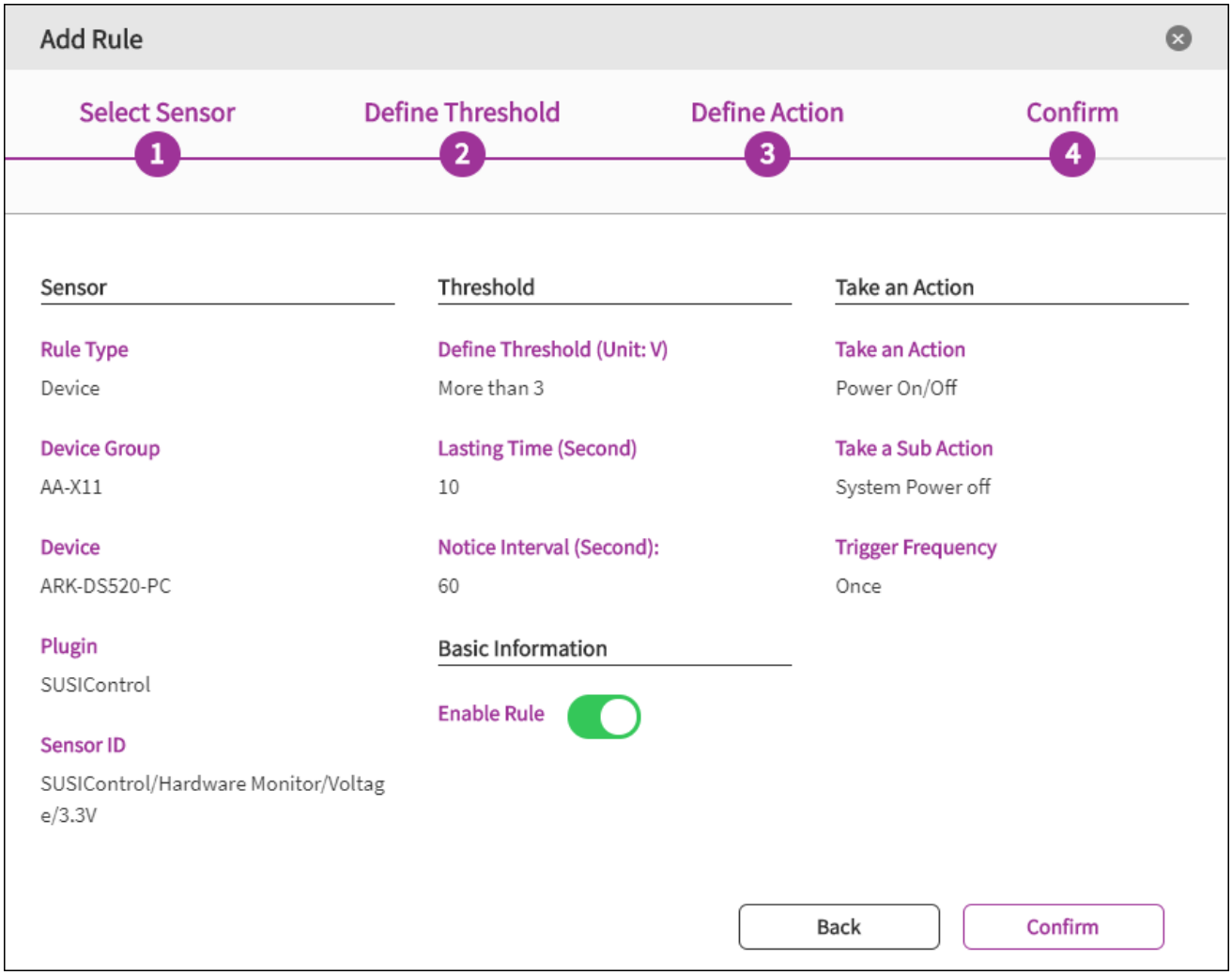
- ✧ Lasting Time (Second): means the sensor over the threshold and continue for a period time, avoid peak value to trigger.
- ✧ Notice Interval (Second): If over the threshold, the WISE-Agent will send a notify event, to avoid lots of message, user could adjust notice interval.



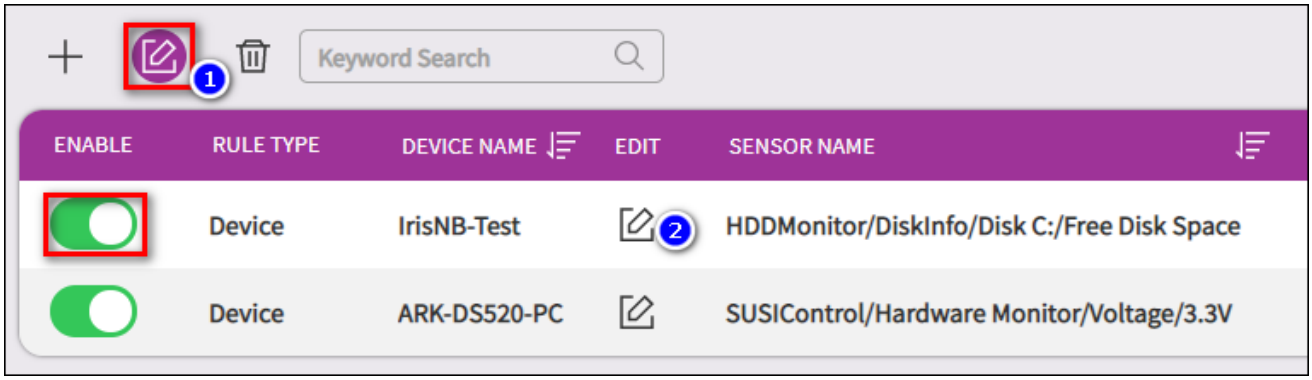
Next, to define the action, if threshold reached. For example, you could power your device off, if the hard drive unhealthy.



Confirm the rule setting and click confirm.



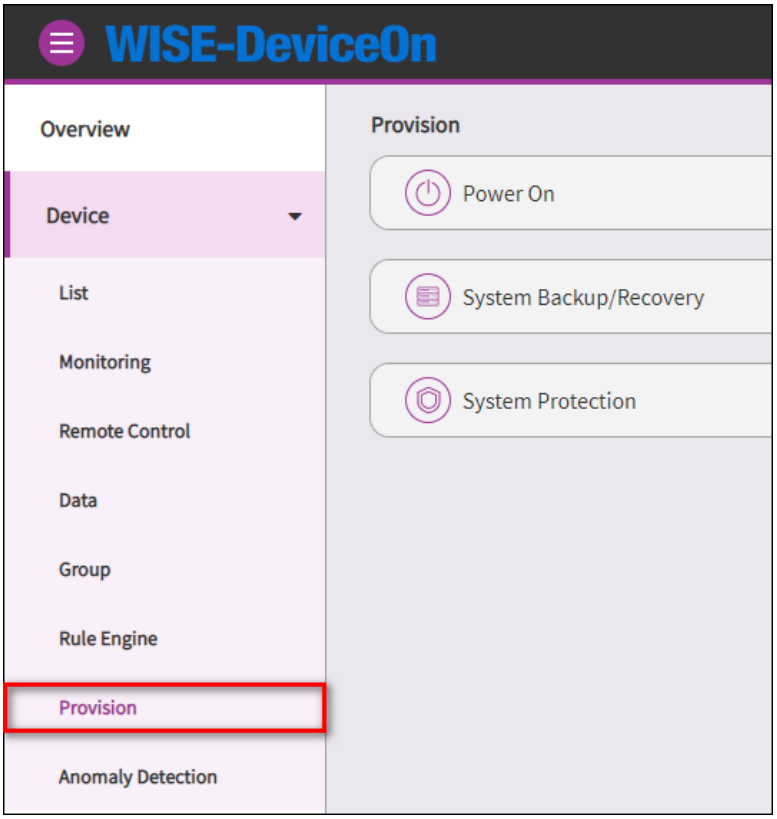
The rule list shown as below, user could edit or disable through the switch.



- Provision

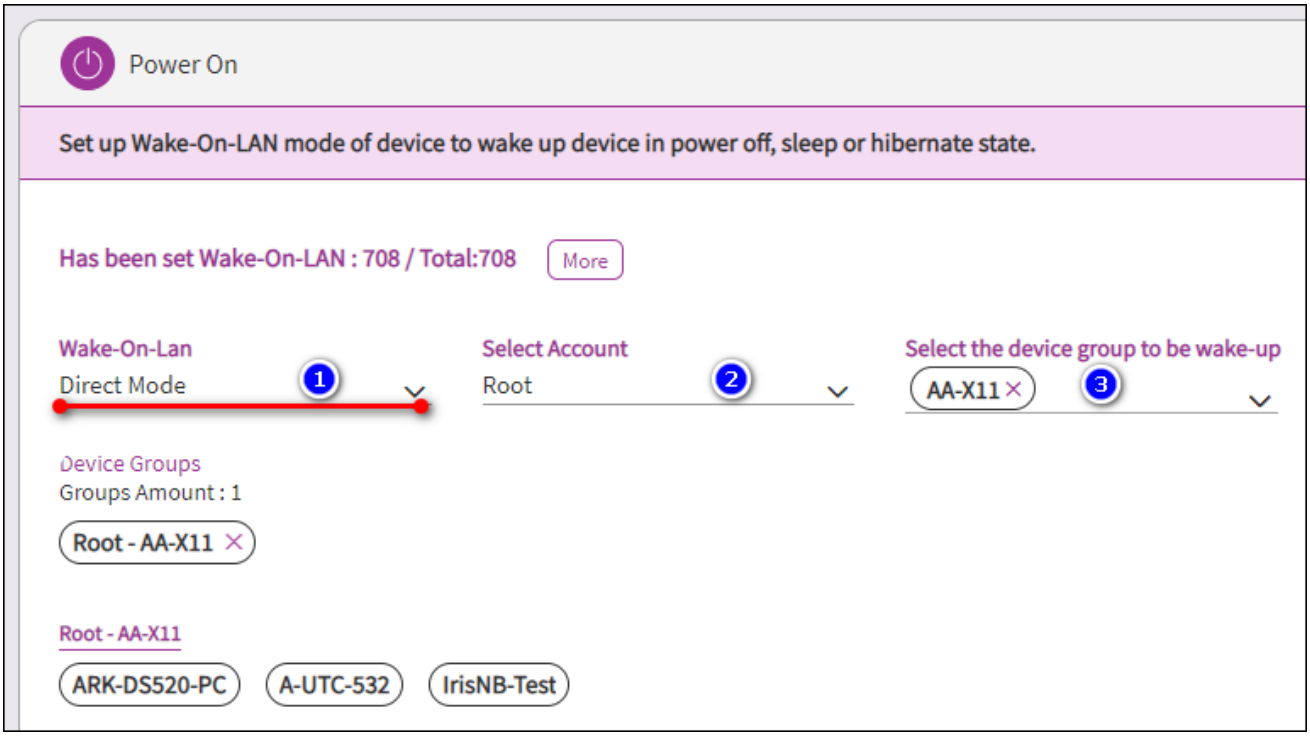
For device provision, 3 types need be pre-configured. One is "**Power On**", select which mode to enable device wake up. The others are 3rd party tool integration, **Acronis** to **backup/recovery** your device system and **McAfee** for white-list security **protection**. To install 3rd tools, you must purchase

the license and activate the product.

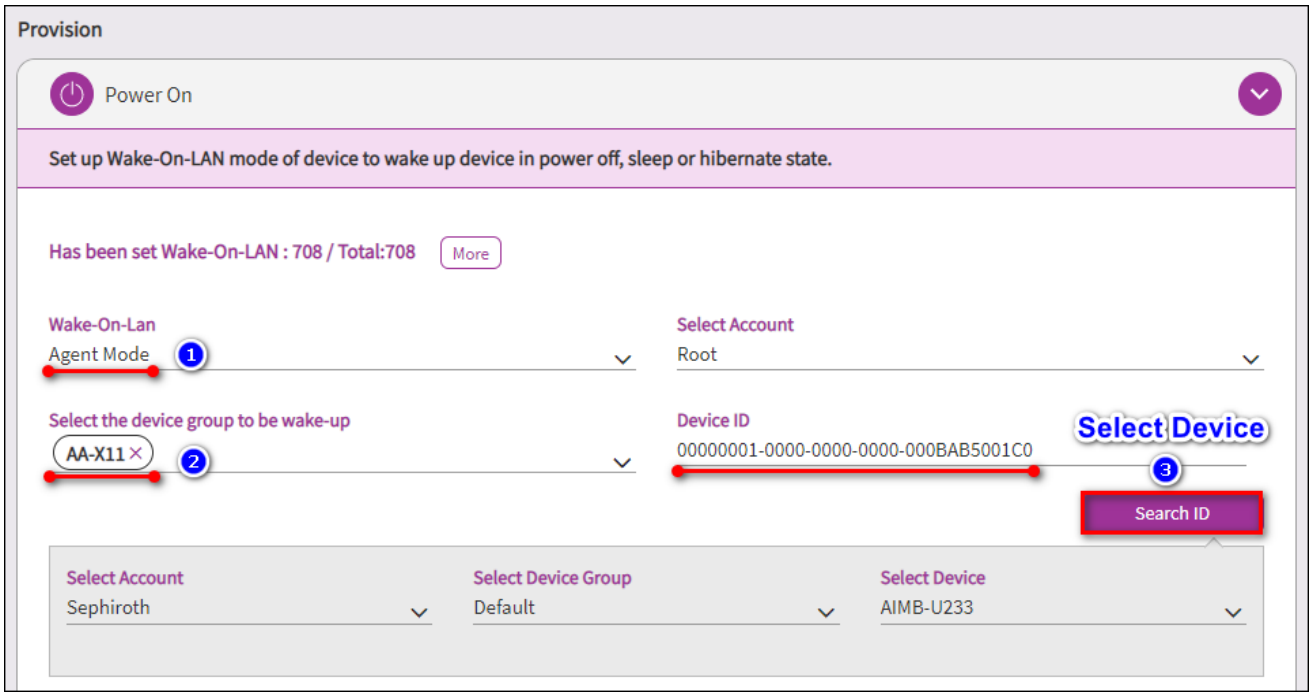


✧ Power On

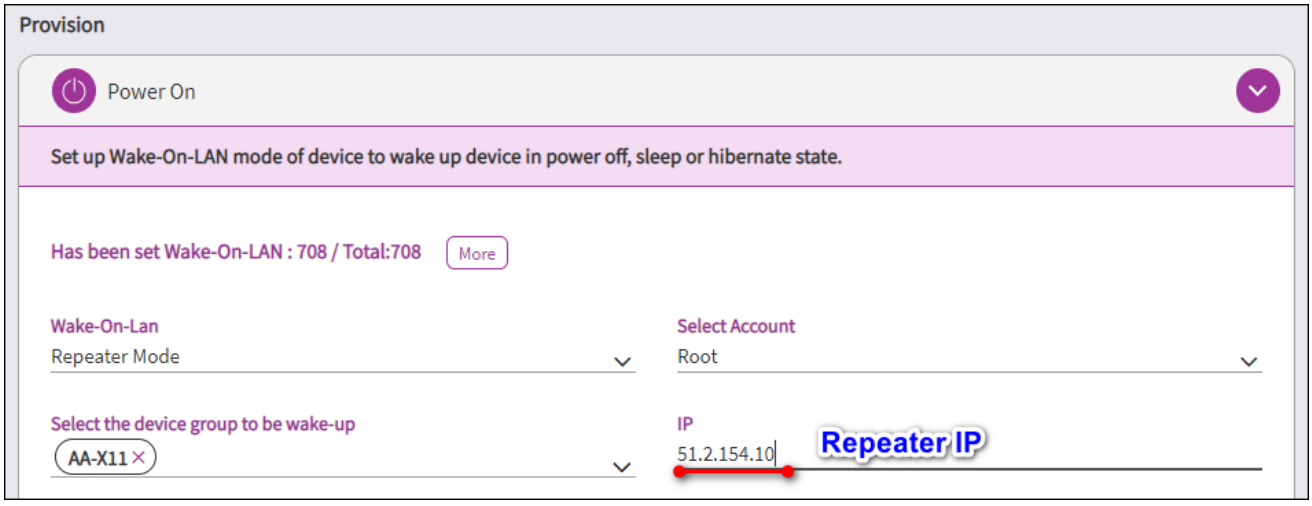
To power your device up, you might to configure the mode for your device. The mechanism is based on Wake-on-LAN to send magic packet to your device. There is a limitation on "**Direct Mode**", the DeviceOn server and edge device must be on the same network.



However, through the "**Agent Mode**" or "**Repeater**" could overcome the limitation. You need to pick-up a device that **always on** and on the same network with other devices.

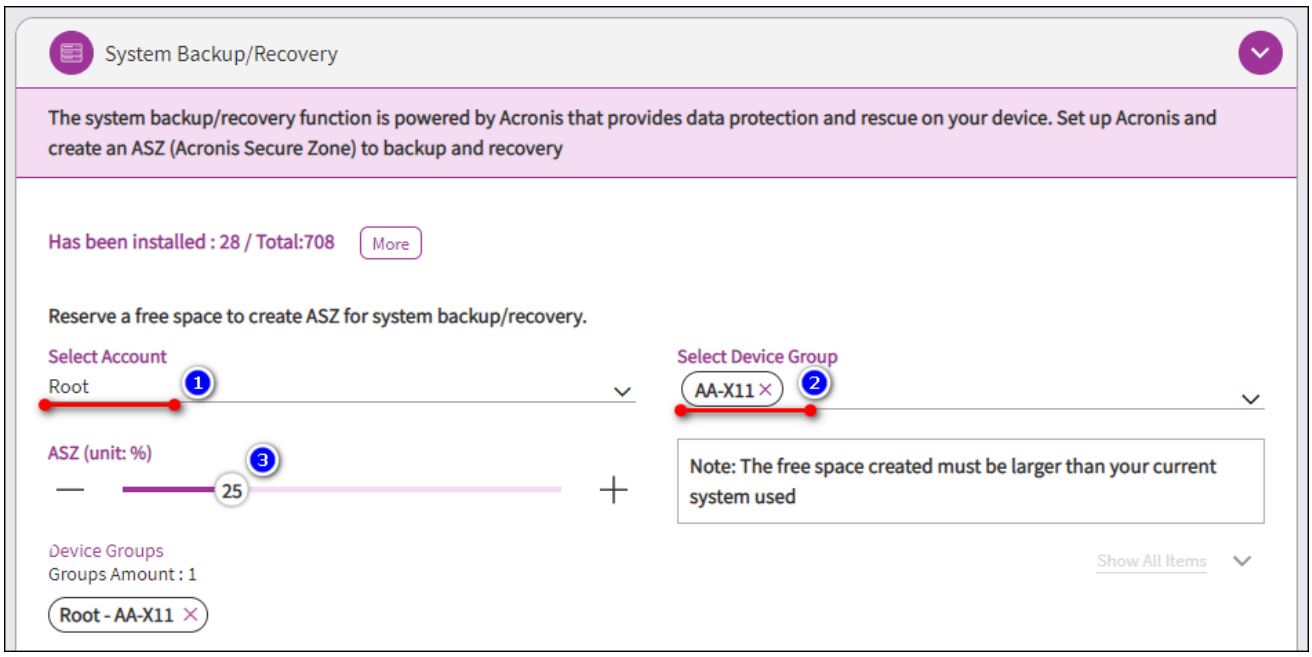


For **Repeater** mode, not only enter your repeater IP, but set your repeater to allow port forwarding (uses UDP port 7 and 9) and permit the packet to be broadcast to the entire LAN.



✧ System Backup/Recovery

Select the free space size to create Acronis Secure Zone (Hidden Partition) to backup system partition. The free space size must larger than system used.



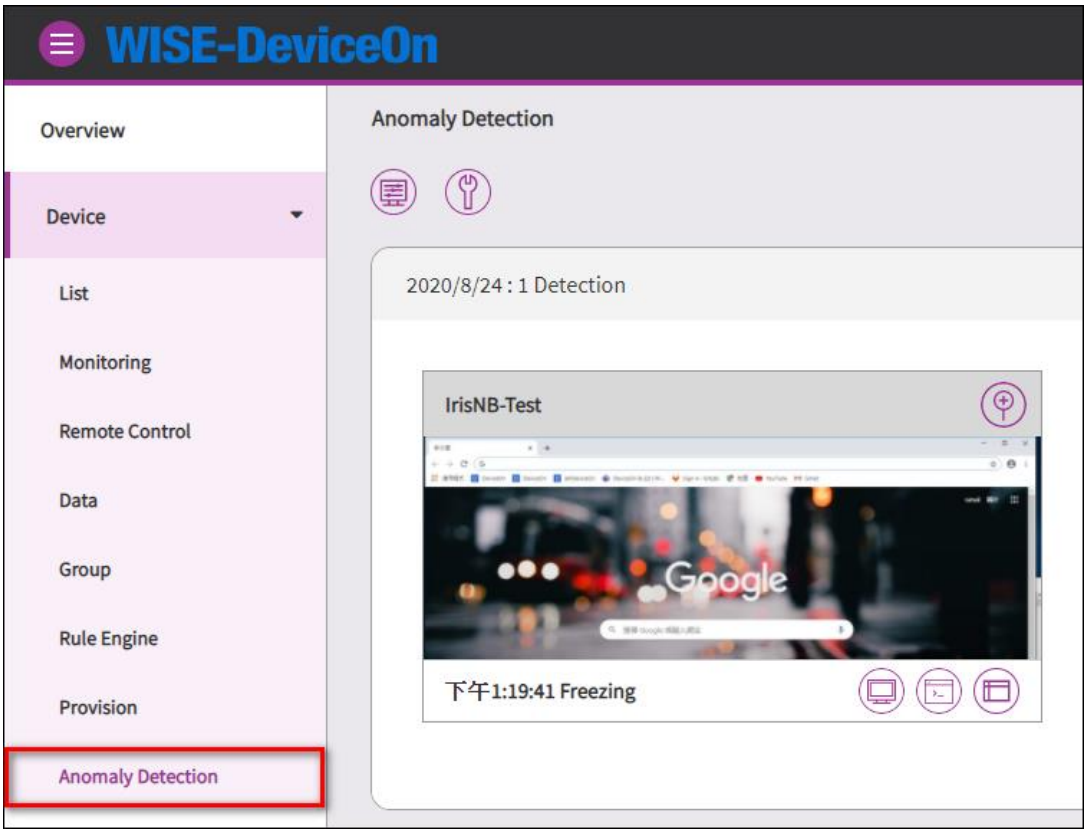
✧ System Protection

Select the device group to install.

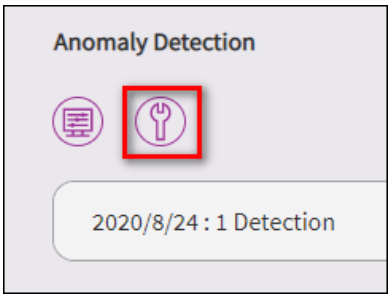
- Anomaly Detection

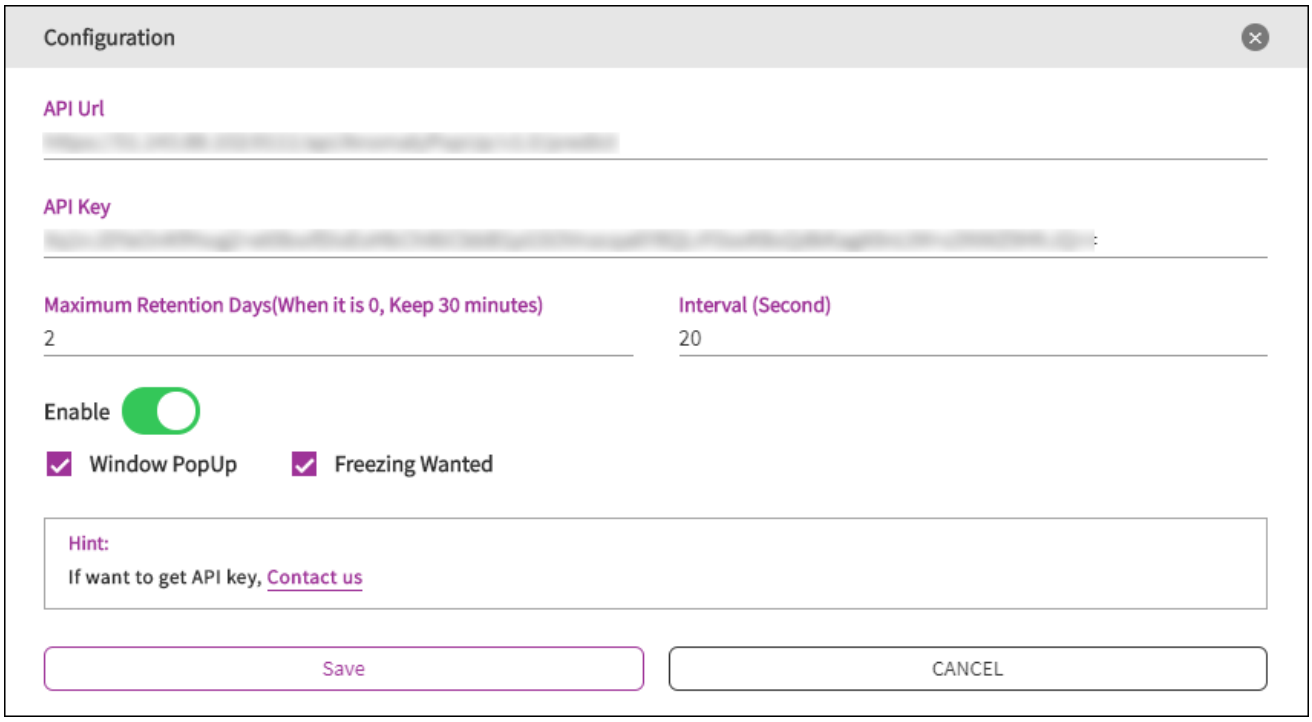
DeviceOn ADS is one of the service that combines anomaly detection algorithm and DeviceOn function. It not only gives customers the high accuracy identification of the error message when advertising is getting interruption but also provides the IoT device remote monitoring and management. Leverage with Azure Custom Vision to continually train the algorithm in order to overcome various errors pop-up under real field.

Before to detect your anomaly screen, please make sure your device is logged in to capture screen status.



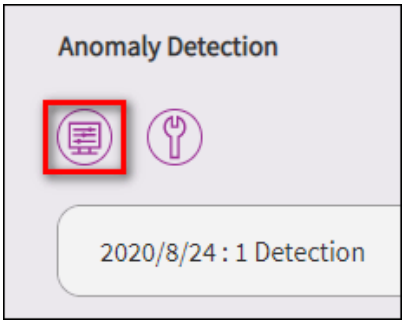
Click on the configuration icon to enter API URL and Key.

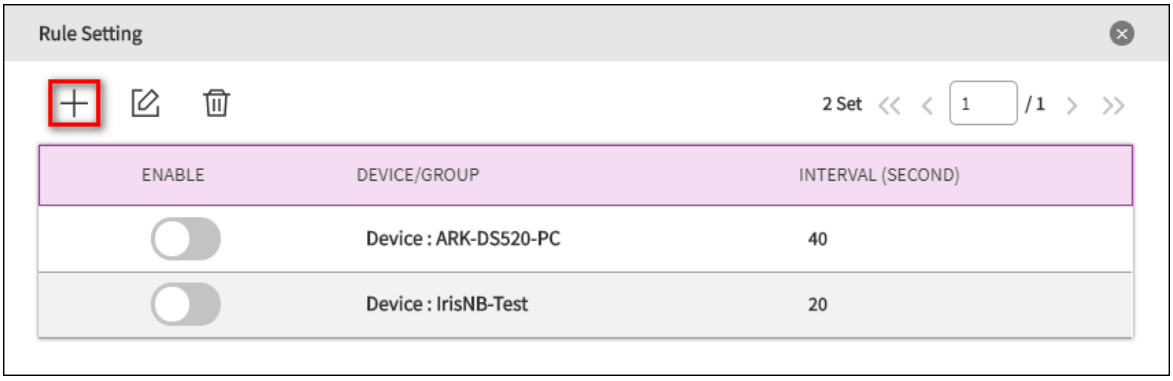




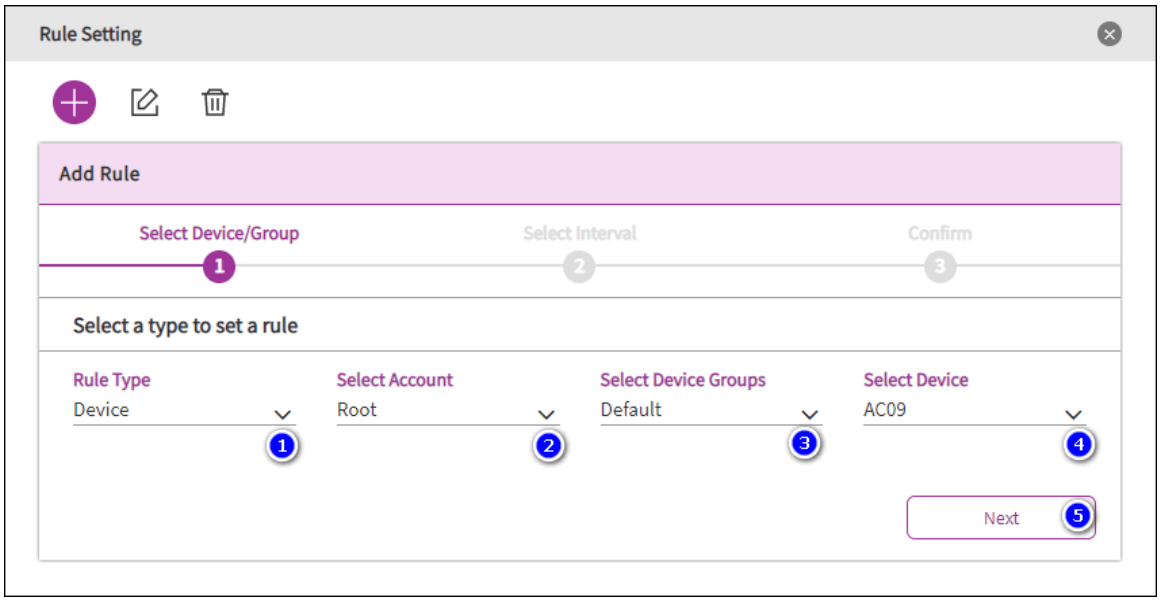
- ✧ API URL & Key: Please contact us for AI machine URL and Key, otherwise, deploy total package from Azure Marketplace.
- ✧ Maximum Retention Days: Maximum retention days for the warning images.
- ✧ Interval: Minimal interval to detect devices screen.
- ✧ Window Popup: Enable to detect popup window on the devices.
- ✧ Freezing Wanted: Enable to detect freezing window on the devices.

Click on the rule icon to enable detection rules.

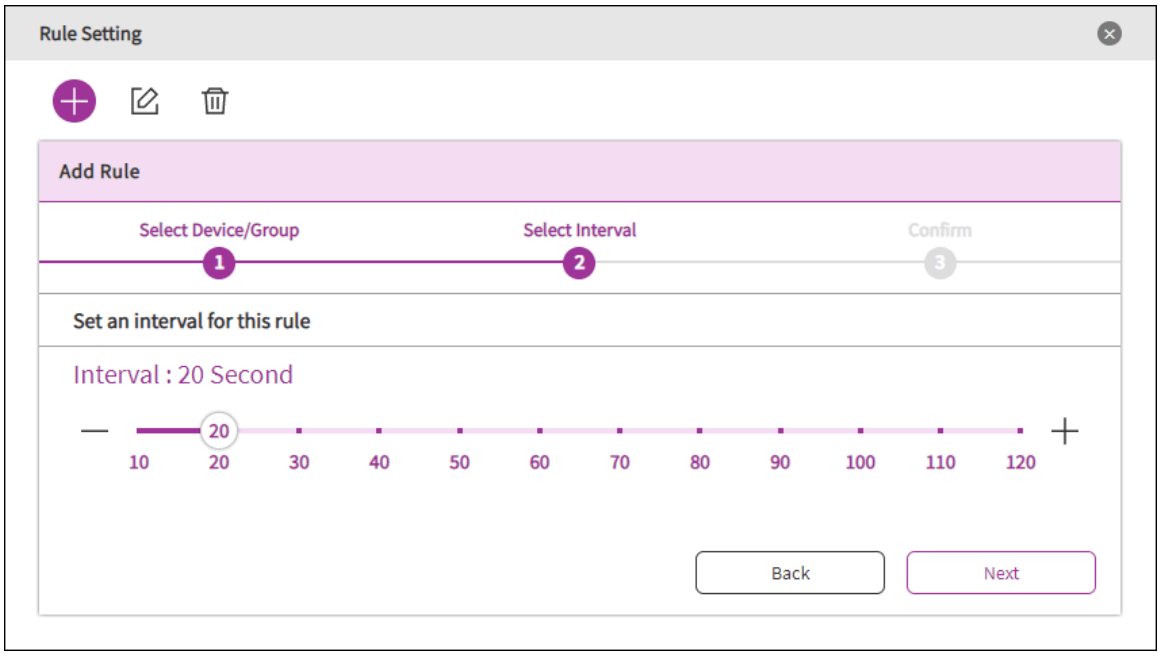




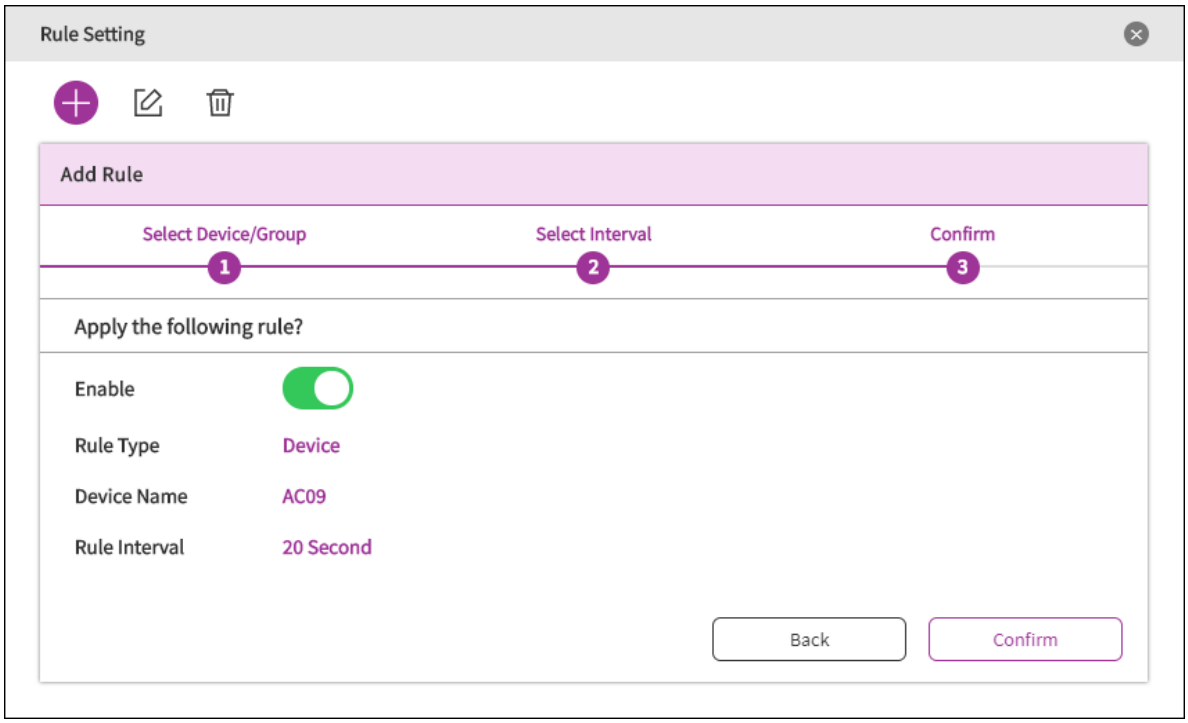
Click on the “+” icon to create anomaly rule, select to your device or device group



Set the detection interval of the rule, the interval cannot be less than the configuration.

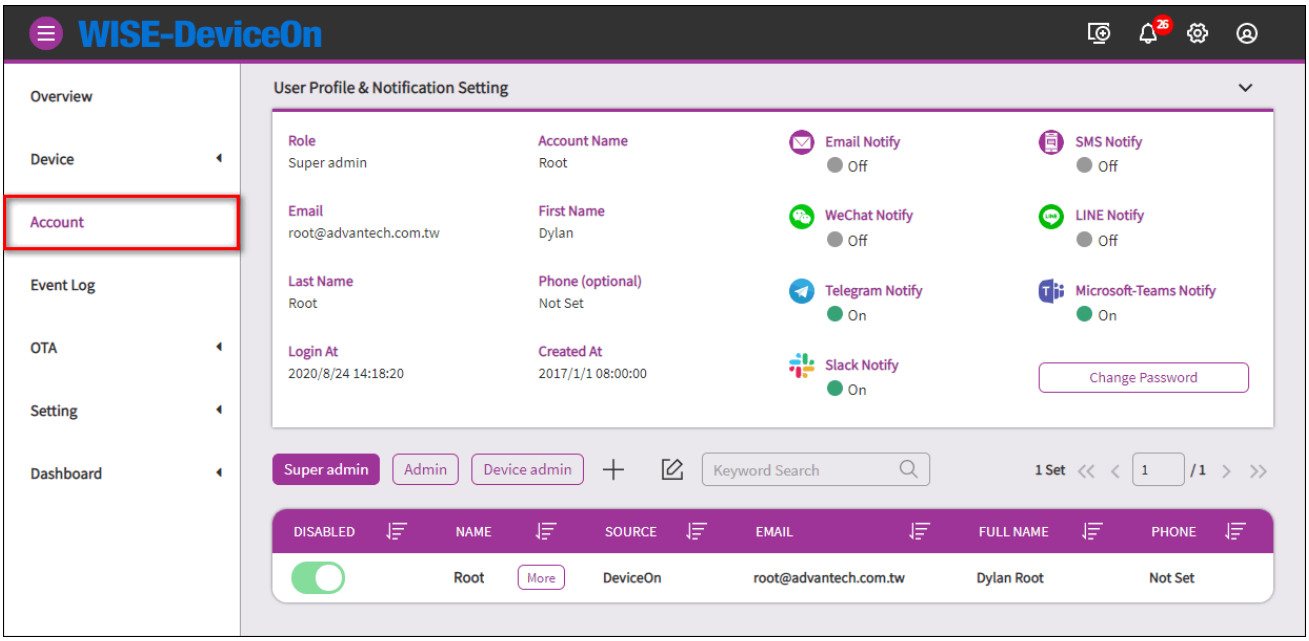


Confirm the rule and enable it.

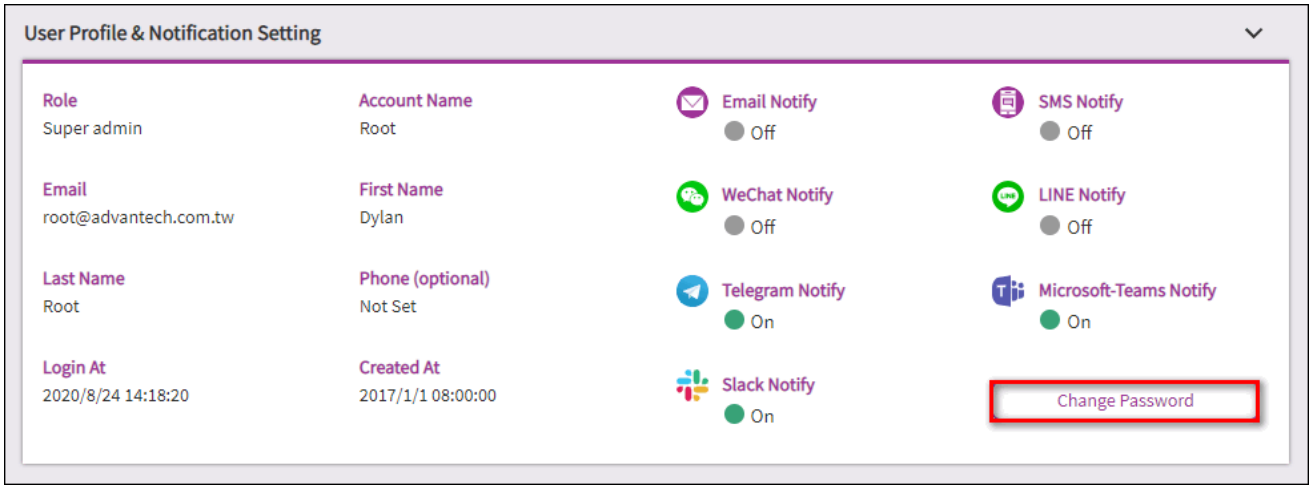


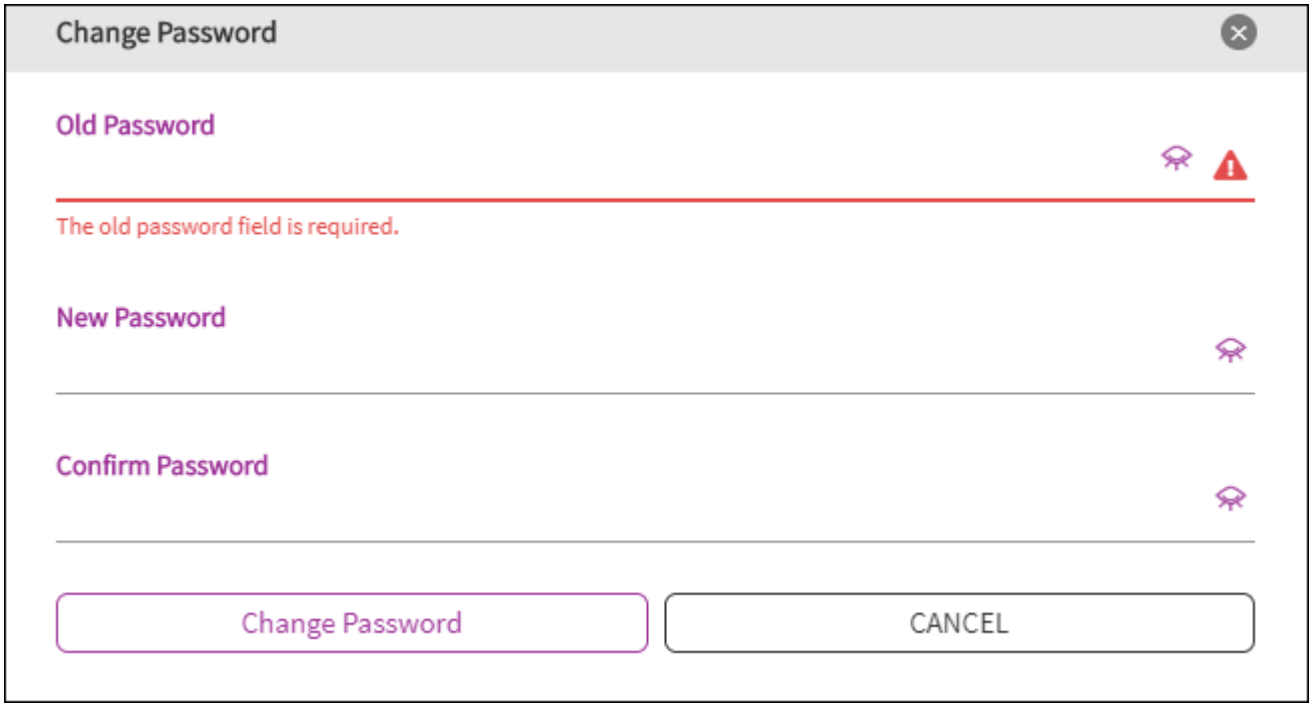
3.3.3 **Account Management**

The first step to manage device is login to DeviceOn, therefore, you could start to invite, edit other accounts on this page. The user profile show your account information and person alert service, such as Email, SMS, WeChat, LINE, Telegram, Slack and Teams status.

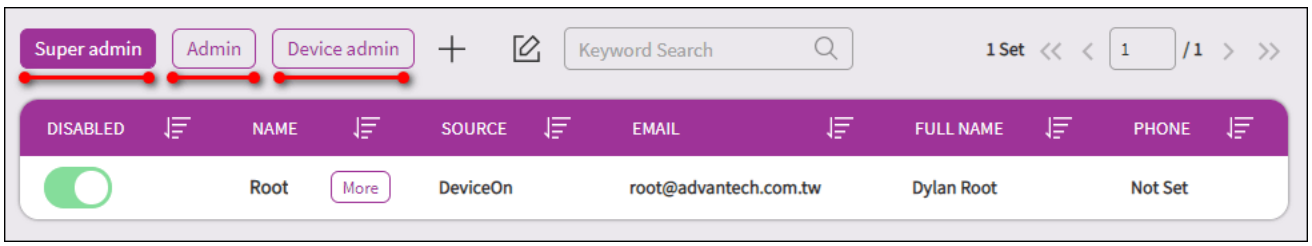


To change your password, please click on the “Change Password” on the profile.

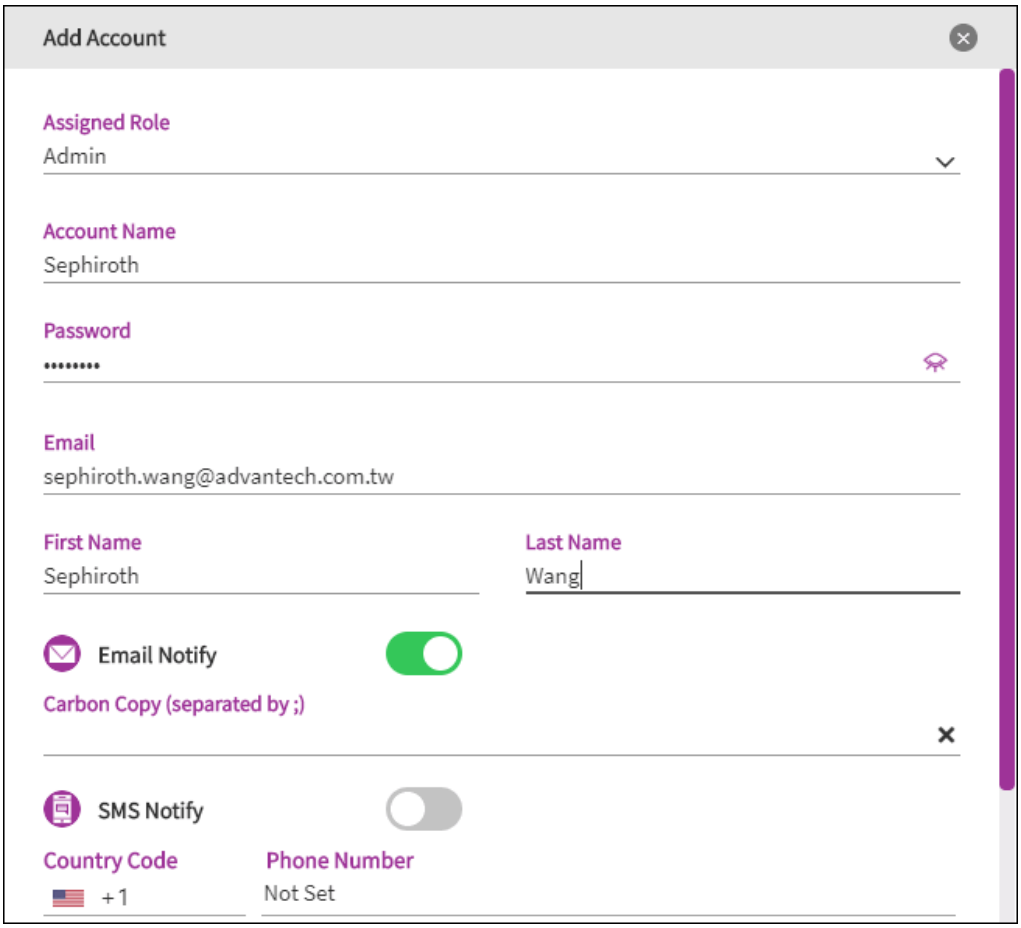




Every account belongs to a role, you could switch the tab to invite/view and edit account. There are 3 roles in the DeviceOn system. One is "**Super Admin**", only one account in the system belongs to "**Super Admin**". The other role is "**Admin**" and "**Device Admin**". For detail role permission, please reference Section 7.1.

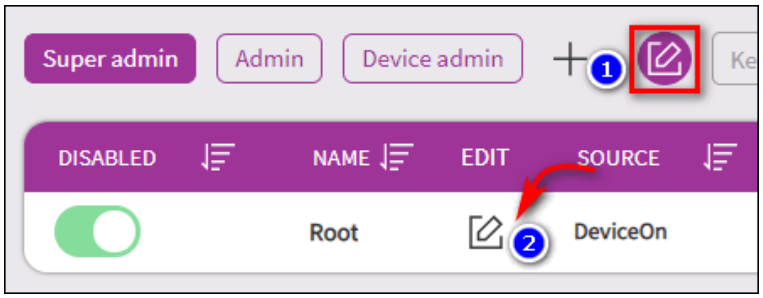


Click on the icon to "Add Account"



Enter your account, role, password, etc. to create an account. If the user would to receive notify from device, system alert, please enable these alert services on "**Mail**", "**SMS**", "**WeChat**", "**LINE**", "**Telegram**", "**Teams**" and "**Slack**". These alert services are personal setting, please make sure the "**Setting** -> **Notification**" is configured, enabled on DeviceOn System.

Click on the icon to "**Edit**" or "**Disable**" account.



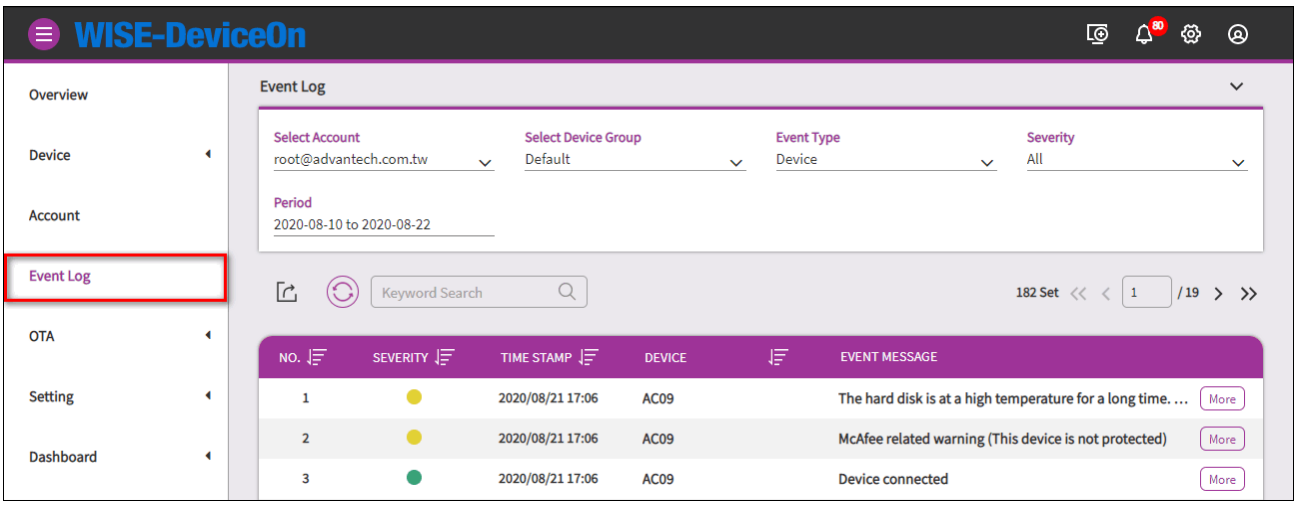
3.3.4 Event Logs

Device management is complex with device log and user behaver. Logging data can provide insights about your devices and help you:

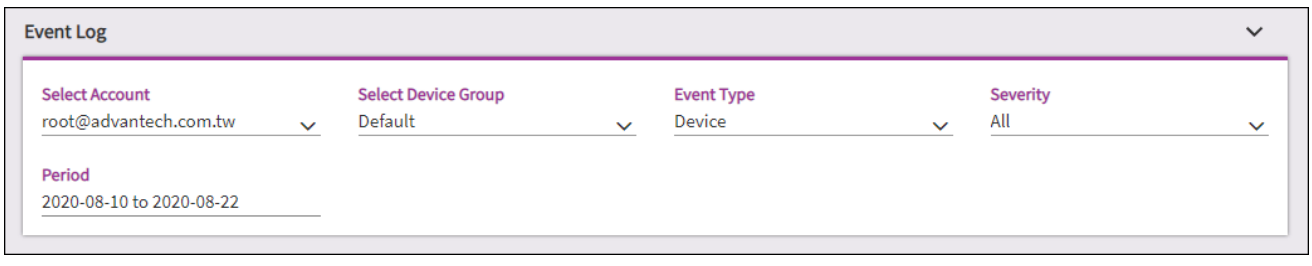
- Troubleshoot past problems or prevent potential ones
- Improve device healthy or maintainability
- Real-time alert through 3rd notification

DeviceOn logs are categorized into the following types:

- Operation log provide information about DeviceOn resource CREATE, UPDATE and DELETE operation, like set device power off, update device name or delete account.
- Device log provide information about events raised as device side resources, like connected, disconnected, over the threshold,
- System log provide information about analyzed; scheduling event/alert that have been process on DeviceOn server. Example of this type are queue buffer alerts where server has processed and measured IoTHub queue and provides concise alerts.



There are three type of Event Logs as mentioned above and each event log with different severity, **Information**, **Warning** and **Error**. Through the filter to find your device log.



Click on the icon to refresh event log by manual.

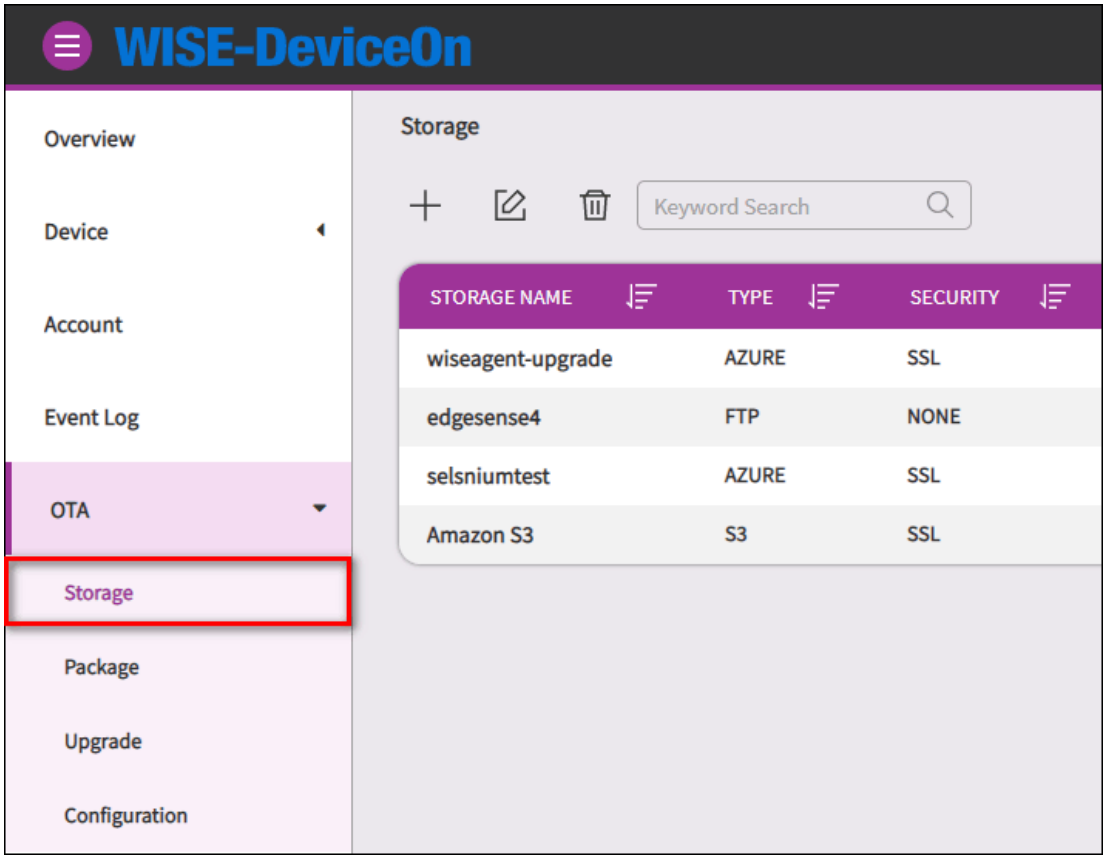
Click on export icon to export devices that in the table as CSV file.

3.3.5 OTA (Remote Provisioning)

OTA (Over-The-Air) is one of powerful feature DeviceOn provides. Users can deploy **software** packages, **configuration**, **Windows QFE** (Quick Fix Engineering), **Advantech BIOS** update onto a device remotely, or even many devices broadly.

- Storage

There is a default Azure blob storage called "**wiseagent-upgrade**", host by Advantech DeviceOn team. If there is a new version of WISE-Agent released, all of user could get the update and upgrade their devices. The storage is read only cannot upload user's OTA package.



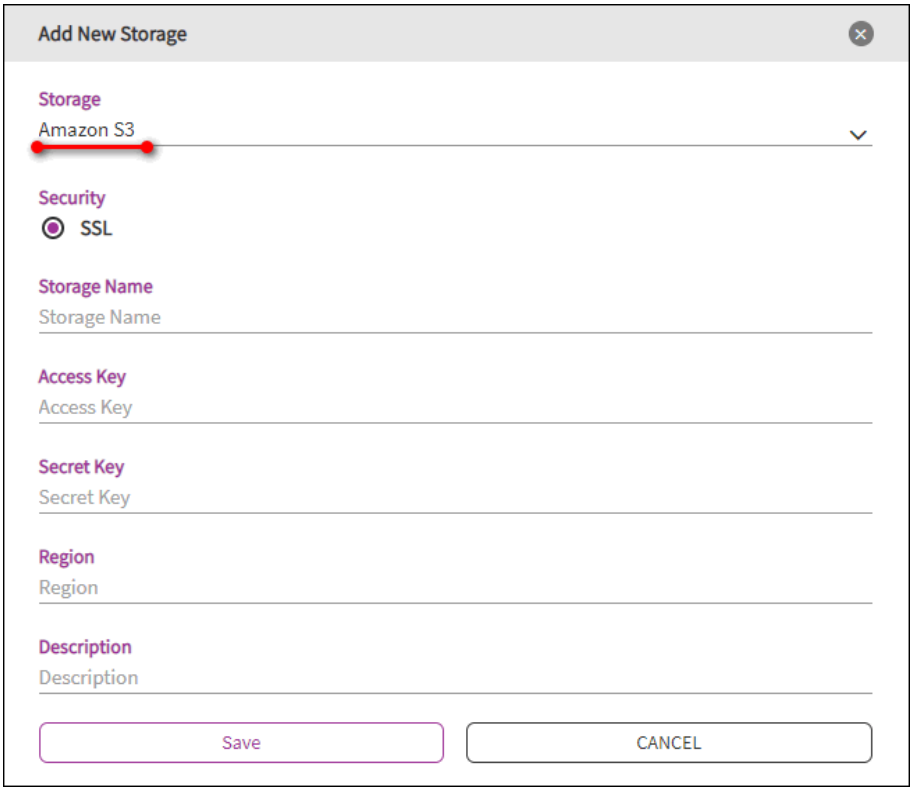
Click on add icon to add new storage.

For cloud storage, DeviceOn provide "**Amazon S3**", "**S3 Compatible**", "**Azure Blob**" and traditional

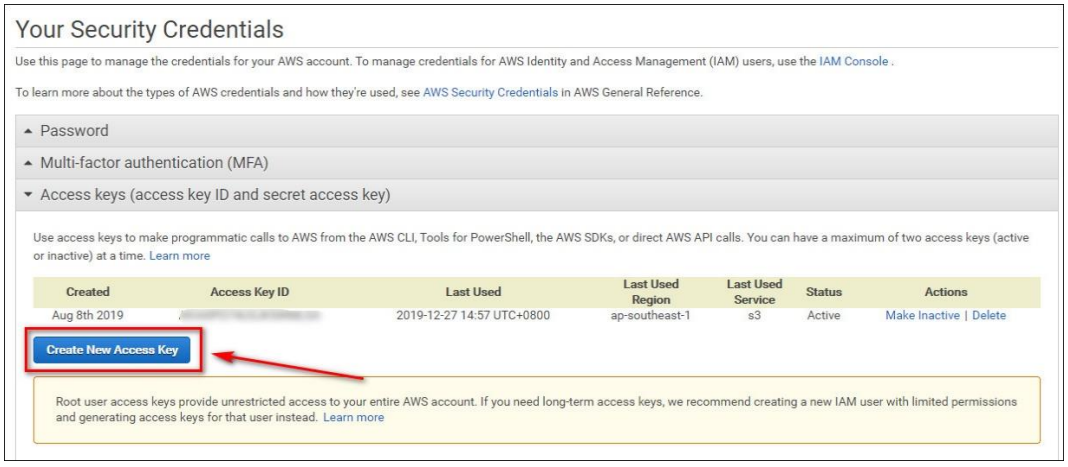
FTP services.

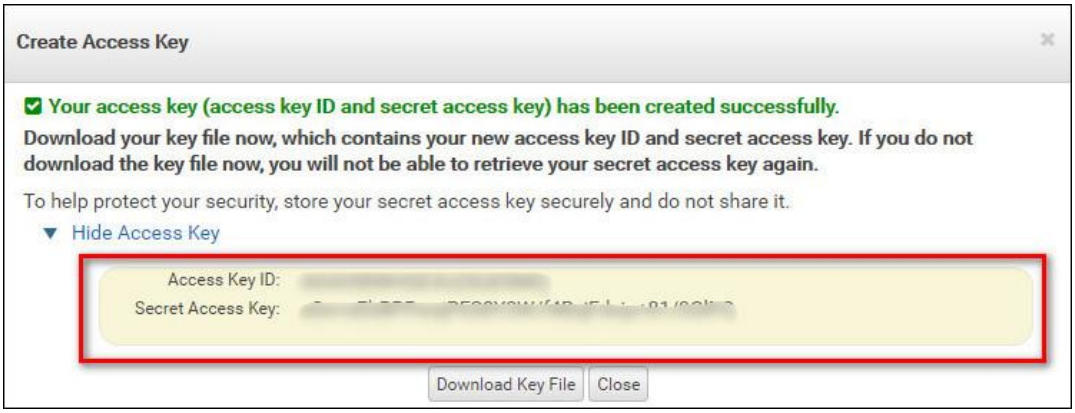
[**Amazon S3**]

You could create and get Access Key, Secret Key from Amazon Web service.



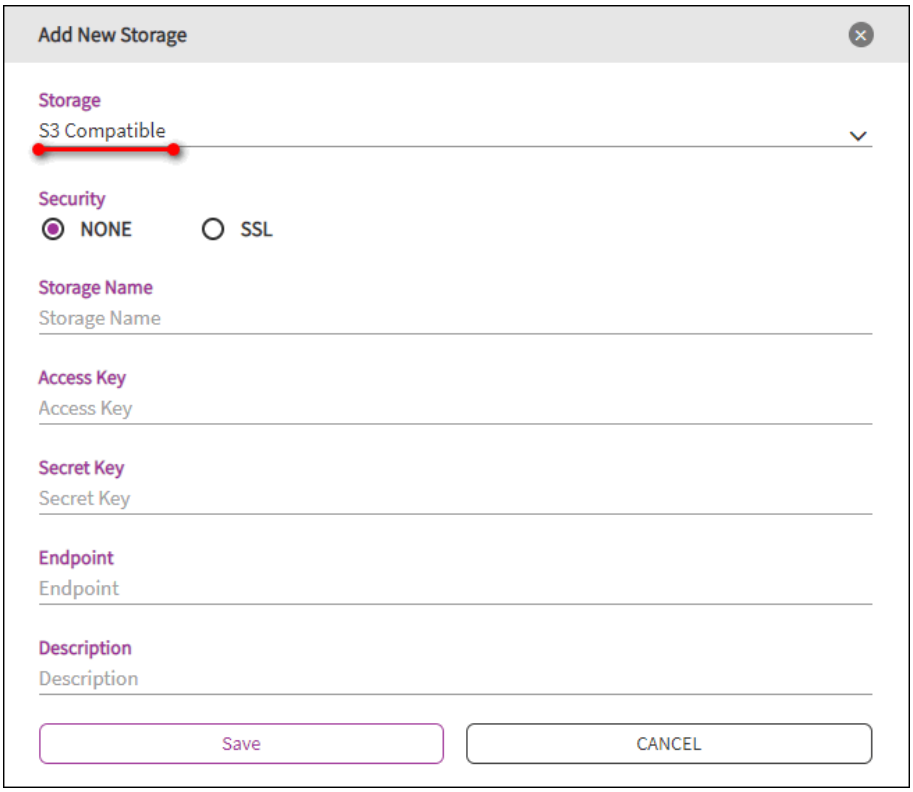
- ✧ Storage Name: Your storage name, define by yourself.
- ✧ Region: Region of AWS S3
- ✧ Access Key: Access Key for AWS S3
- ✧ Secret Key: Secret Key for AWS S3





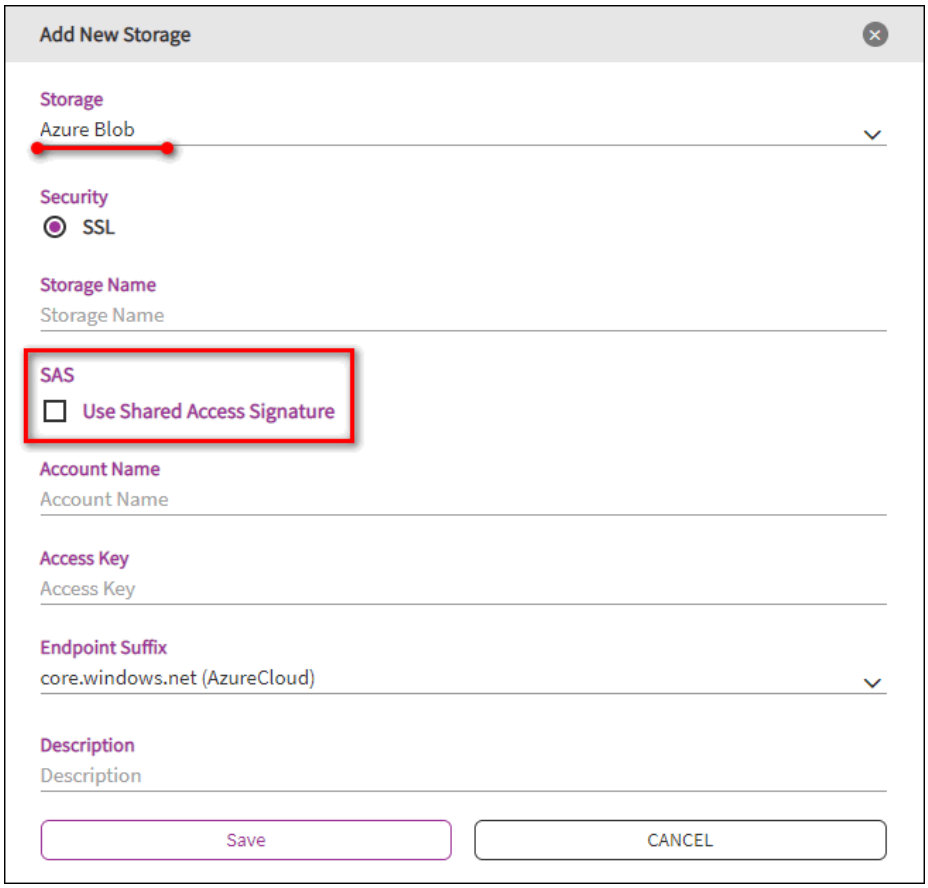
[**S3 Compatible**]

The setting similar to Amazon, only **endpoint** must be configured to yourself.

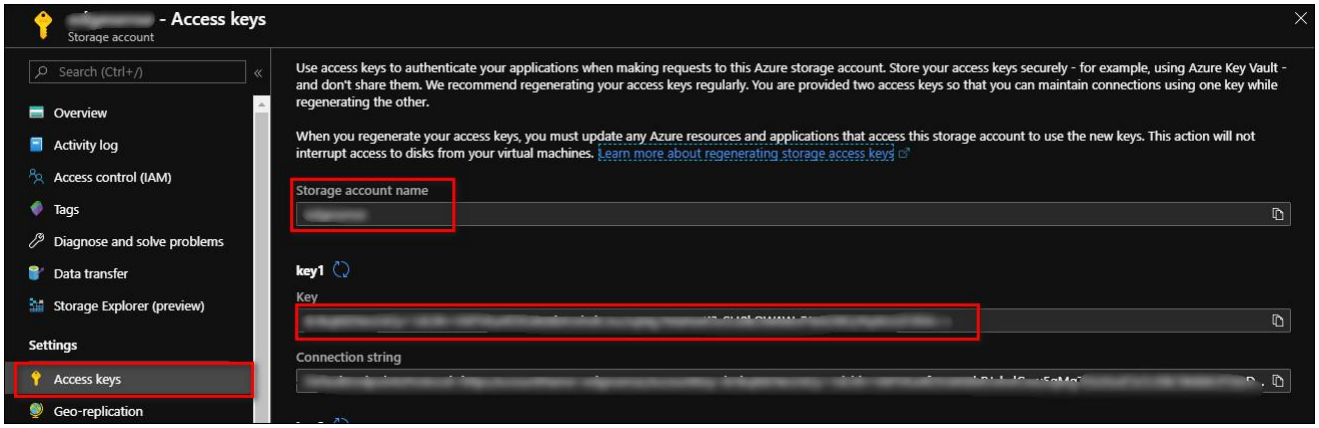


[**Azure Blob**]

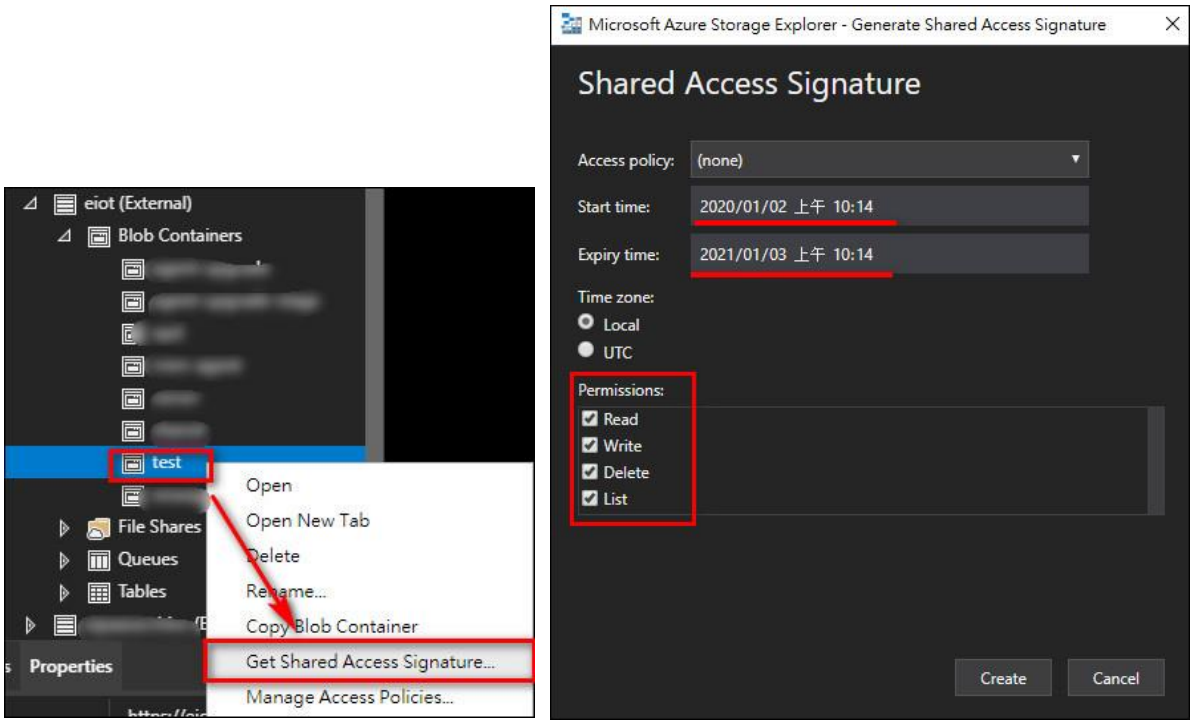
For Azure Blob, supports two mechanisms to access, one is "**Storage Account**" and "**Access Key**" with full access permission of container. The other is "**container**" SAS token generated via Microsoft Azure Storage Explorer.



Through Azure portal to get your **Storage Account** and **Access Key**.

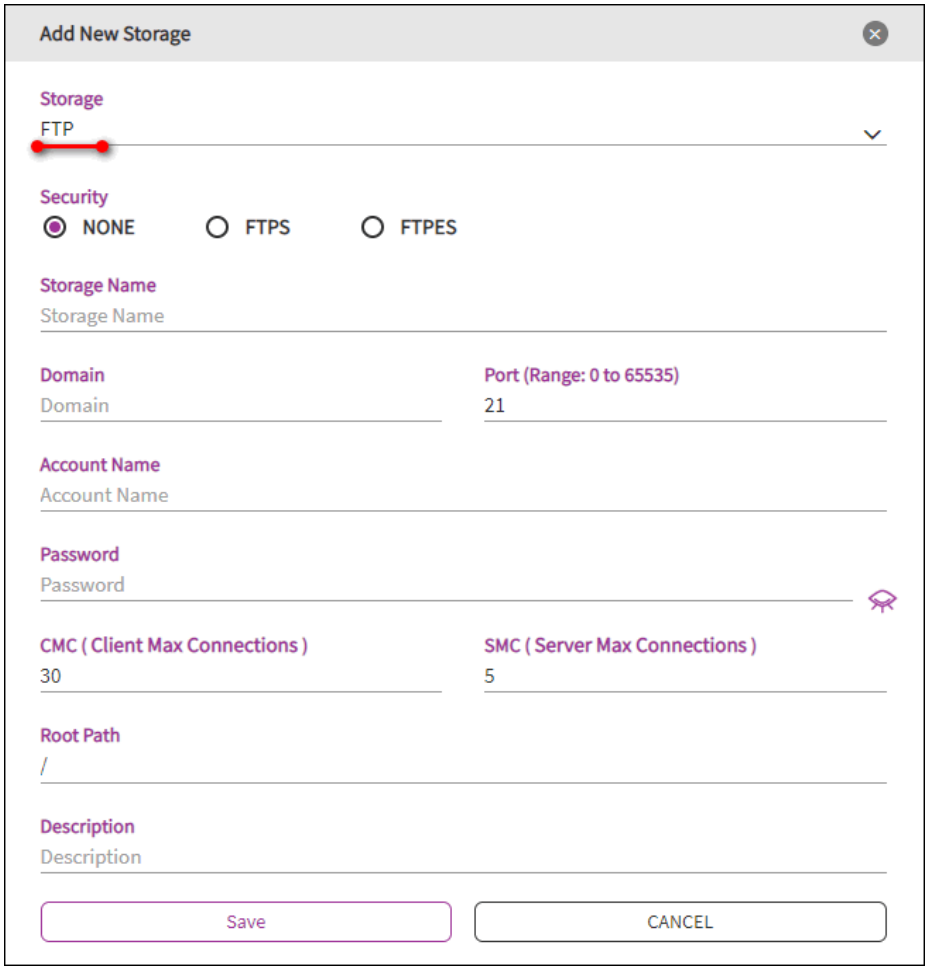


Get **container's SAS token** via Azure Storage Explore, please make sure your permission (Read, Write, Delete, List) and valid period (Start and Expiry time)



[**FTP**]

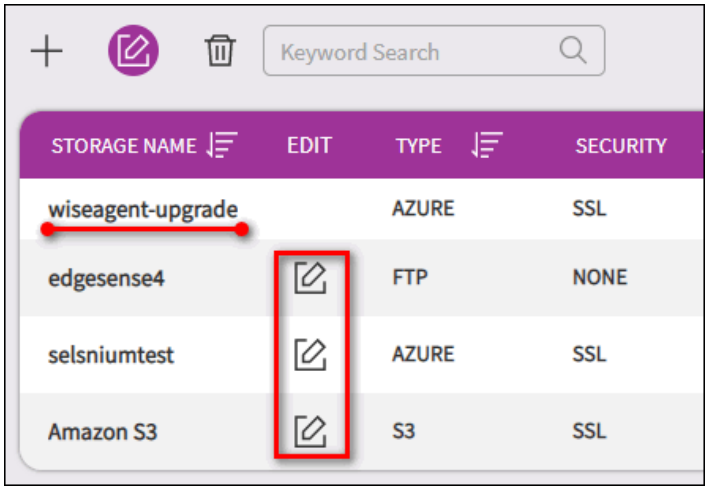
For FTP, you might setup another FTP server with security and account, password.

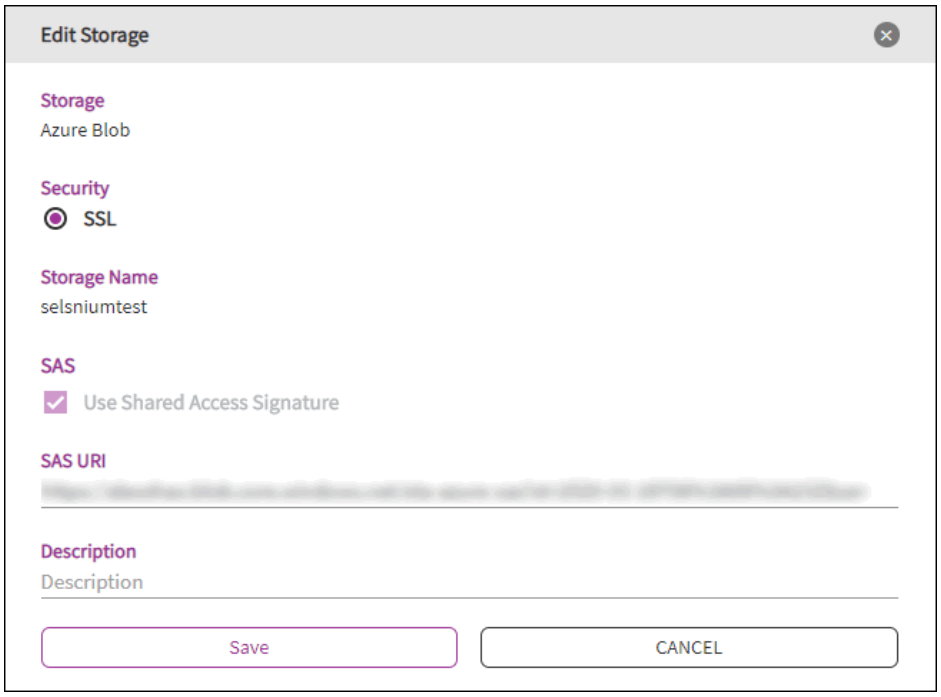


- ✧ **Security:** Leave it as **"NONE"**, the default value. If your FTP server running on FTPS protocol, pick **"FTPS"**.
- ✧ **SOTRAGE NAME:** Enter **"MyFTP"**.
- ✧ **DOMAIN:** Enter the FQDN of your FTP server, or its IP address.
- ✧ **PORT:** Should be **21** if the FTP server runs on a standard port number.
- ✧ **ACCOUNT NAME:** A valid username that can connect to the FTP server, and upload files onto the server as well.
- ✧ **PASSWORD:** The password to login.
- ✧ **CMC/SMC:** Maximum Client & Server Connection.
- ✧ **ROOT PATH:** FTP server access path (root folder)
- ✧ **DESCRIPTION:** It's optional information.

Click on edit icon to adjust a storage.

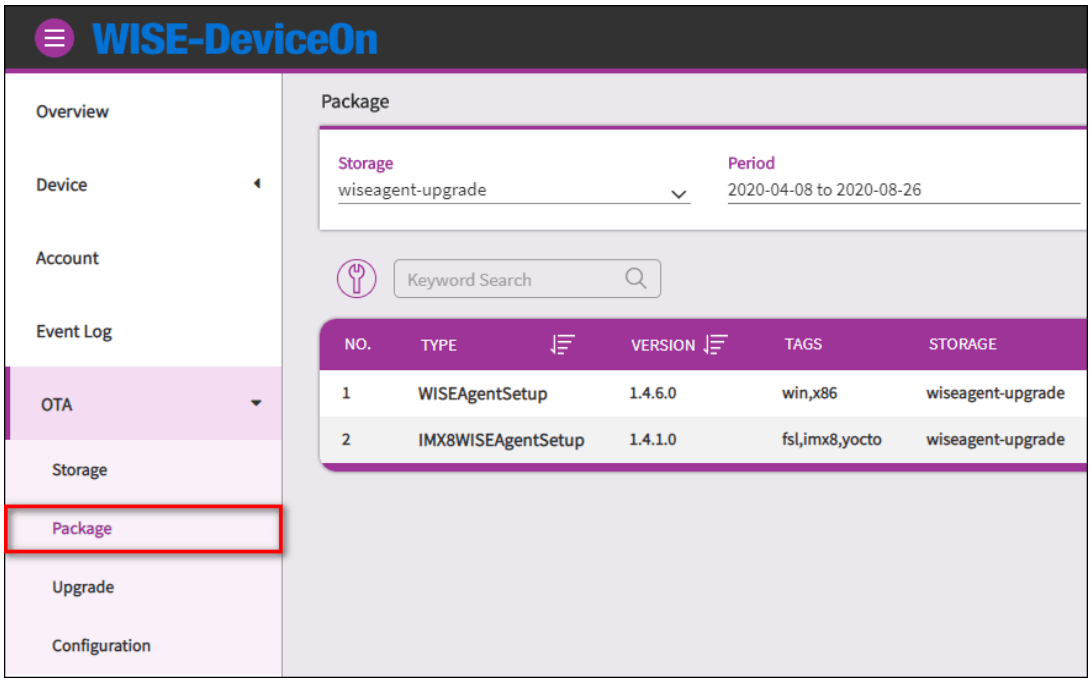
You could edit yourself storage, but the default storage cannot.



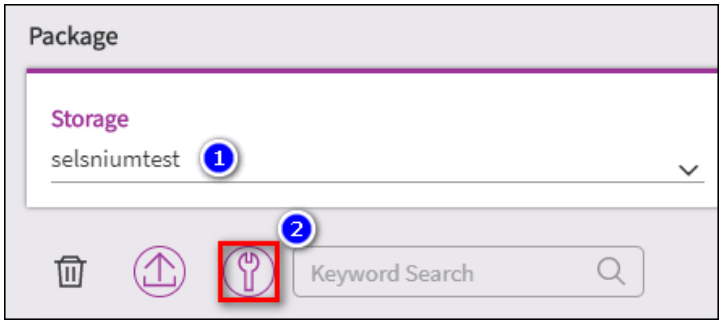


- Package

View and edit OTA package on select storage, user could edit, delete upload their package to selected storage, but default storage (**wiseagent-upgrade**) cannot. To ensure the security and data format on OTA package, user should wrap their software, firmware via DeviceOn toolkit. The toolkit not only command-line tool but support online UI mechanism.



Select to your storage and click on the toolkit icon to start to warp your OTA package.

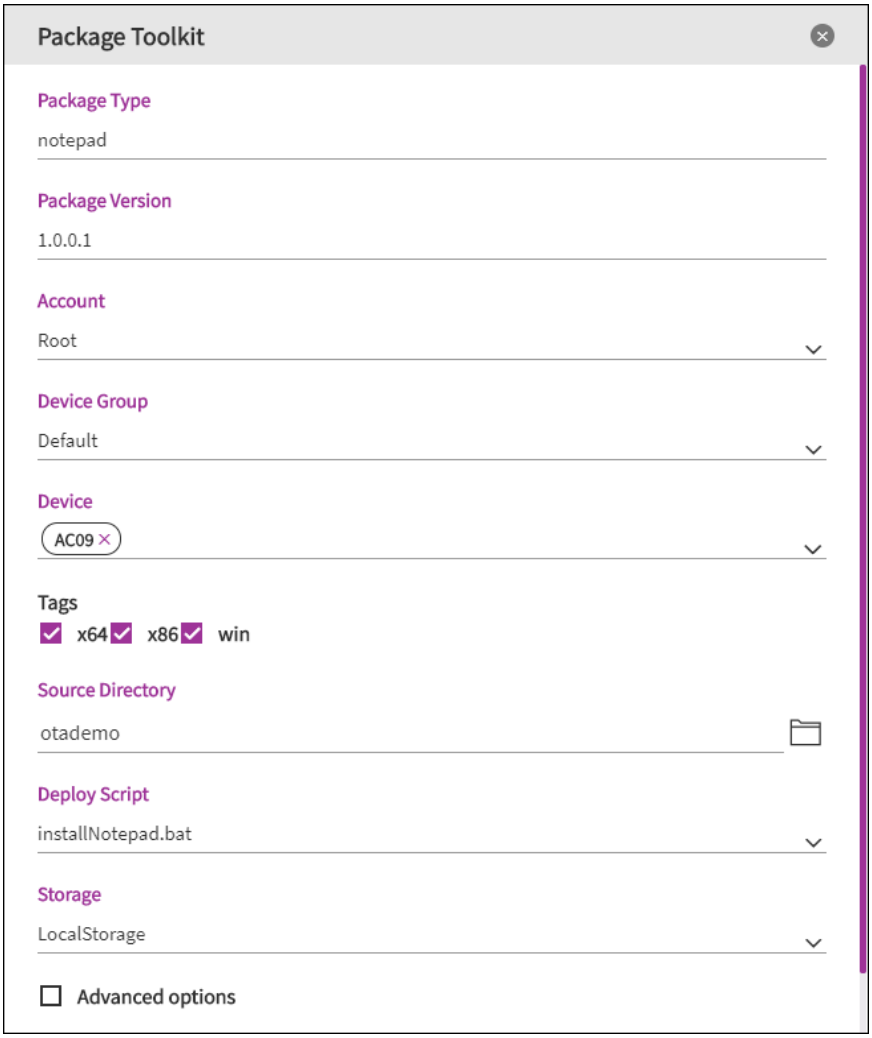


Prepare your software, configuration and installation script first, gives below information. The operation system and architecture might be different. Therefore, to determine the OTA package be deployed on which devices, please pick-up the "**Tag Name**" on "**Supported Arch**". All "**Tags**" must match with devices, the OTA package will be executed. For example, there is two devices (ARK-1123, UTC-520) with different tag attribute. The ARK-1123 device is Windows based and support x64 and x86 OTA package. The UTC-520 is Ubuntu system also support x64, x86.

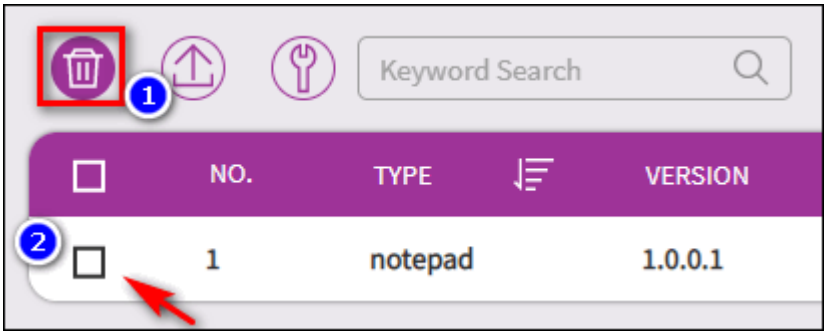
- ARK-1123 (Tags): win, x64, x86
- UTC-520 (Tag): ubuntu, x64, x86

If your OTA package tags are "**win**", "**x64**", "**x86**", the package only support and executed on "ARK-1123". Otherwise, if the tag is "**x64**", both devices could be affected.

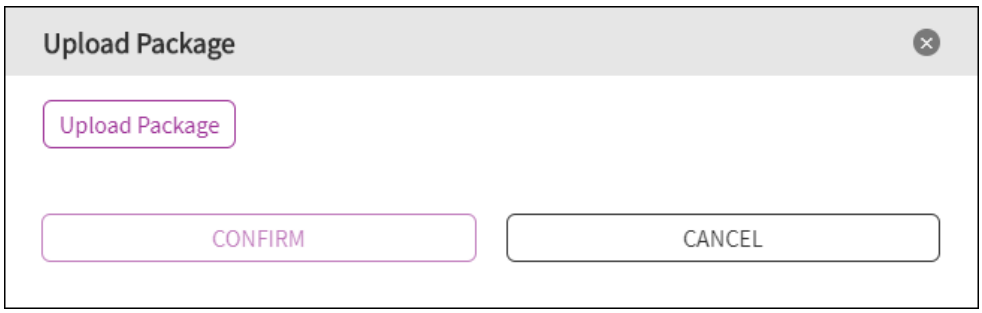
- **Package Type**: Name of package
- **Package Version**: Version of Package
- **Supported Arch**: Select "**Tag Name**" from of device (Account -> Device Group -> Device)
- **Deploy File:** Installation script (batch file or shell script)
- **Storage: Upload to storage or download**
- **Advanced options:** Reboot or run the script after deployed.



Click on the delete icon to delete your OTA package.

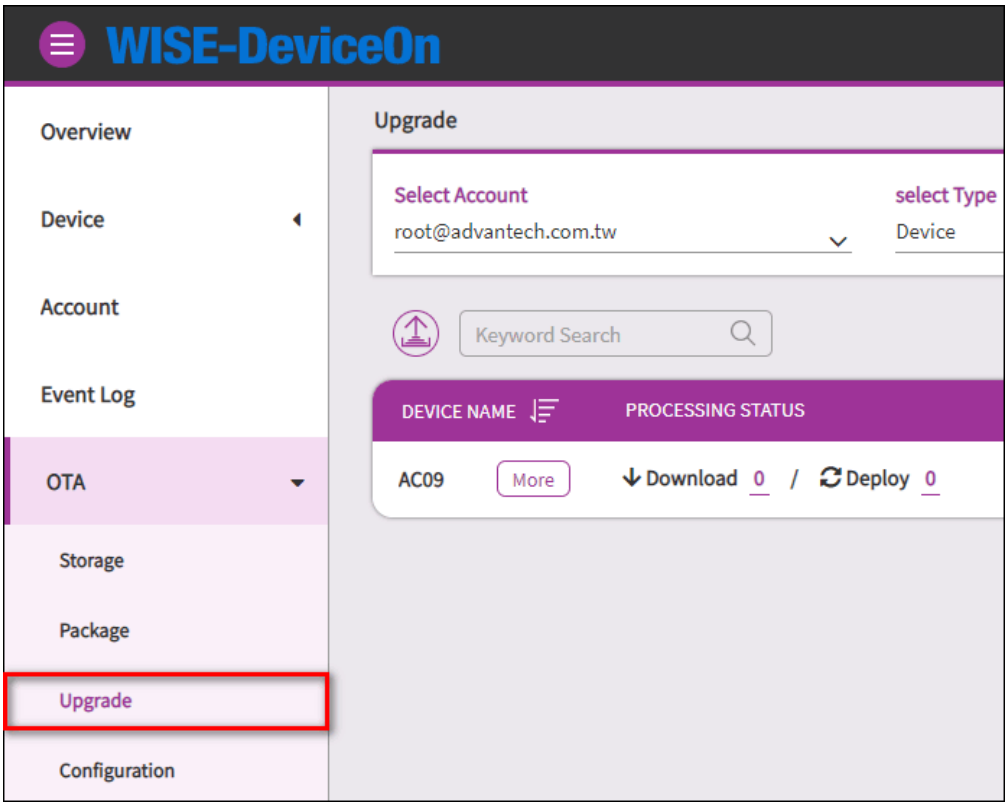


Click on the upload icon to upload your OTA package.

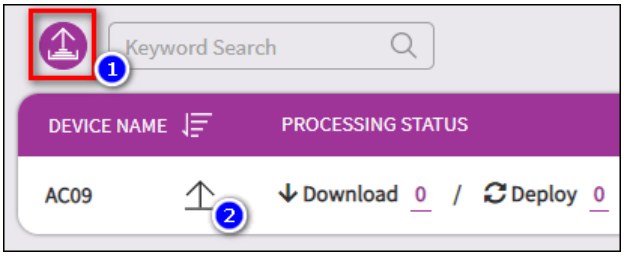


- Upgrade

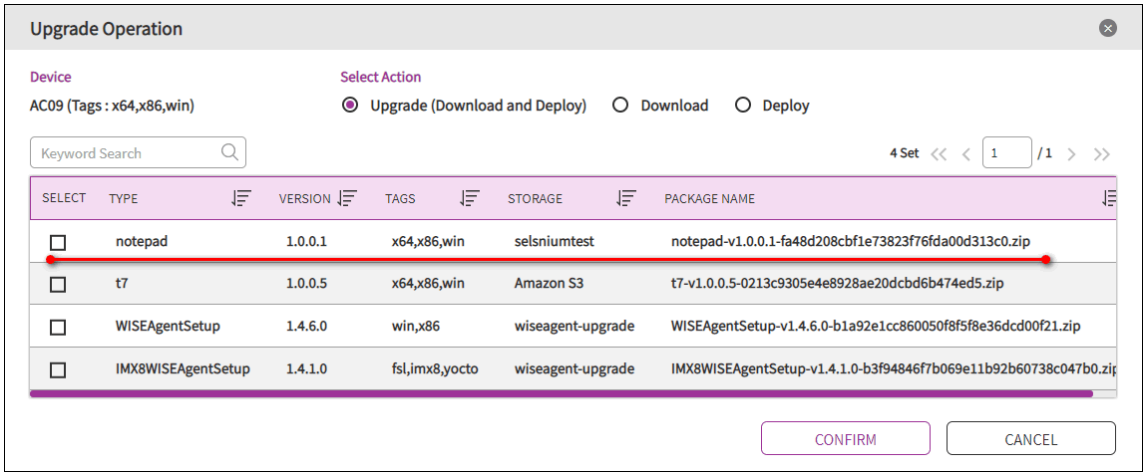
On the upgrade tab, start to select your device or device group and pick-up your OTA package that you upload before. On the device list to configure schedule, check the result status and program list that installed.



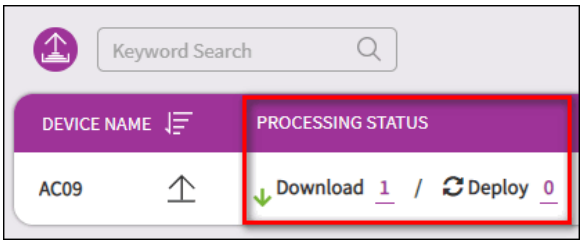
Click on upgrade icon to select OTA package.

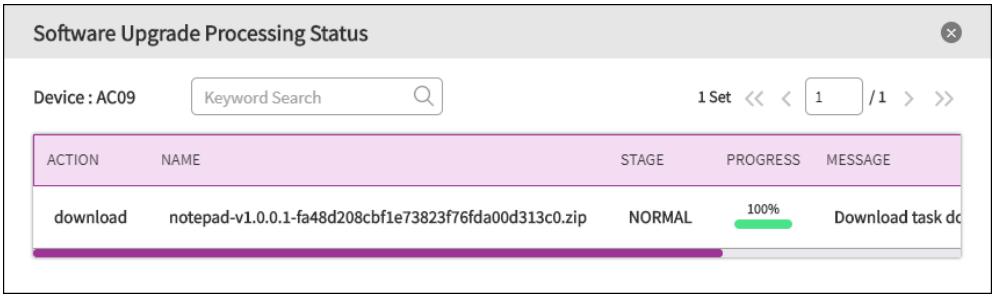


Select your package to “**Upgrade**”, “**Download**” or “**Deploy**”. The “Upgrade” represents download OTA package from storage and execute (Deploy) immediately. Every package would be kept on device side as “Upgrade” or “Download”.

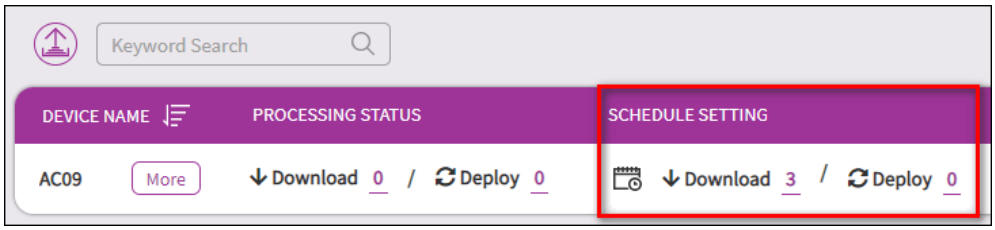


To check the deploy status, please click on process icon.

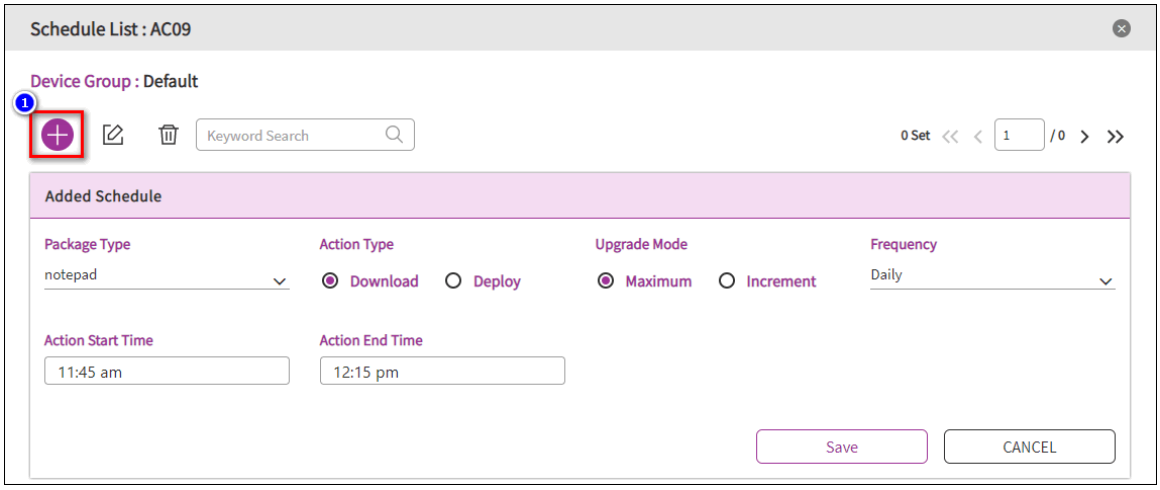




To avoid burst download on large number of devices upgrade at the same time, user could add schedule to check and upgrade by schedule.

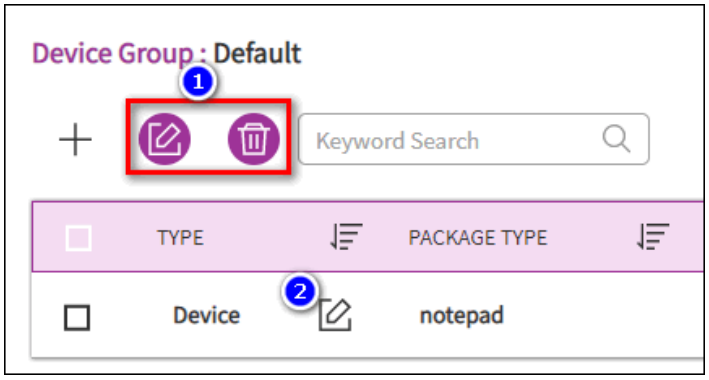


Click on add icon to create a schedule.

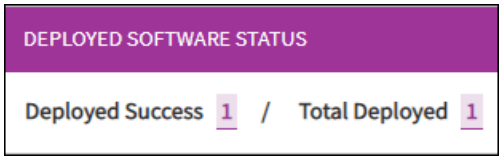


- ✧ Package Type: Select your OTA package from storage.
- ✧ Action Type: Download or Deploy the package.
- ✧ Upgrade Mode: If the mode is **Max**, the action would download/deploy **the latest version** on the package. Otherwise, if the mode "**Increment**", The deploy, or download behavior will gradually increase from the lower version to the latest version.
- ✧ Frequency: **Daily**, **Weekly**, **Monthly** or **Once** to check.
- ✧ Action Start Time: Check time on start.
- ✧ Action End Time: End time for download, if download exceeds the end time, the action will be terminated.

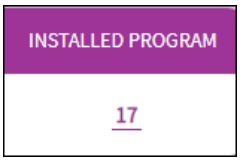
Click on edit icon to modify, delete OTA schedule.

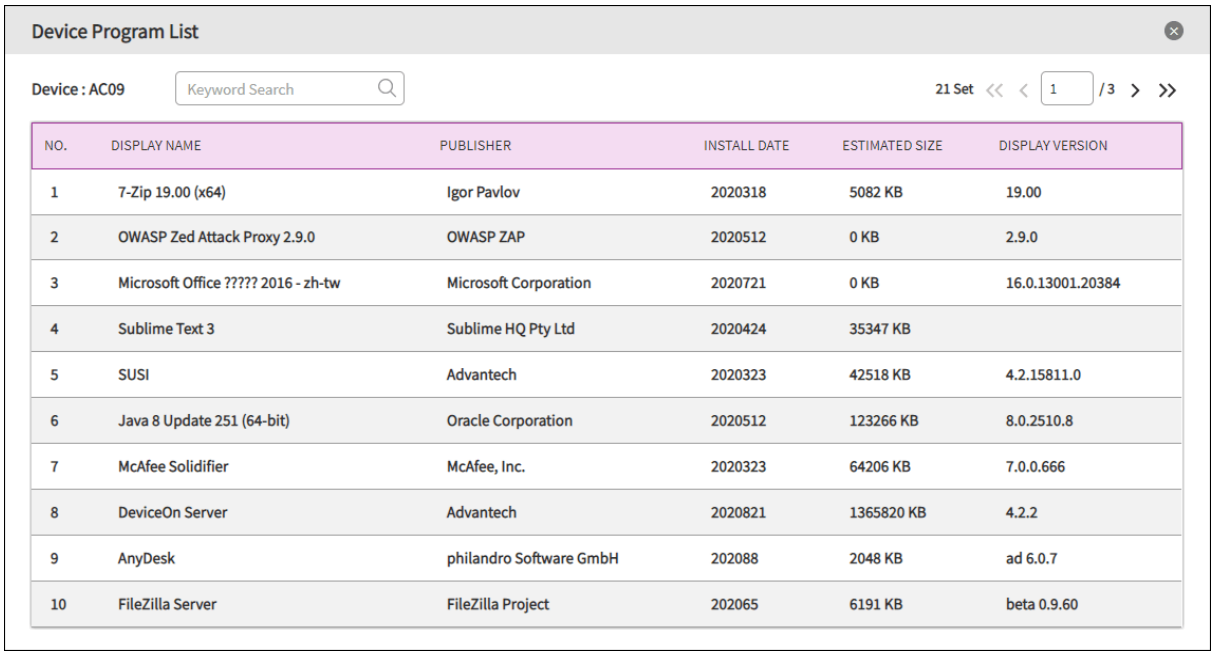


To check deployed software, configure status on device, please click on the numbers.

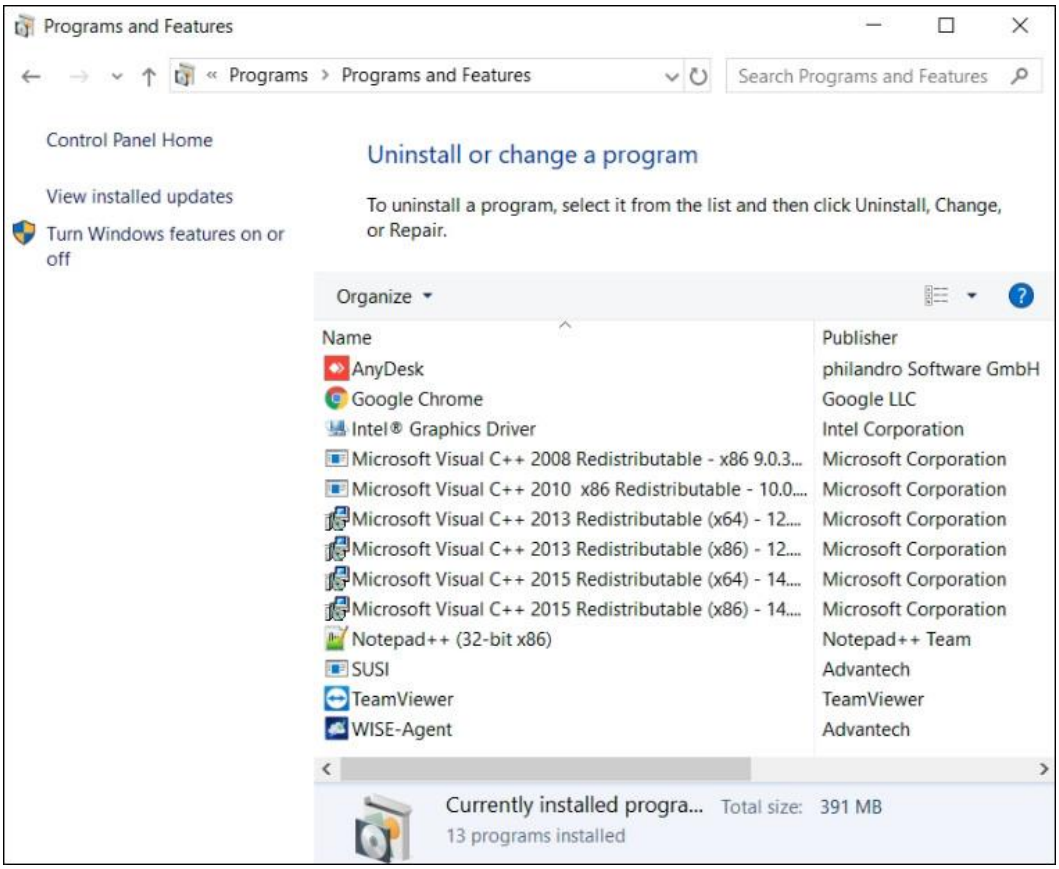


Furthermore, user could view the program list on the device. (Windows Only)



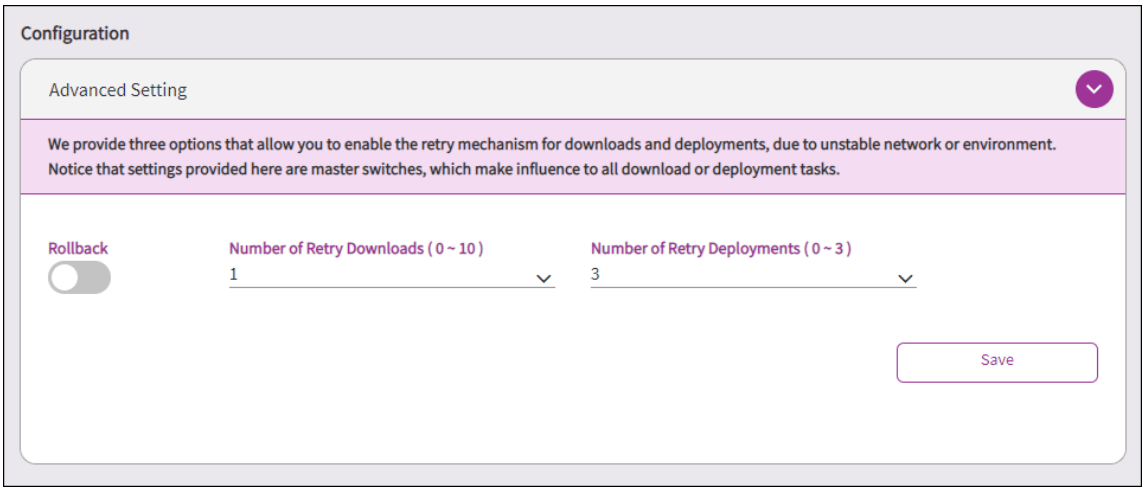


This program information retrieves from device operation system, same as below figure.



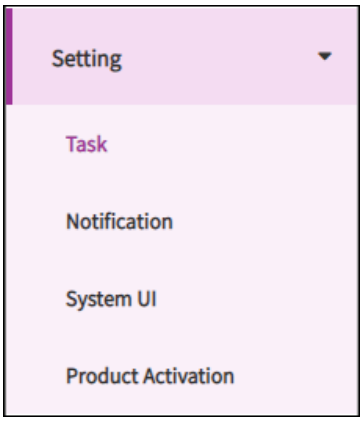
- Configuration

There are three options for OTA to deploy your package, one is “**Rollback**” that means if a new version deploys failed, and the WISE-Agent would try the best to rollback to previous version that successfully. But there is a perquisite, the previous version of the package exists on the device side. The remaining two options are the retry times. Due to network instability or other factors causing the download fail, OTA provide the retry times to ensure successfully deployment as possible.



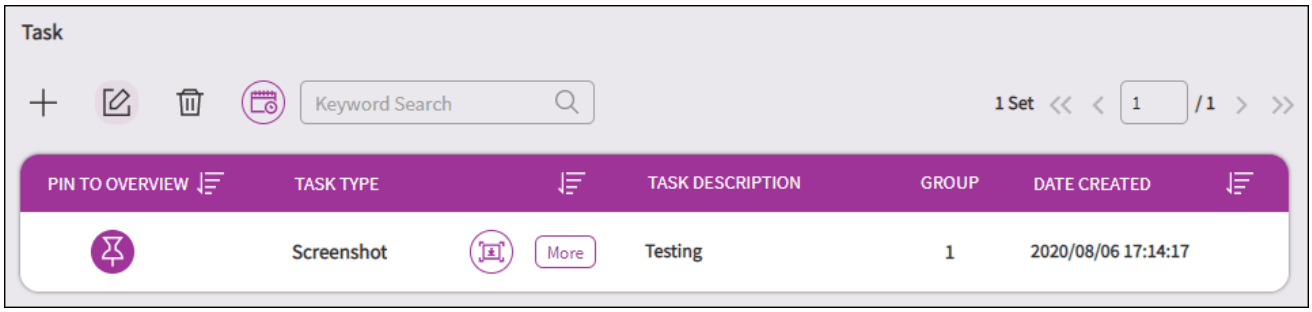
3.3.6 System Configuration

A System Configuration define advanced setting include “**Task**”, “**Notification & Event Alert**”, “**System UI**” and “**Product Activation**. These setting are usually changed less often or only need to be modified once. Some functions require root, admin to modify or be visible, and product activation only shown on prefecture license, such as Standalone, Azure Kubernetes version.



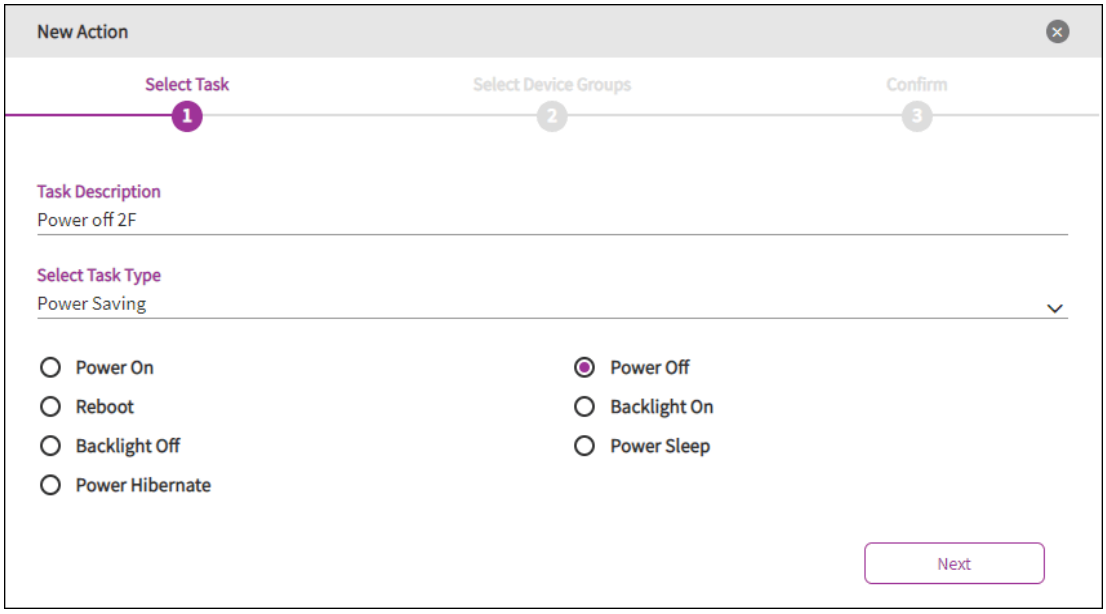
- Task

The real-time actions on the overview that are defined, created on here, you could add a new task and pin to overview. These tasks are binding to personal account, cannot view, edit, and delete others.

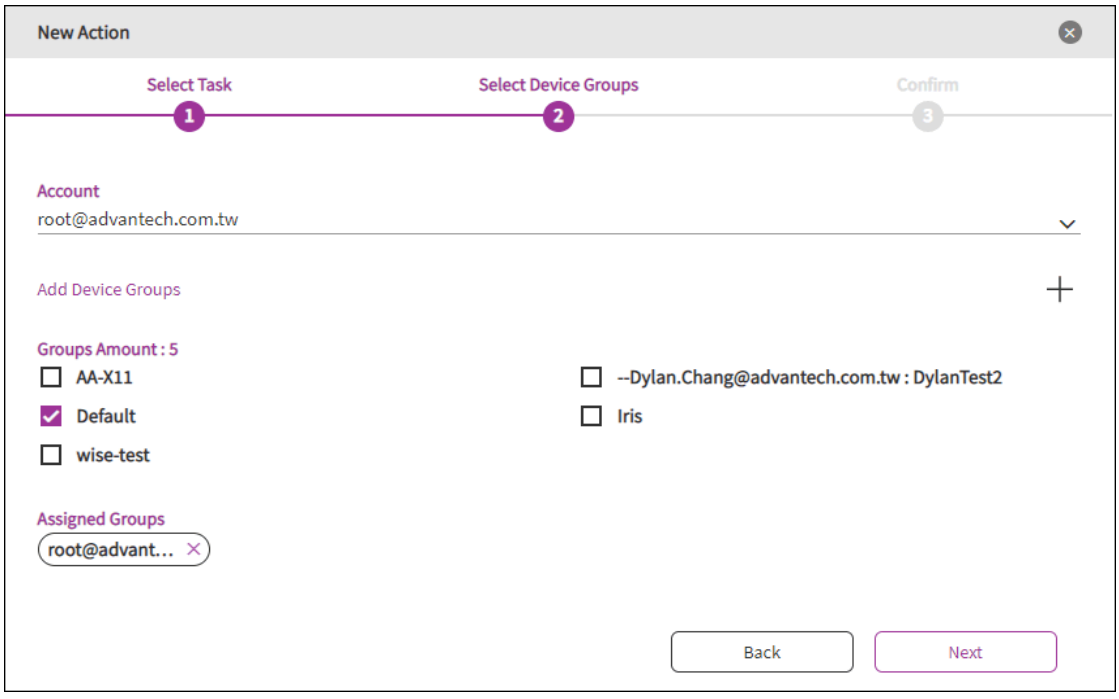


Click on the icon to add action.

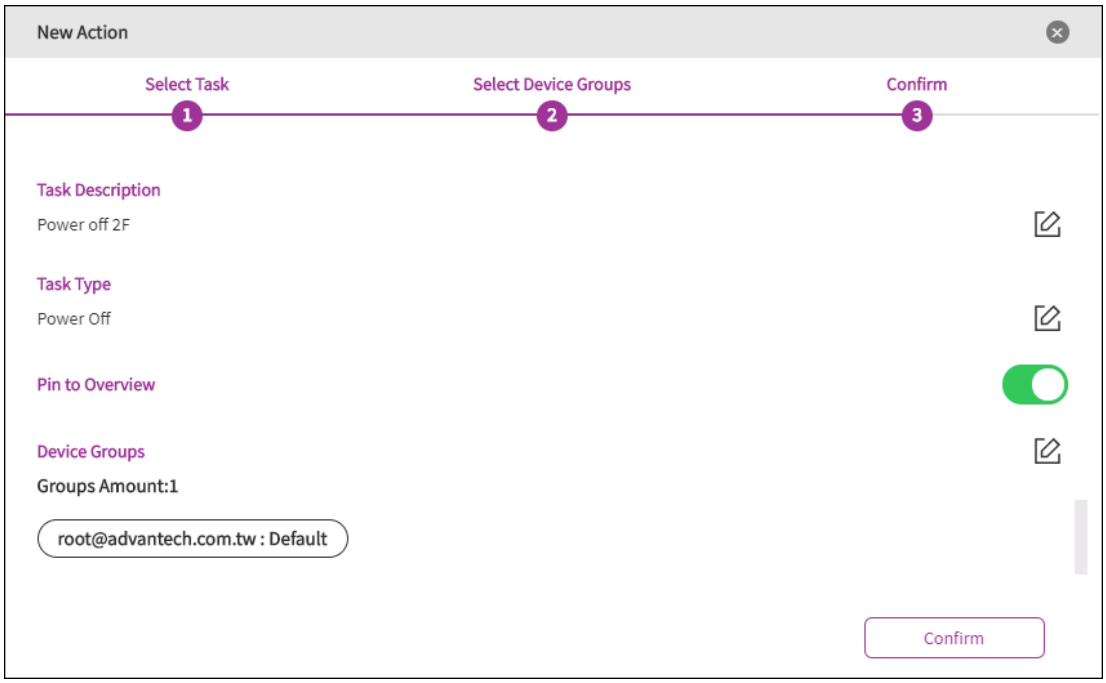
Enter your description and select a “**Task”** from three categories, **Power Saving**, **Security** and **System.**



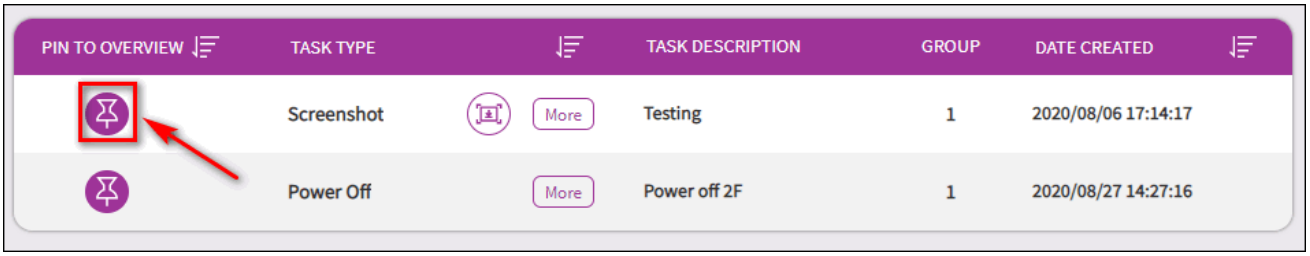
Select "**Device Groups**" for the action that you picked up.



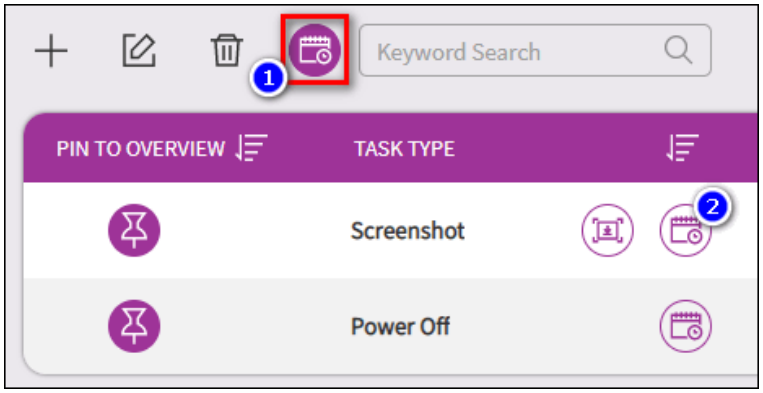
To confirm information, action, group and devices, and enable pin on overview, please click on "**Confirm**" to complete the wizard.



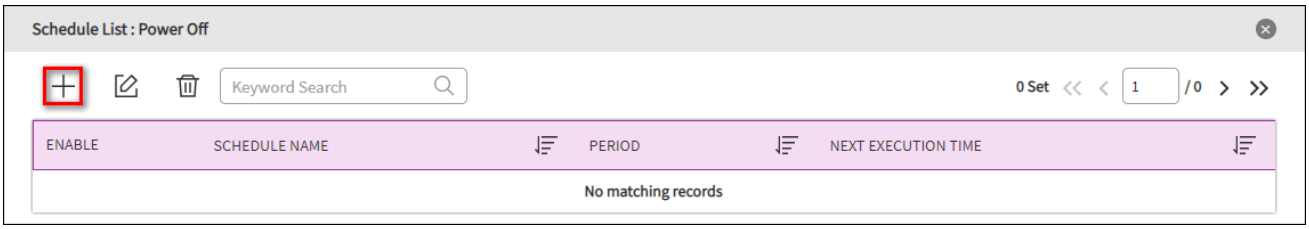
After created, you could find a new action on below actions list, click the PIN icon to determine the action shown on overview or not.



The actions support scheduling, click on the icon to define a schedule, daily, weekly, monthly, yearly or once.

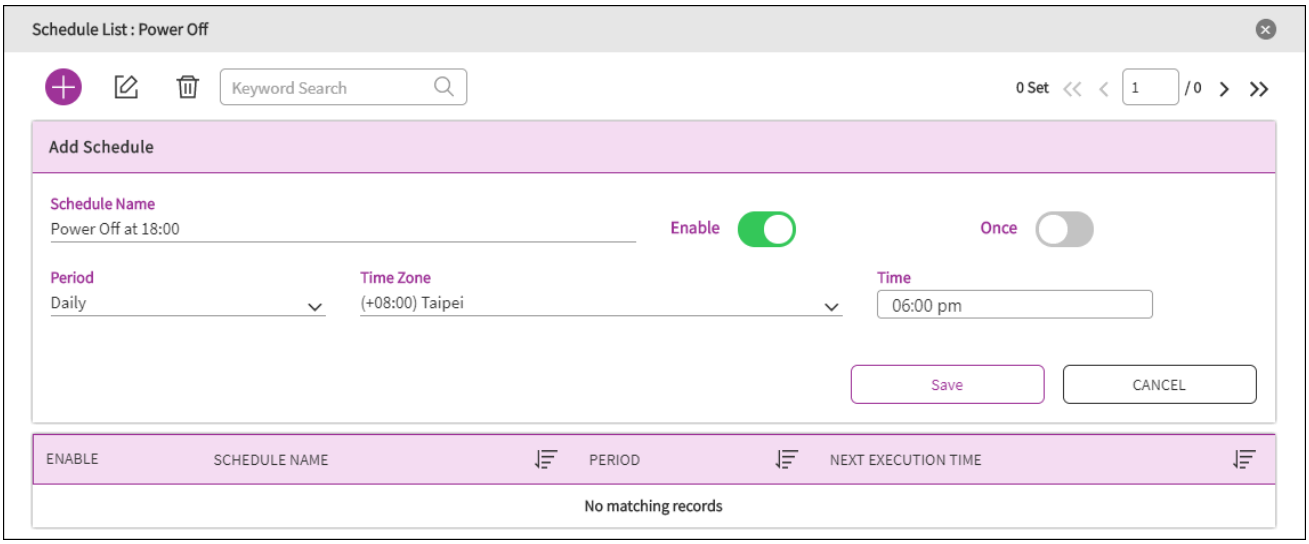


Enter to schedule list for all actions, and click on add icon to create new schedule.

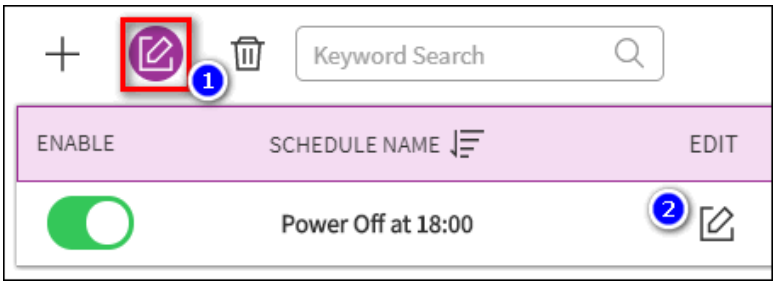


Given your schedule name, time zone, period and time and click **Save**.

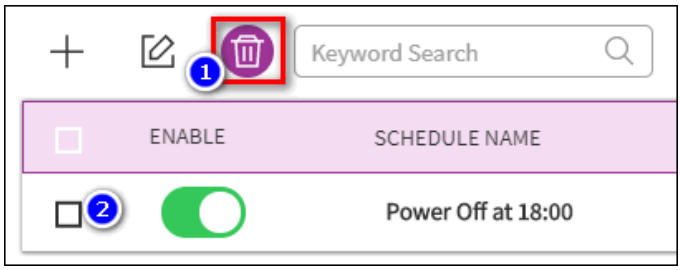
- **Schedule Name**: Name of schedule
- **Time Zone**: Time zones tend to follow the boundaries of countries and their subdivisons instead of longitude, because it is convenient for areas in close commercial or other communication to keep the same time.
- **Period**: Repeat interval for Daily, Weekly, Monthly, Yearly or once at a time.
- **Time**: Execution time.



Click on the edit icon to adjust schedule item.

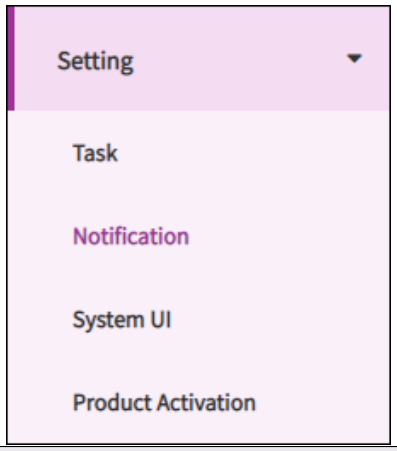


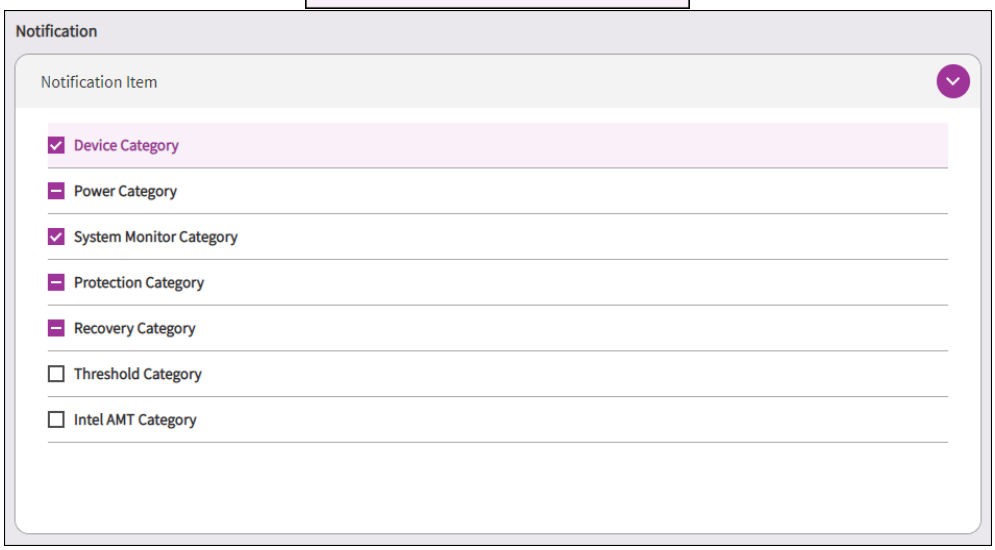
Click on the delete icon to delete schedule item.



- Notification & Event Alert

Here are seven notification services, include tradition service (SMS, Email) and popular social media (LINE, WeChat, Telegram, Microsoft Teams, Slack), if you select the event log type on "**Notification Item**", the notify message will through these services. These notification services are global setting, if your account does not receive, please check the personal setting on **Account**.



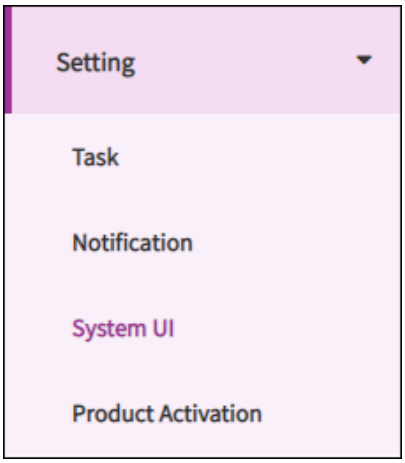


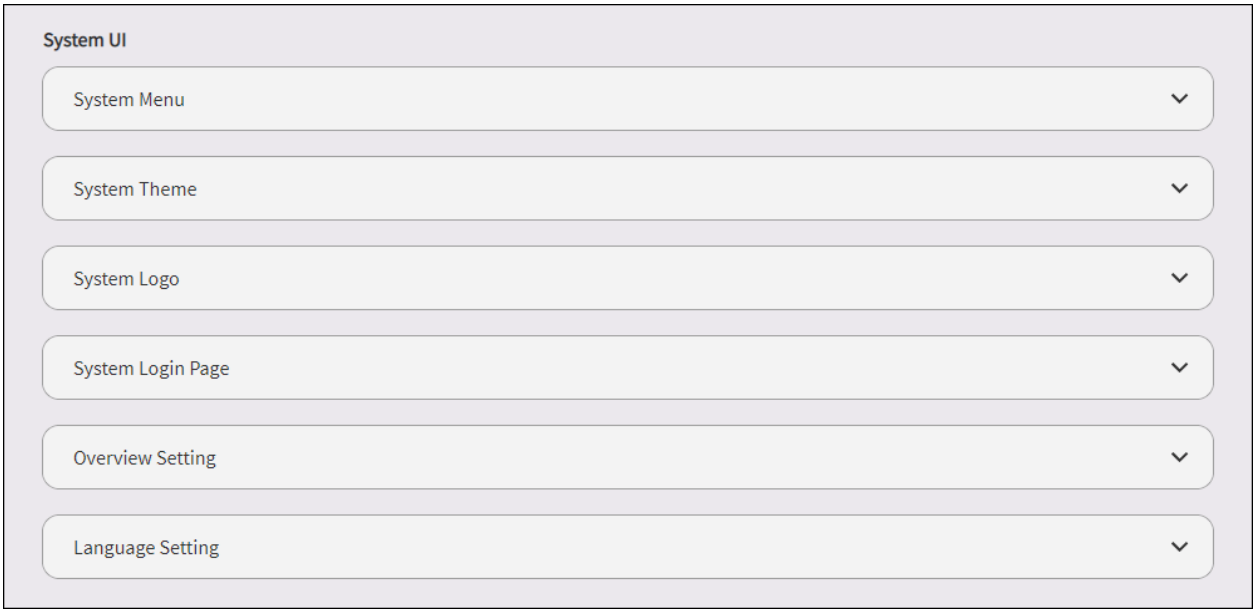
To configure these notification service, please reference Section 4.3.2 ~ Section 4.3.5.

- System User Interface

DeviceOn provide an option to customize menu item, theme, logo and user could by setting up the

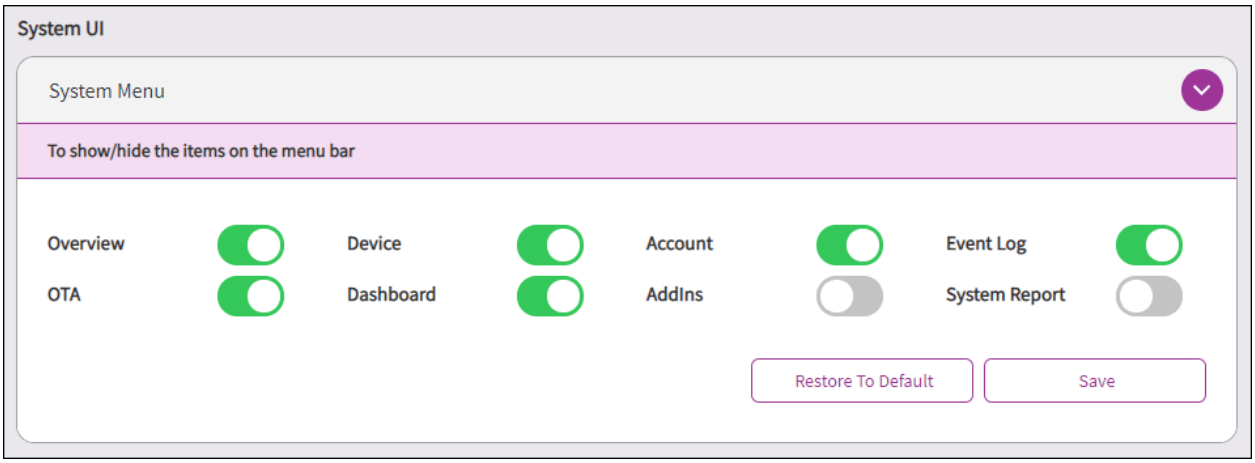
user interface to meet their needs.



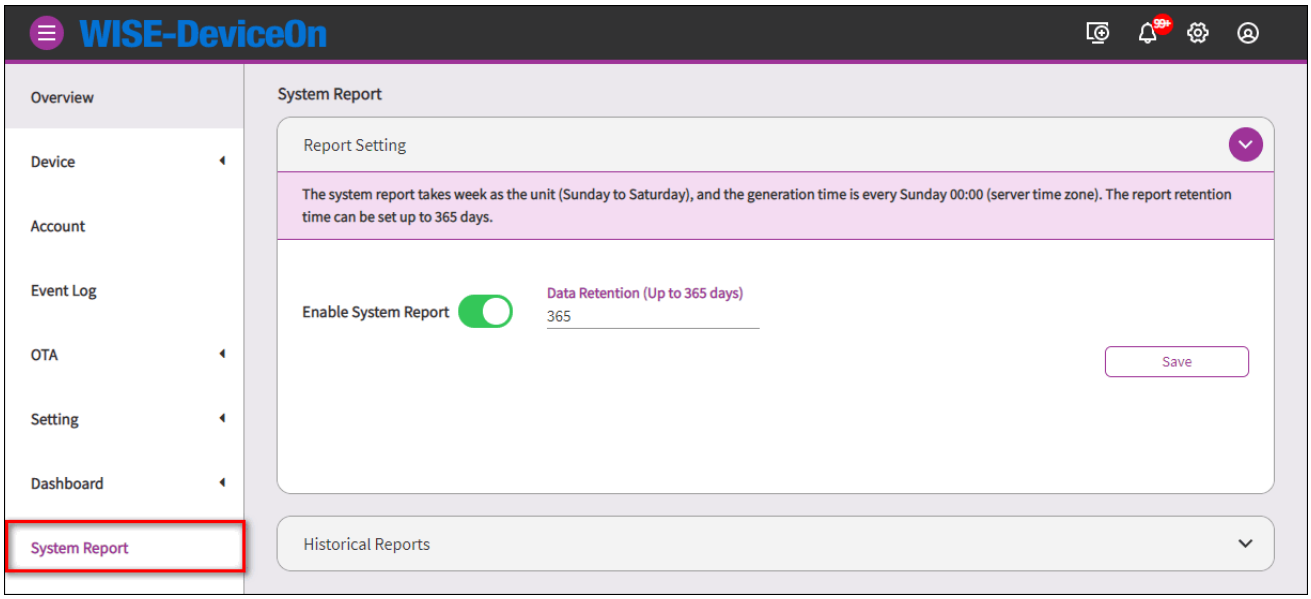


✧ System Menu: To show/hide the items on the menu bar

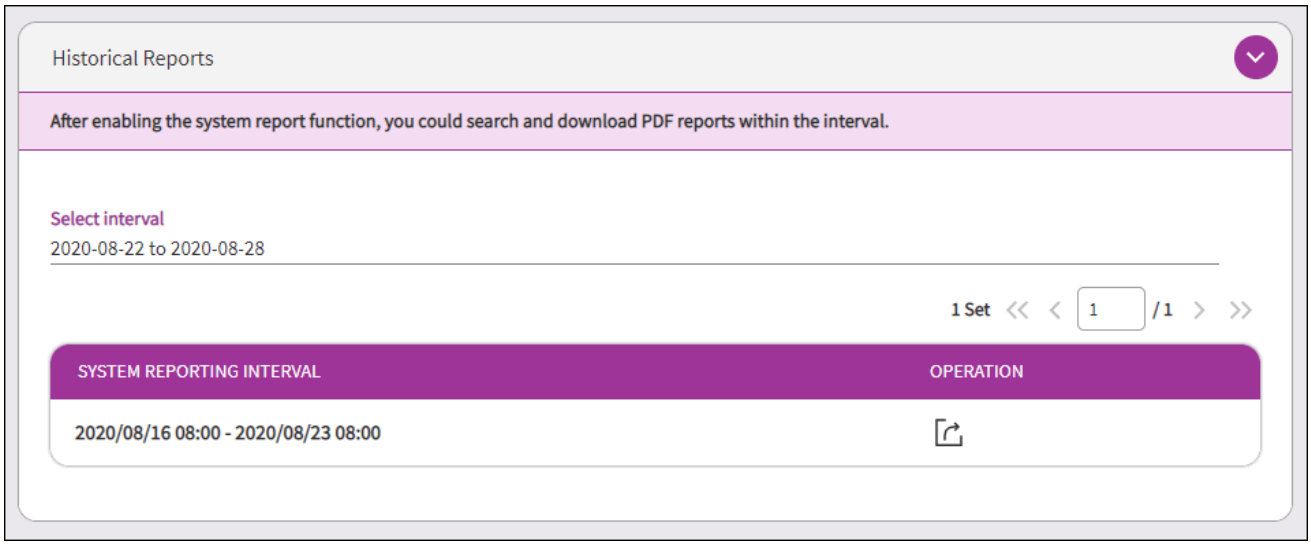
The default setting do not include “**AddIns**” and “**System Report**”, the “AddIns” is used to customized UI page or embed specific website page to integrate with DeviceOn.



Second, if the System Report be enabled will appears to the menu item. The system report takes week as the unit (Sunday to Saturday), and the generation time is every Sunday 00:00 (server time zone). The report retention time can be set up to 365 days.



After enabling the system report function, you could search and download PDF reports within the interval.



From the system report, you may realize the whole status, including server uptime, downtime and managed devices healthy for the pass week.

Weekly report

WISE-DeviceOn

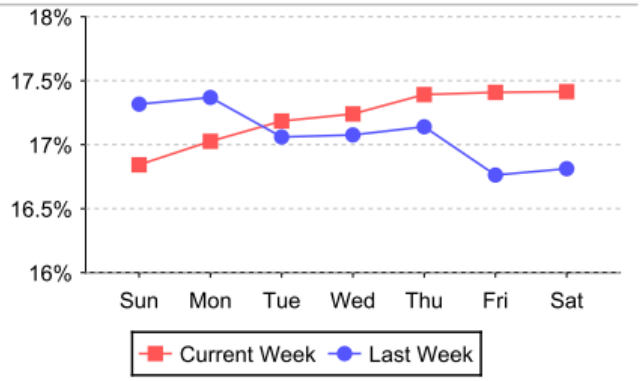
DeviceOn Overview

2020-07-05~2020-07-11

Server Overview

Avg. CPU Usage	8.78 %	percent
Average of CPU usage this week.		
Avg. Mem Usage	74.46 %	percent
Average of memory usage this week.		
Storage Growth	0.58 %	percent
Growth of storage usage this week.		

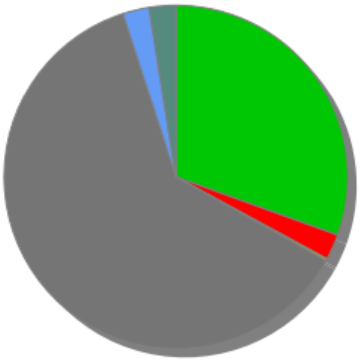
Overall Storage Usage Trend



Event	DateTime	Duration
Up	2020-07-10T09:59:17Z	37 hours, 30 mins
Down	2020-07-10T09:57:50Z	0 hours, 1 mins
Up	2020-07-10T06:21:35Z	3 hours, 36 mins
Down	2020-07-10T06:19:54Z	0 hours, 1 mins
Up	2020-07-09T08:45:27Z	21 hours, 34 mins

Up Time	Down Time
4 Days	5 Minutes
99.92 %	0.08 %

Device Overview



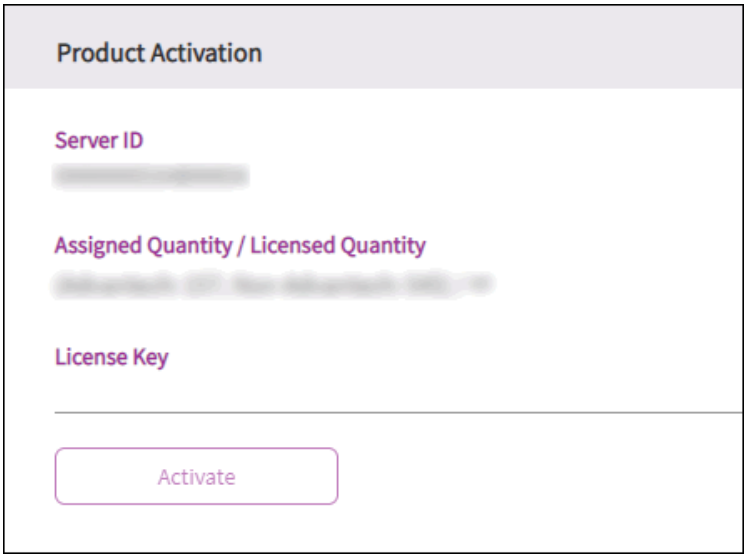
Normal	Error	Warning
989 times	78 times	7 times
30.51 %	30.51 %	30.51 %

Threshold	Disconnect	Loss Connect
2006 times	81 times	81 times
30.51 %	30.51 %	30.51 %

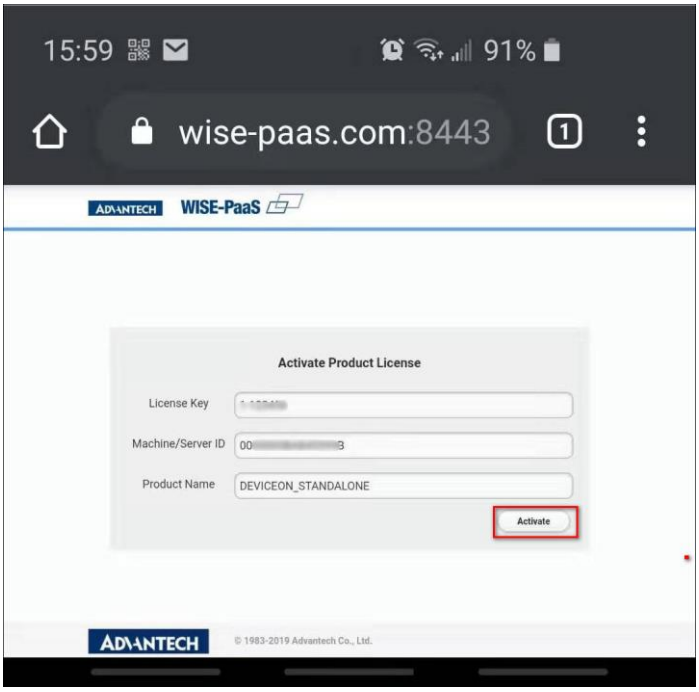
- ✧ System Theme: Select the theme style of the system
- ✧ System Logo: Product logo customized, supported formats: GIF, PNG, JPEG/JPG. We recommend the image with a height is less than 55 pixels.
- ✧ System Login Page: Login page customized, supported formats: PNG, JPEG/JPG. We recommend the image with a resolution is less than 860x840 pixels.
- ✧ Overview Setting: To show/hide the functions on the overview
- ✧ Language Setting: Set display language, (English, Traditional Chinese and Simplified Chinese)

- Product Activation

DeviceOn support online and offline to activate product, if your server could access to Advantech License Server that would be simple. Enter your license key that you purchase the product from WISE-PaaS Marketplace.

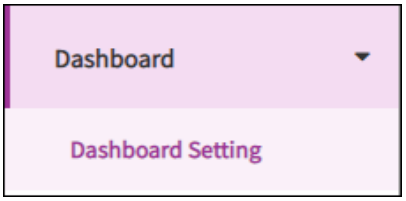


If your server environment without public network accessible, there is a QR code generated after enter the license key. Please leverage your mobile device to scan and retrieve the "**Activation Code**".

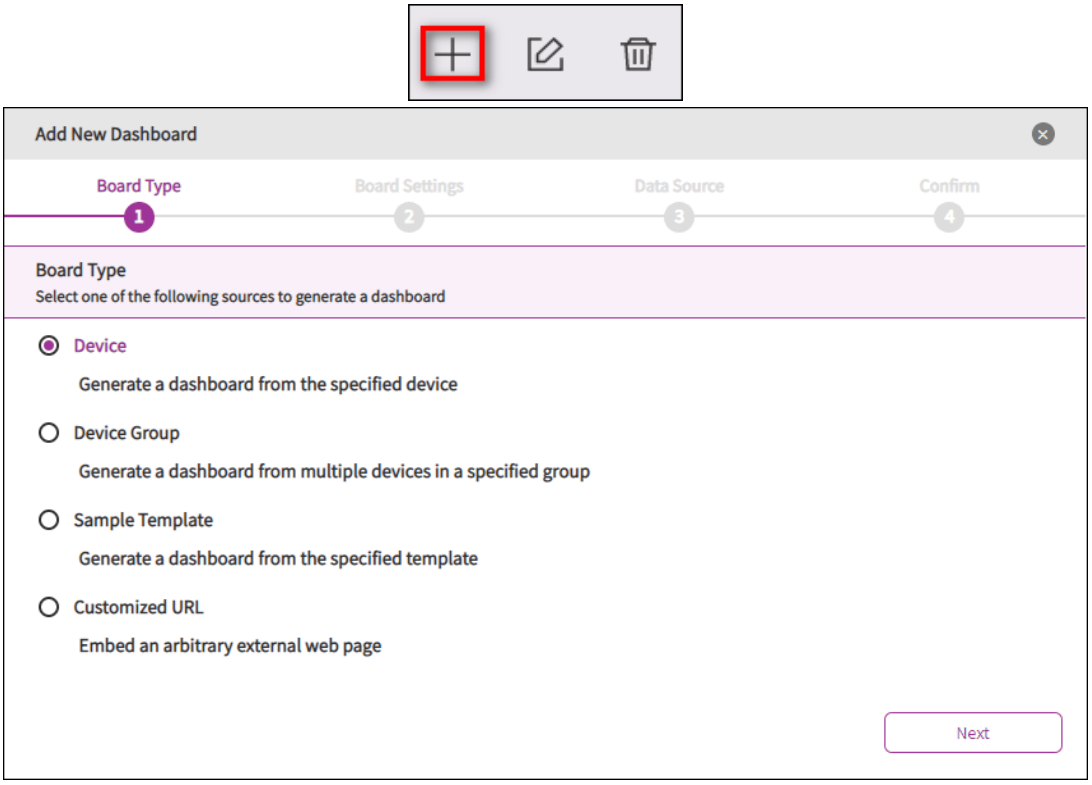


3.3.7 **Dashboard**

The DeviceOn not only native support and build-in Grafana for data visualize, but fully integrate and simplified procedure to 1-Click to generate a dashboard. Grafana is an open-source software for monitoring and analysis. One of its major characteristics is it supports many different data sources, from popular CloudWatch, Elasticsearch, Graphite, and influxDB, to OpenStack Gnocchi or Google Calendar. Its range is very extensive. However, for others data source require to implement SimpleJson to access your data.

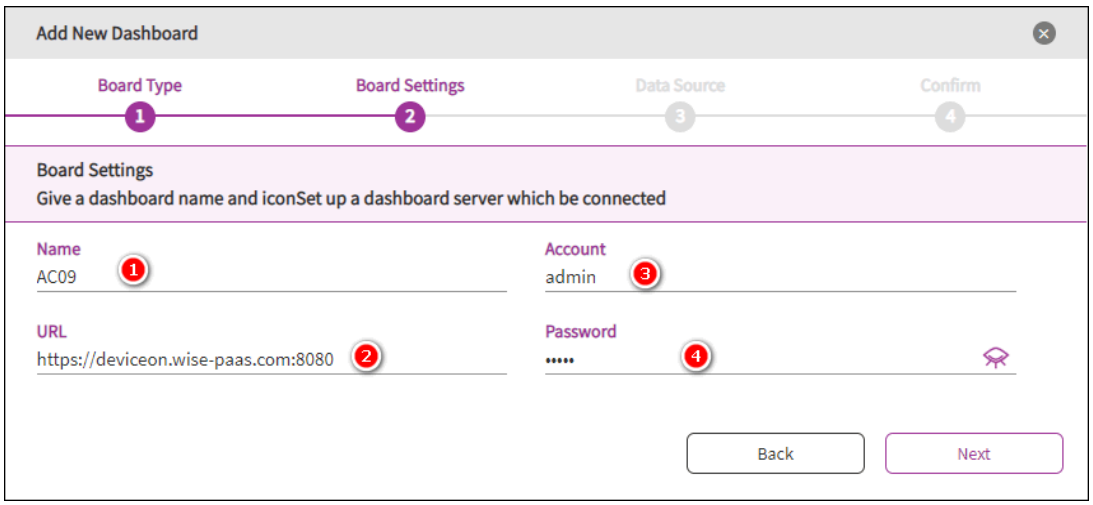


Click on the “+” icon to create dashboard item.

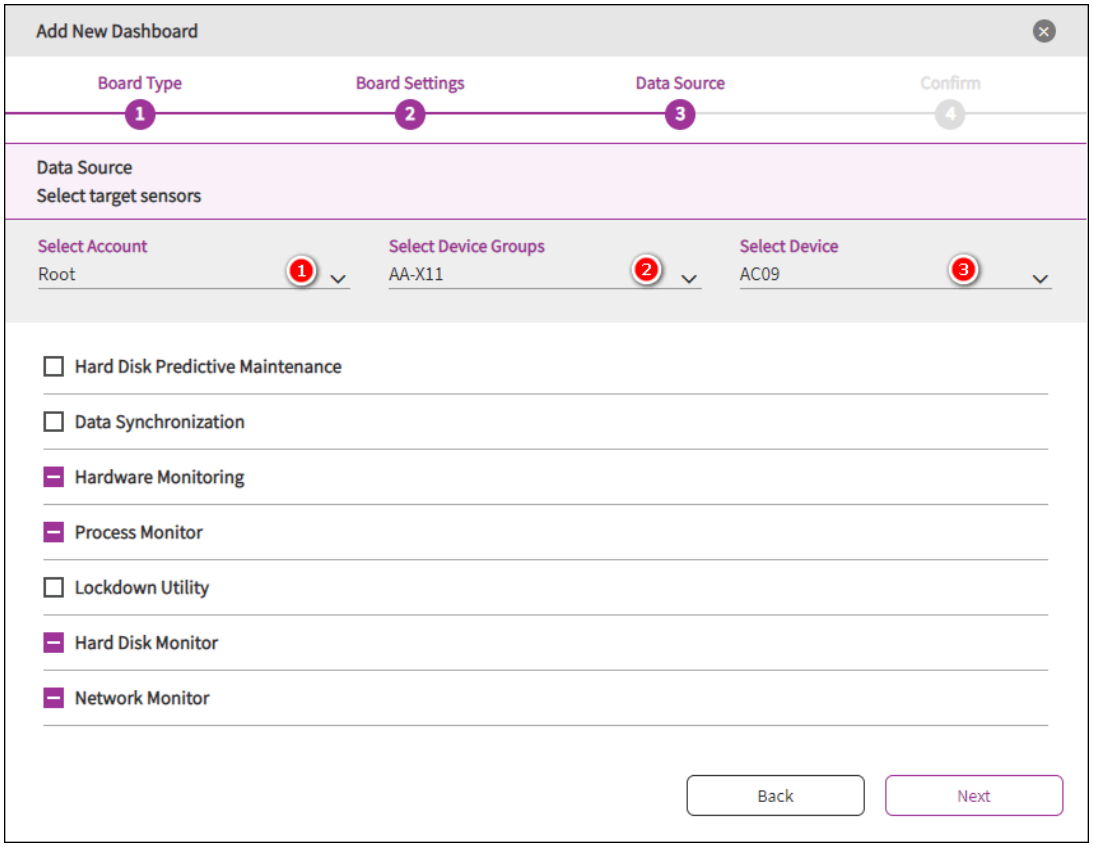


Here, DeviceOn support 4 types of board, select one of method to generate your dashboard. Device, device group, mode or select our default template to generate. Last, embed an arbitrary external web page.

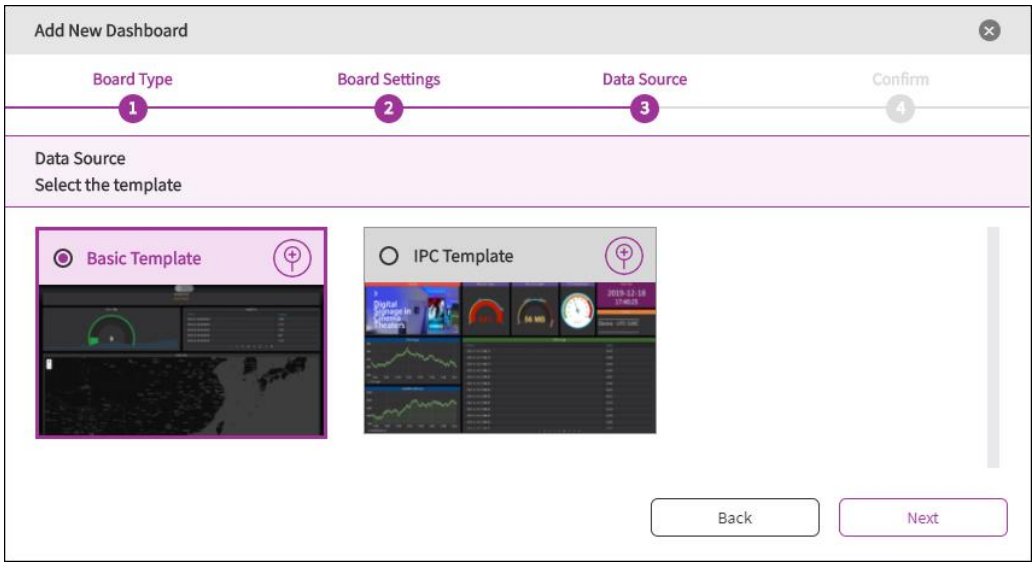
Enter your board name, Grafana URL, account and password. (Default account and password is “**admin**”)



Select target sensor from the device and click next.

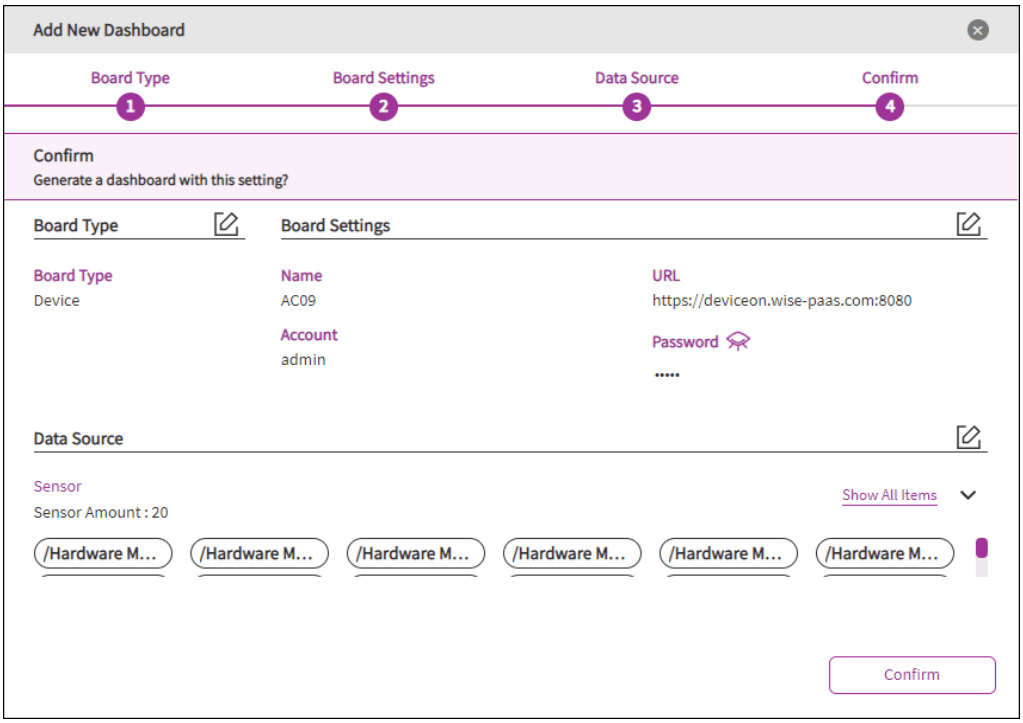


(Device Method)

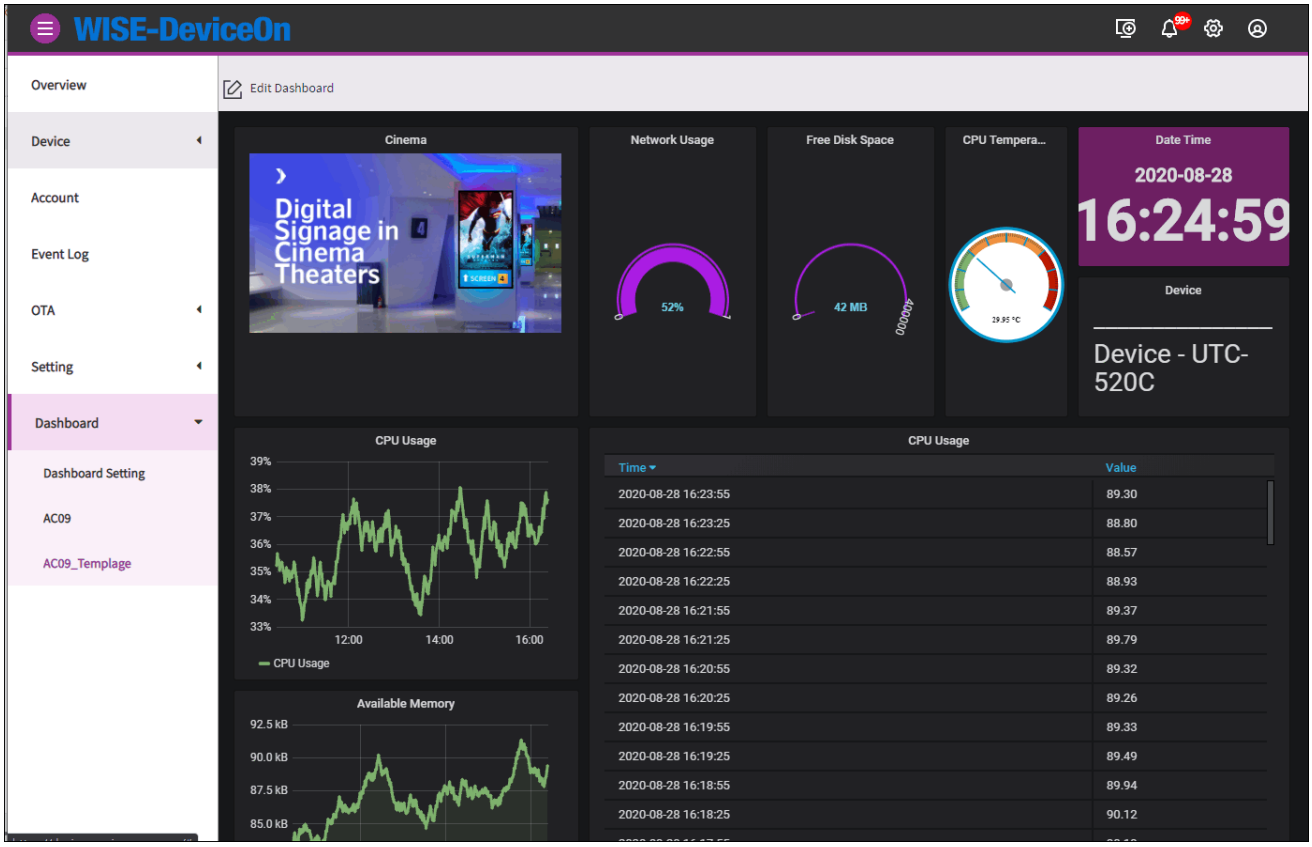


(Template Method)

Confirm the result and information, and start to generate.



After that, the board be generated on the menu item.



4. Hands-On LABs

4.1 How to Create a Real-time Action into Overview

The real-time action is a handy way to execute a specific command to a bunch of devices. This lab guides you how to create a real-time action. And, after this lab, you should:

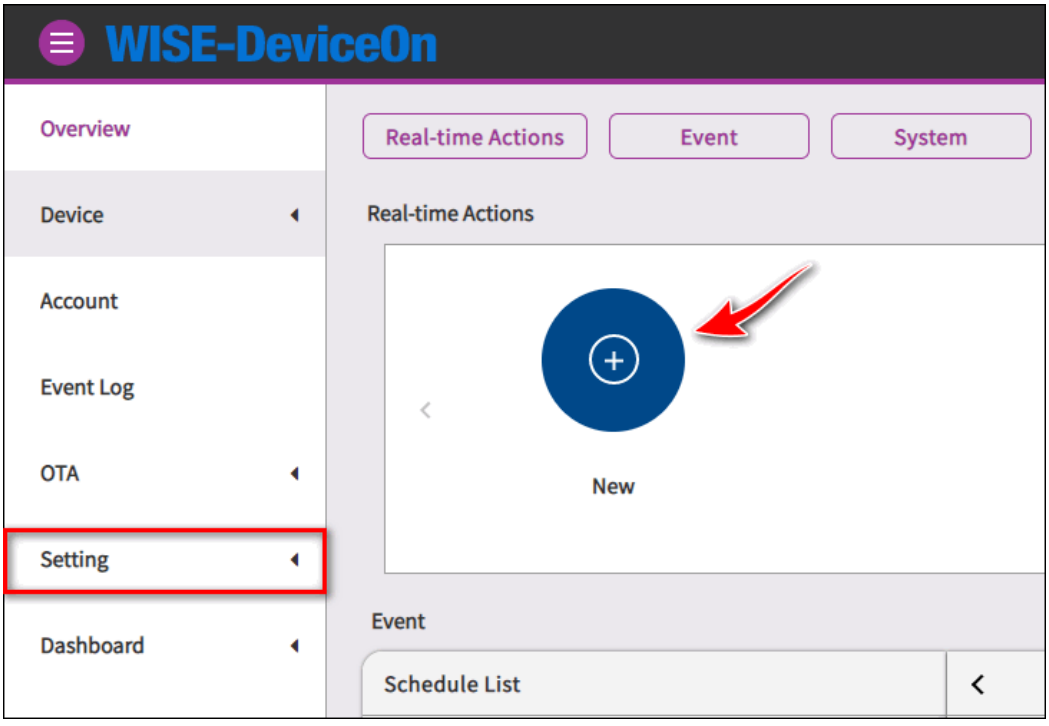
- Learn how to create a real-time action on demand.
- Know of what actions DeviceOn provides.
- Have an action named **"MyTask"** and pinned into your **"Overview"** page, that can reboot devices belong to group **"Default"**.

4.1.1 Prerequisite

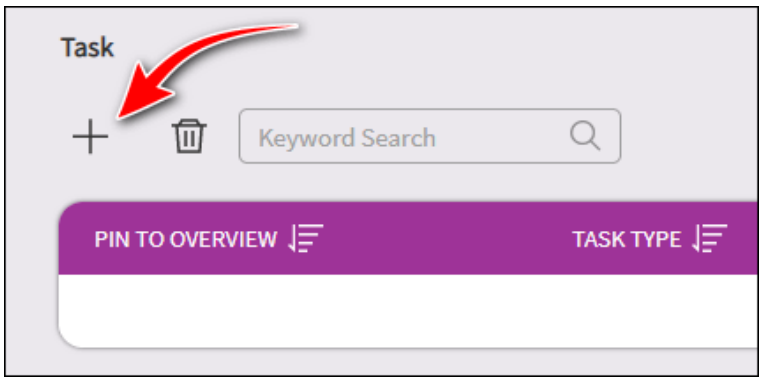
- A running DeviceOn server.
- A device that installed WISE-Agent connects to DeviceOn server.

4.1.2 Step-by-Step

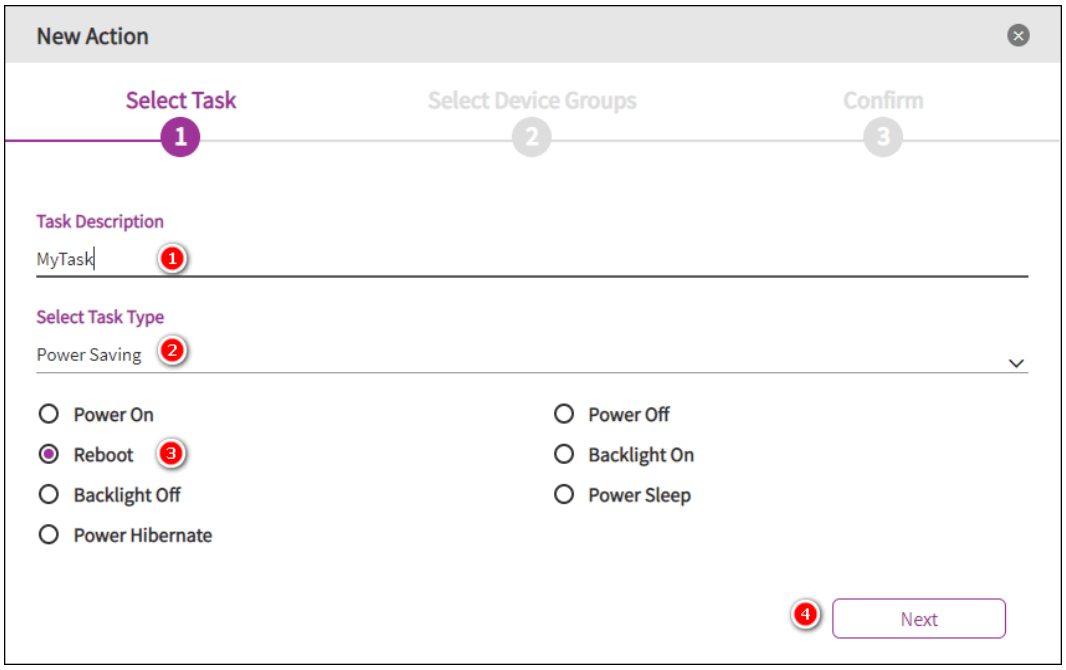
Step 1: To create a real-time action, click the **"New"** icon in **"Overview"**. Alternate, click **"Setting"** from the menu populated in left hand side.



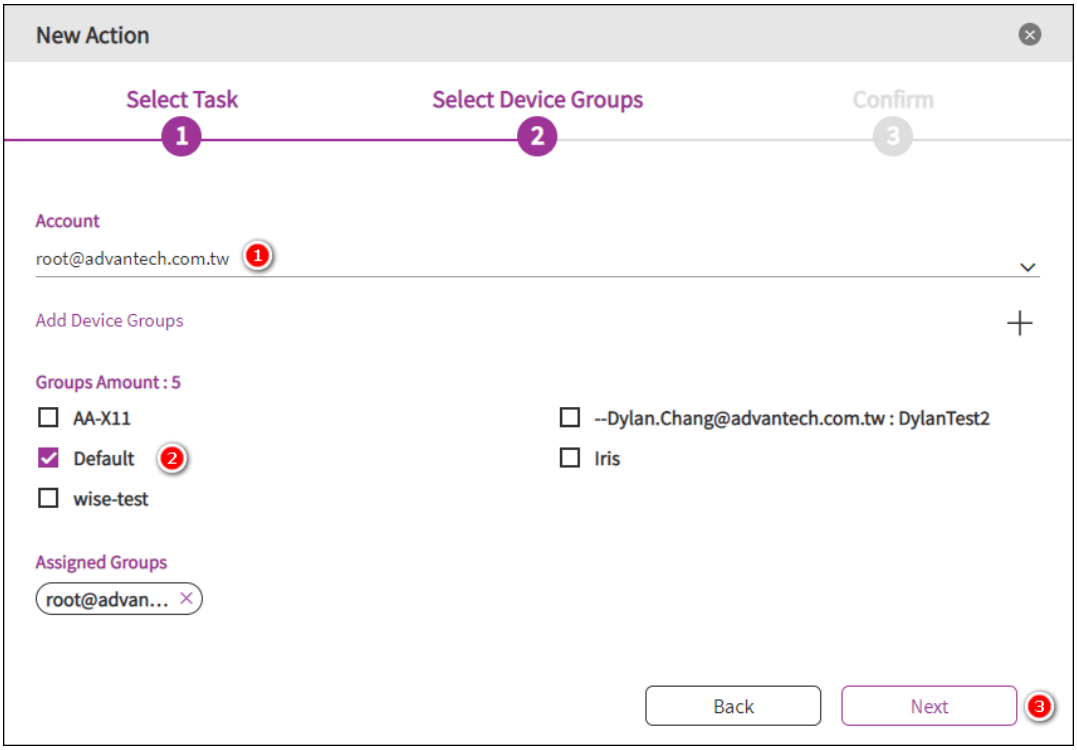
Step 2: Either way you use in step 1, it leads you into the **"Add Task"** page. Click the **"+"** symbol.



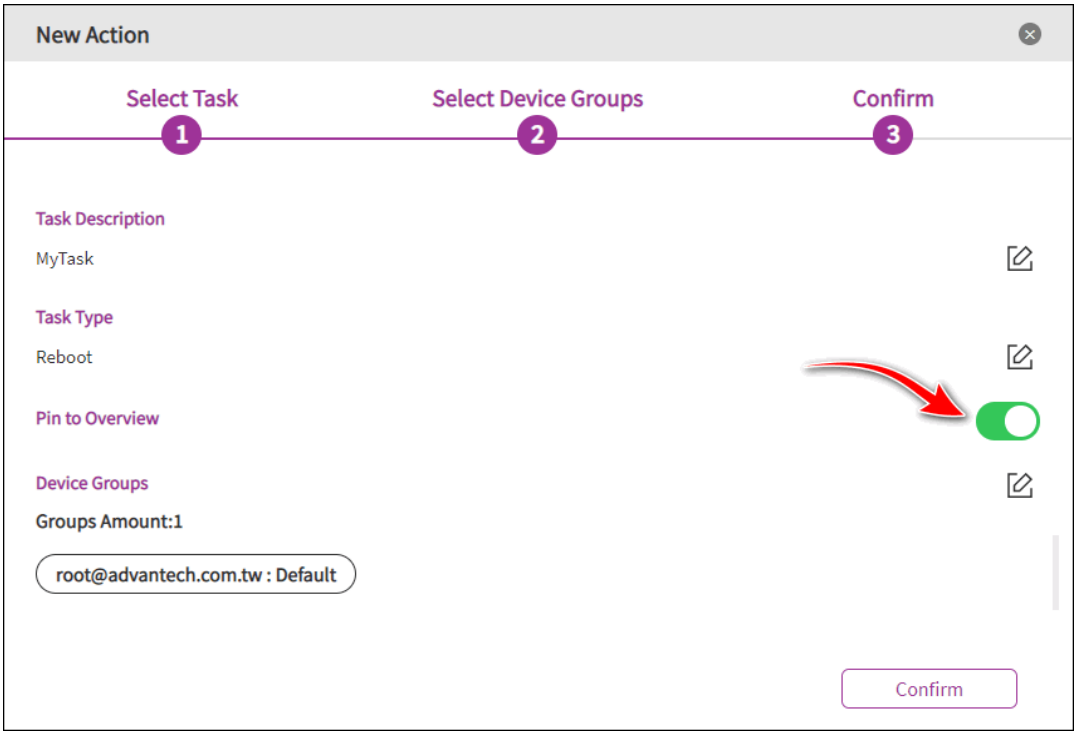
Step 3: You now run into the first page **"Select Task"** to create a new real-time action. Enter the task name **"MyTask"** as well as choose the action **"Reboot"** within this page. From this page you can see all tasksDeviceOn provides. Then end this page up with clicking **"Next"** button.



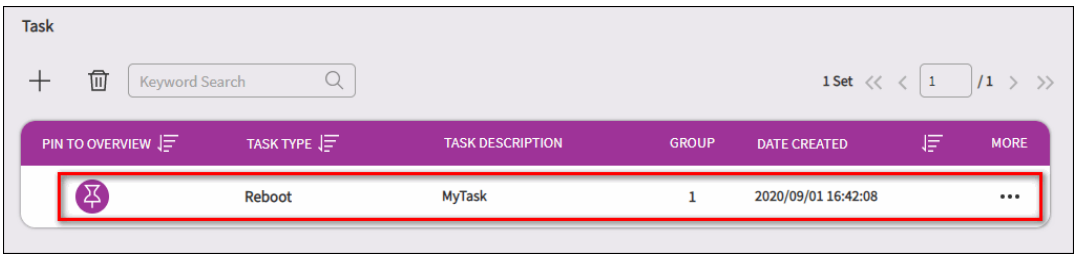
Step 4: Choose the target group **"Default"** to execute the real-time action in **"Select Device Groups"** page.

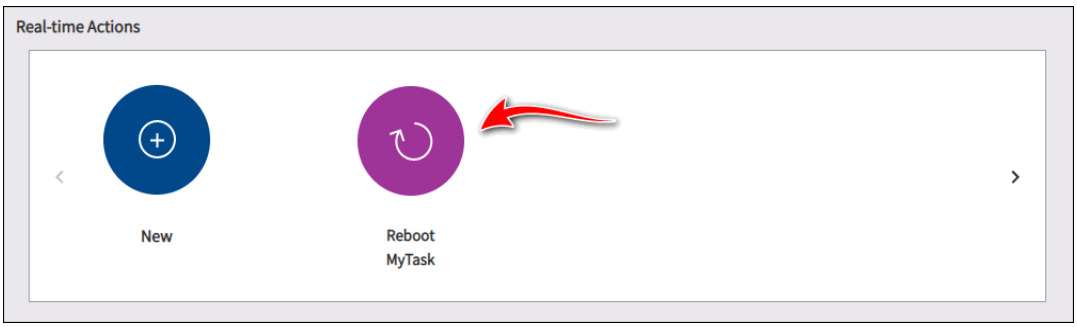


Step 5: The last page **"Confirm"** provides you a summary like information and, more than those, lets you decide whether this action **"Pin"** to your **"Overview"** page or not. DeviceOn turns this feature on by default. Just toggle it if you don't want this action pin to your home. Finally, click **"Confirm"** button to finish.



If everything goes well, you should see there is a new item generated within **"Task"**. Meanwhile, if you go to your home (page **"Overview"**), you can see a new one action icon is populated there.





What should, or can, you do now? Yes, one-click that icon you created from **"Overview"** page, and watch the devices whether they execute reboot action.

4.2 How to Remote Software Provisioning via OTA

OTA (Over-The-Air) is another powerful feature DeviceOn provides. Users can deploy software packages onto a device remotely, or even many devices broadly. This lab guides you how to accomplish remote software provisioning via OTA. And, after this lab, you should:

- Learn how to remote provisioning your software via OTA on demand.
- Learn how to package your software for remote provisioning.
- Have the NotePad++, a popular and famous text editor, populated within the target device.

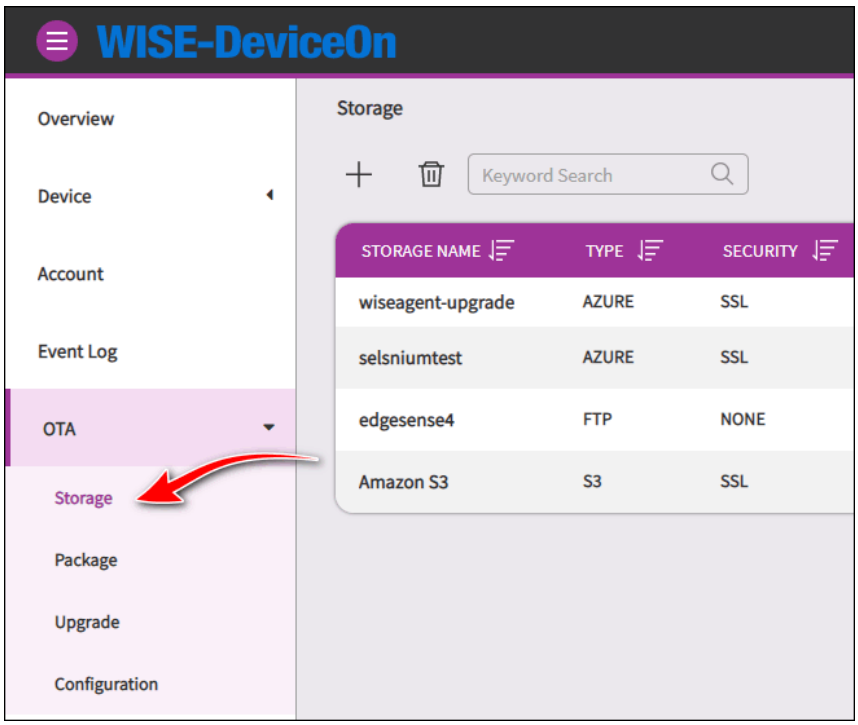
4.2.1 Prerequisite

- A running DeviceOn server.
- A device which running on Windows operating system and installed WISE-Agent, that connects to DeviceOn server.
- A running, with well configured, FTP server as the storage.
- A NotePad++ installer, 32-bit edition is recommended. Its name is **"npp.7.8.2.Installer.exe"**, something like that. It can be downloaded from https://notepad-plus-plus.org/downloads/.
- Automation skills to install target software package. It is because that user intervention is not possible during provisioning via OTA. For Windows it can be batch file or power shell, while for

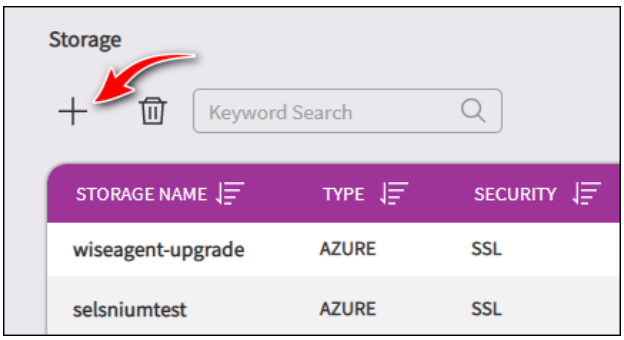
Ubuntu it may be shell scripts.

4.2.2 Step-by-Step

Step 1: Click **"OTA"** from the menu on left hand side. It leads you into the **"Storage"** page.



Step 2: In **"Storage"** page, click the plus (**+**) sign. This step leads you into the **"Add New Storage"** page. You have to add a new storage to upload new packages.



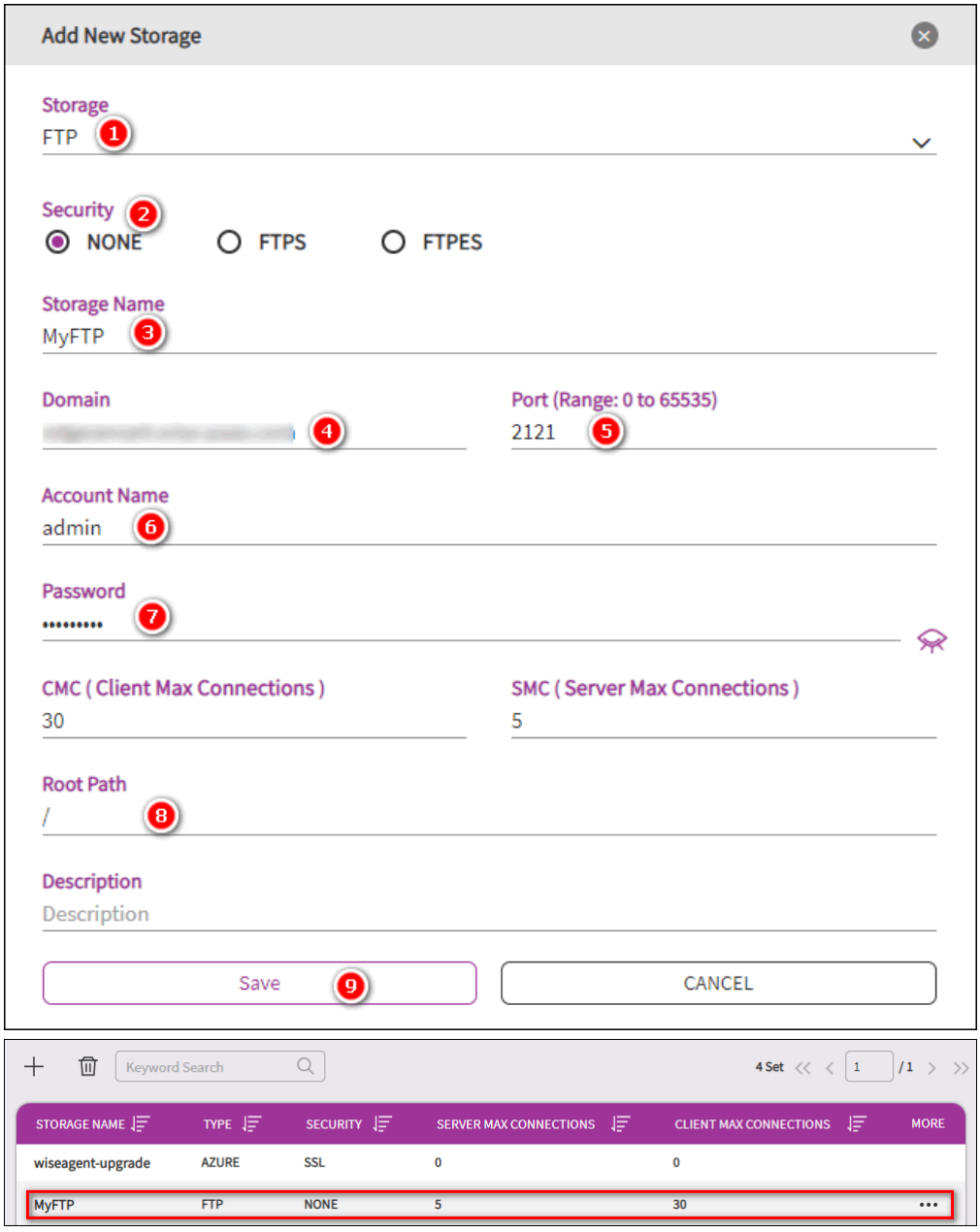
Step 3: Fill all fields in with proper values like following:

- **SOTRAGE:** Pick **"FTP"** from the dropdown lists.
- **Security:** Leave it as **"NONE"**, the default value. If your FTP server running on FTPS protocol, pick **"FTPS"**.
- **SOTRAGE NAME:** Enter **"MyFTP"**.
- **DOMAIN:** Enter the FQDN of your FTP server, or its IP address.
- **PORT:** Should be **21** if the FTP server runs on a standard port number.
- **ACCOUNT NAME:** A valid username that can connect to the FTP server, and upload files onto

the server as well.

- **PASSWORD:** The password to login.
- **CMC/SMC:** Use defaults.
- **ROOT PATH:** Simply uses **"/"**.
- **DESCRIPTION:** Leave it empty. It's optional information.

Finally, click **"Save"** button to finish this step. If it goes well, you should see a new table row regarding this FTP storage populated in **"Storage"** page.

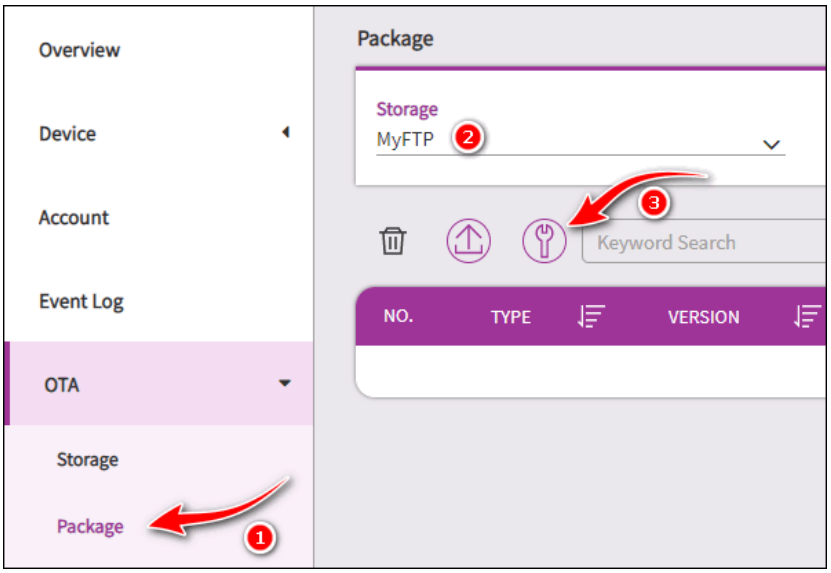


Step 4: An extra step we need to execute prior to next step: prepare a valid package for OTA. DeviceOn provides users a toolkit to pack all stuff to be a valid OTA package.

1. Create a new folder names **"NPP"** in, say, your desktop.

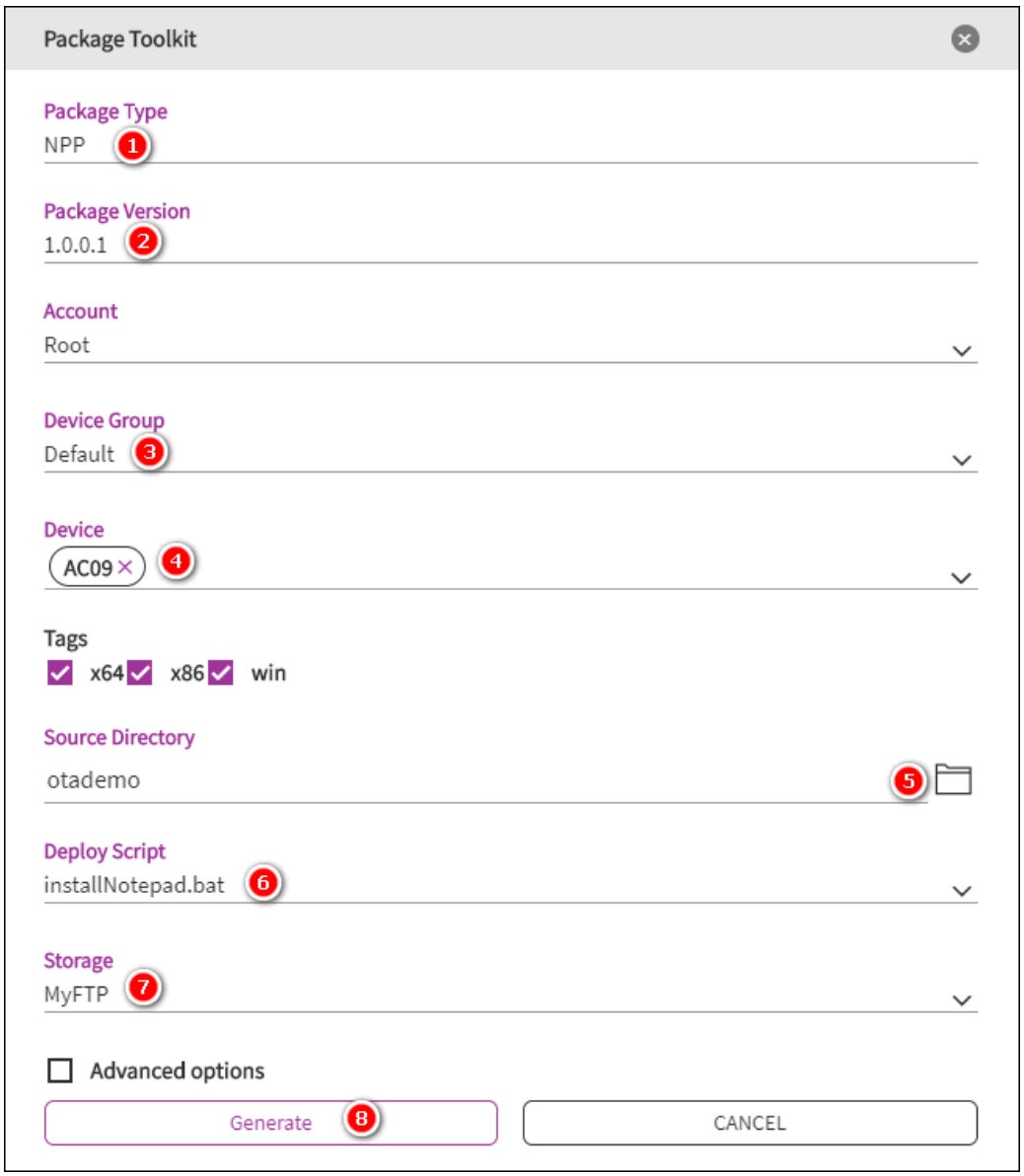
2. Move the downloaded file **"npp.7.8.2.Installer.exe"** into.
3. Create a new file **"install.bat",** contains only **start /wait npp.7.8.2.Installer.exe /S**, inside. This command, per its document in official web site, installs the downloaded NotePad++ software silently.

Step 5: Now click the **"Package"** item. And, then, choose **"MyFTP"** from **"STORAGE"** field. Last, click the **"Package Toolkit"** icon to enter **"Package Toolkit"** page.

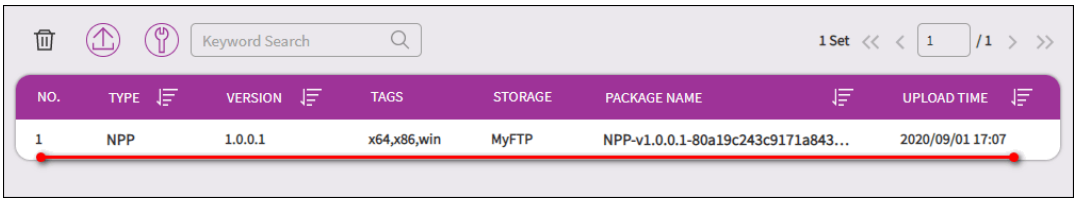


Step 6: In **"Package Toolkit"** page, fill all mandatory field up with proper values. At last, click **"Generate"** button to package **"NPP"** software, and upload onto **"MyFTP"** storage as well.

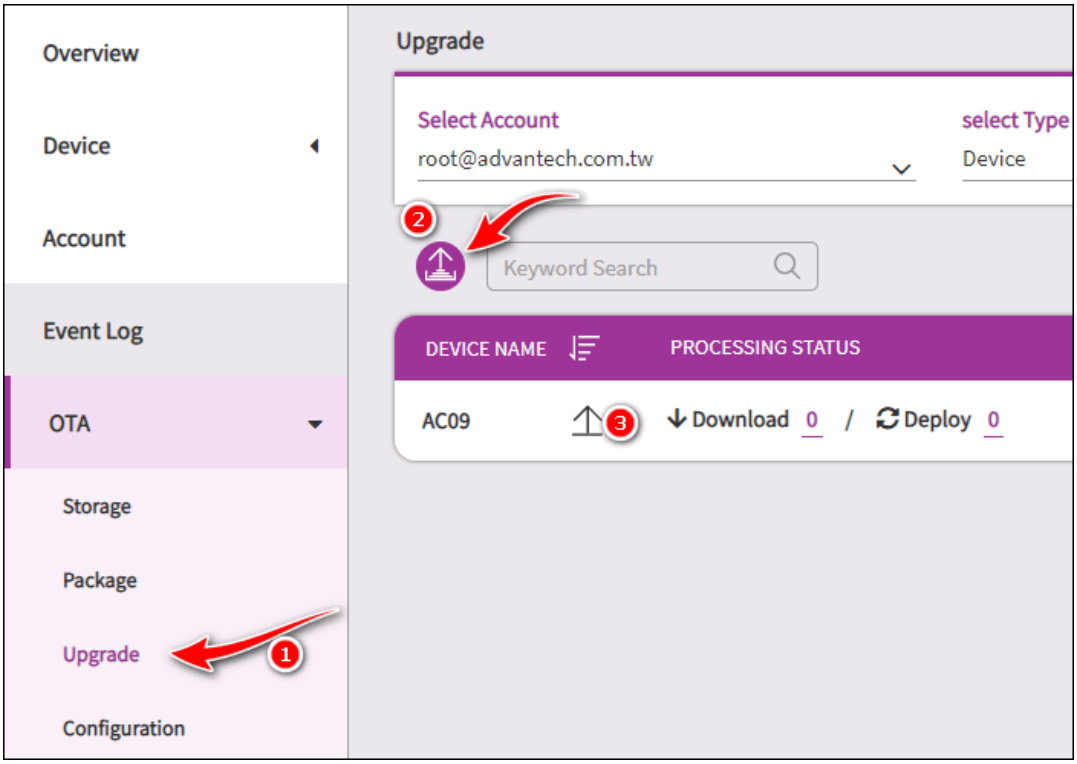
- **Package Type:** Fill **"NPP"** up.
- **Package Version:** Fill **"1.0.0.0"** up.
- **Device Group:** Choose **"Default"**.
- **DEVICE:** Choose the target device. **"AA-Win"** in this lab environment.
- **SOURCE DIR:** Click **"Browser"** to point to the location of **"NPP"** folder we created in step 4.
- **DEPLOY FILE:** DeviceOn chooses **"install.bat"** for you.
- **STORAGE:** Choose **"MyFTP"** from dropdown list.



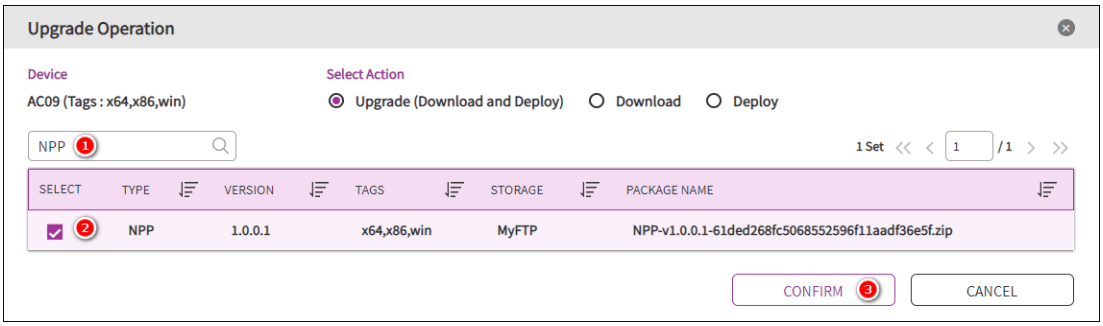
Step 7: Now, in **"Package"** page, a new one table row represents the **"NPP"** package has been added.



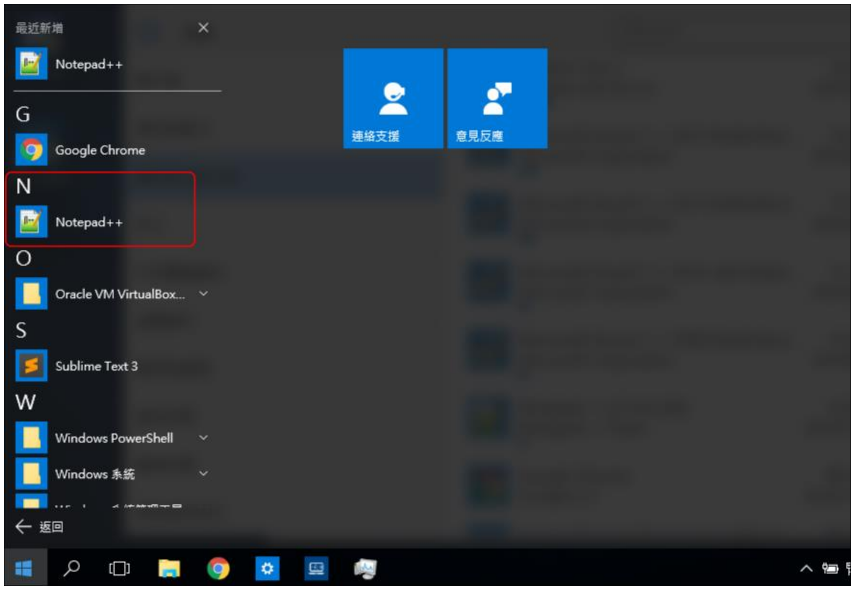
Step 8: It is time to install NotePad++ onto the target device remotely. Based on previous step, click **"Upgrade"** tab next to **"Package"** tab. You should find the target device shows there within the table view. Click the icon locates in target device row and **"UPGRADE"** column. It leads you into the **"Upgrade Operation"** page.



Step 9: In **"Upgrade Operation"** page, fill **"NPP"** up in **"KEYWORD SEARCH"** field so that the package can be filtered out of all packages. Check the box accordingly and click **"FONFIRM"** button.



Step 10: Now the NotePad++ should been be installing and, after a while, if everything went well, a corresponding application item should be created in Windows menu.



4.3 How to Set a Device Threshold and Event Notify Services

For devices monitoring, DeviceOn provides the rule engine. Users can acquire anomaly situations by means of setting thresholds to those interested devices, and, once one or more thresholds meets, receive alerts via event notification services, another one indispensable feature for users. This lab guides you how to set thresholds to a device and how to set event notification services as well. As such, after this lab, you should:

- Learn how to set thresholds to a device on demand.
- Learn how to set event notification services, including email, LINE, and WeChat as well.

4.3.1 Prerequisite

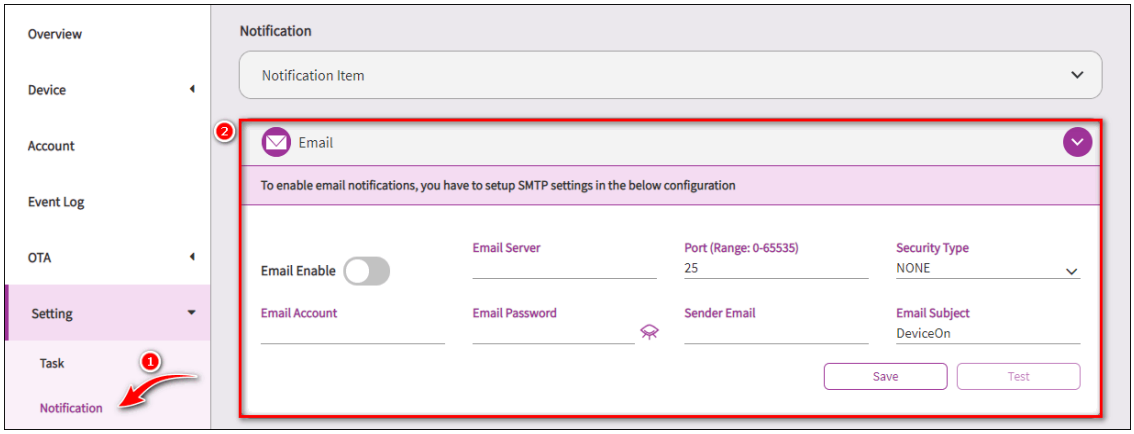
- A running DeviceOn server.
- A device that installed WISE-Agent connects to DeviceOn server.
- A valid, send-able, email account to enable Email notification service.
- A valid LINE account to enable LINE notification service.
- A valid WeChat account, as well as a valid GitHub account, to enable WeChat notification service.

4.3.2 Steps to Set Event Notification Service – Email

The configuration of using email as one of event notification services is a system-wide setting. This means DeviceOn uses the server, the one you set in this step, to send all emails. Therefore, uses email settings from your organization is recommended, rather than uses your personal Gmail. If you really want to use Gmail, the situations you are running into may vary and depends on your google account

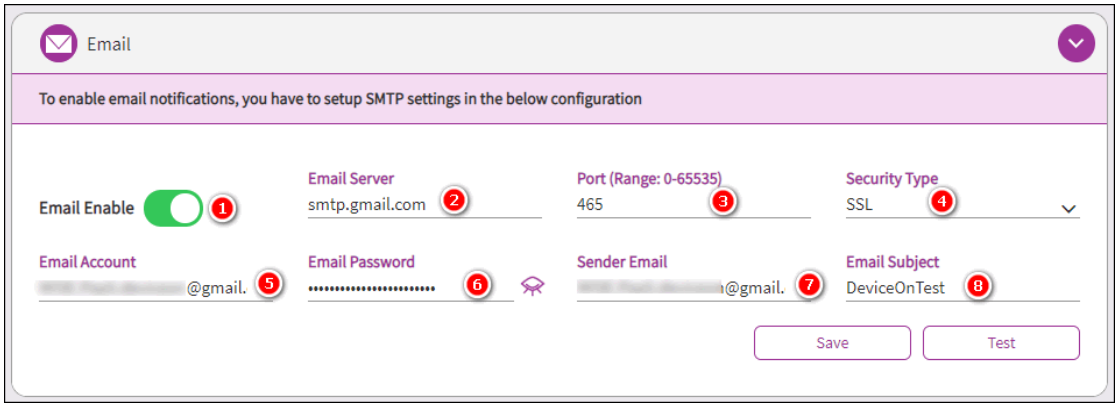
settings. So, in this lab, we assume that you have already a valid business email address from your company.

Step 1: Click **"Setting"** menu on the left-hand side of DeviceOn portal and, then, **"Notification"**. Click **"EMail"** bar to open settings regarding email notification service.

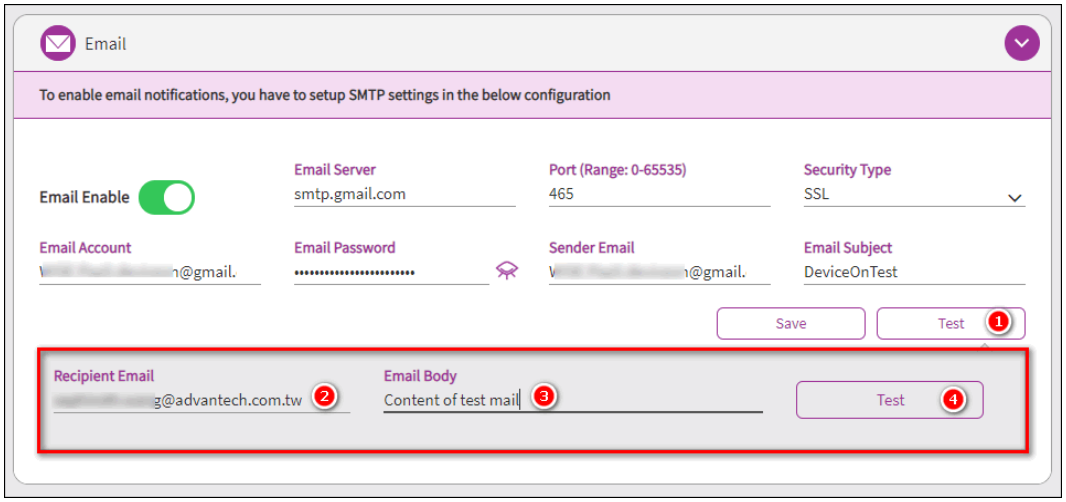


Step 2: Toggle **"On/Off"** switch to enable this feature. Then fill fields up with proper values. And end up this step by clicking **"Test"** button.

- **EMAIL SERVER:** The email server host name.
- **PORT:** The email server port. Normally this is 25.
- **SSL/TLS:** Toggle to a proper setting.
- **EMAIL ACCOUNT:** Your email account name. If takes the windows domain into account, a value format like **"DOMAIN\USER"** should be used.
- **EMAIL PASSWORD:** Your password to sign in to the email server.
- **SENDER EMAIL:** Your email address.
- **EMAIL SUBJECT:** Leave it the default.



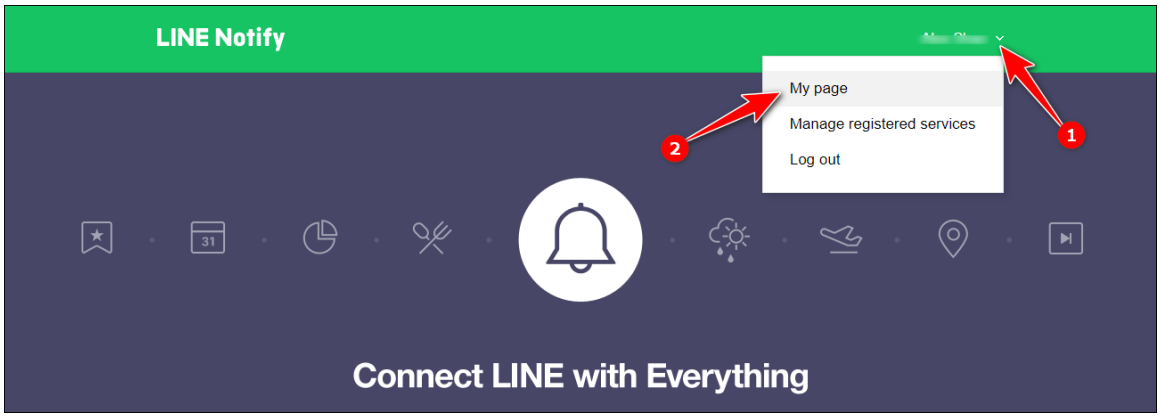
Step 3: To assert all values are correct, click **"Test"** button, on the bottom right of the page, to open the **"Send Email for Testing"** dialog for testing purpose. And fill a recipient email as well as email body. Then click "Test" on this dialog. An email you should receive in a while later. Revise them until you got a test email.



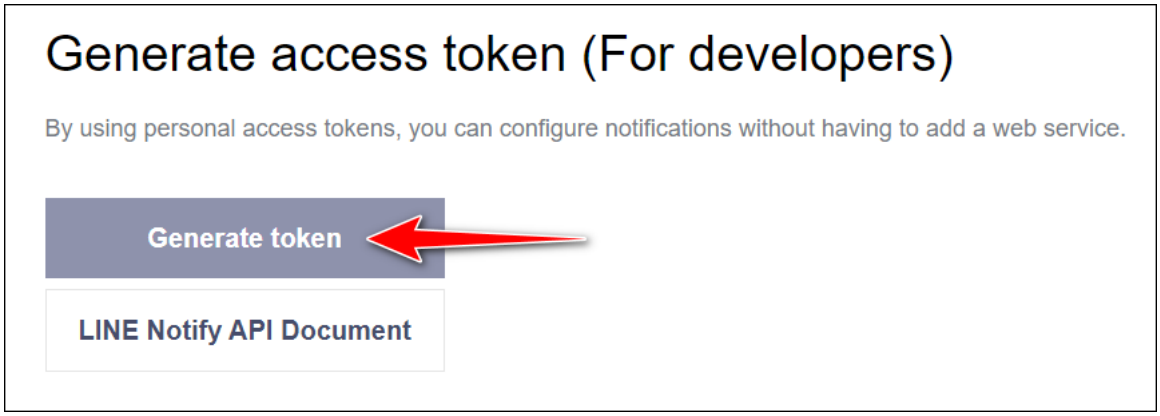
Step 4: Click **"Save"** on the bottom right of the page that shows in step 2 to keep all settings and enable email notification service.

4.3.3 Steps to Set Event Notification Service – LINE

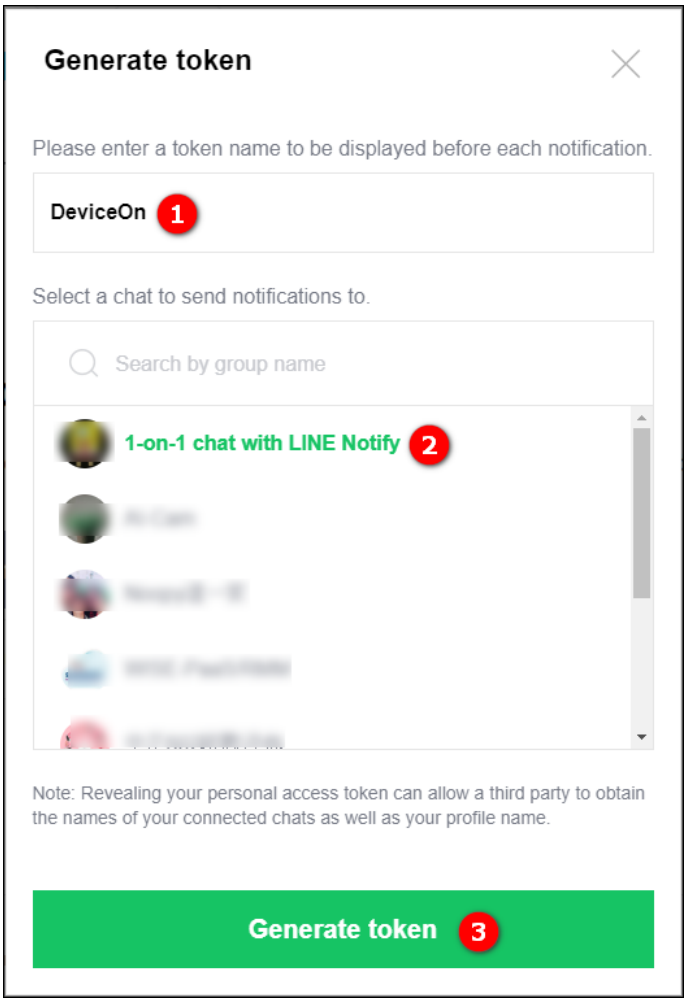
Step 1: Go to https://notify-bot.line.me/ and sign in with your LINE account. Click **"My Page"** from your account's dropdown menu in the upper right of the page.



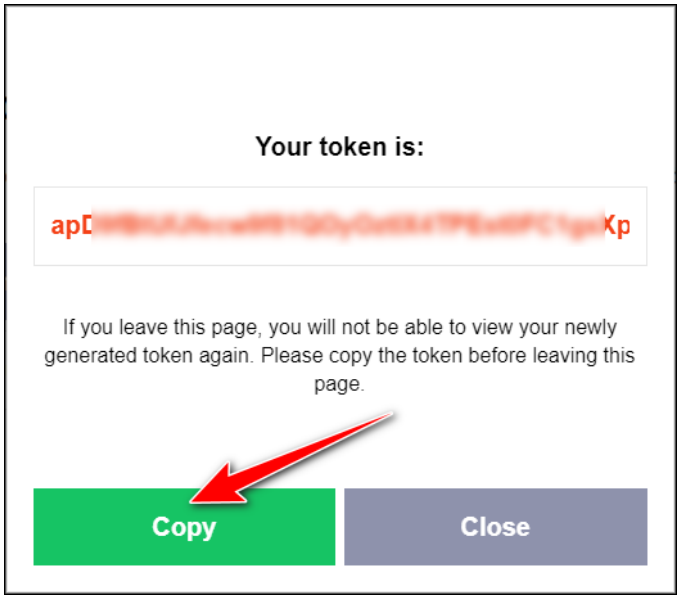
Step 2: Click **"Generate token"** under **"Generate access token (For developers)"**. It pops up the **"Generate token"** dialog.



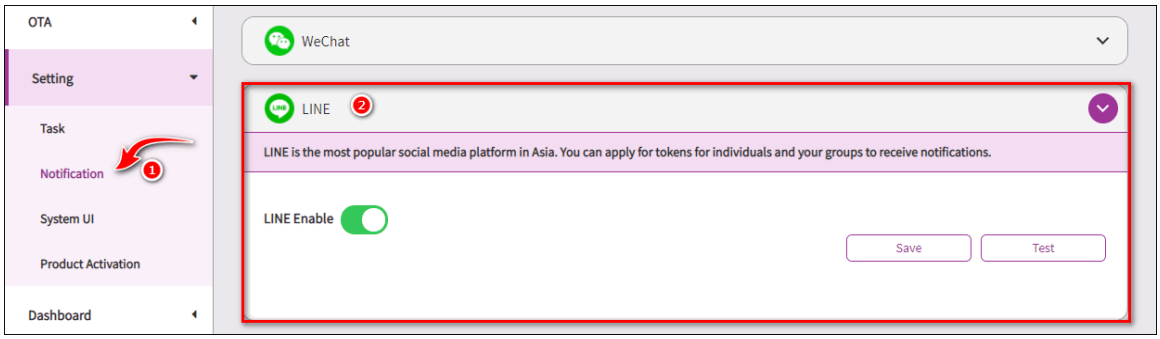
Step 3: Fill token field up with **"DeviceOn"** and click the **"1-on-1 chat with LINE Notify"** item. Then click the **"Generate token"** button in green at bottom.



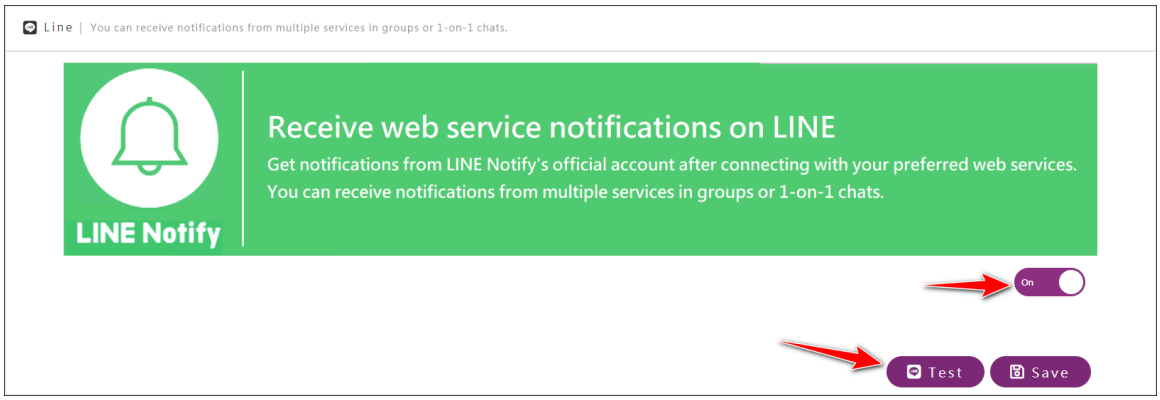
Step 4: A new window pops up with token. Meanwhile, a LINE message about this token generation received immediately. Click **"Copy"** to keep the token in memory, or any file you like.



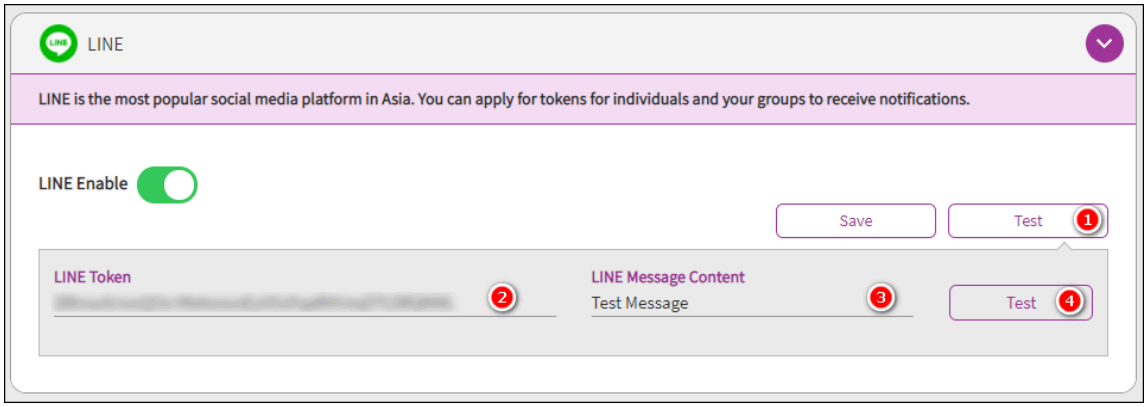
Step 5: Now switch your browser to DeviceOn portal. Click **"Setting"** menu on the left-hand side, then **"Notification"**, and last **"LINE"** bar to open settings regarding LINE event notification service.



Step 6: Toggle **"On/Off"** switch to enable this feature. Click **"Test"** to show the test dialog up.



Step 7: Paste the copied token into the first field (LINE Token) and write something into the second field (LINE Message Content). Click **"Test"**, you should receive the messages you wrote with "DeviceOn" as the prefix.

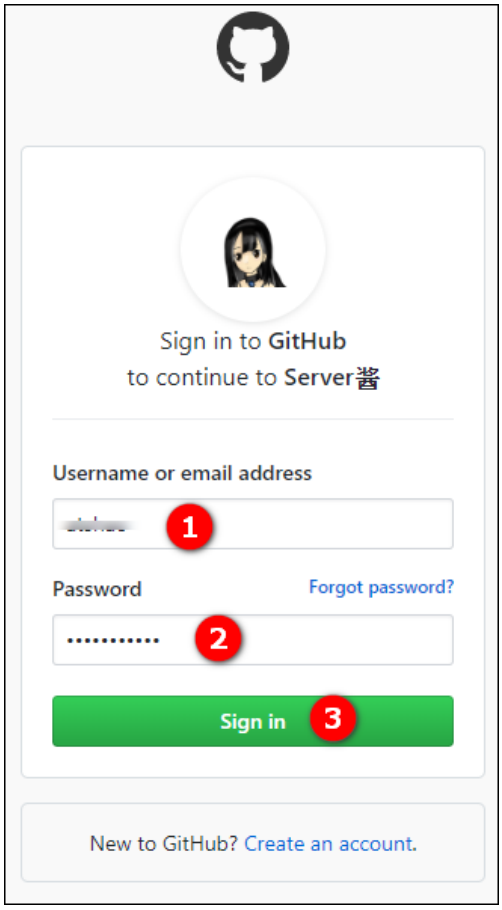


Step 8: Click **"Save"** button that shows in **Step 6** to keep your settings and enable LINE event notification service.

4.3.4 Steps to Set Event Notification Service – WeChat

Step 1: Go to http://sc.ftqq.com/3.version. Click **"登入网站"** hyperlink.

Step 2: Sign in with your GitHub account.



Step 3: Click “微信推送” hyperlink.

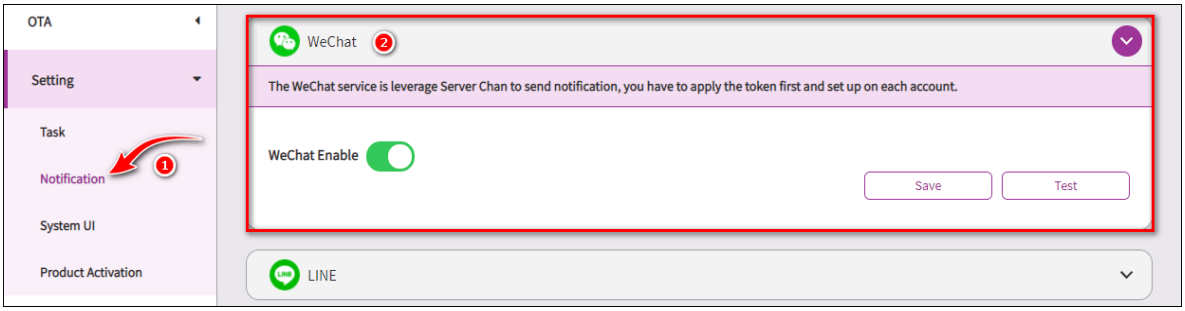
Step 4: Click “开始绑定”. It opens a QR code image.

Step 5: Take your mobile up, swipe and open WeChat App to scan this generated QR code so that the service can bind with your WeChat account.

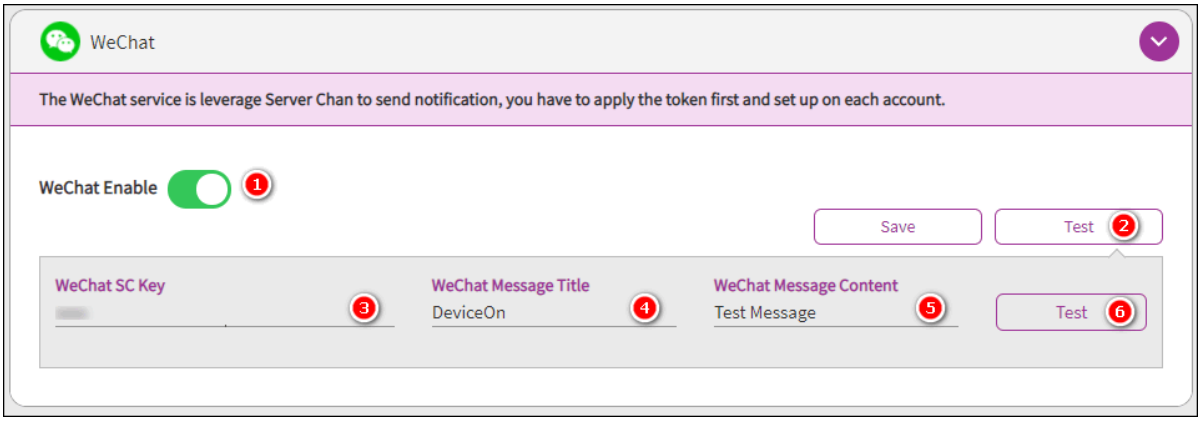
Step 6: Once it is done. The page changes, like below.

Step 7: Click **"SCKEY"** hyperlink and copy, from the opened page, the SCKEY value.

Step 8: Now switch your browser to DeviceOn portal. Click **"Setting"** menu on the left-hand side, then **"Notification"**, and **"WeChat"** to open settings regarding WeChat event notification service.



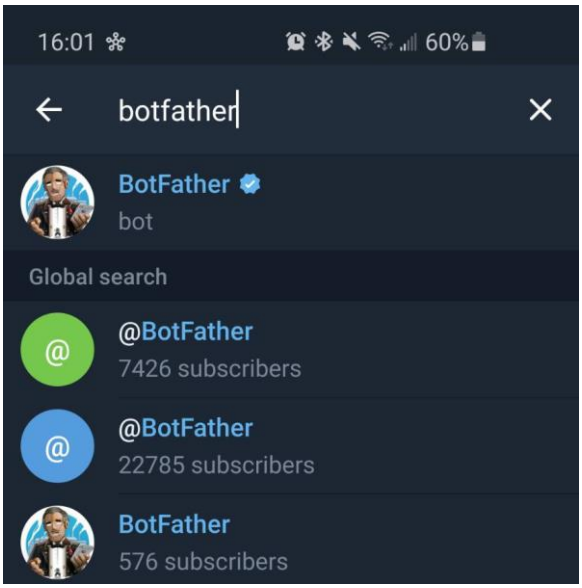
Step 9: Toggle **"On/Off"** switch to enable this feature. Click **"Test"** to show the test dialog up. Paste the copied SCKEY, copied in step 7, into the first field **"WECHAT SC KEY"**. Give a title to the second field **"WECHAT MESSAGE TITLE"**. Write some message content to the last field **"WECHAT MESSAGE CONTENT"**. And click **"Test"** to see if it works or not.



Step 10: Click **"Save"** button that shows in step 9 to keep your settings and enable WeChat event notification service.

4.3.5 Steps to Set Event Notification Service – Telegram

Step 1: Search "BotFather" and start to chat on your Telegram App.

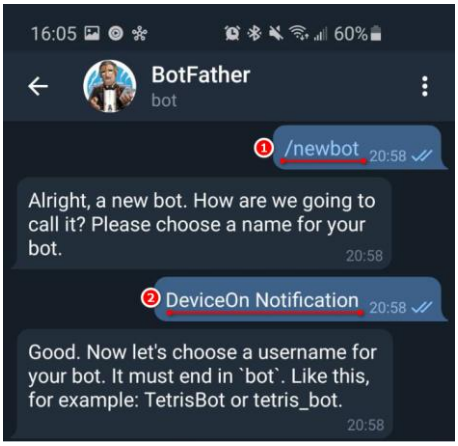


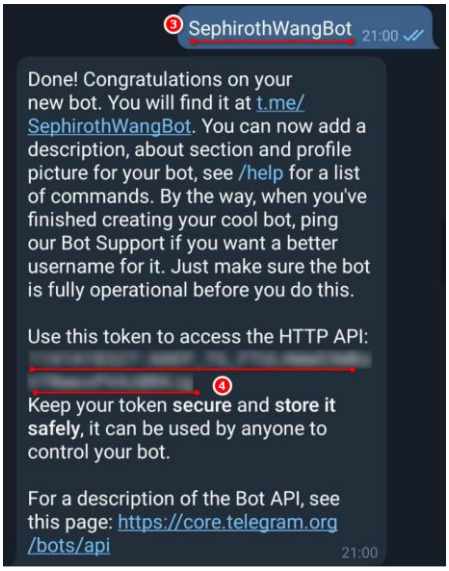
Step 2: Create a new bot and generate an **authorization token**.
Use the **/newbot** command to create a new bot. The BotFather will ask you for a name and username, then generate an authorization token for your new bot. The name of your bot is displayed in contact details and elsewhere.

The Username is a short name, to be used in mentions and t.me links. Usernames are 5-32 characters long and are case insensitive, but may only include Latin characters, numbers, and underscores. Your bot's username must end in '**bot**', e.g. 'tetris_bot' or 'TetrisBot'.

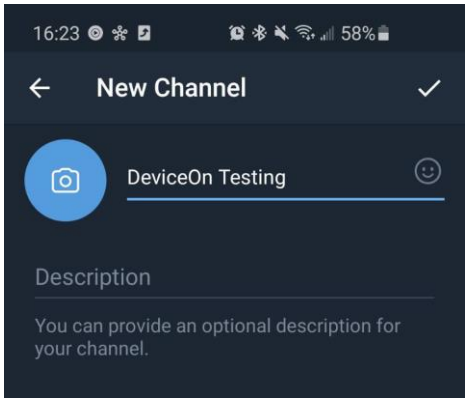
The token is a string along the lines of 110201543:AAHdqTcvCH1vGWJxfSeofSAs0K5PALDsaw that is

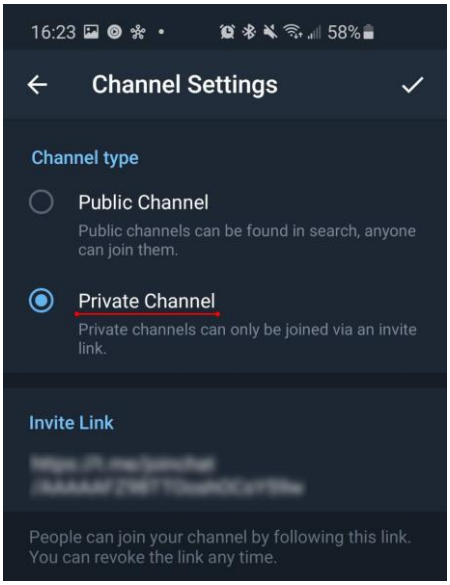
required to authorize the bot and send requests to the Bot API. Keep your token secure and store it safely, it can be used by anyone to control your bot.



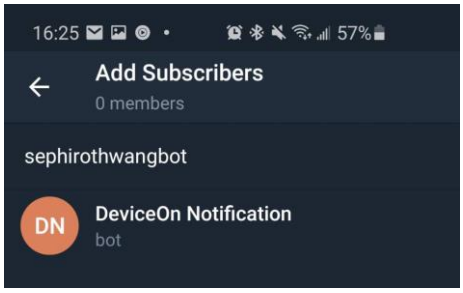


Step 2: Create your private channel on Telegram

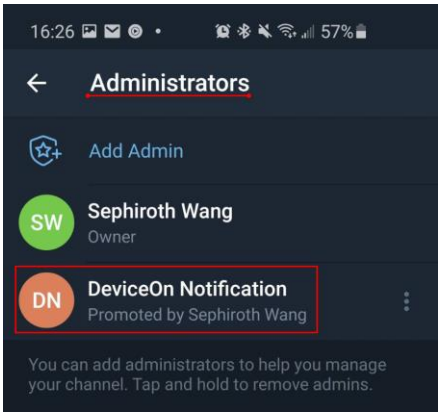




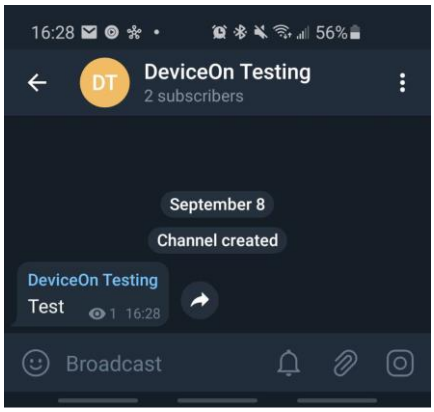
Step 3: Invite your bot into the channel.



Step 4: Set your bot as "**Administrators**"



Step 5: Enter any txt message in the channel.

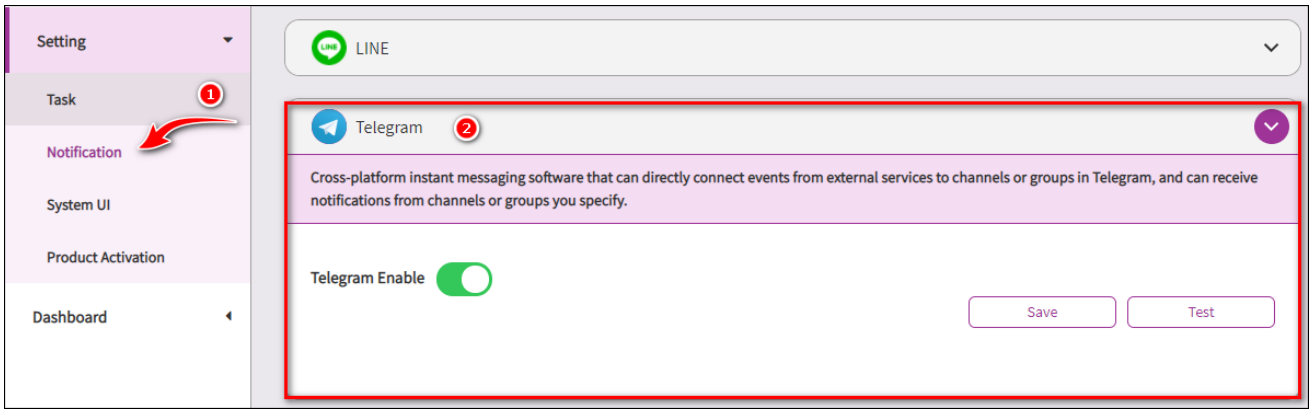


Step 6: Retrieve the **chat id** via below URL with your authorization token (Step 2).

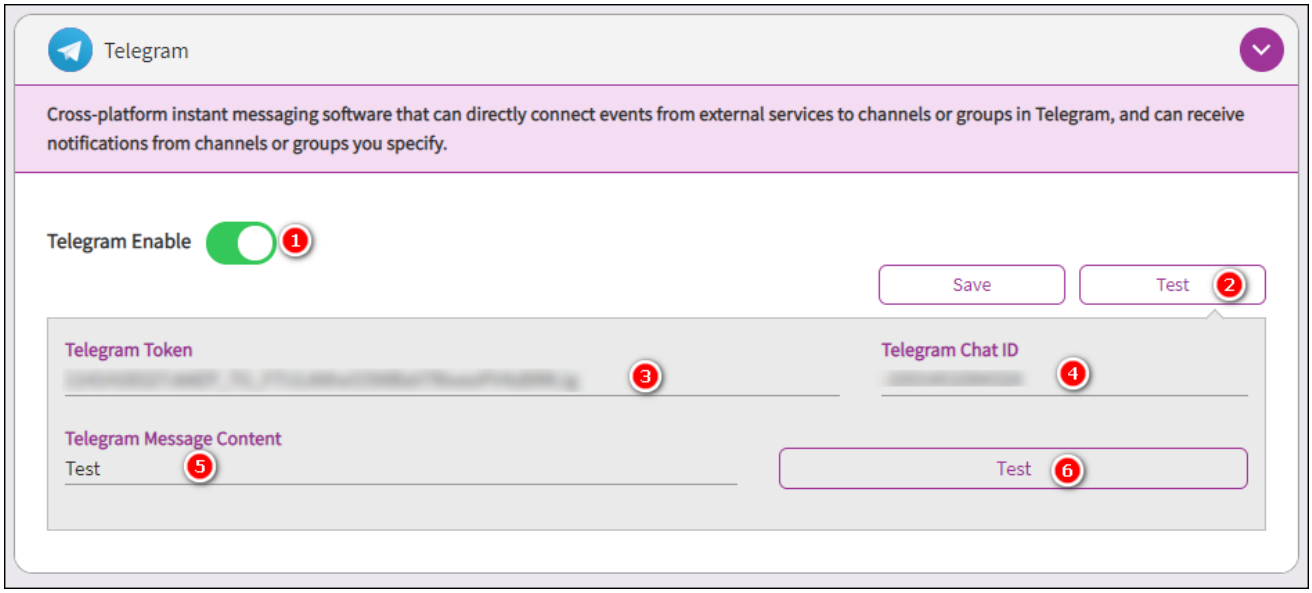
⇨ https://api.telegram.org/bot**TOKEN**/getUpdates

The response that include your chat id as below example.

Step 7: Now switch your browser to DeviceOn portal. Click **"Setting"** menu on the left-hand side, then **"Notification"**, and **"Telegram"** to open settings regarding Telegram event notification service.



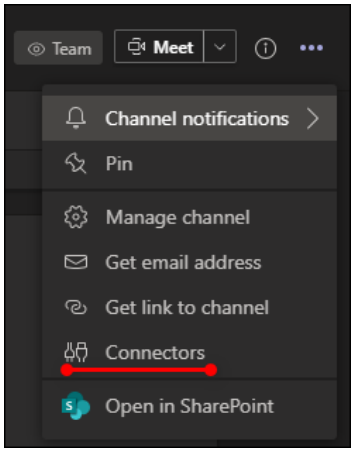
Step 8: Toggle **"On/Off"** switch to enable this feature. Click **"Test"** to show the test dialog up. Paste the copied Token and chat id, copied in step 2 and step 6. Give a title to the second field **"Telegram Message Content"**. Write some message content to the last field **"Test"**. And click **"Test"** to see if it works or not.



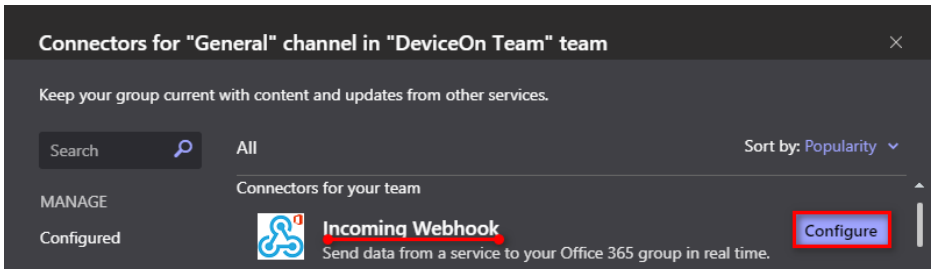
Step 10: Click **"Save"** button that shows in step 8 to keep your settings and enable Telegram event notification service.

4.3.6 Steps to Set Event Notification Service – Microsoft Teams

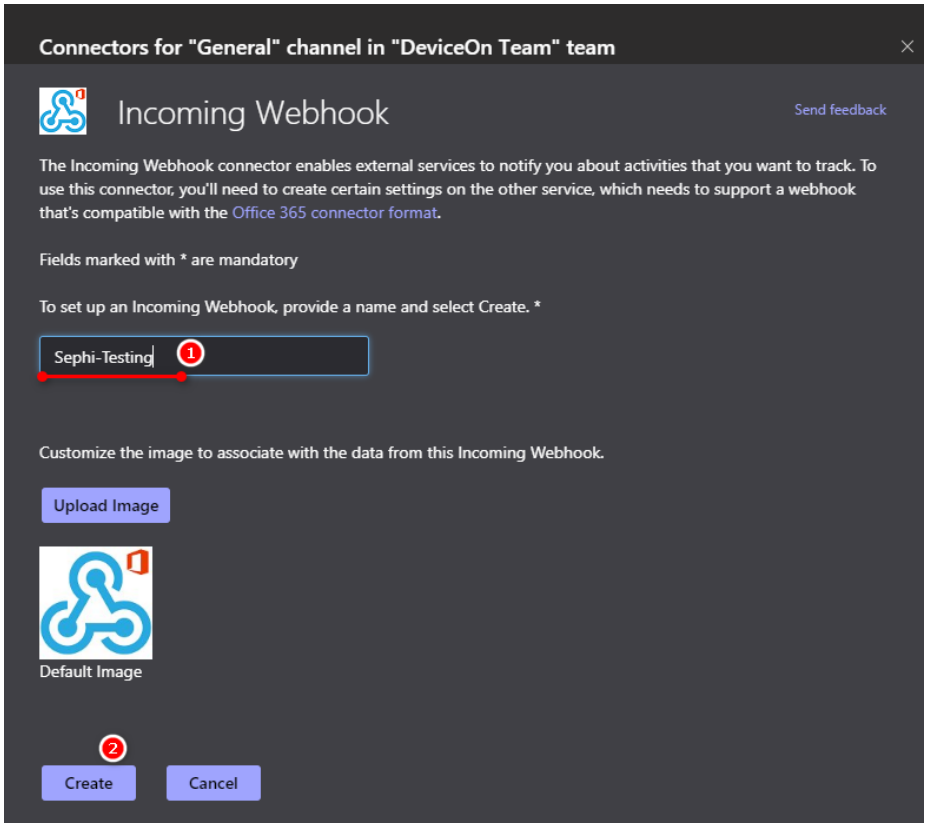
Step 1: In the function menu of the channel where you want to send the message, select...(Other), and select the connector in the menu.



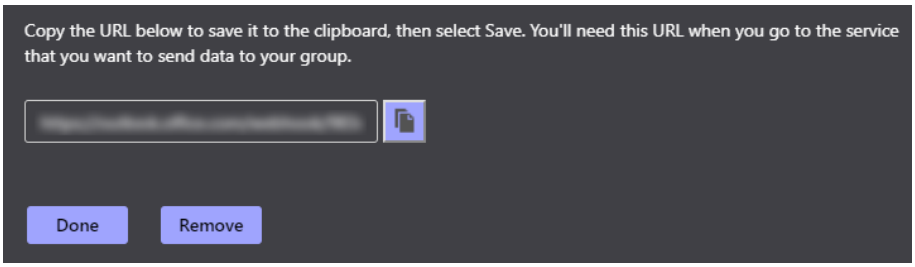
Step 2: Select “Incoming Webhook”



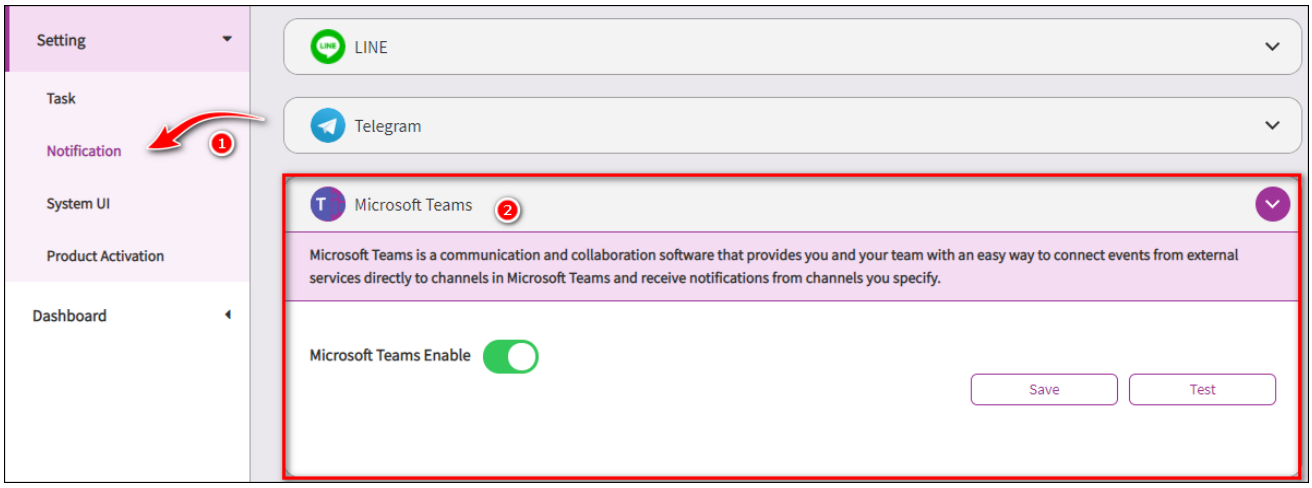
Step 3: Give this connector a name, then press the create button



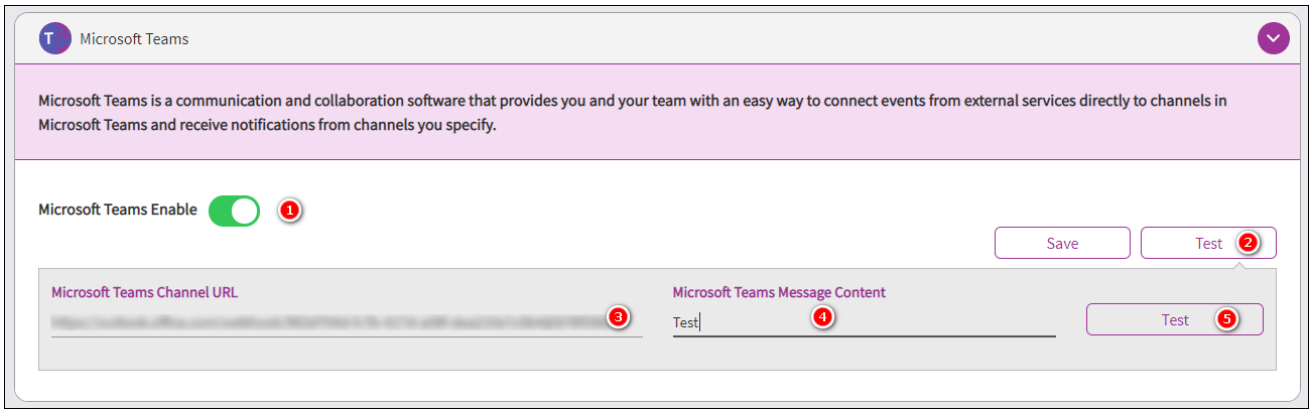
Step 4: At this time, a set of URLs will appear, which are used to transfer message. After copying, press the "**Done**" button.



Step 5: Now switch your browser to DeviceOn portal. Click **"Setting"** menu on the left-hand side, then **"Notification"**, and **"Microsoft Teams"** to open settings regarding Teams event notification service.



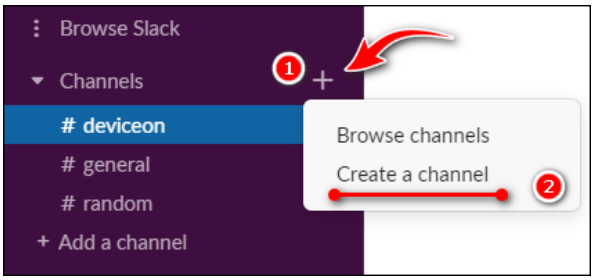
Step 6: Toggle **"On/Off"** switch to enable this feature. Click **"Test"** to show the test dialog up. Paste the URL, copied in step 4. Give a title to the second field **"Microsoft Teams Message Content"**. Write some message content to the last field **"Test"**. And click **"Test"** to see if it works or not.



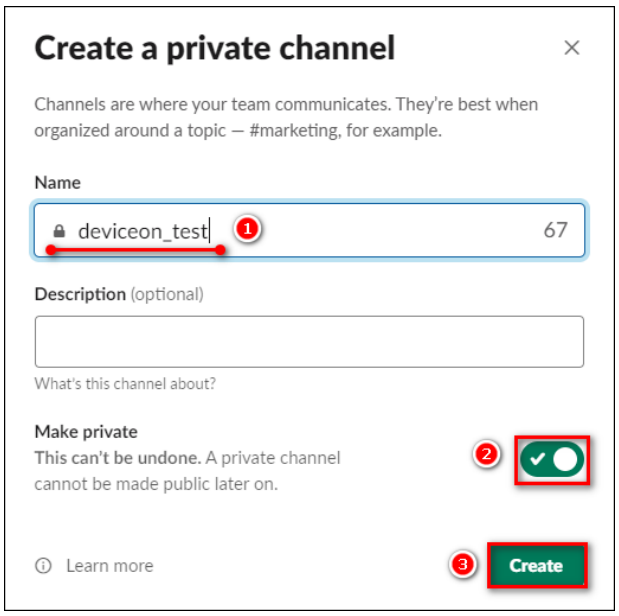
Step 7: Click **"Save"** button that shows in step 6 to keep your settings and enable Microsoft Teams event notification service.

4.3.7 Steps to Set Event Notification Service – Slack

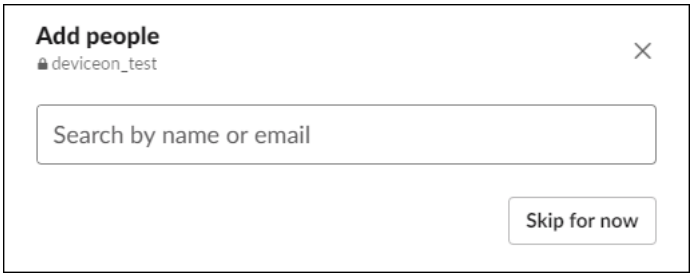
Step 1: Create your channel on your Slack.



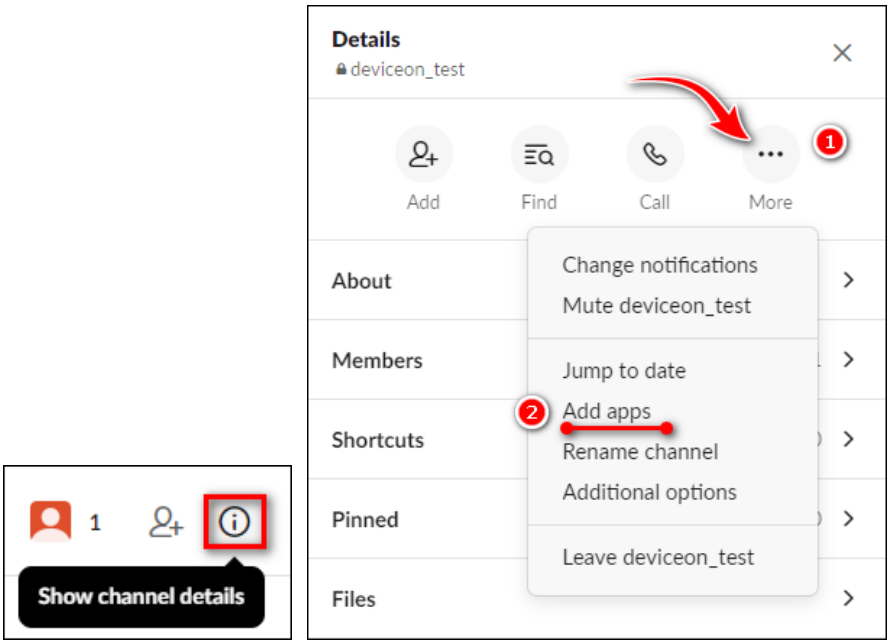
Step 2: Give this channel name and set as private.



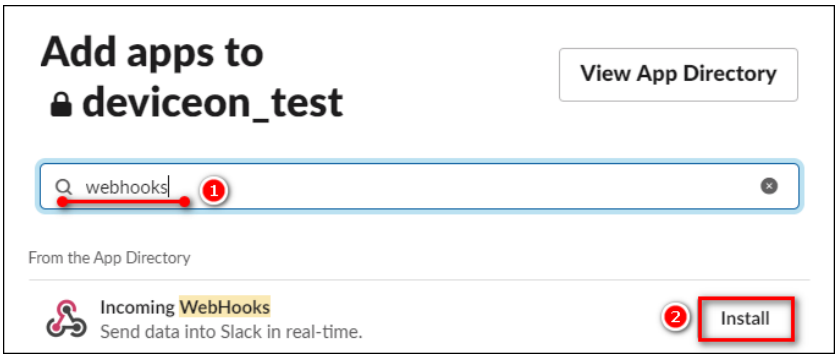
Step 3: Skip or add your member into channel.

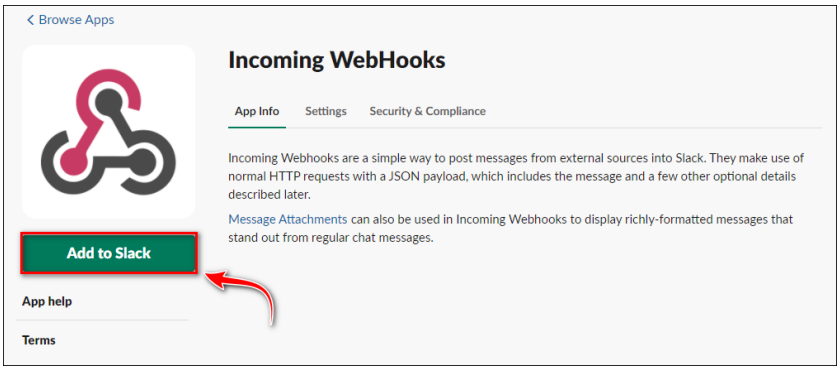


Step 4: After logging in to slack, there will be a row of menus on the right, click "**Apps**" to expand the sub-menu, and then click "**Add Apps**"

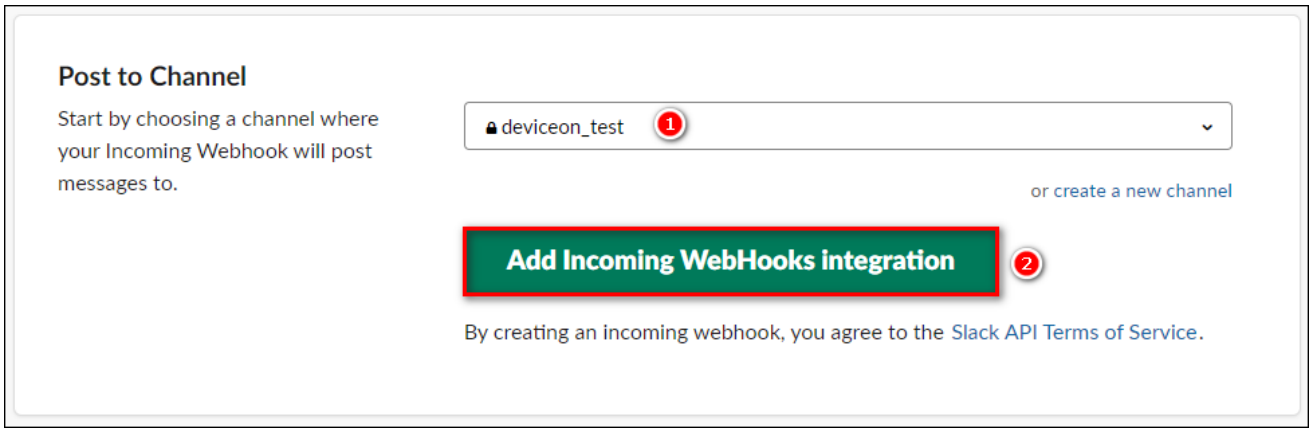


Step 5: A search box will appear, type "**webhooks**" and you will see the first result is "**Incoming WebHooks**", then click to install and "**Add to Slack**".

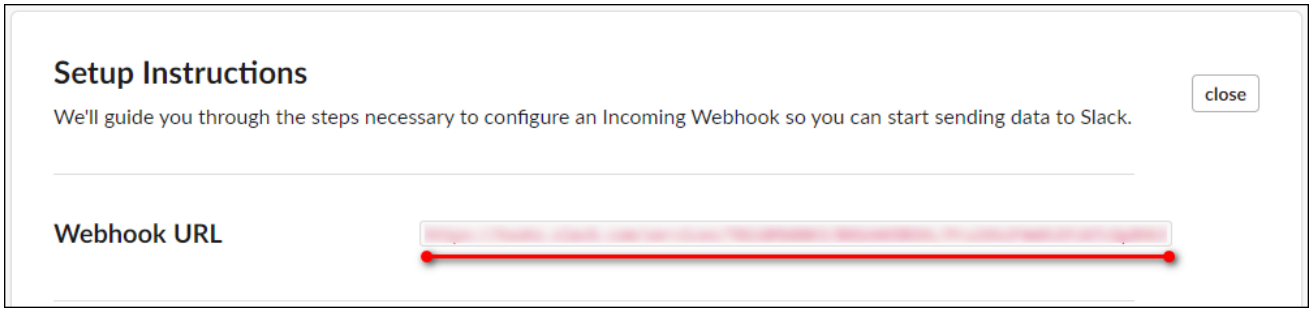




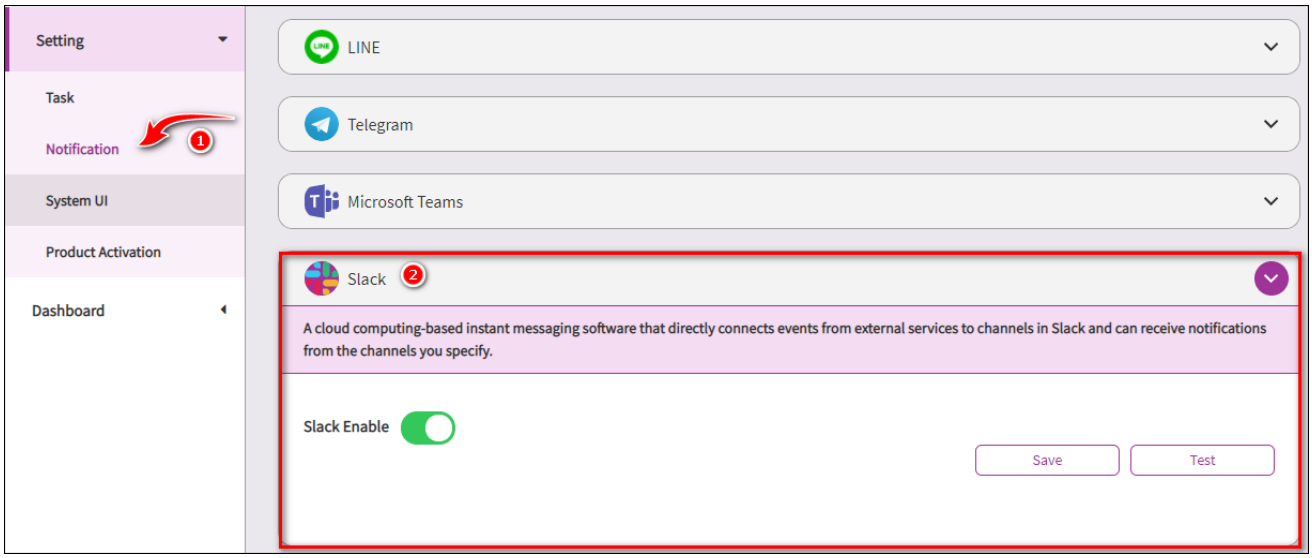
Step 6: Click "**Add to Slack**", and a menu will appear asking which channel to install on. After selecting it, click "Incoming WebHooks integration".



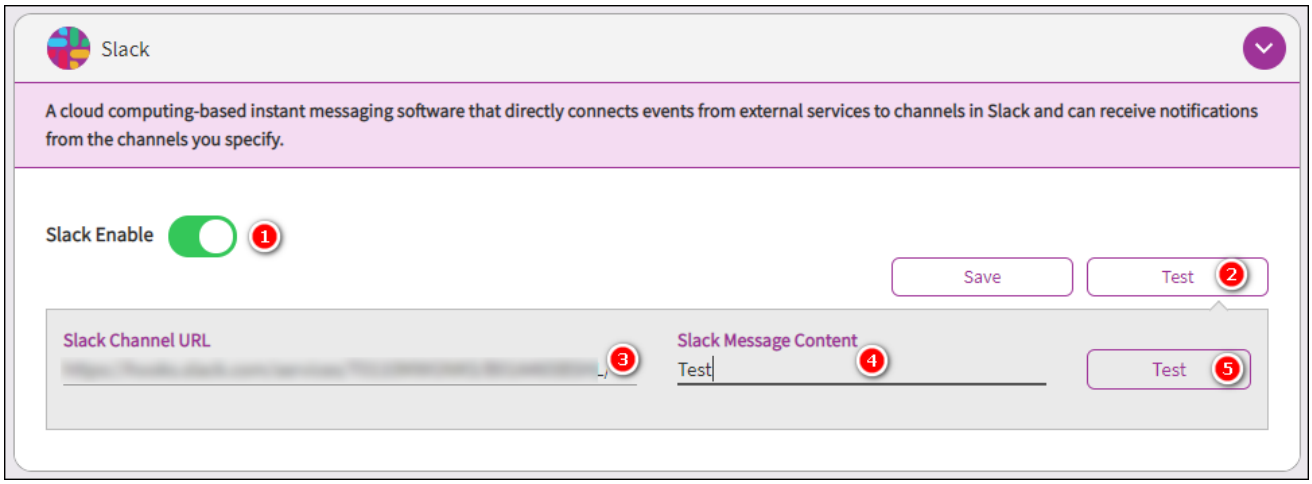
Step 7: After installation, you will enter the setting page of incoming webhooks. The first line of the page "**Webhook URL**" is the most important. We can send out automatic notification messages as long as we post to this url.



Step 8: Now switch your browser to DeviceOn portal. Click **"Setting"** menu on the left-hand side, then **"Notification"**, and **"Slack"** to open settings regarding Slack event notification service.

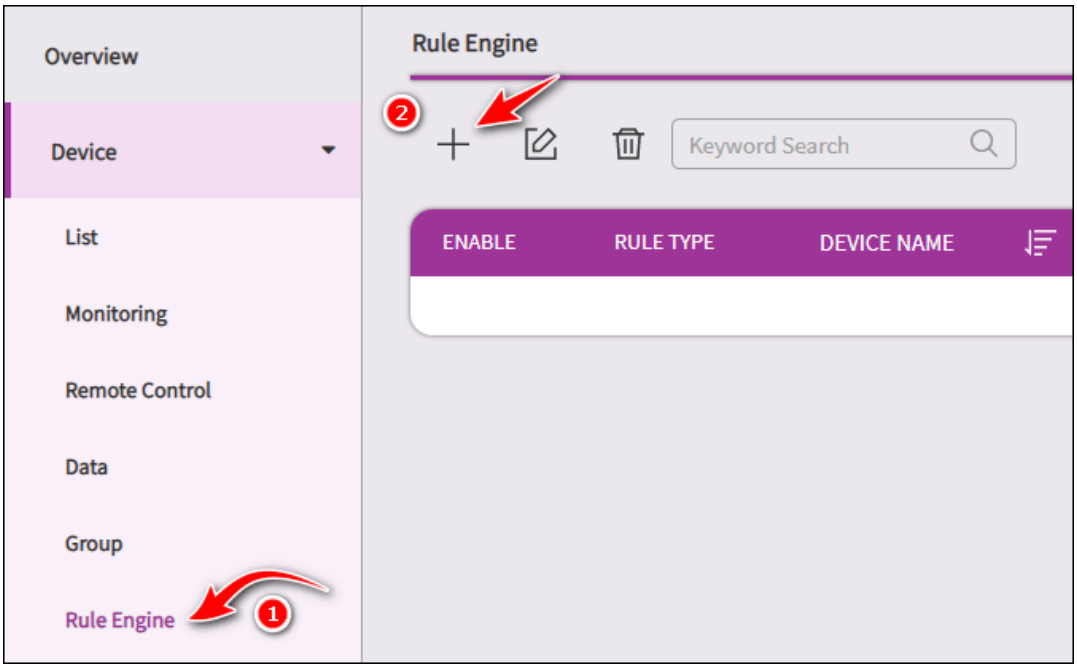


Step 9: Toggle **"On/Off"** switch to enable this feature. Click **"Test"** to show the test dialog up. Paste the URL, copied in step 4. Give a title to the second field **"Slack Message Content"**. Write some message content to the last field **"Test"**. And click **"Test"** to see if it works or not.



4.3.8 Steps to Set Thresholds to a Device

Step 1: Click **"Setting"** menu on the left-hand side of DeviceOn portal and, then, **"Rule Engine",** click the plus (+) sign to enter **"Rule Engine"** page.

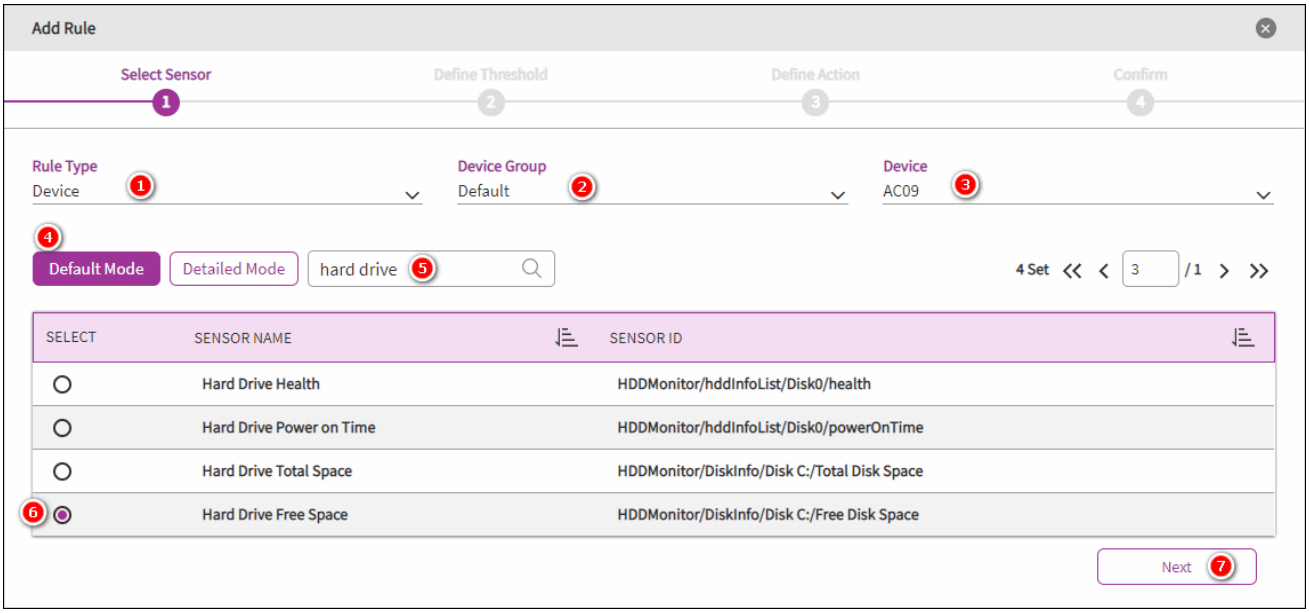


Step 3: Choose each setting with a proper value within step 1 – Select Sensor.

- **SELECT RULE TYPE:** Shows the new rule engine applies to a single device or a device group. Please pick **"Device"** here.
- **SELECT DEVICE GROUP:** Also, leave it the default, **"Default"**.
- **SELECT DEVICE:** Which device the new rule engine will apply? We choose **"AA-Win"** in this lab environment.

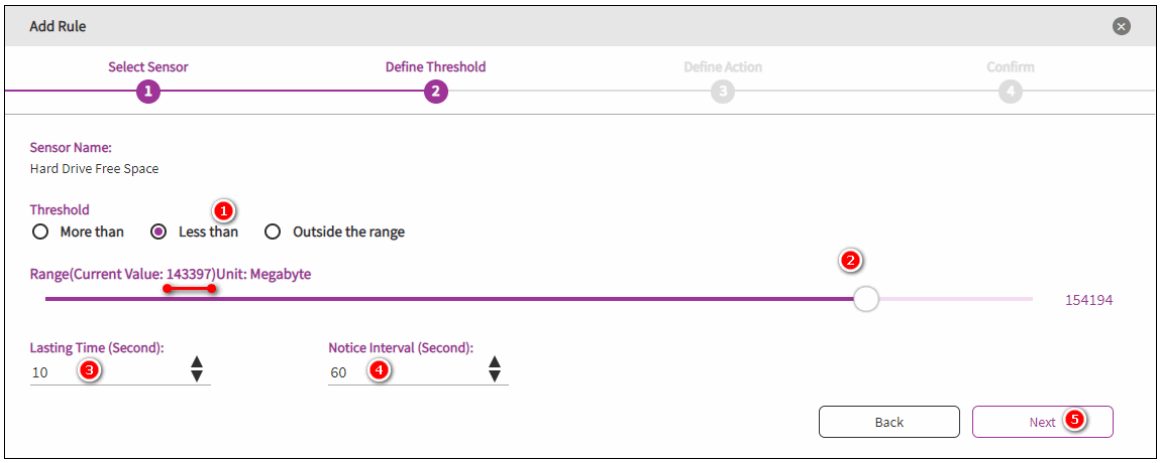
- **KEYWORD SEARCH:** Please enter **"hard drive"** so that only hard drive relevant items available.

Here, to ease this lab, we pick **"Hard Drive Free Space"** as a threshold of the rule engine. In addition, like the picture shows, it illustrates the disk C is the target hard drive in this lab. Click **"Next"** to go to next step.



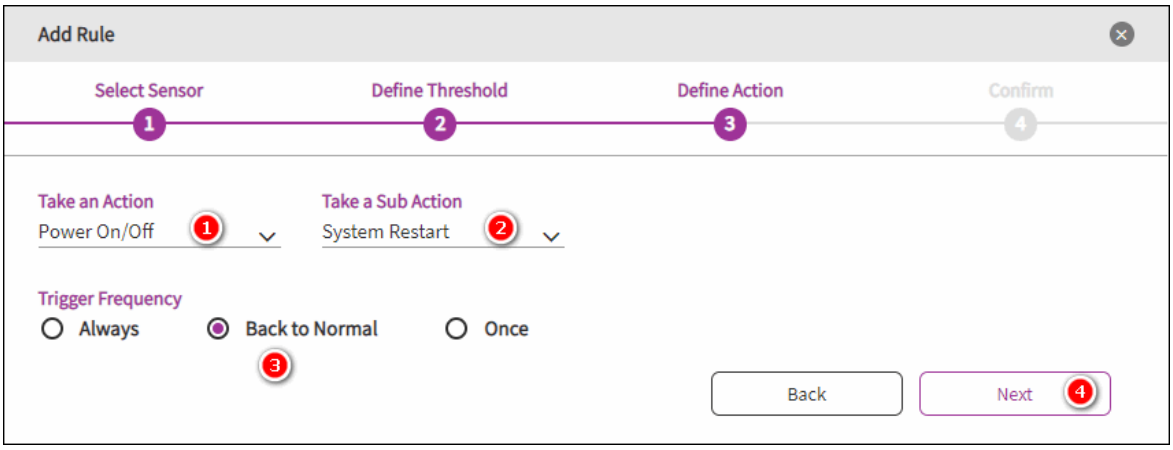
Step 4: Now we need to define a threshold for this rule engine in this step. Based on **"Current Value"** shows on top right, check the **"Less than"** radio button and slide to a maximum value that just on less than **"Current Value"**.

Leave **"Lasting Time"** as well as **"Notice Interval"** the defaults. **"Lasting Time"** indicates that the target device runs into the abnormal condition only when it reaches the set threshold and last the set time. While **"Notice Interval"** tells the interval of users receive an event, until the condition back to normal. Then click **"Next"** to go to next page.

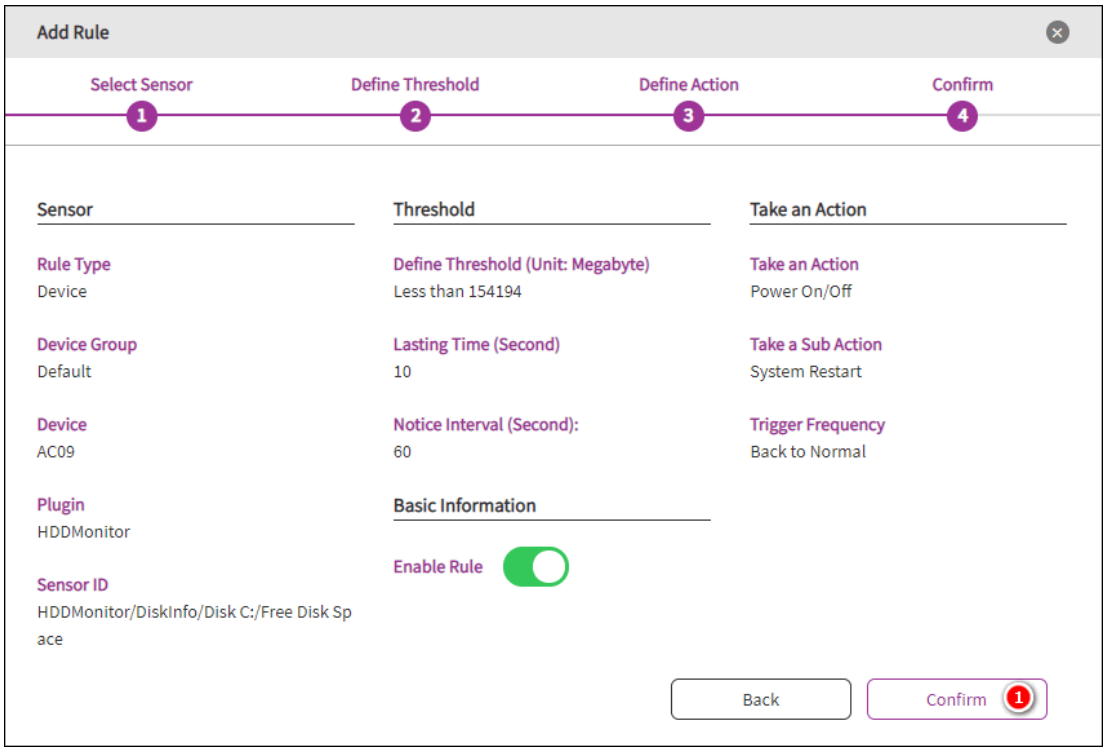


Step 5: We are now in **"Define Action"** step. Pick **"Power On/Off"** from **"TAKE A ACTION"**, **"System Restart"** from **"TAKE A SUB ACTION"**, and **"Back to Normal"** for **"Trigger Frequency"**. These

combination means that the target device will reboot once it backs to normal, after it enters the threshold we set. Also, click **"Next"** to go to next page.

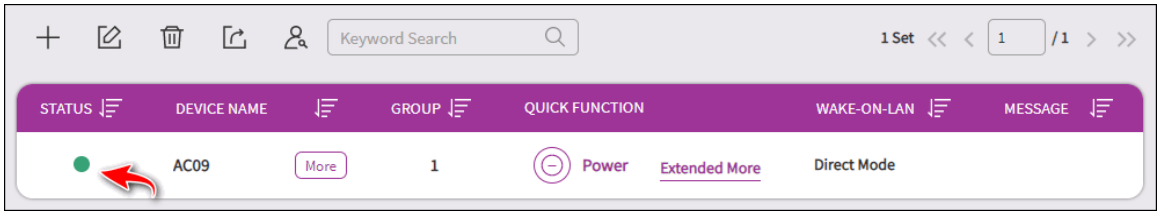


Step 7: Review all information within this page. Leave **"Enable"** the default and click **"Confirm"** button to set this rule, and apply it to the target device as well.

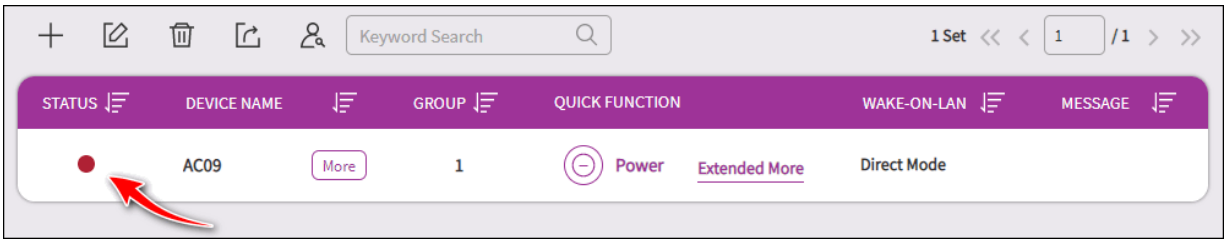


Step 8: The new item should be populated as the image shows.

Step 9: Click **"Device"** menu item on left hand side of DeviceOn portal. You can see a green circle represents the target device accordingly.



Step 10: We can do something so that the target device meets the threshold we set previous. Here we download the newest Ubuntu ISO image to the target device. The green circle shows in step 9 changes, a while later, to an orange one, of which indicates it runs into an abnormal condition.



Step 10: Interrupt the download action at any time, or wait until it finishes. Purge the downloaded file so that the target device has free space more than the threshold we set previous. After a while, the target device should reboot due to the rule engine we set. Note here that it may necessary to purge the recycle bin to achieve our goal.

4.4 How to Visualize Device Data via Grafana Dashboard

Grafana is an open-source software for monitoring and analysis. One of its major characteristics is it supports many different data sources, from popular CloudWatch, Elasticsearch, Graphite, and influxDB, to OpenStack Gnocchi or Google Calendar. Its range is very extensive. However, for others data source require to implement SimpleJson to access your data. The DeviceOn native support SimpleJson APIs and data source plugin on Grafana. This lab guides you how to visualize device data via Grafana dashboard.

4.4.1 Prerequisite

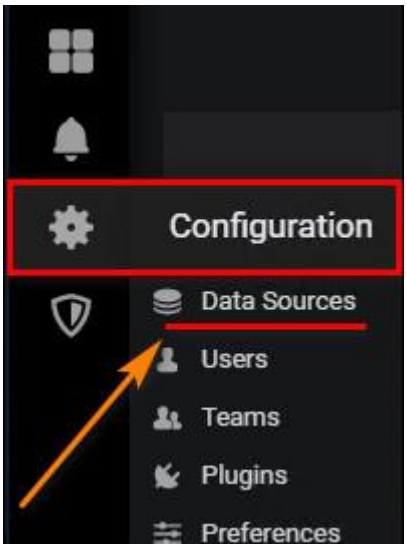
- A running DeviceOn server.
- A running Grafana service with DeviceOn data source plugin.
- A device which installed WISE-Agent, that connects to DeviceOn server.

4.4.2 Step-by-Step

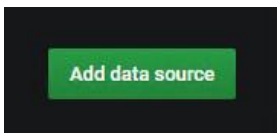
Step 1: Launch Grafana Web Service Shortcut on Desktop, or access the Grafana service endpoint.

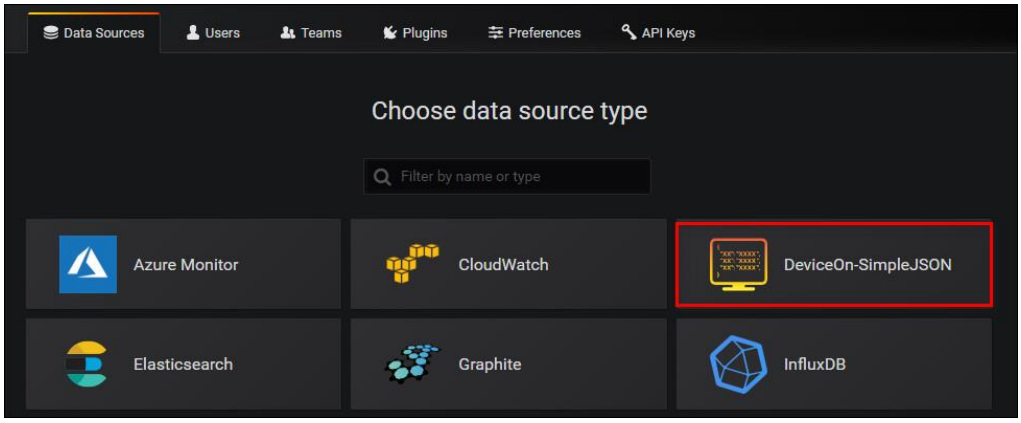
Step 2: Login to Grafana portal with your account, password (Default: admin/admin)

Step 3: Create a data source to access DeviceOn SimpleJson API.



Click on "**Add data source**" and select "**DeviceOn-SimpleJson**", (for previous version might be RMM-SimpleJson)





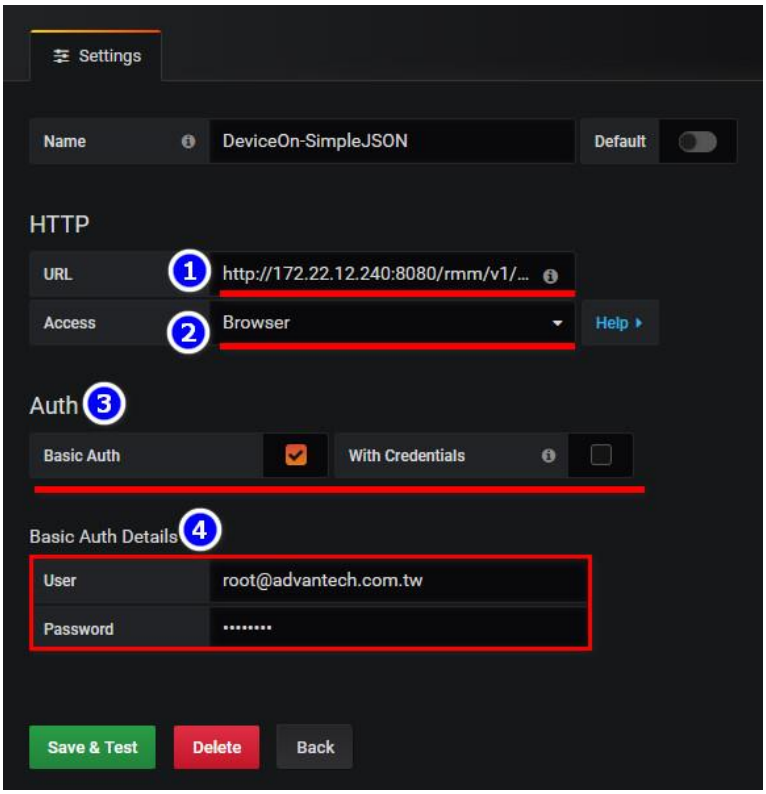
Step 4: Given below parameters for data source plugin to retrieve device data from DeviceOn APIs.

URL: **http://<DEVICEON_SERVER>:8080/rmm/v1/grafana/simplejson**

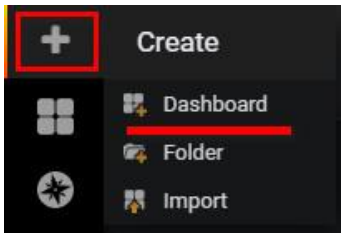
Access: Browser

Auth: Basic Auth (Support on prefecture version)

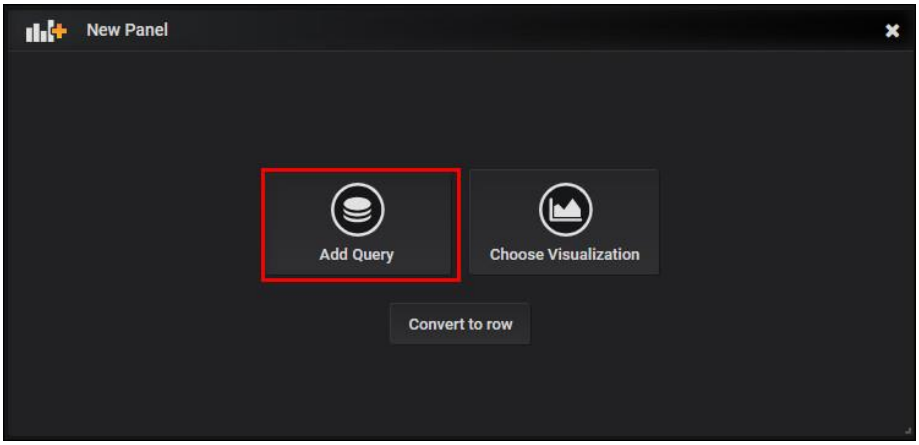
Basic Auth: DeviceOn Account & Password



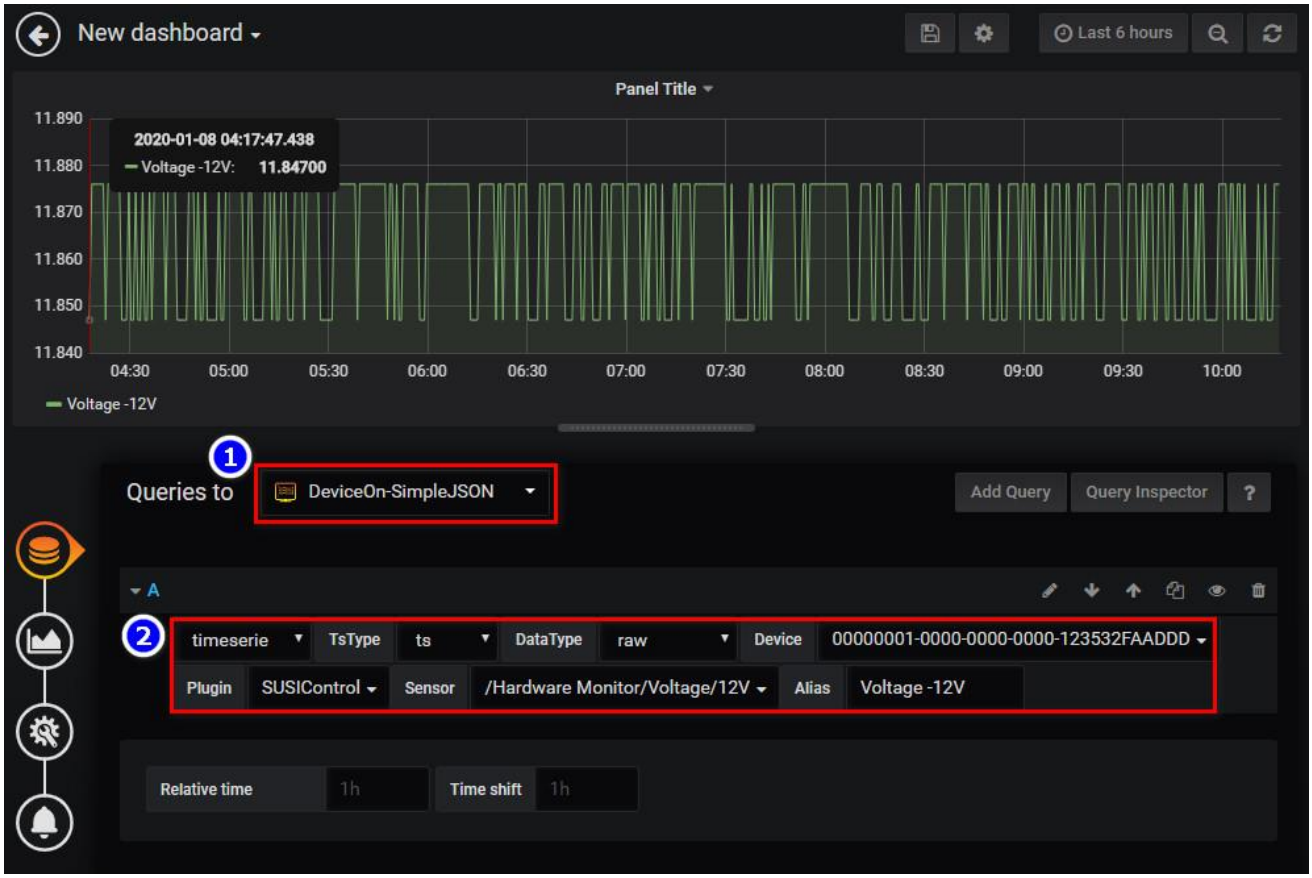
Step 5: Create a dashboard to visualize your device data.



Select “Add Query” for your device.



Select **DeviceOn-SimpleJson** from “**Queries to**”, and pick-up your device with **AgentID**, **Plugin**, **Sensor** and **Alias Name** (Option).



4.5 How to Enable/Disable Windows Lockdown Features

For devices protection, Windows built many nice features in natively. For instance, function key protection disables Ctrl, Alt, and WinKey. UWF protection guarantees your disk C (System Partition) rollbacks to the original state after you reboot the Windows operating system. This lab guides you how to enable Windows lockdown features, and how to active/inactive them via DeviceOn portal. After this lab, you should:

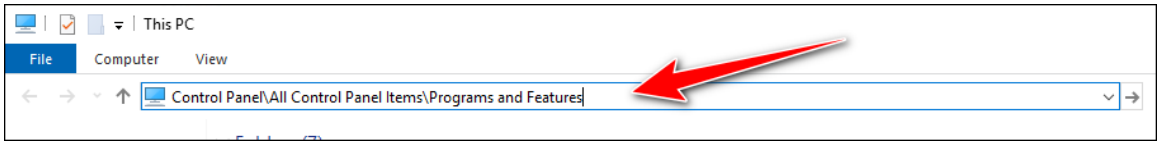
- Learn how to enable **"Keyboard Filter"** and **"Unified Write Filter"** (a.k.a. UWF) in Windows lockdown features.
- Know what lockdown features can be controlled via DeviceOn portal.

4.5.1 Prerequisite

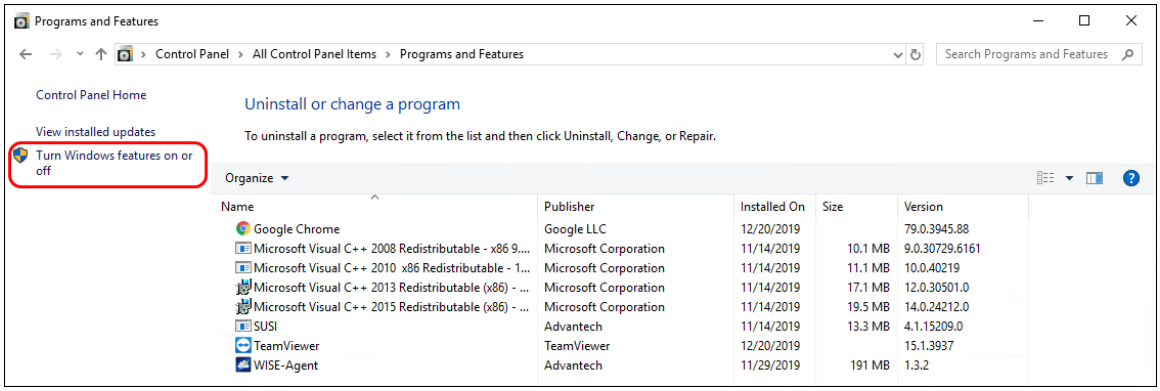
- A running DeviceOn server.
- A device which running on Windows 10 operating system (LTSB, LTSC) and installed WISE-Agent, that connects to DeviceOn server. Besides, this agent must install Advantech SUSI driver, or lockdown feature should not work properly.

4.5.2 Step-by-Step

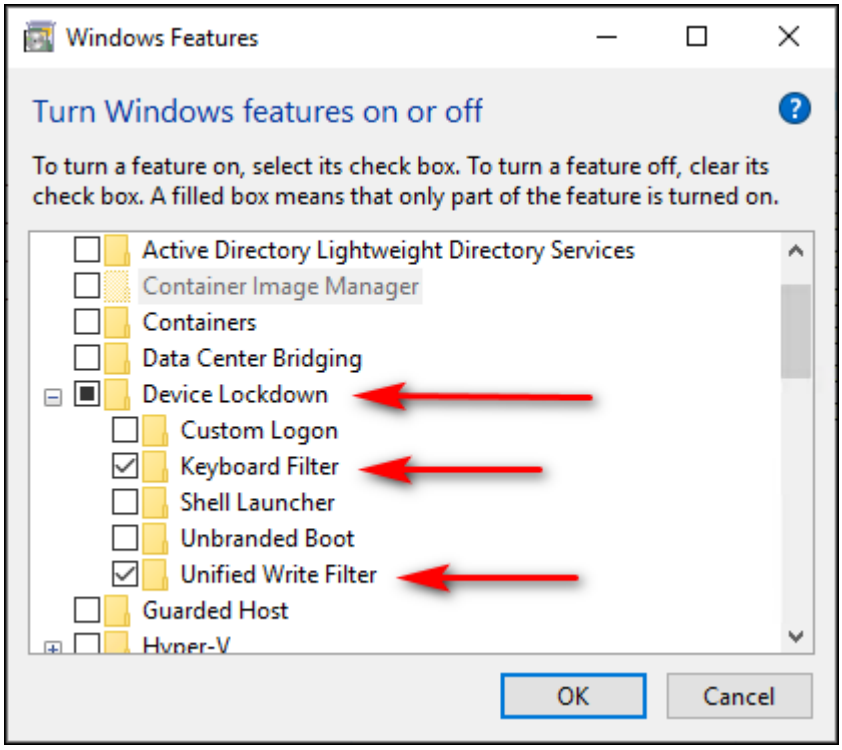
Step 1: Go to the target agent device and open the file explorer window. In address bar, key **"Control Panel\All Control Panel Items\Programs and Features"** in and followed by pressing **"ENTER"**. It opens the **"Programs and Features"** window.



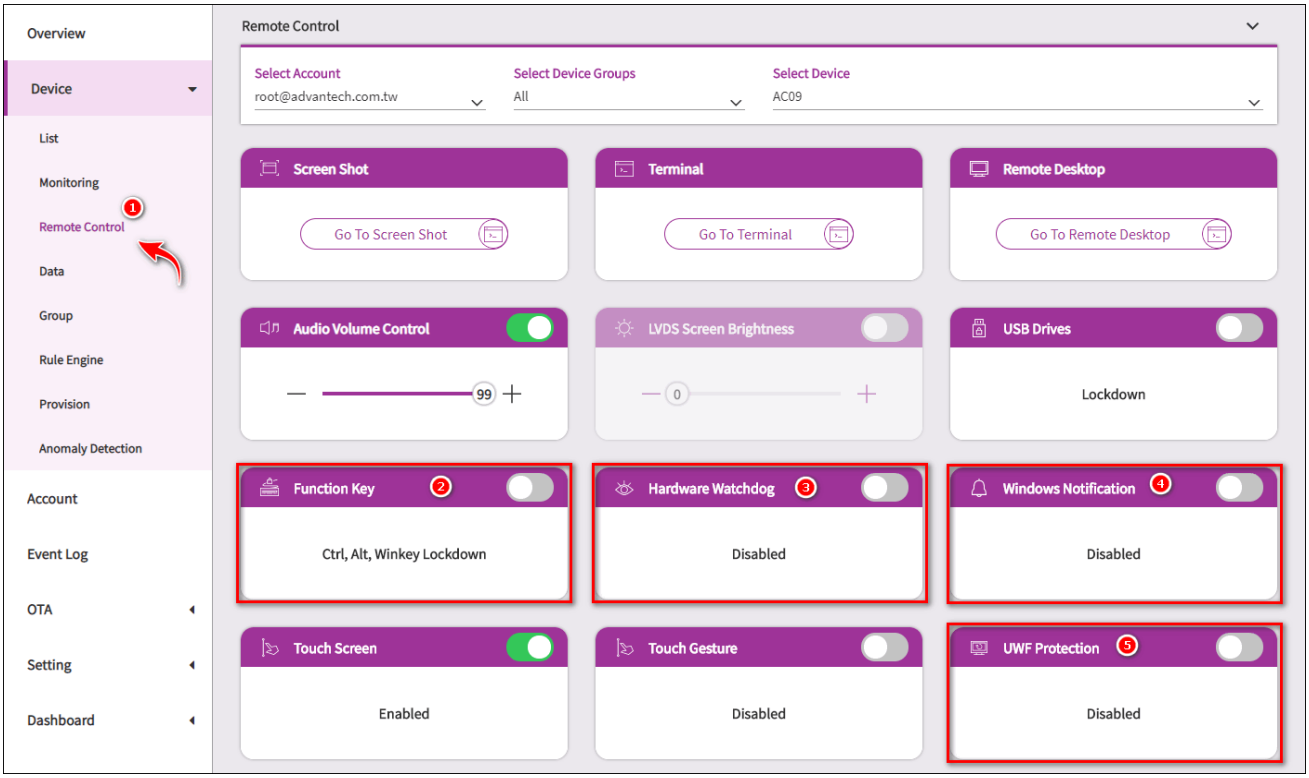
Step 2: Click **"Turn Windows features on or off"** on left hand side to open **"Windows Features"** window.



Step 3: Scroll down the window, find and open the **"Device Lockdown"** item. Make sure both **"Keyboard Filter"** and **"Unified Write Filter"** are checked. Then click **"OK"**.



Step 4: Now back to DeviceOn portal. Click **"Device"** menu item, then **"Remote Control"** tab. And choose proper account, group, and device from **"SELECT ACCOUNT"**, **"SELECT DEVICE GROUPS"**, and **"SELECT DEVICE"** fields accordingly. You can see **"Function Key"**, **"UWF Protection"** control buttons there. Also, other than these two mentioned, **"WatchDog Protection"**, **"Windows Notification"** and more relevant features are available as you can see.



Step 5: Click **"Function Key"** control button. You would find, after a while, the description of **"Function Key"** changes from **"Available"** to **"Ctrl, Alt, WinKey Lockdown"**. If you try to press such keys on the target device, they should not work as expected. Okay, you learned how to enable, disable **"Function Key"** lockdown. Let's go ahead and learn something regarding UWF.

Step 6: Click **"UWF Protection"** control button. A dialog pops up and the message shows that this action will reboot the device. Click **"CONFIRM"**, its description changes from **"Disabled"** to **"Enabled"**. Just wait for the reboot completed.

Step 7: Now, write some data into disk C. You can, for example, download files into disk C, copy files into disk C. Or even generate by programmatically. Just do whatever you can do to mimic that you are working on disk C.

Step 8: Once you finish your tasks, reboot the target device. You would find that all those data you made at previous step disappear. The disk C rollbacks to the original state and just like you did nothing at all.

4.6 How to Manage DeviceOn on AKS

Since the DeviceOn service container are running the Azure Kubernetes, that's much easier to upgrade to new version, if released. There

4.6.1 Prerequisite

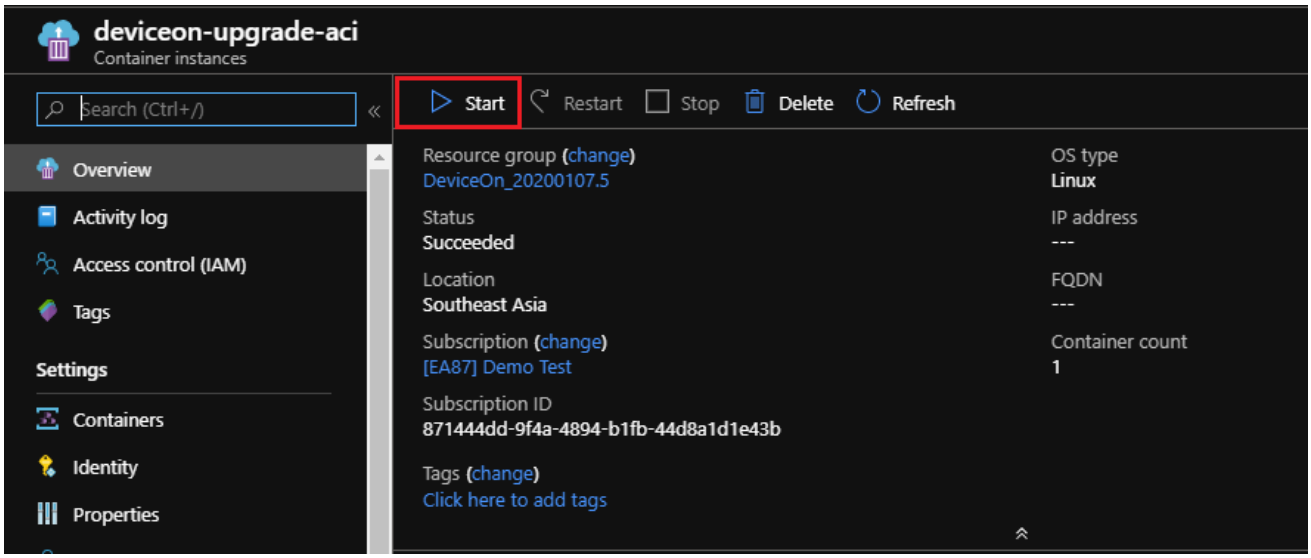
- A running DeviceOn server on Azure Kubernetes
- Azure Account

4.6.2 Steps to Upgrade DeviceOn

Step 1: Login to Azure Portal and find your AKS solution resource group.

Step 2: Click **deviceon-upgrade-aci** service.

Step 3: Click **Start** button to upgrade latest version DeviceOn server.

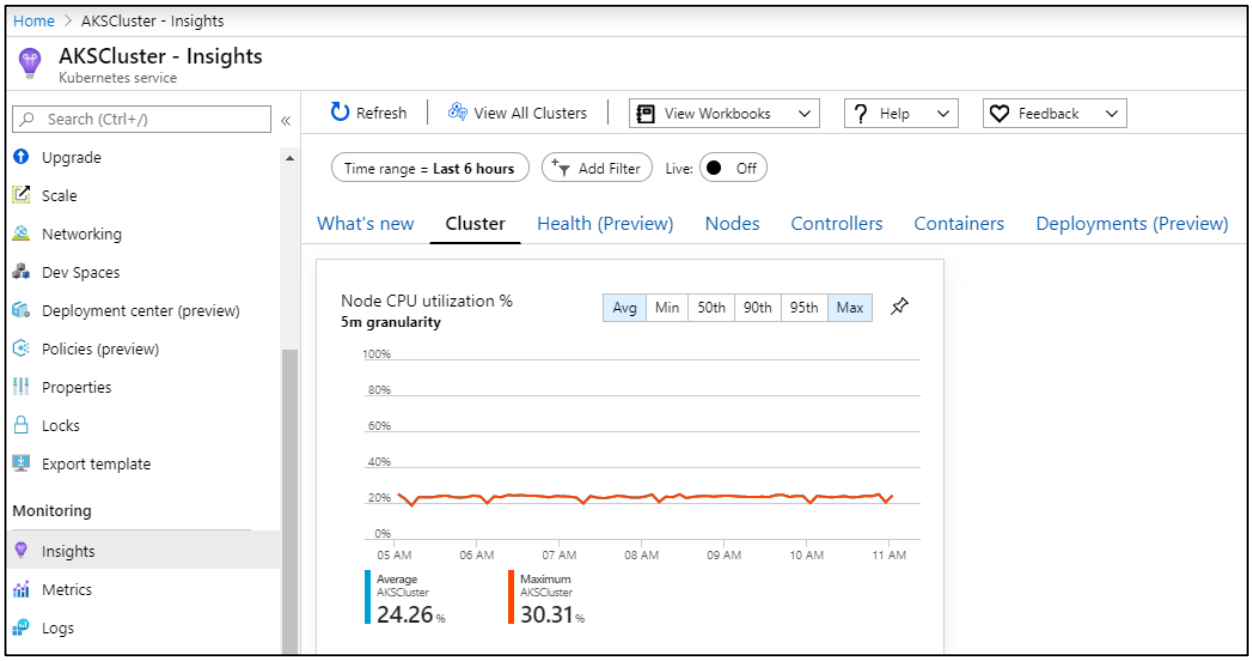


Step 4: After upgrade container instances finish, please go to DeviceOn portal check server version.

4.6.3 Step to Monitor Container Healthy and Status

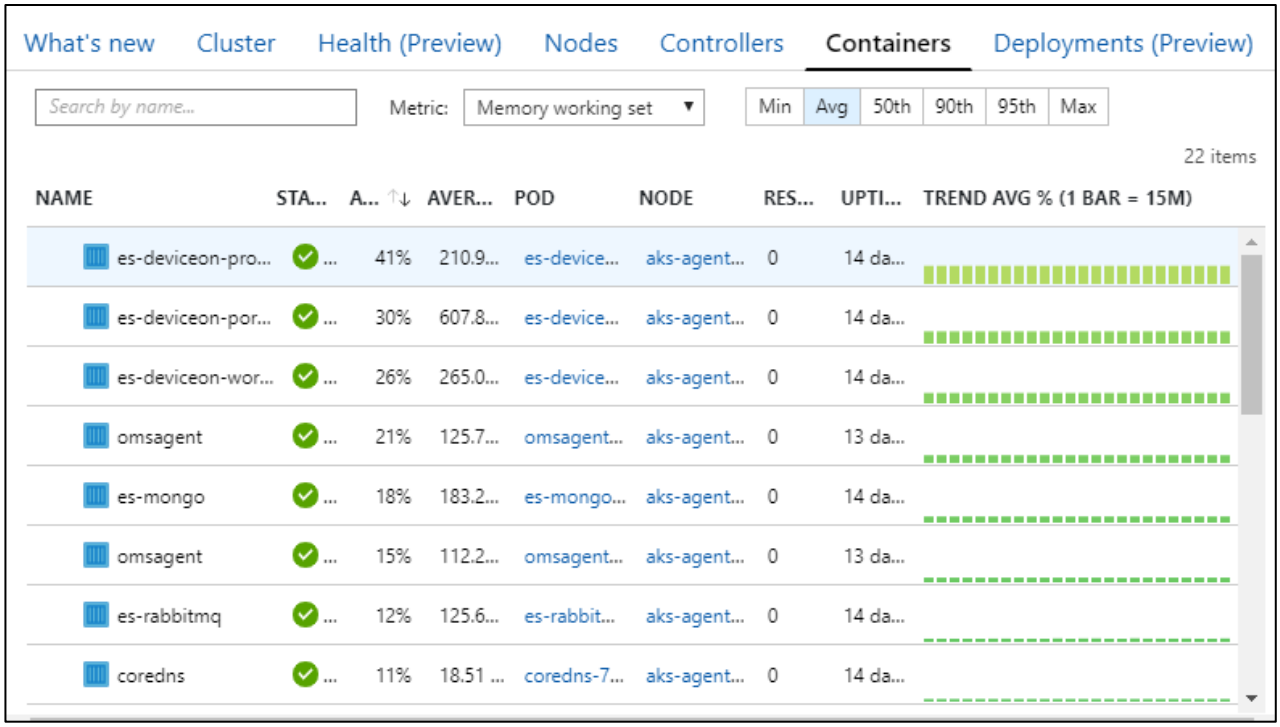
- **Monitor Container Status**

Step 1: In AKS service, select “Insights” on the left tab.

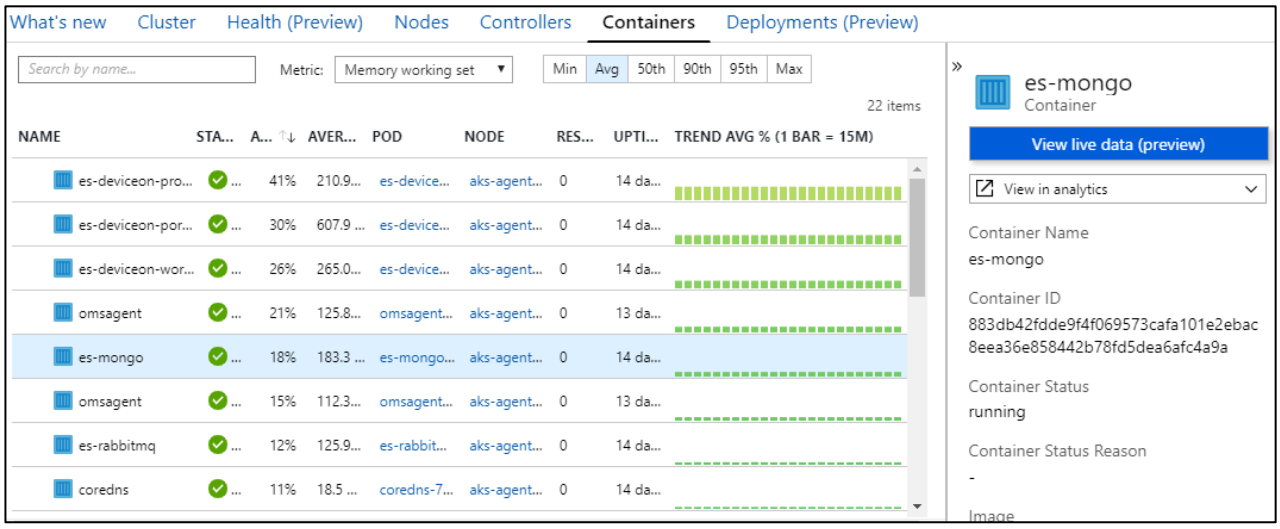


Step 2: Click on “**Containers**” on the top tab. Check if status of each container is running.

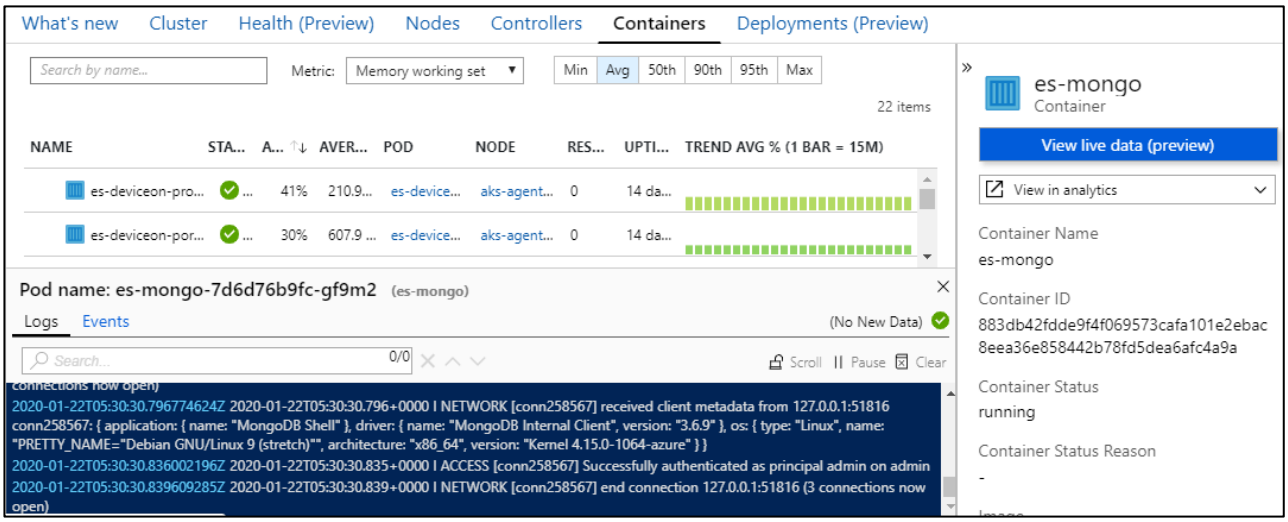
** Init container will show completed. **



Step 3: View the container log by clicking on container name. Take es-mongo for example.



Step 4: Click on “View live data” to view live log of container.



- **Monitor Usage of Persistent Volume (PV)**

Step 1: Open PowerShell and login

Follow the instruction to login your account

```
# az login
```

Select your subscription, if you have multiple subscription, you have to set which subscription AKS service is located.

```
# az account set --subscription "SUBSCRIPTION_NAME”
```

Try to install AKS command line.

```
# az aks install-cli
```

To get AKS credential to access.

```
# az aks get-credentials --resource-group RESOURCE_GROUP --name K8S_CLUSTER
```

Step 2: Use kubectl command to get k8s information

```
# kubectl get pod --all-namespaces (check all pods)
```

```
PS Azure:\> kubectl get pod --all-namespaces
NAMESPACE     NAME                                      READY   STATUS    RESTARTS   AGE
default       es-deviceon-portal-6d6574b65-tp744        1/1     Running   0          13d
default       es-deviceon-provisioning-fc74b9cd7-cnwvl  1/1     Running   0          13d
default       es-deviceon-worker-5f88997ccb-292pl       1/1     Running   0          13d
default       es-mongo-7d6d76b9fc-gf9m2                 1/1     Running   0          13d
default       es-rabbitmq-7cb946db6c-5sqk5              1/1     Running   0          13d
default       rmm-iothub-bridge-6cf98f5f4-cvqf4         1/1     Running   13         13d
kube-system   coredns-7fc597cc45-ht4jc                  1/1     Running   0          13d
kube-system   coredns-7fc597cc45-r9814                  1/1     Running   0          13d
kube-system   coredns-autoscaler-7ccc76bfbd-bjwpd       1/1     Running   0          13d
kube-system   kube-proxy-zsfmx                          1/1     Running   0          7d9h
kube-system   kubernetes-dashboard-6fbc7f598b-klcbg     1/1     Running   5          13d
kube-system   metrics-server-58b6fcfd54-8jpn8           1/1     Running   0          13d
kube-system   omsagent-nwrnl                            1/1     Running   0          12d
kube-system   tiller-deploy-59b99695d8-2x4pf            1/1     Running   0          13d
```

Try to shows all PVs in the cluster, it also includes which PVCs (Persistent Volume Claim) are requesting for the resources.

```
# kubectl get pv --all-namespaces (check PV)
```

```
PS Azure:\> kubectl get pv --all-namespaces
NAME                                       CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM
     STORAGECLASS      REASON   AGE
pvc-e967a374-3115-11ea-a6bc-6201b0e63488   500Gi      RWO            Delete           Bound    default/es-mongo
     managed-premium            13d
pvc-e9eb4084-3115-11ea-a6bc-6201b0e63488   8Gi        RWO            Delete           Bound    default/es-postg
es   managed-premium            13d
pvc-ea66f281-3115-11ea-a6bc-6201b0e63488   32Gi       RWO            Delete           Bound    default/es-rabbi
mq   managed-premium            13d
```

Take es-mongo as example, the capacity of es-mongo PV is 500Gi, and default/es-mongo is requesting it as PVC. If you'd like to know the PV usage, you need to access es-mongo container.

```
# kubectl exec -it es-mongo-7d6d76b9fc-gf9m2 -- /bin/bash
```

Please replace the **pod** name to yours. To display disk available space on the file system, you could enter:

```
# df -h
```

```
root@es-mongo-7d6d76b9fc-gf9m2:/# df -h
Filesystem      Size  Used Avail Use% Mounted on
overlay          97G   19G   79G  20% /
tmpfs            64M     0   64M   0% /dev
tmpfs           7.9G     0  7.9G   0% /sys/fs/cgroup
/dev/sdc        493G  448M  492G   1% /data/db
/dev/sda1        97G   19G   79G  20% /etc/mongo
shm              64M     0   64M   0% /dev/shm
tmpfs           7.9G   12K  7.9G   1% /run/secrets/kubernetes.io/serviceaccount
tmpfs           7.9G     0  7.9G   0% /proc/acpi
tmpfs           7.9G     0  7.9G   0% /proc/scsi
tmpfs           7.9G     0  7.9G   0% /sys/firmware
```

4.6.4 Steps to Expose Database/RabbitMQ to Access

Step 1: Download Database/RabbitMQ yaml files.

Step 2: Open PowerShell and login

Follow the instruction to login your account

```
# az login
```

Select your subscription, if you have multiple subscription, you have to set which subscription AKS service is located.

```
# az account set --subscription "SUBSCRIPTION_NAME”
```

Try to install AKS command line.

```
# az aks install-cli
```

To get AKS credential to access.

```
# az aks get-credentials --resource-group RESOURCE_GROUP --name K8S_CLUSTER
```

Step 3: Expose MongoDB/PostgreSQL/RabbitMQ

```
# kubectl create –f service-mongodb.yaml

# kubectl create –f service-postgres.yaml

# kubectl create –f service-rmq.yaml
```

Step 4: Use Kubernetes Dashboard to check service public address

```
# az aks browse --resource-group RESOURCE_GROUP --name K8S_CLUSTER
```

Port for DeviceOn Server Used

Name & Description	Inbound Port

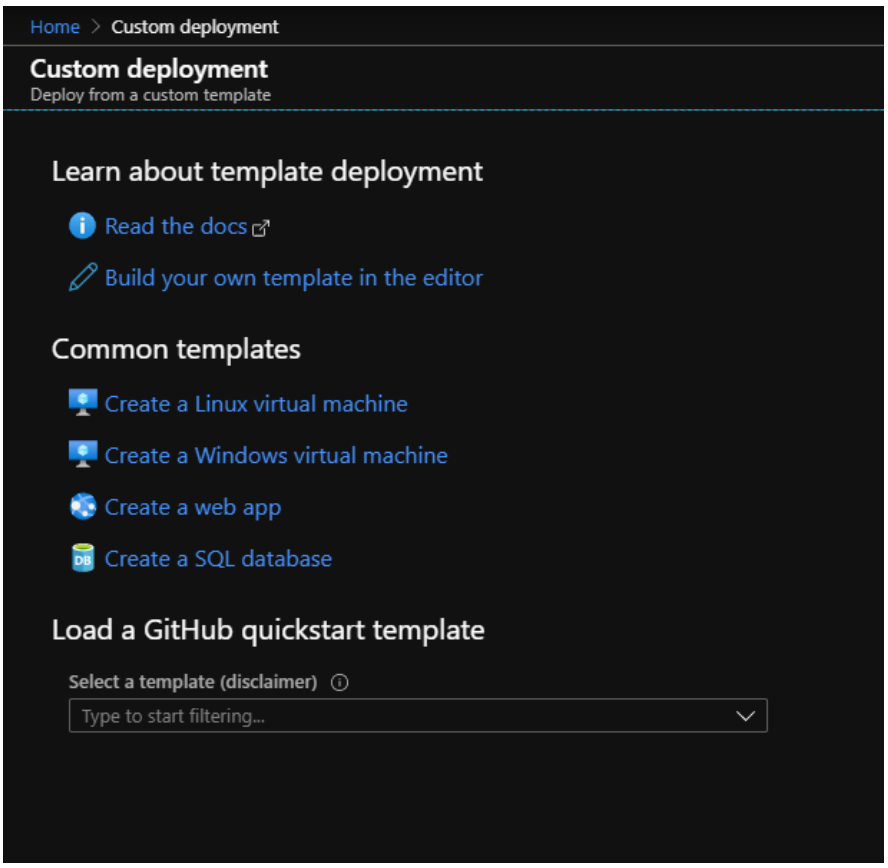
1	Message Broker (RabbitMQ) MQTT, MQTTs	1883, 8883
2	Message Broker (RabbitMQ) AMQP, AMQPs	5671, 5672
3	Message Broker (RabbitMQ) Management Console	15672
4	Database for MongoDB	27017
5	Database for PostgreSQL	5432

4.6.5 Steps to Deploy DeviceOn to AKS by Manual

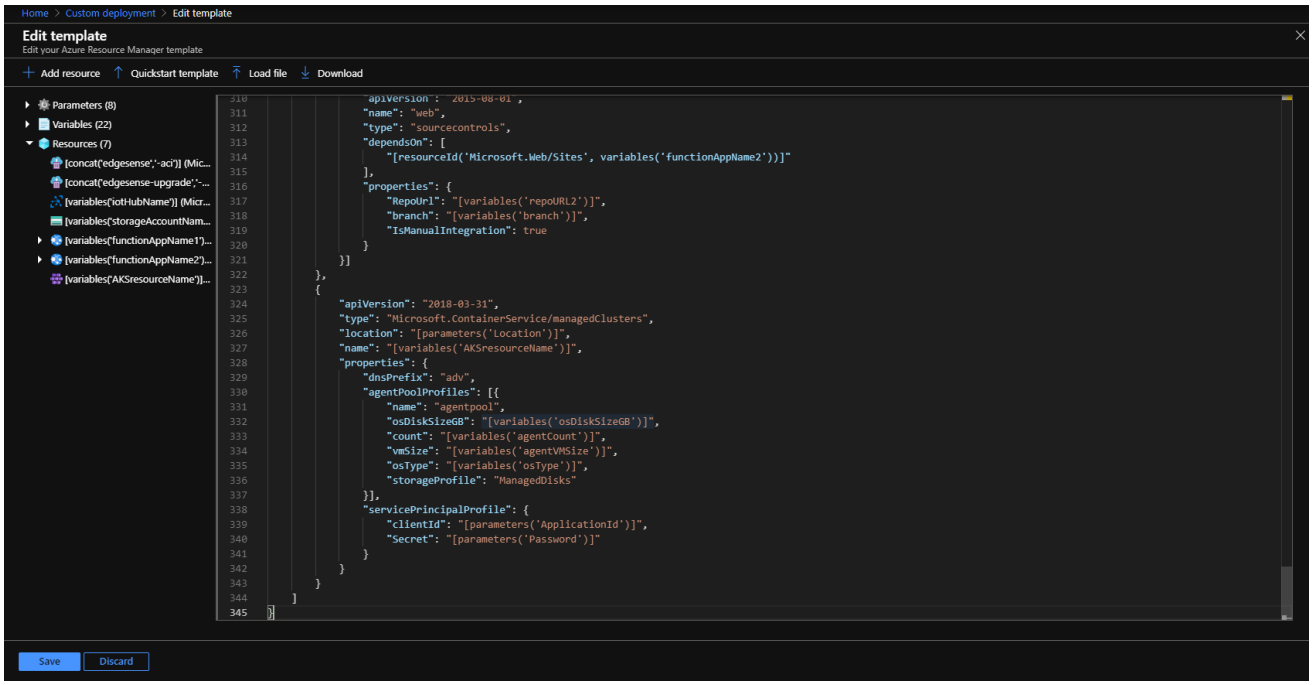
Step 1: Download ARM template and use notepad to open the file.

Step 2: Copy all content from the template

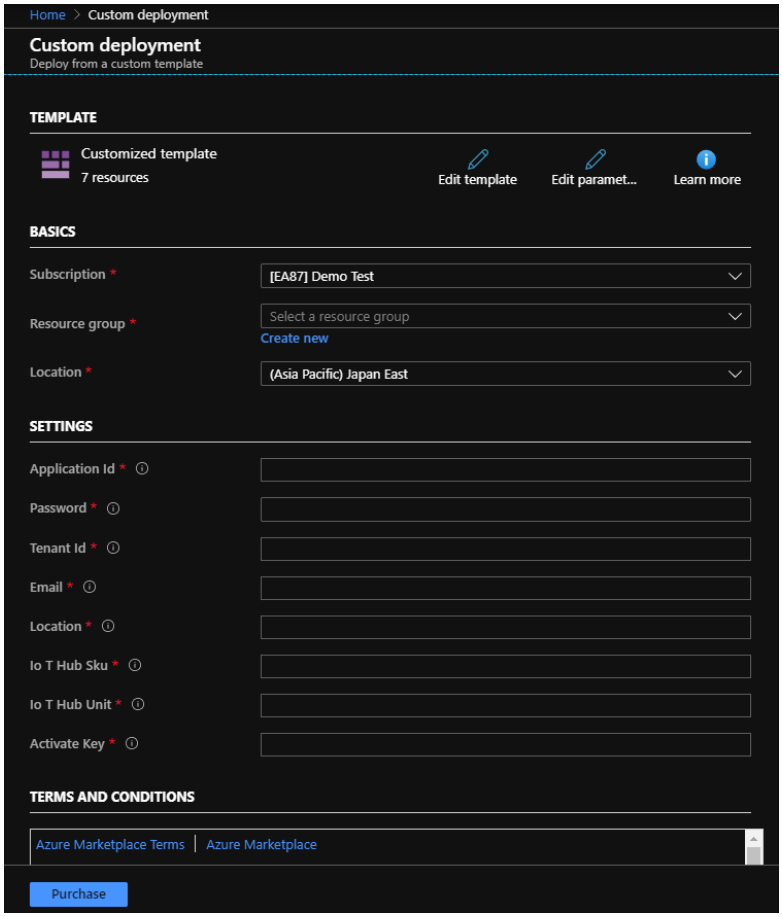
Step 3: Login Azure Portal to start custom deployment



Step 4: Select **Build your own template in the editor** and paste the content from Step 2, then click save button.



Step 5: Please enter these parameters to start deploy, the **Application Id**, **Password (Client Secrets)**, **Tenant Id**, please refer to Section 2.1.2.

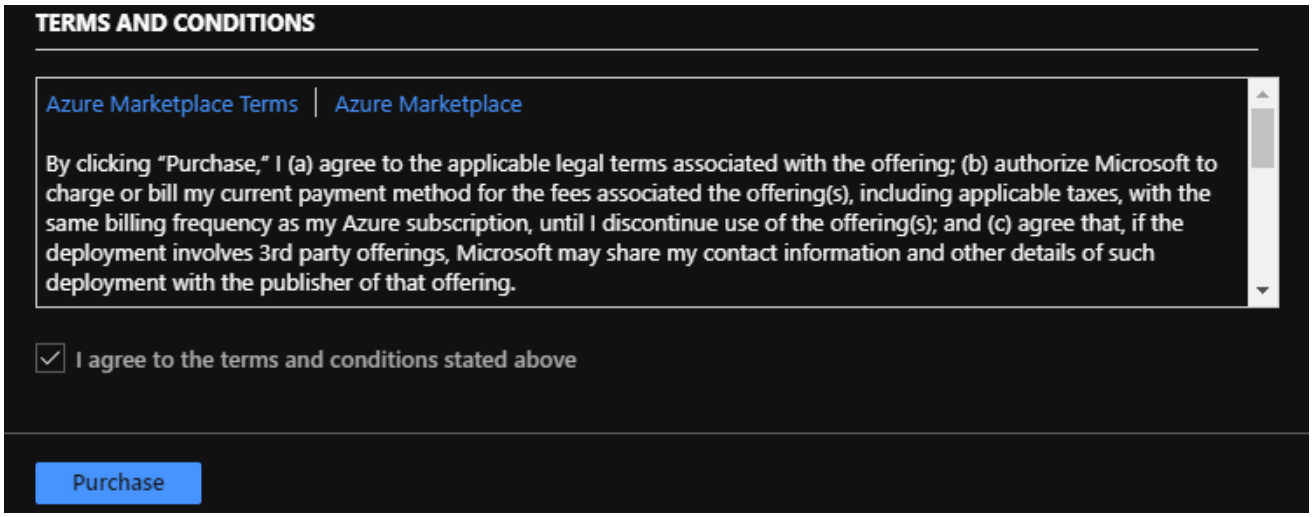


- Email: The email address to get deployment status.
- Location: Please refer below table

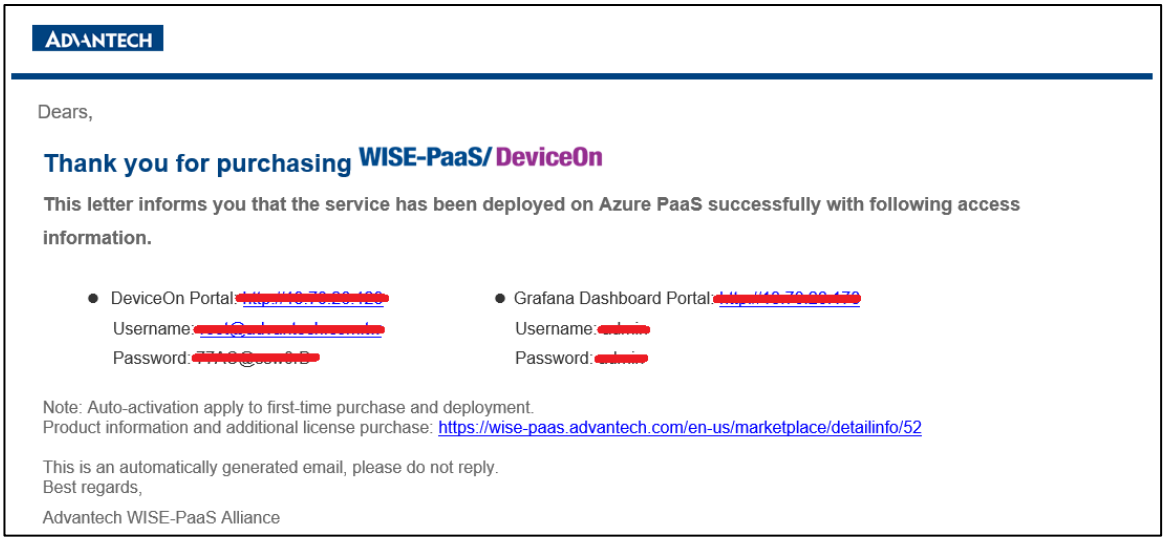
Data Center	Location name
Asia East	eastasia
Asia Southeast	southeastasia
Japan East	japaneast
US East	eastus
Europe North	northeurope

- IoT Hub SKU: S1/S2/S3, default is S1
- IoT Hub Unit: 1 to 10, default is 1
- Activate key: Enter **N/A** to skip activate DeviceOn server automatically or please contact us to purchase license key.

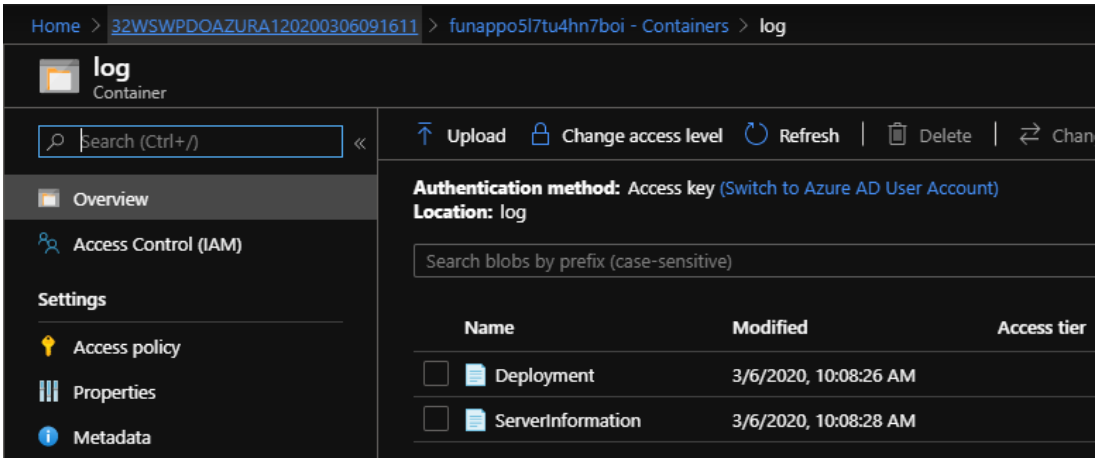
Step 6: Pick-up agreement item and click **Purchase** button to start deployment process.



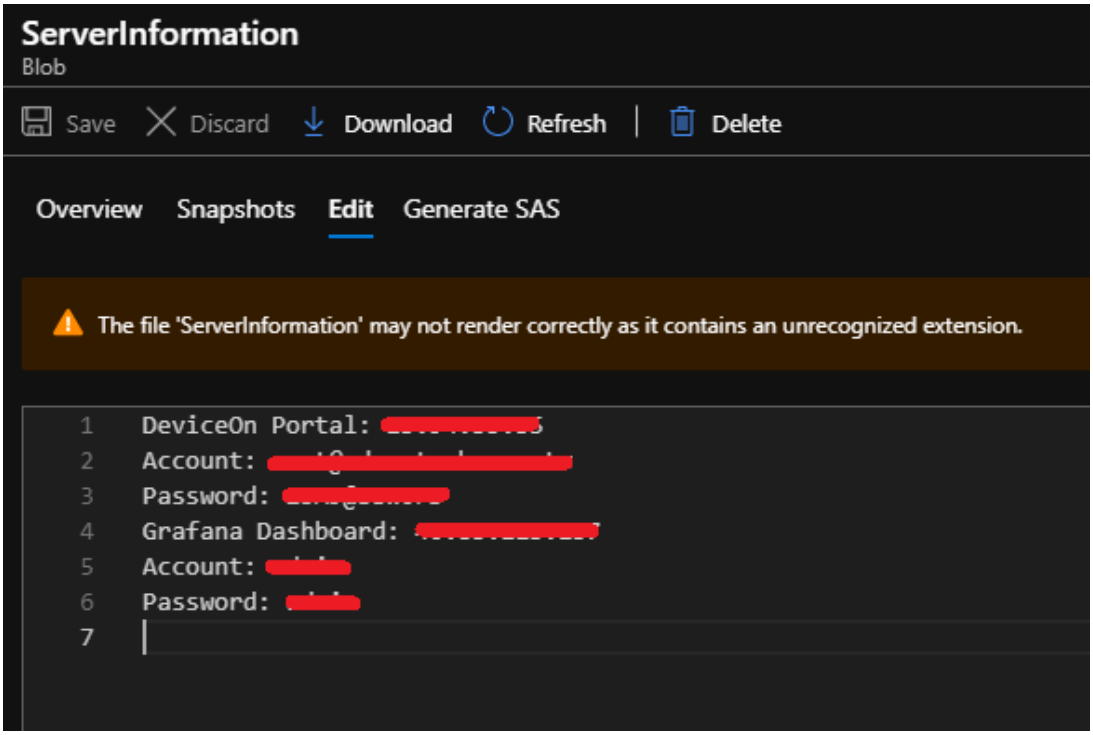
Step 7: After deployment, you will receive a mail to get server information, including account, password and URL.



To prevent your mail blocked, we write the server information in Azure blob simultaneously.



The credential and access information also on the “ServerInformation”.

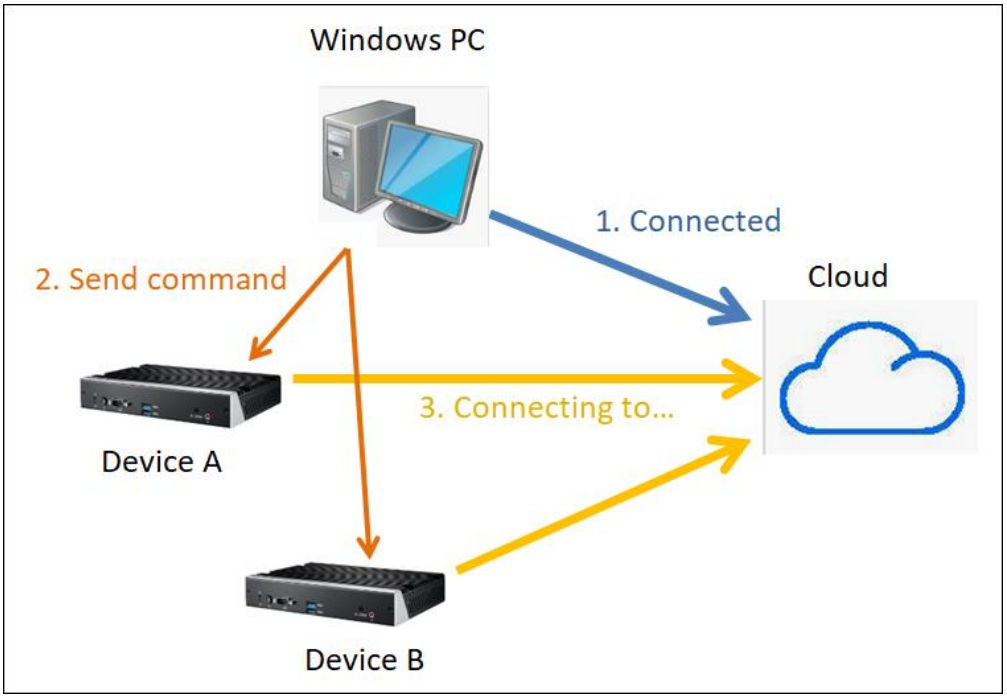


4.7 How to Batch Provision to Your Devices

WISE-Agent will connect to DeviceOn server through **Credential URL** and **IoT Key** and those setting in **agent_config.xml**, if you have many devices (that has WISE-Agent in it) need to connect to the server, it takes time to modify agent_config.xml in each device. Here, we build-in the “**Local Provision**” Plugin to speed up this process. You will learn how to trigger all local devices to connect to the server with the same Credential URL and IoT Key.

The WISE-Agent local provision plugin will send Credential URL and IoT key to other local agent devices,

and the local agent devices can connect to the server successfully. In following figure, you can send trigger command to make device A and B connect to a server with a Windows GUI tool.



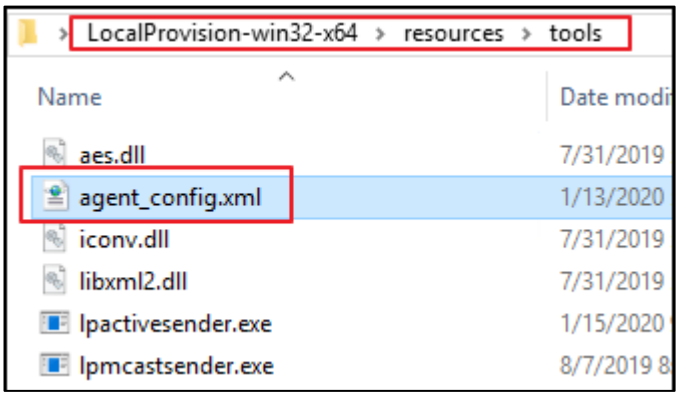
4.7.1 Prerequisite

- All devices must install WISE-Agent in it.
- All devices and the control PC must in the same local network (The multicast packet will not be filtered)
- All devices have the capability to connect to DeviceOn server.

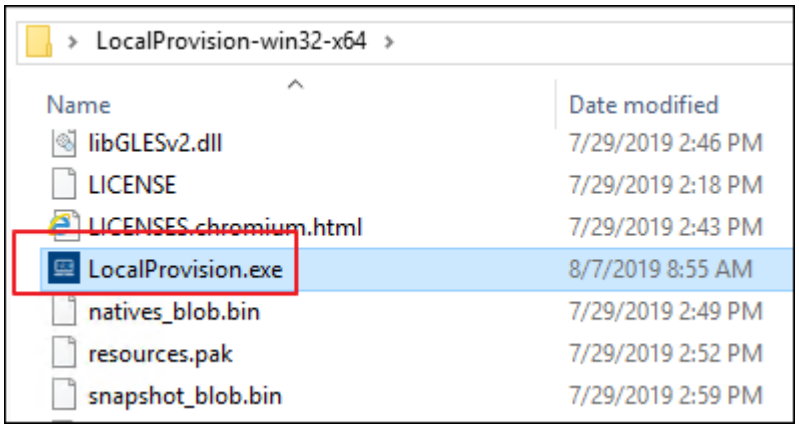
4.7.2 Steps to Local Provisioning

Step 1: Download and unzip the local provision GUI tool.

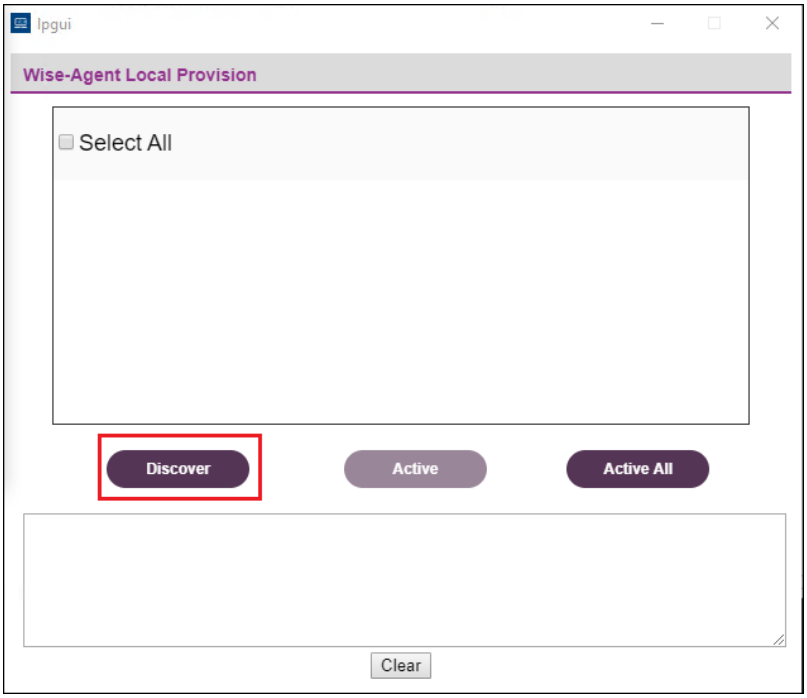
Step 2: Place valid "**agnet_config.xml**" file (with correct Credential URL and IoT Key) to "**GUI tool\resources\tools**" folder



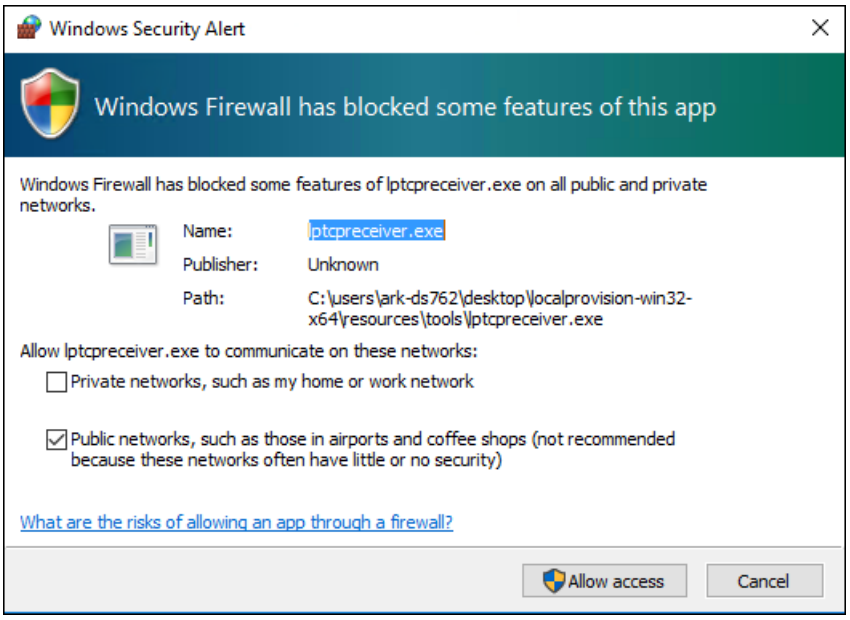
Step 3: Double click "**LocalProvision.exe**"



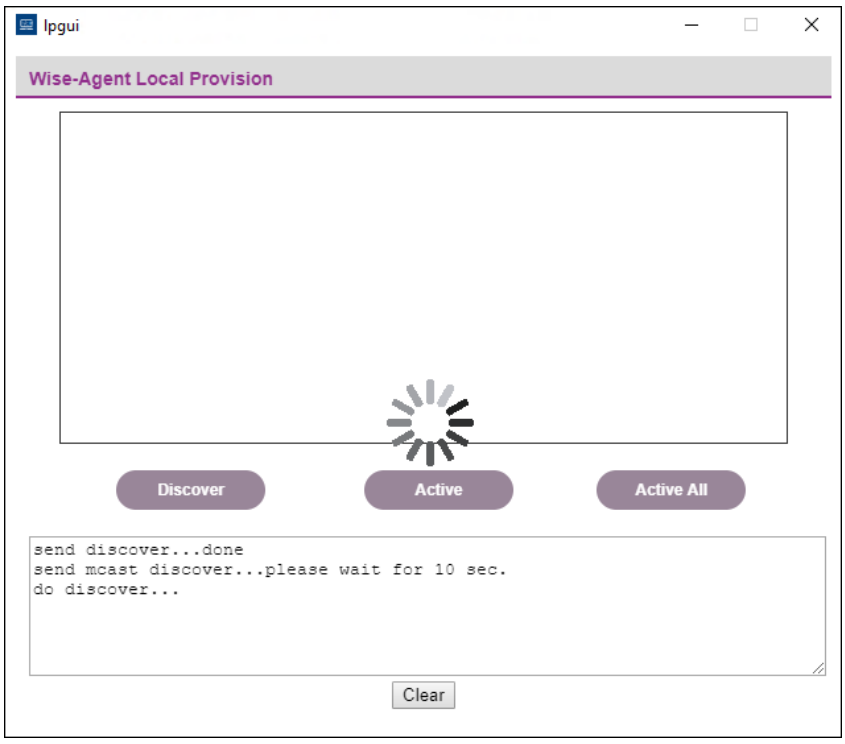
Step 4: Click **Discover** button

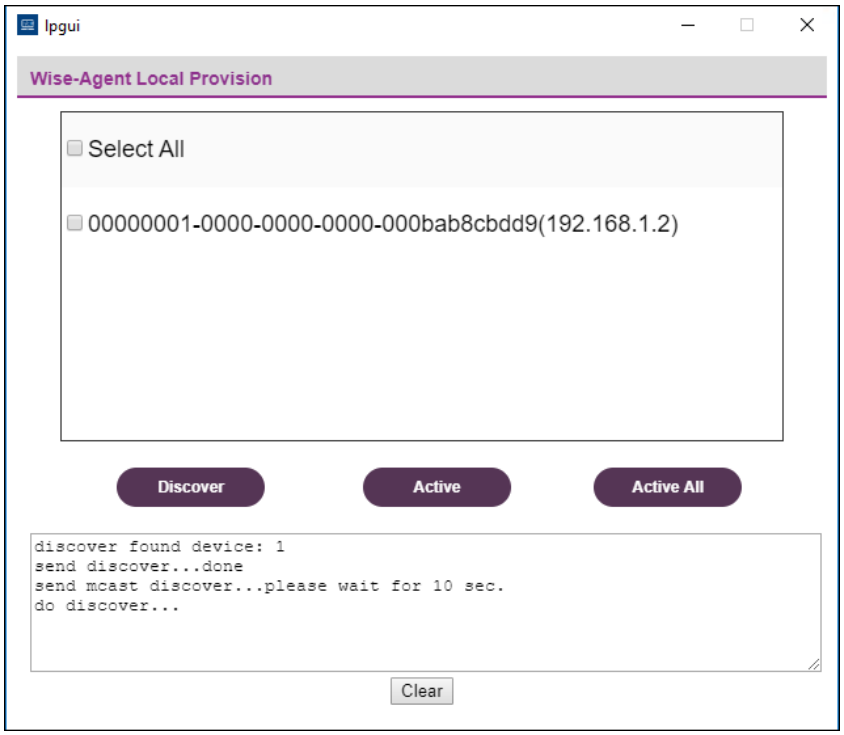


If windows display a firewall dialog, please click allow to enable TCP server permission in tool.

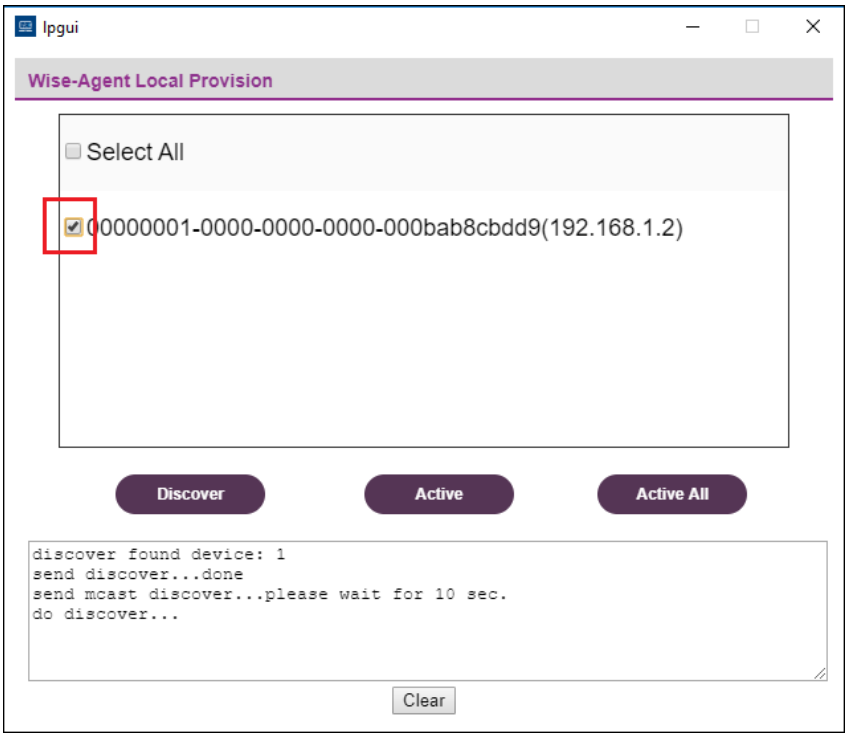


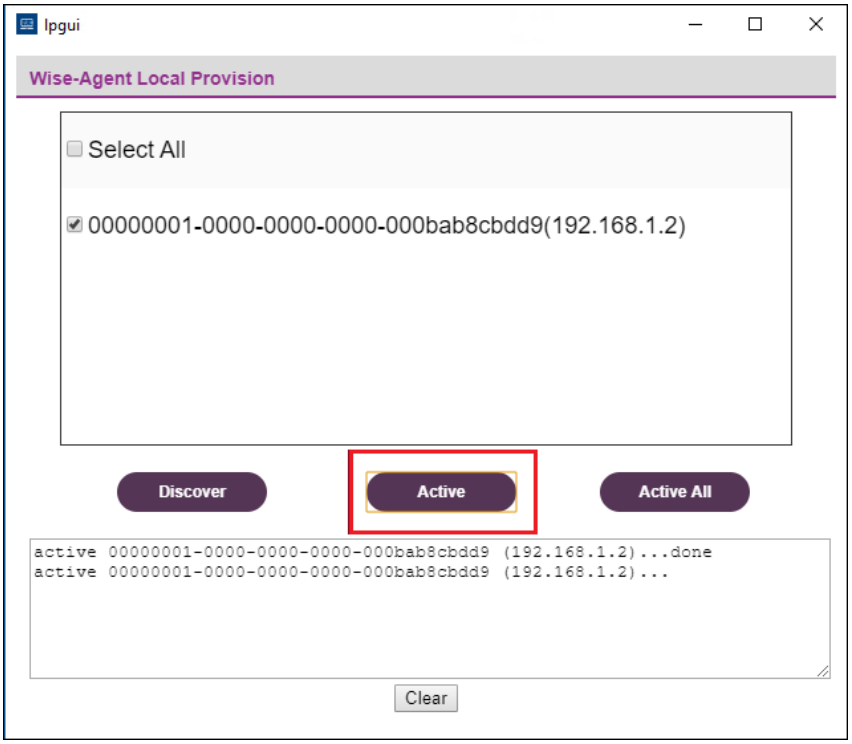
Step 5: Wait for 10 second and then you can get the devices on checkbox list.





Step 6: Pick-up the device that you would like to connect to the server and click **Active.**





Until now, the checked devices should connect to server after few second later.

4.7.1 **Troubleshooting**

Why can't I find some WISE-Agent devices? Please help check following:

A. Please check if your local provision plugin is enabled.
Open the **module_config.xml** in "Installation path\module\" to check if local provision handler is enabled.

```
1  <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2  <XMLConfigSettings>
3    <BaseSettings>
4      <ModuleNum>17</ModuleNum>
5      <ModuleName1>HDDHandler</ModuleName1>
6      <ModulePath1>\module\HDDHandler.dll</ModulePath1>
7      <ModuleEnable1>TRUE</ModuleEnable1>
8      <ModuleName2>PowerOnOffHandler</ModuleName2>
9      <ModulePath2>\module\PowerOnOffHandler.dll</ModulePath2>
```

```
50      <ModuleName16>EmbIPC</ModuleName16>
51      <ModulePath16>\module\EmbIPC.dll</ModulePath16>
52      <ModuleEnable16>TRUE</ModuleEnable16>
53      <ModuleName17>HDDPMQ</ModuleName17>
54      <ModulePath17>\module\HDDPMQ.dll</ModulePath17>
55      <ModuleEnable17>TRUE</ModuleEnable17>
56      <ModuleName18>LocalProvision</ModuleName18>
57      <ModulePath18>\module\LocalProvisionHandler.dll</ModulePath18>
58      <ModuleEnable18>TRUE</ModuleEnable18>
59    </BaseSettings>
60  </XMLConfigSettings>
```

B. Please check if your device and windows PC is in the same local network and can transfer multicast packets.

C. Because the local provision discovers wise-device by UDP port **9178** and TCP port **9177**, please check if your IT block these ports in your local network.

5. DeviceOn Development Guide

5.1 WISE-Agent Plugin Development

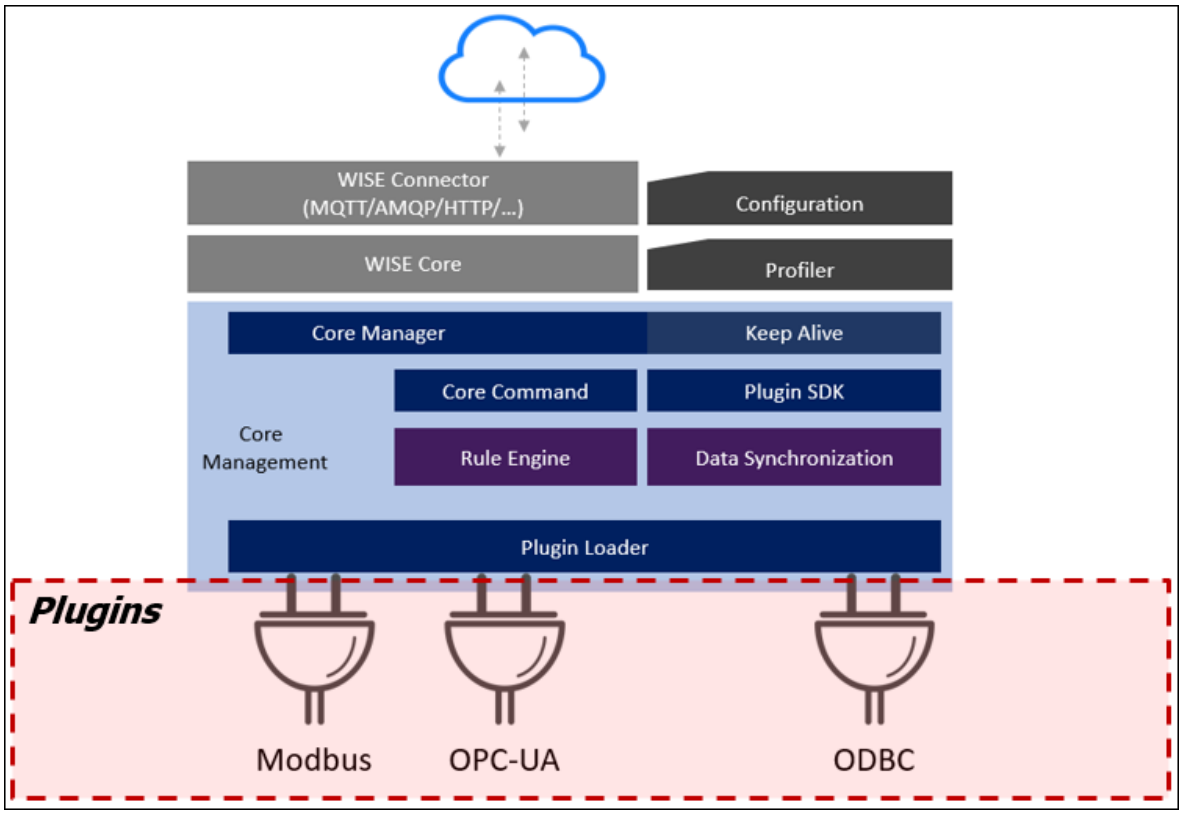
Advantech provides an edge software tool to communicate and exchange information between IoT (Internet of Thing) devices and DeviceOn cloud, called a WISE-Agent. The WISE-Agent not only provides a rich set of users friendly, intelligent, standardization and scalability.

- **Standardization**
  The communication protocol is based on the MQTT protocol to communicate and exchange data with DeviceOn cloud. The IoT sensor data report format is following the IPSO Alliance. in JSON format.
- **Portability**
  The whole framework is written in C language and follow the ANSI C Standard that C compilers are available for most systems and are often the first compiler provided for a new system, such as OpenWRT, Yacto and Linux based system.
- **Scalability**
  The WISE-Agent is modular design and offering plugin concept to Plug & Play (PnP) which is one with a specification that facilitates the discovery of a Plugin in a system without the need for a physical device to advanced configuration or user intervention in resolving resource conflicts.

Besides the basic device connectivity, the WISE-Agent provides an advanced heartbeat solution to synchronize device status. On the different network environment, how to keep your device data without loss? The WISE-Agent has built-in "**Data Synchronization**" to avoid and overcome the disconnect for a long time. For various protocols, we offer a plugin SDK, users only focus on how to retrieve the data, do not worry about the connectivity and stability.

5.1.1 WISE-Agent Architecture

WISE-Agent includes two parts, one is the **Core Framework** and **Plugins**.



- **Core Framework**

  The main library used to communicate with WISE-PaaS IoTHub or standard MQTT broker and include below components.

  - ✧ **Platform Profiler**: describes the target platform (e.g., OS version, SN, Device name, MAC address)
  - ✧ **Configuration**: describes how to connect to MQTT broker (e.g., Credential URL, IoTKey, TLS/SSL settings)
  - ✧ **Core Manager:** integrates and manages the resources and keeps them alive.
  - ✧ **Core Command:** responsible for handling commands that interact with internal components (e.g., rename, update, get capability, auto report start/stop)
  - ✧ **Plugin SDK:** A plugin framework that makes plugin implement more easily.

- **Keep Alive:** A component to detect the connection between WISE-Agent and DeviceOn Server.
- **Data Synchronization:** kernel plugin that caches and restores data to ensure zero downtime.
- **Rule Engine:** kernel plugin that supports the threshold rule check and then sends event or trigger actions
- **Plugin Loader:** responsible for loading and managing plugins indicated in module_config.xml

- **The plugins**

  The plugins include IPC monitoring (Advantech Hardware, HDD/SSD, Networks, Process…etc.), control function (Backup/Recovery, Protection, Remote Desktop, Terminal…), and sensor protocol collection. Following are the list of supported plugins in WISE-Agent.

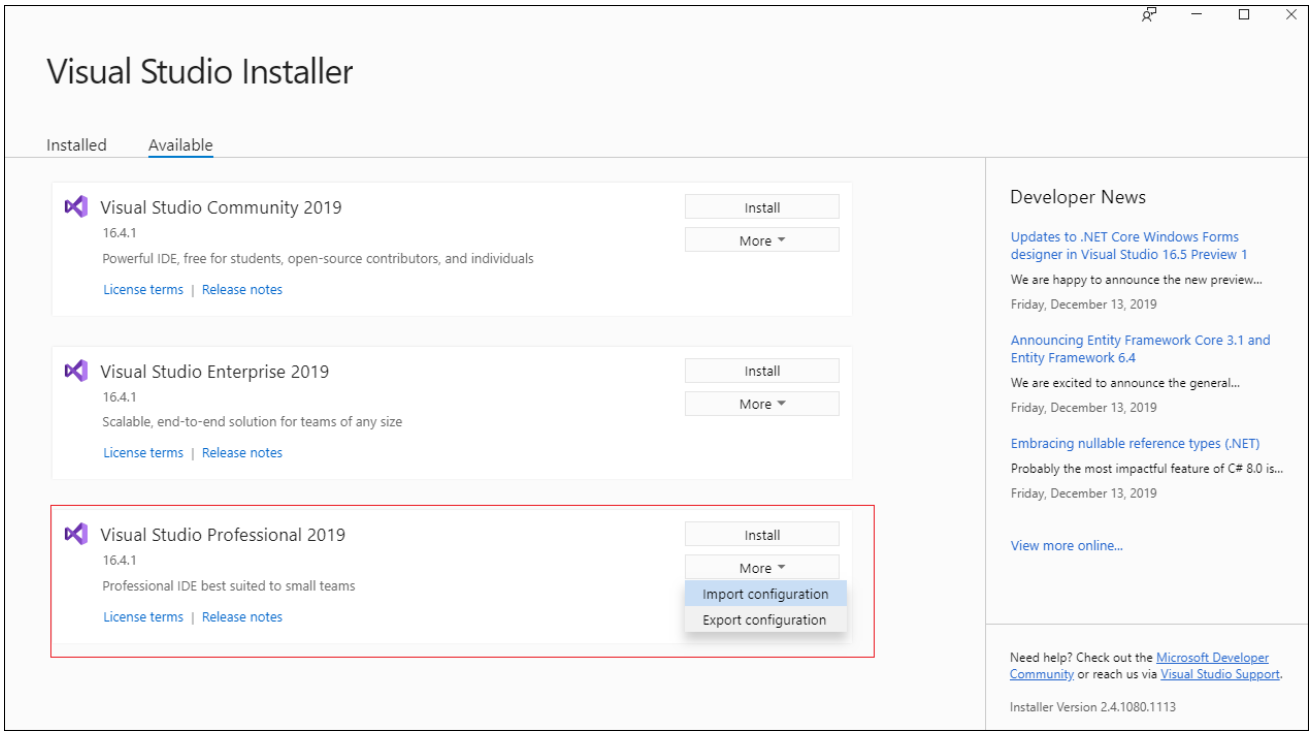
  - **SUSI Control**: Monitoring and Control Advantech Hardware Platform
  - **HDD Monitoring:** Monitoring Hard Drives (HDD, SSD) Usage, Healthy and S.M.A.R.T Information, especially for Advantech SQFlash.
  - **Network Monitoring:** Monitoring Network Interface Usage, Throughput…
  - **Process Monitoring:** Monitoring System Process Status, CPU, Memory Usage.
  - **Power Management:** Remote Control Power On, Off, Reboot, Sleep, Hibernate.
  - **Backup/Recovery:** Remote Backup/Recovery System via Acronis
  - **Protection:** Remote System Protection via McAfee
  - **Remote Desktop:** Remote Desktop via VNC Viewer
  - **Remote Terminal:** Remote Terminal Command
  - **Remote Screenshot:** Remote Screenshot on Current Screen
  - **OTA (Over-the-Air):** Remote Software, Firmware Update
  - **System Program Monitoring:** System Program Information
  - **Embedded Control:** Advanced Control (UWF, USB Lock, Keyboard Filter, …etc.) for Windows 10 Embedded, LTSC, LTSB
  - **HDD Prediction:** Build-in Hard Drives (HDD, SSD) Failure Prediction Model
  - **Modbus:** Modbus Device Data Gathering
  - **Service Plugin:** Bridge Southbound Device Service

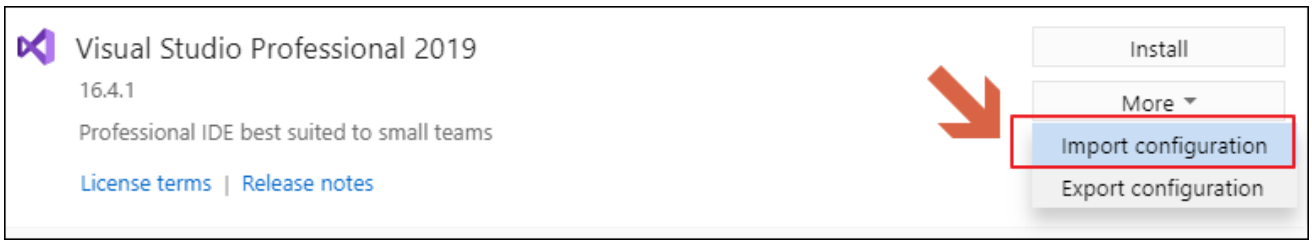
5.1.2 Prerequisite

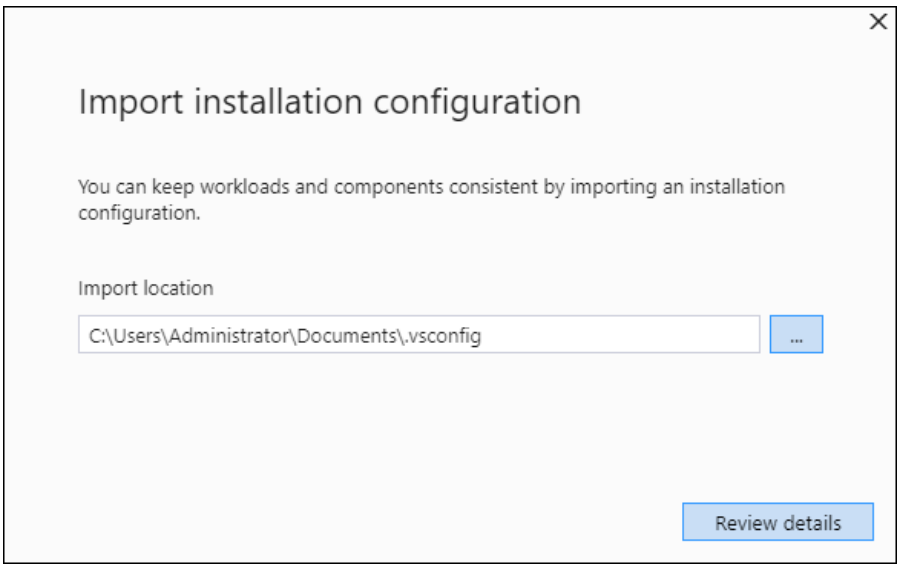
- Visual Studio 2019 for Windows Plugin
- Android NDK for Android Plugin
- A WISE-Agent that is running on your system.

5.1.3 Develop a Plugin on Windows Environment

Step 1: You can configure Visual Studio across your organization with installation configuration files, .[vsconfig]



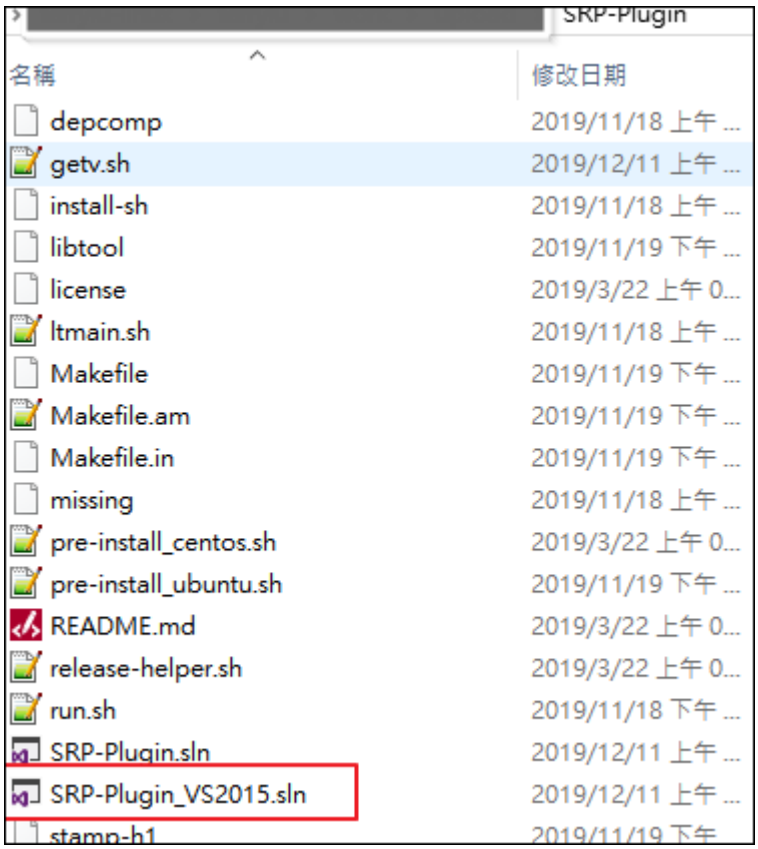




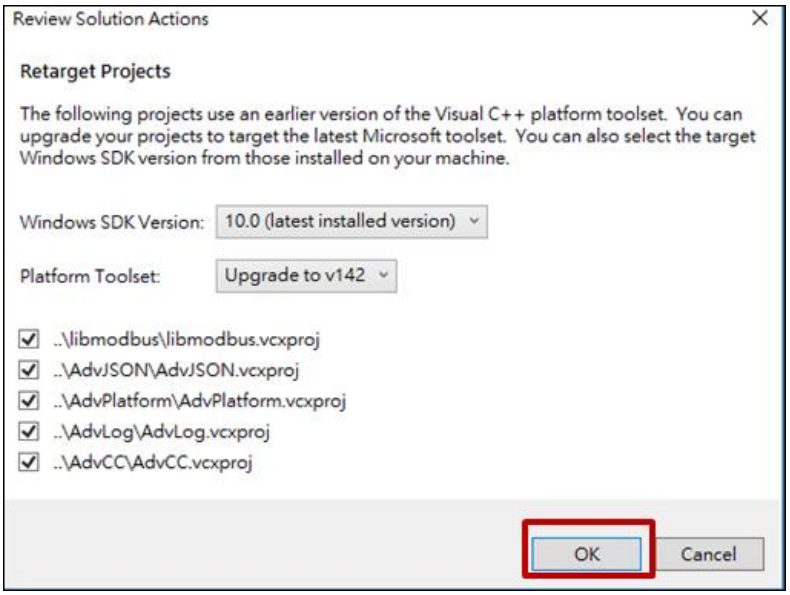
Step 2: Download SRP-Plugin,

git clone http://advgitlab.eastasia.cloudapp.azure.com/SRP-Connect/SRP-Plugin.git

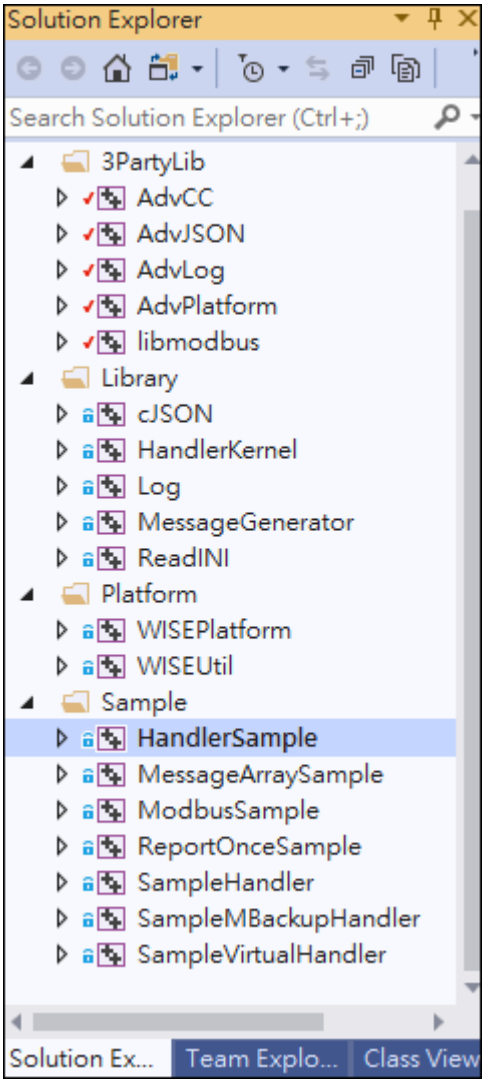
Step 3: Open SRP-Plugin solution file, **SRP-Plugin-V2015.sln**



Step 4: Click OK to update the SDK and Toolset for current compile environment



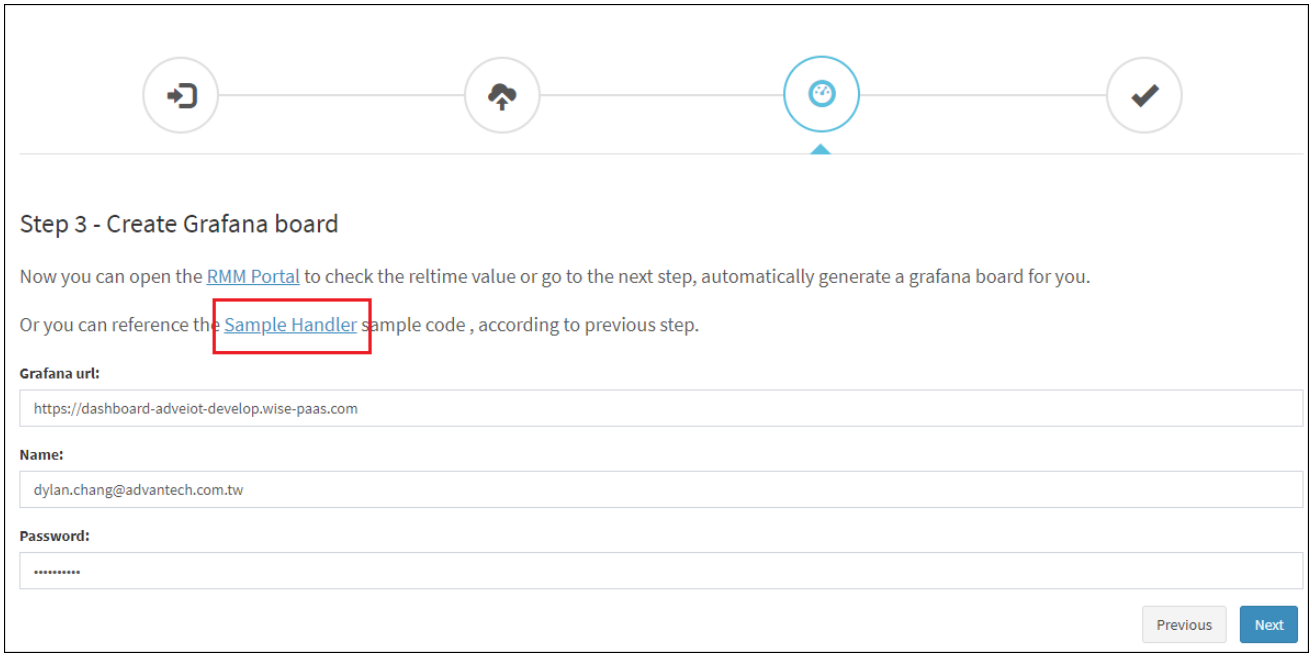
Step 5: You can implement new plugin base on plugin sample project.



Step 6: It is more easily to create a new plugin by Web-Simulator tools. Web-Simulator is an auxiliary tool that helps you quickly simulate data on the cloud via MQTT over WebSocket (network port: 15675) and directly generate the corresponding code. Following step will introduce how to create a new plugin by Web-Simulator tools. If you want to know exactly how this tool is used, you can refer Web-Simulator QuickStart.

Step 7: Download Web-Simulator tools.

Step 8: The sample code can be generated in the fourth step. Please save it as **handler_data.c** and replace it in the "**SRP-Plugin\Sample\HandlerSample**" path.



Step 9: Right click the "**HandlerSampe**" project in Step 5 and choose "**Solution**".

Step 10: Check output without error message. If appear error message, suggest to copy the error message search in google or ask Advantech technical people.

Step 11: After successfully completing the compilation, you can find all the **.dll** files in below path "**SRP-Plugin\Debug\module**"

Step 12: Download and install WISE-Agent for Windows. The default installation path is **C:\Program Files (x86)\Advantech\WISE-Agent**

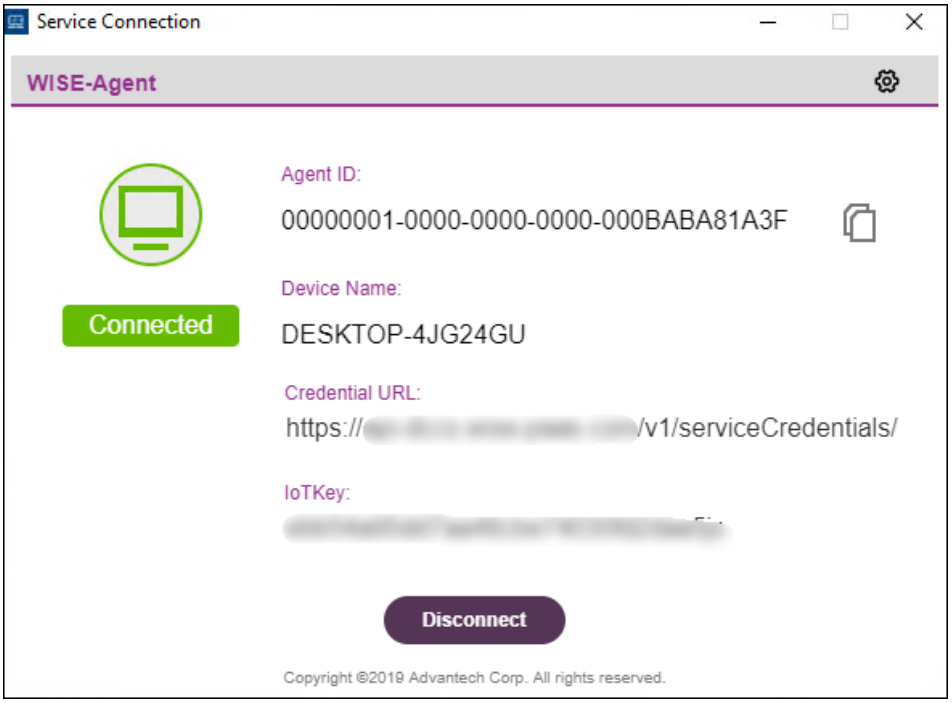
Step 13: After install the WISE-Agent, copy "**HandlerSample.dll**" file to "C:\Program Files (x86)\Advantech\WISE-Agent**module**" folder.

Step 14: Modify **module_config.xml** on "C:\Program Files (x86)\Advantech\WISE-Agent\module**module_config.xml**"

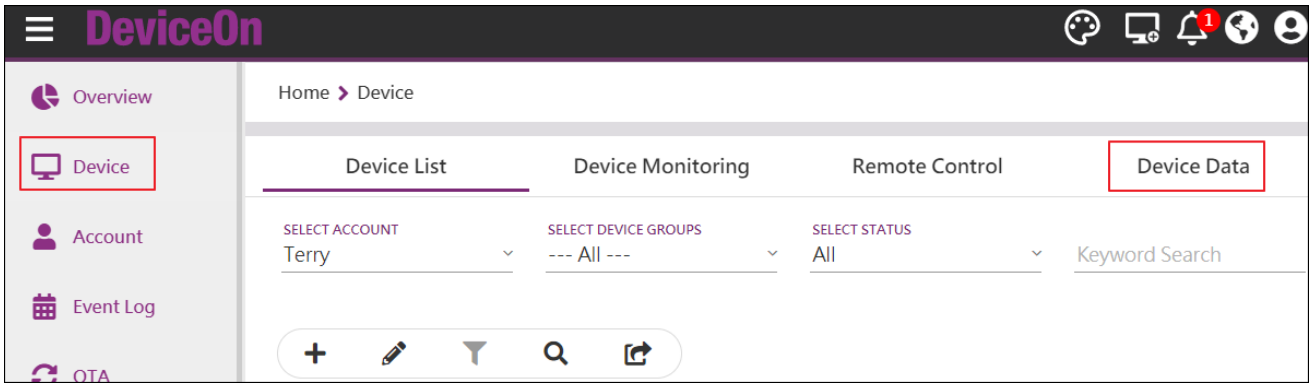
- Increase **ModuleNum** value in below line 3
- Add **HandlerSample.dll** item in below line 7.

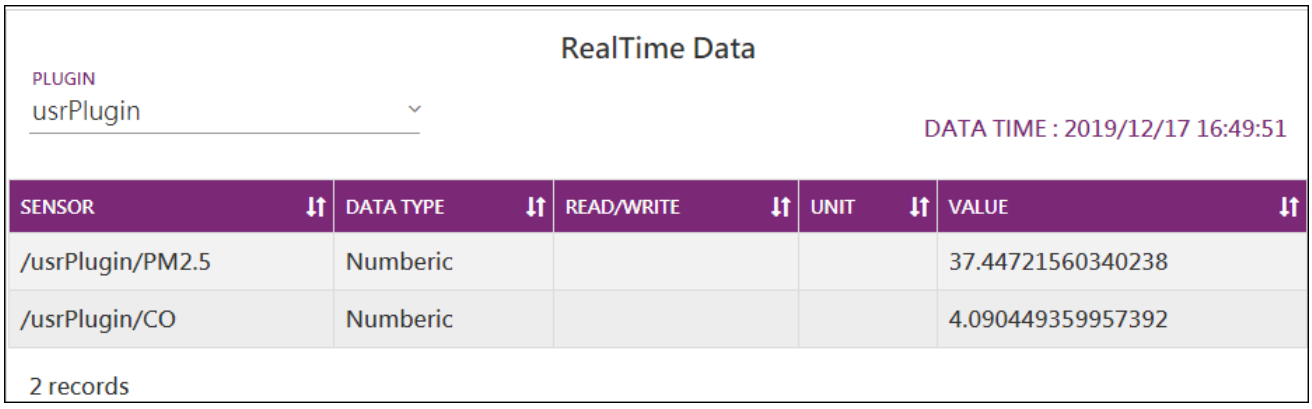
```
01. <?xml version="1.0"?>
02. <XMLConfigSettings><BaseSettings>
03. <ModuleNum>15</ModuleNum>
04. <ModuleName1>HDDHandler</ModuleName1><ModulePath1>module/HDDHandler.so</ModulePath1><ModuleEnable1>TRUE</ModuleEnable1>
05. ...
06. <ModuleName14>ServiceHandler</ModuleName14><ModulePath14>module/ServiceHandler.so</ModulePath14><ModuleEnable14>TRUE</ModuleEnable14>
07. <ModuleName15>HandlerSample</ModuleName15><ModulePath15>module/HandlerSample.so</ModulePath15><ModuleEnable15>TRUE</ModuleEnable15>
08. </BaseSettings>
09. </XMLConfigSettings>
```

Step 15: Reconnect WISE-Agent by “Server Connection” tools. Press “Disconnect” then “Connect”.



Step 16: Check if your plugin appears in DeviceOn Page, (**Device** -> **Device Data** -> **PLUGIN**)





5.1.4 **Develop a Plugin on Linux Environment**

The following steps are handled in Ubuntu or Debian system. If your target device is Yocto Linux, you have to set up cross-compile environment on your host PC. The example below shows how to set up for NXP i.MX8 projects.

$ /opt/fsl-imx-xwayland/4.14-sumo

$ source environment-setup-aarch64-poky-linux

From now, the DeviceOn supports the following RISC platform, please refer to the SDK links relating to the platform you are developing for details.

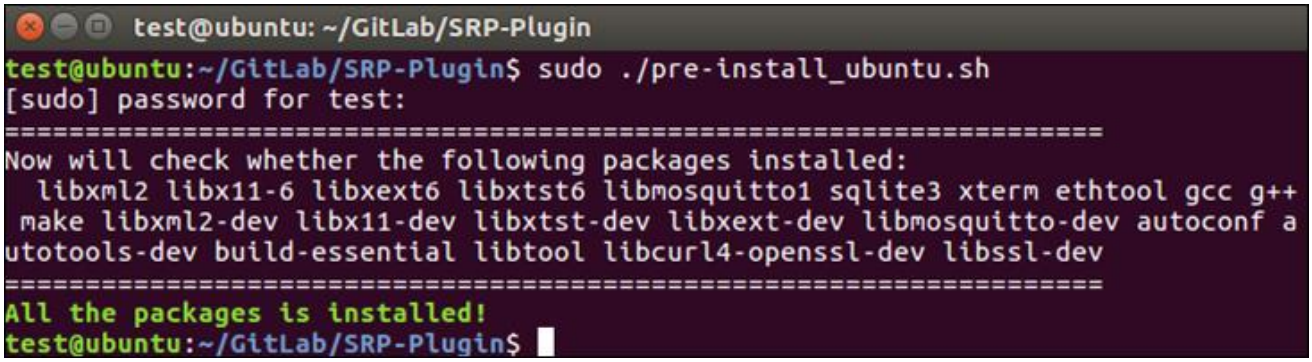
Platform	OS	Architecture	SDK
NXP i.MX8	Yocto 2.5.2	aarch64	Link
NXP i.MX6	Yocto 2.1.1	armv7-a	Link
Qualcomm APQ8016	Yocto 2.1.3	aarch64	Link
TI AM335x	Yocto 2.4	armv7-a	Link
RK3288	Debian 9.8	arm	N/A
RK3399	Debian 9.9	aarch64	N/A

http://ess-wiki.advantech.com.tw/view/AIMLinux/AddOn/DeviceOn#Supported_Platforms

Step 1: Download SRP-Plugin as Section 5.1.3 Step 2.

Step 2: In Plugin SDK (SRP-Plugin) folder, execute '**sudo ./pre-install_ubuntu.sh**' with root user authority to install compile tools and dependency libraries

Note: If you are developing with cross-compile, you can skip this step.



Step 3: You can implement new plugin base on plugin sample project or Web-Simulator in Section 5.1.3 Step 6 to Step 8.

Step 4: Copy **handler_data.c** that generated by Web-Simulator to

"SRP-Plugin/Sample/**HandlerSample**".

Step 5: Build SRP-Plugin by "**build-srpplugin.sh**"

```
$ ./build-srpplugin.sh
```

Step 6: You can find the release build file in "**SRP-Plugin/Release/module**" folder.

```
test@ubuntu: ~/GitLab/SRP-Plugin/Release/module
test@ubuntu:~/GitLab/SRP-Plugin$ cd Release/
test@ubuntu:~/GitLab/SRP-Plugin/Release$ ls
libAdvCC.so                 libAdvJSON.so.0.0.0  libmodbus.so.5
libAdvCC.so.0               libAdvLog.so         libmodbus.so.5.1.0
libAdvCC.so.0.0.0           libAdvLog.so.0       libWISEPlatform-1.1.1.so
libAdvCompression.so        libAdvLog.so.0.0.0   libWISEPlatform.so
libAdvCompression.so.0      libLog.so            libWISEUtil-1.1.1.so
libAdvCompression.so.0.0.0  libLog.so.0          libWISEUtil.so
libAdvJSON.so               libLog.so.0.0.0      module
libAdvJSON.so.0             libmodbus.so
test@ubuntu:~/GitLab/SRP-Plugin/Release$ cd module/
test@ubuntu:~/GitLab/SRP-Plugin/Release/module$ ls
HandlerSample.so        MessageArrayHandler.so.0      ModbusSample.so.0.0.0
HandlerSample.so.0      MessageArrayHandler.so.0.0.0  module_config.xml
HandlerSample.so.0.0.0  ModbusSample.so
MessageArrayHandler.so  ModbusSample.so.0
```

Step 7: Please download [WISE-Agent] for Ubuntu 16.04 x64. The default installation path is /usr/local/AgentService.

Step 8: After install the WISE-Agent, copy "**HandlerSample.so.0.0.0**" and "**HandlerSample.so**" files to "/usr/local/AgentService/module/" folder.

```
$ sudo cp -a Release/module/HandlerSample.so* /usr/local/AgentService/module/
```

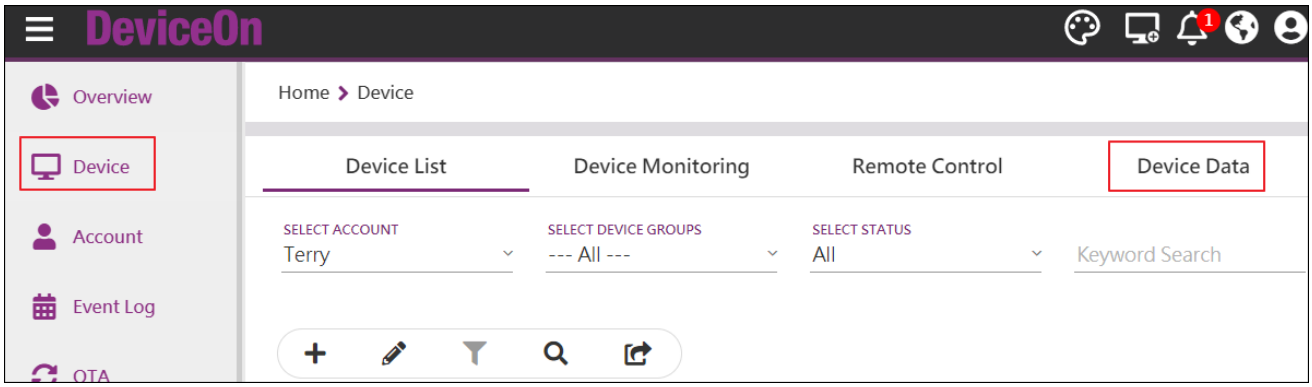
Step 9: Modify "/usr/local/AgentService/module/module_config.xml" Increase **ModuleNum** value in below line 3. Add HandlerSample.so item in below line 7.

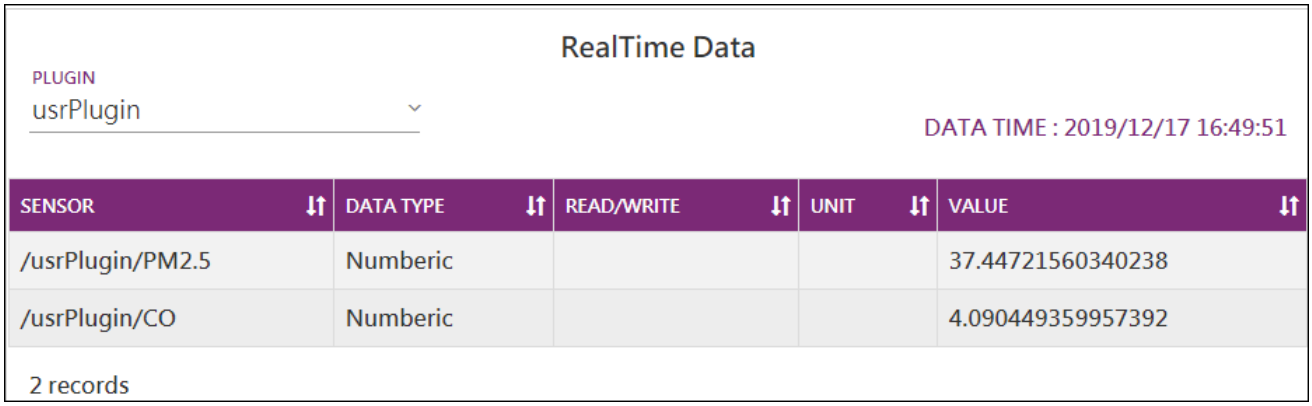
```
01. <?xml version="1.0"?>
02. <XMLConfigSettings><BaseSettings>
03. <ModuleNum>15</ModuleNum>
04. <ModuleName1>HDDHandler</ModuleName1><ModulePath1>module/HDDHandler.so</ModulePath1><ModuleEnable1>TRUE</ModuleEnable1>
05. ...
06. <ModuleName14>ServiceHandler</ModuleName14><ModulePath14>module/ServiceHandler.so</ModulePath14><ModuleEnable14>TRUE</ModuleEnable14>
07. <ModuleName15>HandlerSample</ModuleName15><ModulePath15>module/HandlerSample.so</ModulePath15><ModuleEnable15>TRUE</ModuleEnable15>
08. </BaseSettings>
09. </XMLConfigSettings>
```

Step 10: Restart WISE-Agent

```
$ sudo systemctl restart saagent
```

Step 11: Check if your plugin appears in DeviceOn Page, (**Device** -> **Device Data** -> **PLUGIN**)





5.1.5 Develop a Plugin on Android Environment

Step 1: Download SRP-Plugin as Section 5.1.3 Step 2.

Step 2: In Plugin SDK (SRP-Plugin) folder, execute **android_build.sh**, where

- CMD : -b : build, -c : clean
- APP_ABI : x86, armeabi-v7a and so on

For example, to build an armeabi-v7a arch plugin you may enter:

source android_build.sh -b armeabi-v7a &

Step 3: You can find the release build file in

"**SRP-Plugin/~/obj/local/armeabi-v7a/libHandlerSample.so**" folder.

Step 4: Copy **libHandlerSample.so** to DUT in the **/system/lib/module/**.

Step 5: Modify **module_config.xml in** DUT as 5.1.3 step 14 described.

Step 6: Check handler as 5.1.3 step 16 described.

5.2 DeviceOn UI Plugin Development

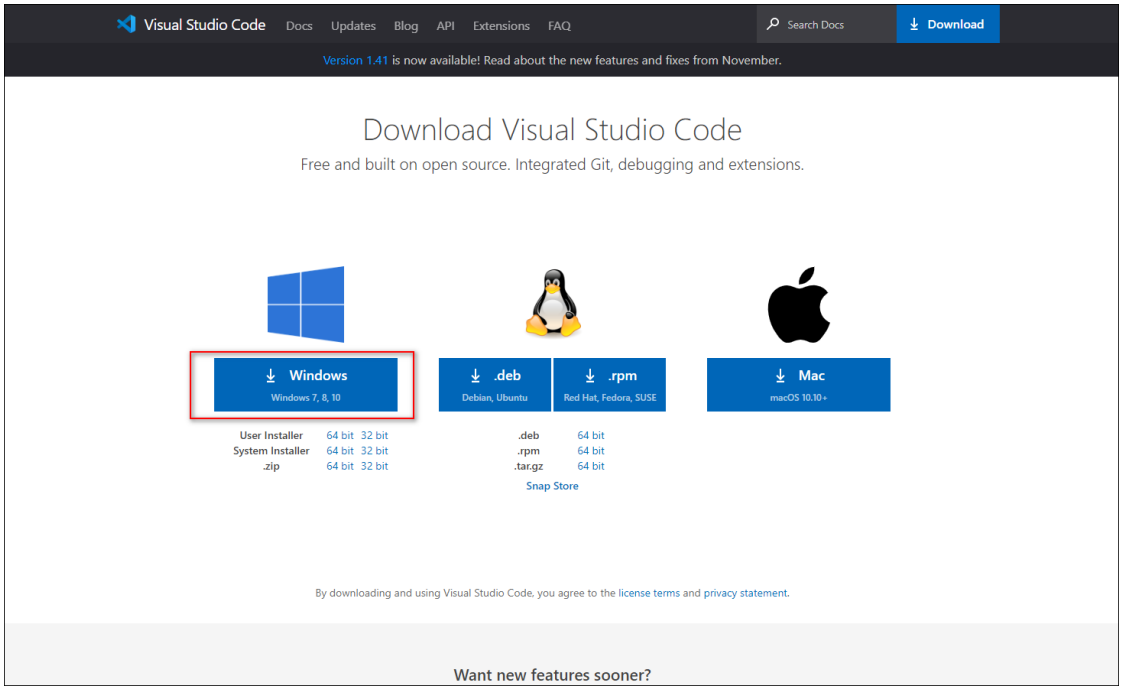
Actually, DeviceOn provide plenty of features to remote management, control to your edge devices, but it's hard to meet all domains application, such as, medical, traffic, energy system and etc. Fortunately, DeviceOn provide APIs and Addins (web user interface) for users to develop their own solution.

5.2.1 Prerequisite

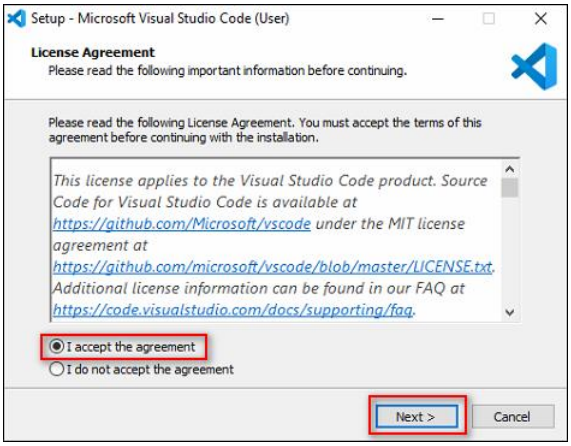
- Visual Studio Code V 1.4.1
- DeviceOn Server

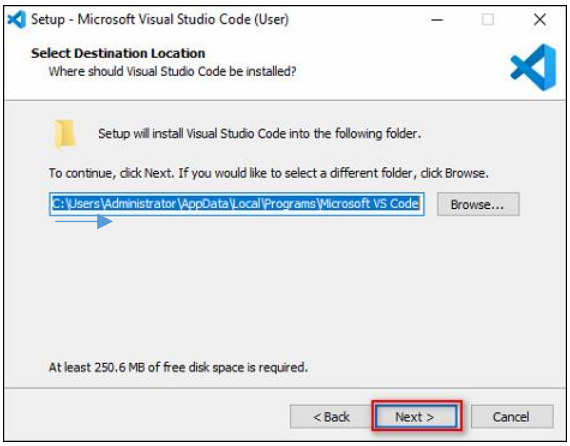
5.2.2 Environment Setup

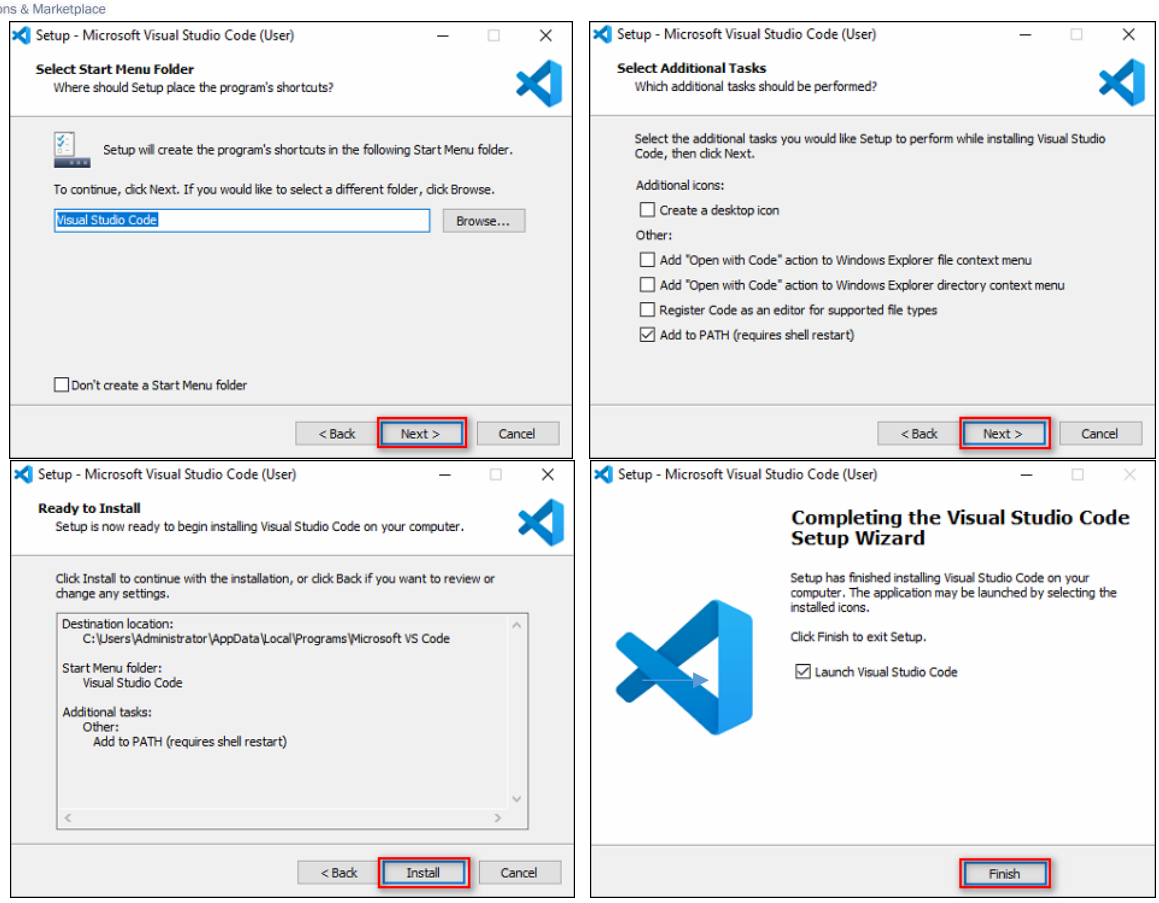
Step 1: Download Visual Studio Code v-1.4.1 and launch VSCodeUserSetup-x64-1.41.1.exe.



Step 2: Install Visual Studio Code, step by step.







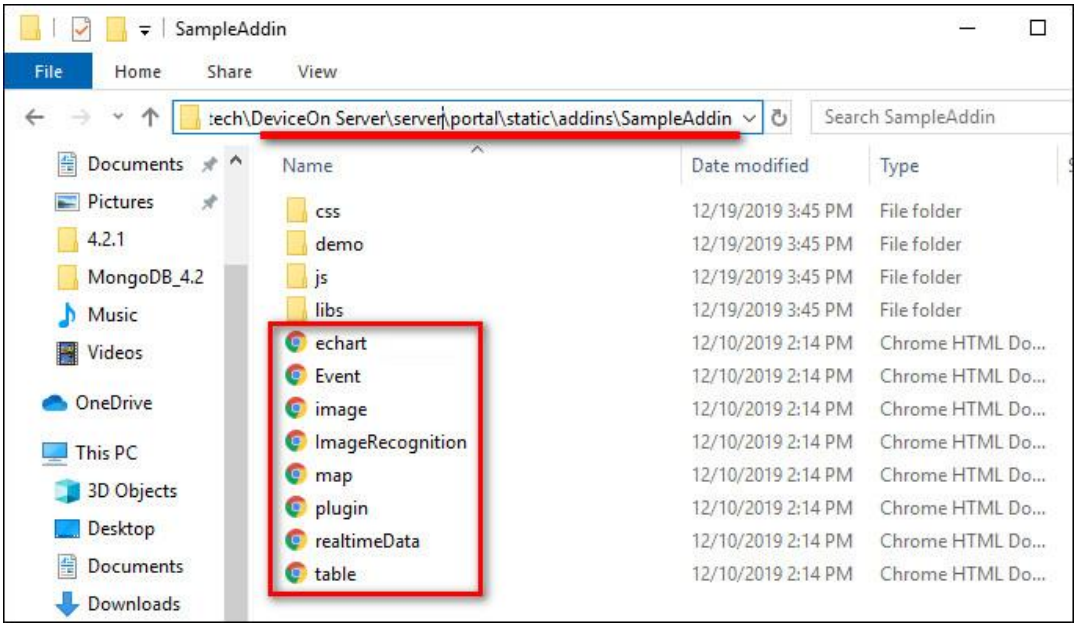
Step 3: Install DeviceOn Server, if you don't install DeviceOn Server before, please reference Section 2.2.

5.2.3 **Develop a Sample Add-in**

Step 1: Open DeviceOn Server folder and go to the installation path:

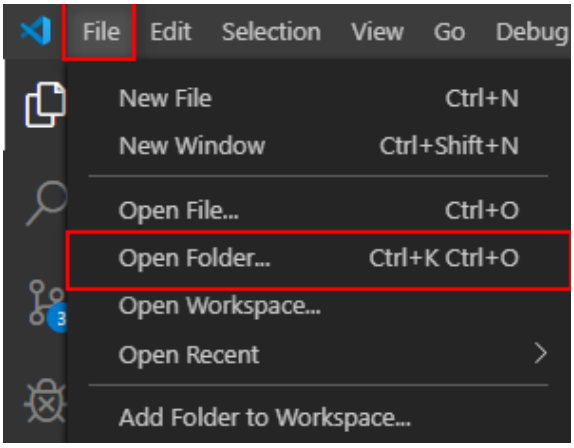
\DeviceOn Server\server\portal\static**addins\SampleAddin**.

Here are serval Add-in examples (***.html**) that we provide, for your reference.

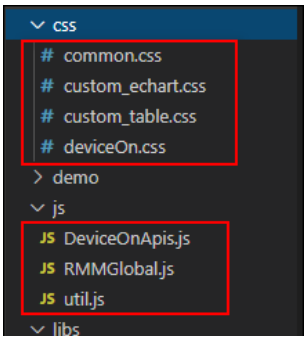


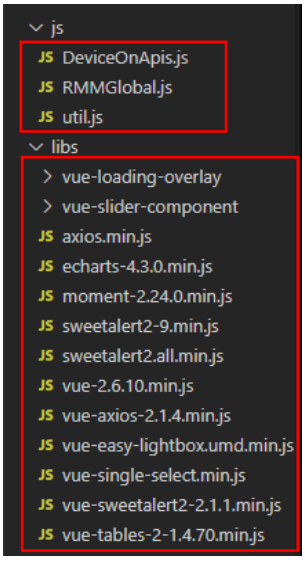
Step 2: Open Visual Studio Code -> Open the path:

\DeviceOn Server\server\portal\static\addins\SampleAddin\



Step 3: Here are serval resources for you to develop your function.





- **CSS folder** that include *.css style to describes how HTML elements are to be displayed on screen, paper, or in other media.
- **js folder** provides DeviceOnApis.js which is the API for get or set Data from Database on the server and RMMGlobal.js which is the function to get or set the data from the local storage of Website.
- **libs** folder provides simple library, if you need another library, please download from CDN.js and place in this folder.

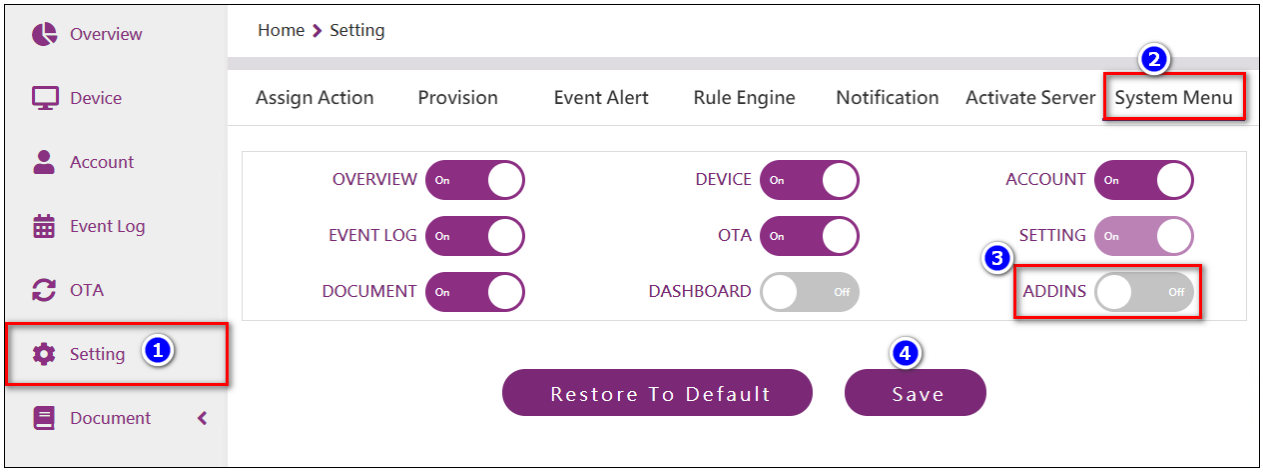
Step 4: Download sample code, there are two files (demo.html, demo2.html), please place **demo.html** into "**SampleAddin**" folder.

Line 18 to 30 (demo.html) to include java script library, you could place your library in the relative path, or alternatively, given library URL from CDNjs.

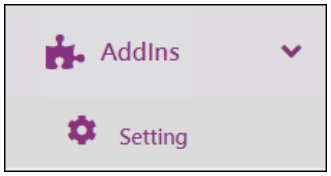
```
17      <!-- javascript plugins -->
18      <script src="/static/addins/SampleAddin/libs/vue-2.6.10.min.js"></script>
19      <script src="/static/addins/SampleAddin/libs/vue-tables-2-1.4.70.min.js"></script>
20      <script src="/static/addins/SampleAddin/libs/axios.min.js"></script>
21      <script src="/static/addins/SampleAddin/libs/sweetalert2.all.min.js"></script>
22      <script src="/static/addins/SampleAddin/libs/vue-sweetalert2-2.1.1.min.js"></script>
23      <script src="/static/addins/SampleAddin/libs/echarts-4.3.0.min.js"></script>
24      <script src="/static/addins/SampleAddin/libs/moment-2.24.0.min.js"></script>
25      <script src="/static/addins/SampleAddin/libs/vue-single-select.min.js"></script>
26
27      <!-- javascript common plugins -->
28      <script src="/static/addins/SampleAddin/js/RMMGlobal.js"></script>
29      <script src="/static/addins/SampleAddin/js/DeviceOnApis.js"></script>
30      <script src="/static/addins/SampleAddin/js/util.js"></script>
31
```

```
1. <!-- CDNjs-->
2. <script src="https://code.jquery.com/jquery.js"></script>
3. <script src="https://cdnjs.cloudflare.com/ajax/libs/twitter-
   bootstrap/3.3.7/js/bootstrap.min.js"></script>
```

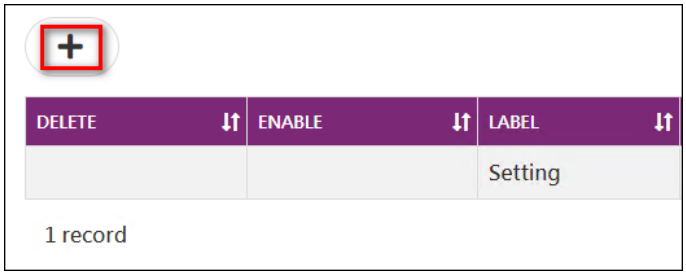
Step 5: Enable "**AddIN**" option from DeviceOn Server. (**Setting** -> **System Menu** -> **ADDINS**)

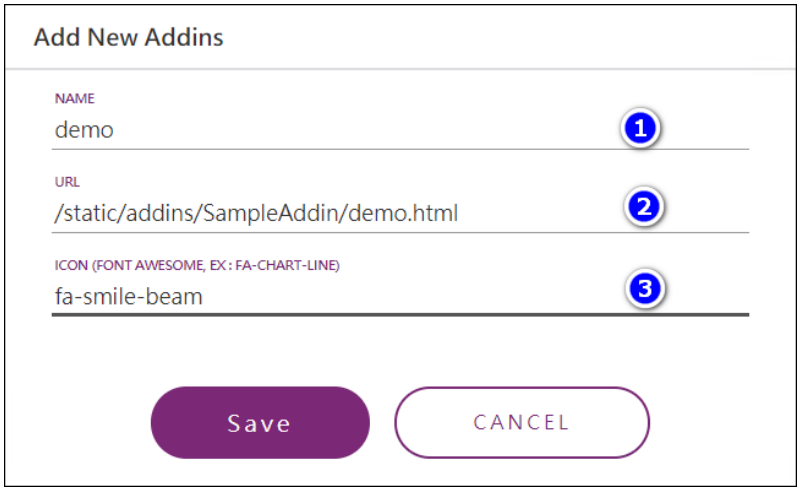


After the option is enabled, the "**Addins**" will appear in the menu item.



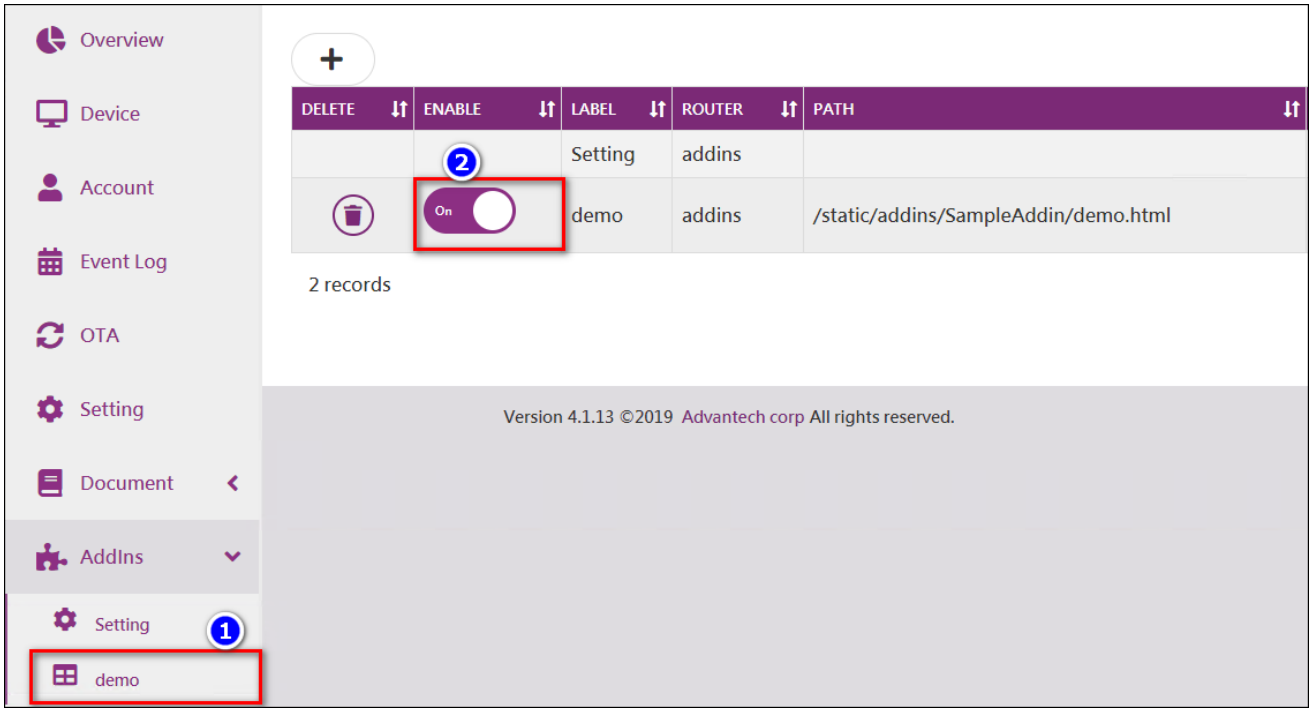
Step 6: Click on the "**Setting**" (**Addins** -> **Setting**) to add your Addins.



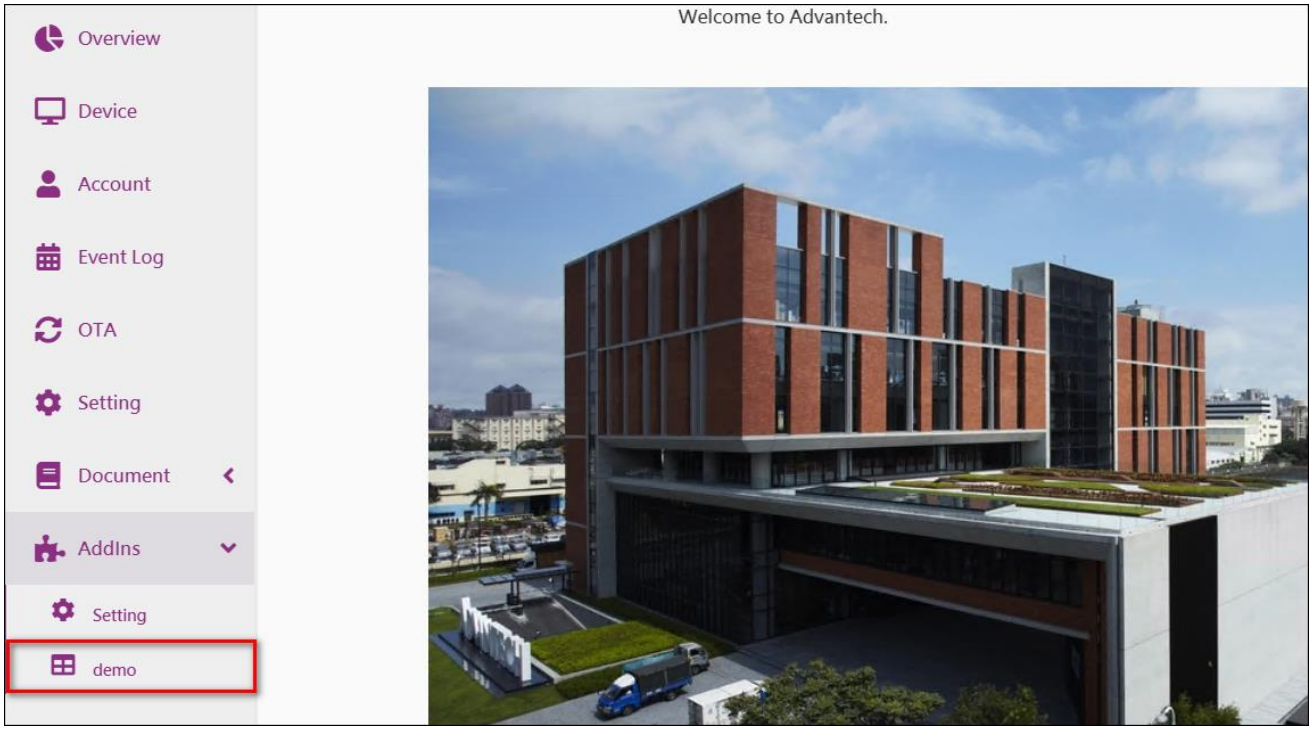


- **Name**: Label name on the menu item
- **URL**: Relative path, **/static/addins/SampleAddin/demo.html**
- **Icon**: Reference Fontawesome site to get the string of icon

After that, the "**demo**" shown on the menu item, if not, please enable the "**Addin**" on Setting page.



Step 7: Click on the "**demo**" addins.



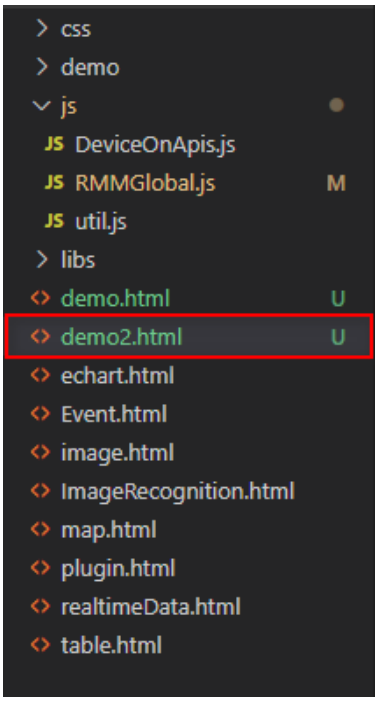
5.2.4 Develop an Add-in to Access DeviceOn API

This example will show you how to get all accounts, groups and devices.

APIs used on below sample

1. DeviceOnApis.accounts.get.accounts(aid)
   To get all accounts information from database.
2. DeviceOnApis.accounts.get.deviceGroups(aid)
   To get all groups which under this aid's account from database.
3. DeviceOnApis.devicegroups.get.devicesAll(data)
   To get all devices which under this aid's account from database.
4. DeviceOnApis.devicegroups.get.devices(gid, data)
   To get all devices which under this gid's group from database.

Step 1: Download sample code, there are two files (demo.html, demo2.html), please place **demo2.html** into "**SampleAddin**" folder.



Step 2: Line 10 ~22 (demo2.html) that describe library used in the Add-in.

```
8
9     <!-- css plugins -->
10    <link rel="stylesheet" href="/static/addins/SampleAddin/css/deviceOn.css">
11    <link rel="stylesheet" href="/static/addins/SampleAddin/css/common.css">
12    <link rel="stylesheet" href="/static/addins/SampleAddin/css/custom_echart.css">
13
14    <!-- javascript plugins -->
15    <script src="/static/addins/SampleAddin/libs/vue-2.6.10.min.js"></script>
16    <script src="/static/addins/SampleAddin/libs/axios.min.js"></script>
17    <script src="/static/addins/SampleAddin/libs/vue-single-select.min.js"></script>
18
19    <!-- javascript common plugins -->
20    <script src="/static/addins/SampleAddin/js/RMMGlobal.js"></script>
21    <script src="/static/addins/SampleAddin/js/DeviceOnApis.js"></script>
22    <script src="/static/addins/SampleAddin/js/util.js"></script>
23
```

Use single-select component to build demo view. (Line 27 ~ 57)

```
26 <body style="background: #FAFAFA;">
27     <div id="app">
28         <div class="content">
29             <div class="row">
30                 <div class="col-md-4">
31                     <div class="cus-label">Account: </div>
32                     <vue-single-select v-model="selectedAccount" :options="accountOptions" option-label="name">
33                         <template slot="option" slot-scope="{option, index}">
34                             <div>
35                                 <span style="margin-left: 1rem;">{{option.name}}</span>
36                             </div>
37                         </template>
38                     </vue-single-select>
39                 </div>
40                 <div class="col-md-4">
41                     <div class="cus-label">Device Group: </div>
42                     <vue-single-select v-model="selectedGroup" :options="groupOptions" option-label="name"></vue-single-select>
43                 </div>
44                 <div class="col-md-4">
45                     <div class="cus-label">Device: </div>
46                     <vue-single-select v-model="selectedDevice" :options="deviceOptions" option-label="name">
47                         <template slot="option" slot-scope="{option, index}">
48                             <div>
49                                 <i :class="option.iconClass" :style="{'color': option.iconColor}" aria-hidden="true"></i>
50                                 <span style="margin-left: 1rem;">{{option.name}}</span>
51                             </div>
52                         </template>
53                     </vue-single-select>
54                 </div>
55             </div>
56         </div>
57     </div>
58 </body>
```

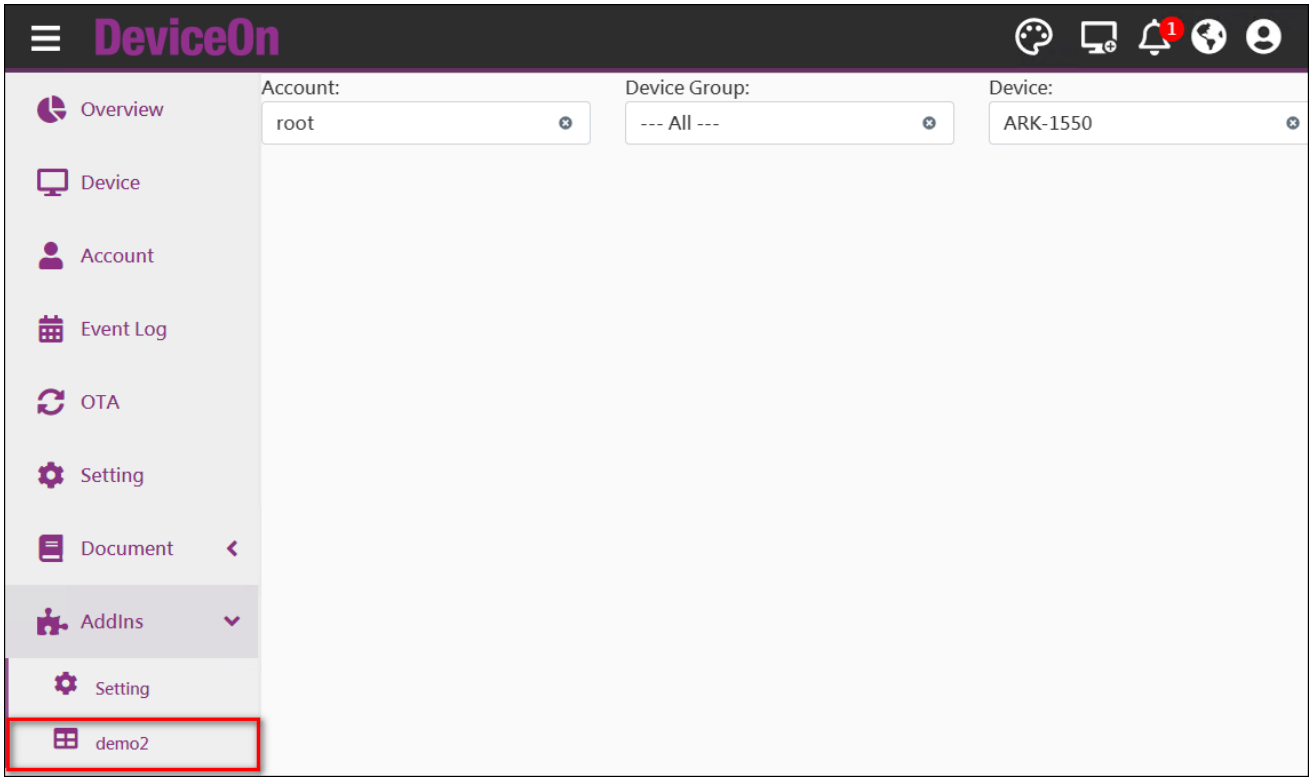
Use RMMGlobal() to get your login account ID (aid), through the aid as parameter to request API.

```
70         mounted: function () {
71             //get current user aid
72             var aid = RMMGlobal.get().Login.aid;
73             this.getAccounts(aid);
74         },
```

The API (**DeiceOnApis.accouts.get.accouts(aid)**) will send request to server, and return all account data.

```
83         methods: {
84             getAccounts: function (aid) {
85                 DeviceOnApis.accounts.get.accounts(aid)
86                     .then(function (xhr) {
87                         if (xhr && xhr.data && xhr.data.accounts) {
88                             vue.accountOptions = xhr.data.accounts;
89                             let aAccount = vue.accountOptions.filter(function (g, i) {
90                                 return g.aid === Number(aid);
91                             });
92                             if (aAccount.length === 0 && vue.accountOptions.length > 0) {
93                                 vue.selectedAccount = vue.accountOptions[0];
94                             } else {
95                                 vue.selectedAccount = aAccount[0];
96                             }
97                         }
98                     });
99             },
```

Step 3: Add an Addin (demo2) as before steps.



6. FAQ

6.1 Why Some of Devices Cannot Power On

REF: https://www.lifewire.com/wake-on-lan-4149800/

The DeviceOn leverage Wake-on-LAN (WoL) mechanism to remote power your device on, there are 2 steps to should be configured at first. Wake-on-LAN (WoL) is a network standard that allows a computer to be turned on remotely, whether it's hibernating, sleeping, or even completely powered off. It works by receiving what's called a "magic packet" that's sent from a WoL client.
It also doesn't matter what operating system the computer will eventually boot into (Windows, Mac, Ubuntu, etc.), Wake-on-LAN can be used to turn on any computer that receives the magic packet. A computer's hardware does have to support Wake-on-LAN with a compatible BIOS and network interface card, so not every computer is automatically able to use Wake-on-LAN.

Two-step WoL Setup

Enabling Wake-on-LAN is done in two steps, both of which are described below. The first sets up the motherboard by configuring Wake-on-LAN through BIOS before the operating system boots, and the next logs into the operating system and makes some small changes there. The first step with the BIOS is valid for every computer, but after following the BIOS setup, skip down to your operating

system instructions, whether it be for Windows, Mac, or Linux.

Step 1: BIOS Setup

The first thing you need to do to enable WoL is to set up BIOS correctly so that the software can listen for incoming wake up requests.

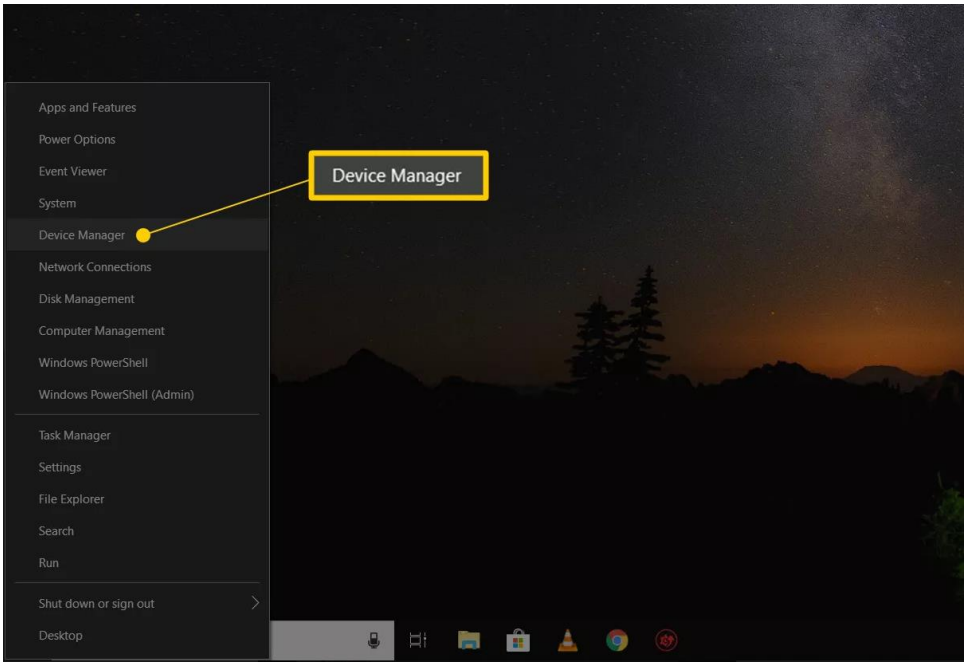
Every manufacturer will have unique steps, so what you see below may not describe your setup exactly. If these instructions aren't helping, find out your BIOS manufacturer and check their website for a user manual on how to get into BIOS and find the WoL feature.

1. Enter BIOS instead of booting to your operating system.
2. Look for a section that pertains to power, such as Power Management. This may be under an Advanced section. Other manufacturers might call it Resume On LAN, such as on the Mac. Most BIOS screens have a help section off to the side that describes what each setting does when enabled. It's possible that the name of the WoL option in your computer's BIOS isn't clear.
3. Once you find the WoL setting, you can most likely press **Enter** to either immediately toggle it on or to show a small menu that allows you to toggle it on and off, or enable it and disable it.
4. Save the changes. This isn't the same on every computer, but on many the **F10** key will save and exit BIOS. The bottom of the BIOS screen should give some instructions about saving and exiting.

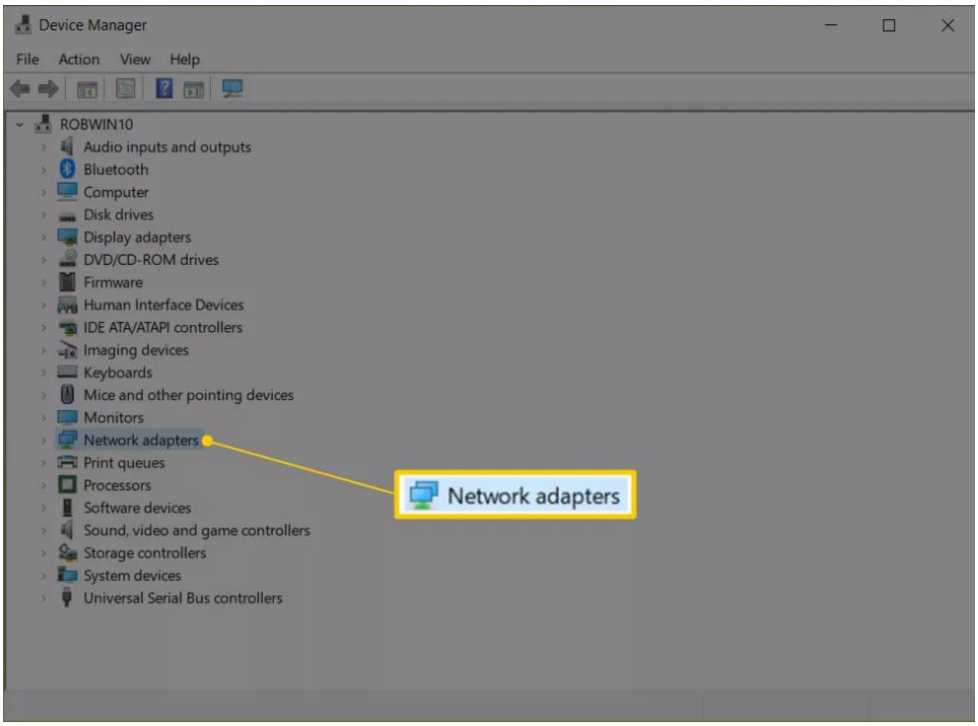
Step 2: Windows operating system WoL setup

Windows Wake-on-LAN is set up through Device Manager. There are a few different settings to enable here:

1. Open Device Manager

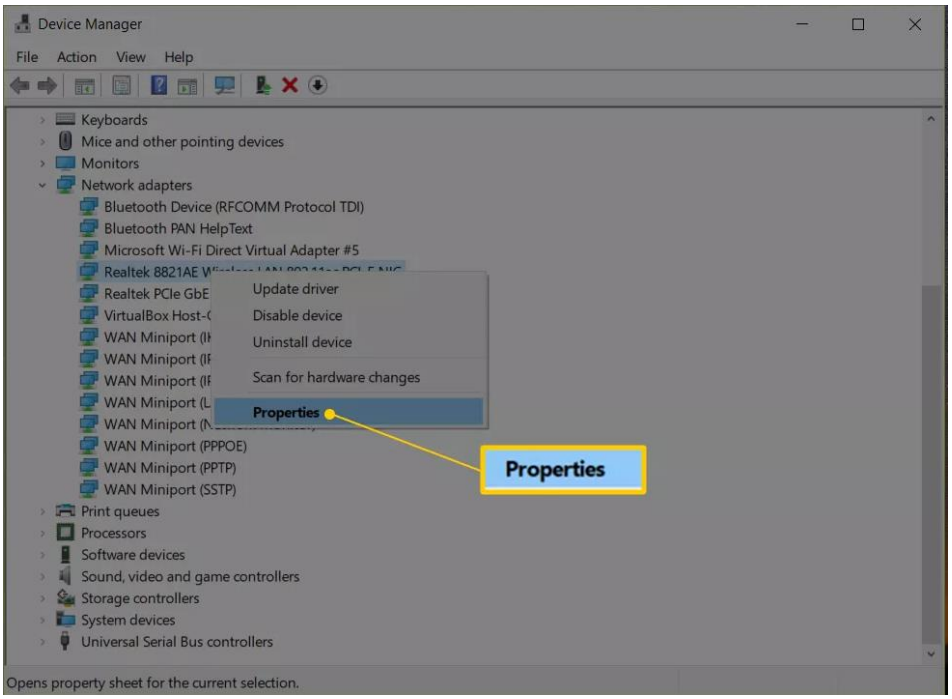


2. Find and open the Network adapters section.

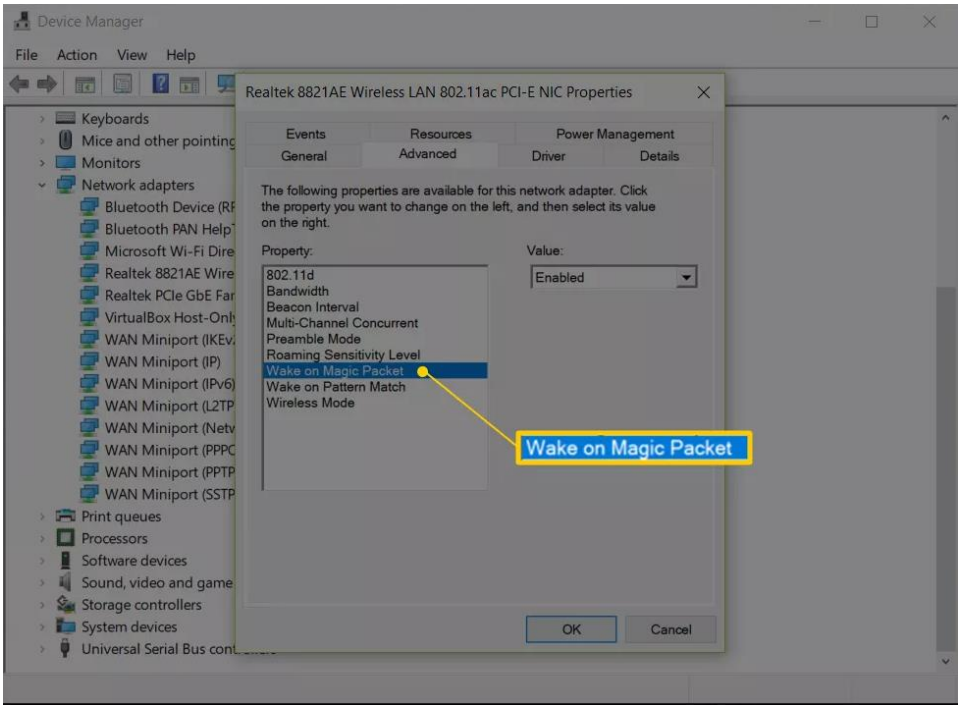


You can ignore any Bluetooth connections and virtual adapters. Double-click (or double-tap) **Network adapters** or select the small **+** or **>** button next to it to expand that section.

3. Right-click or tap-and-hold the adapter that belongs to the active internet connection. Examples of what you might see are **Realtek PCIe GBE Family Controller** or **Intel Network Connection**, but it will vary depending on your computer.
4. Choose **Properties**.

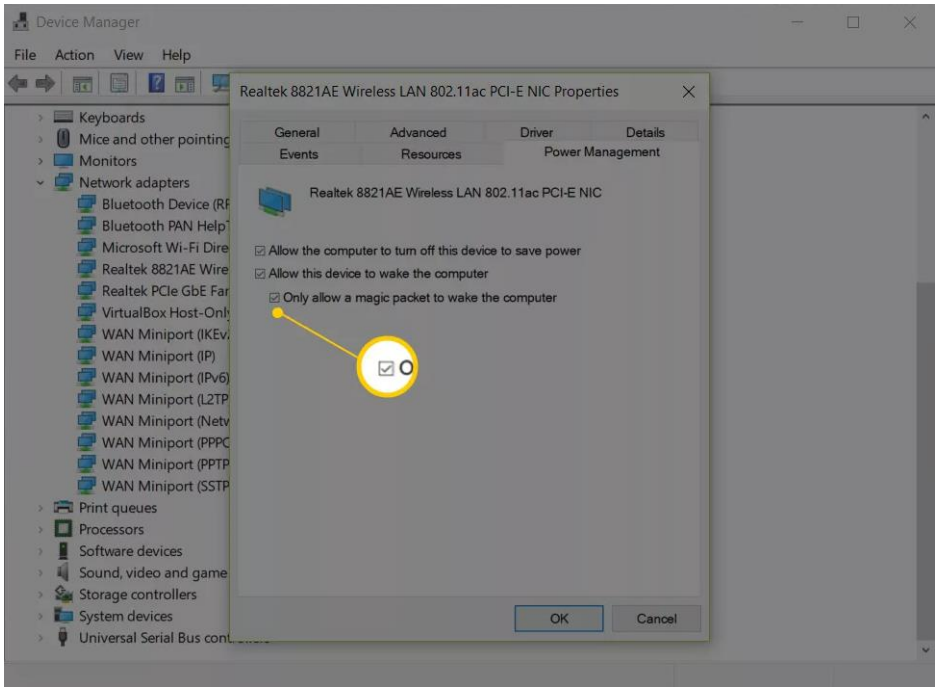


5. Open the **Advanced** tab.
6. Under the **Property** section, click or tap **Wake on Magic Packet**. If you can't find this, skip to Step 8; Wake-on-LAN might still work anyway.



7. From the **Value** menu on the right, choose **Enabled**.
8. Open the **Power Management** tab. It might be called **Power** depending on your version of Windows or network card.
9. Make sure these two options are enabled: **Allow this device to wake the computer** and **Only**

allow a magic packet to wake the computer.

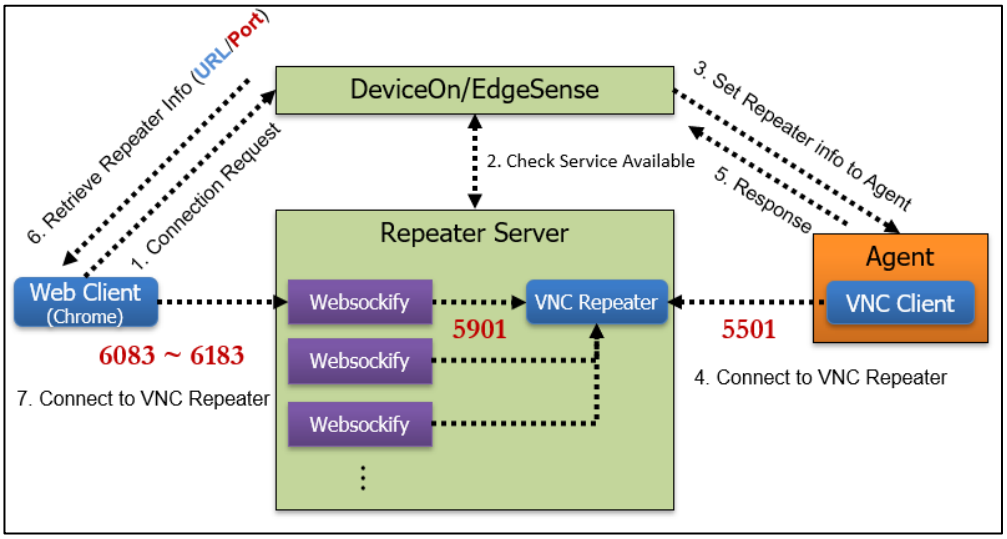


These settings might instead be under a section called Wake-on-LAN and be a single setting called **Wake on Magic Packet**.

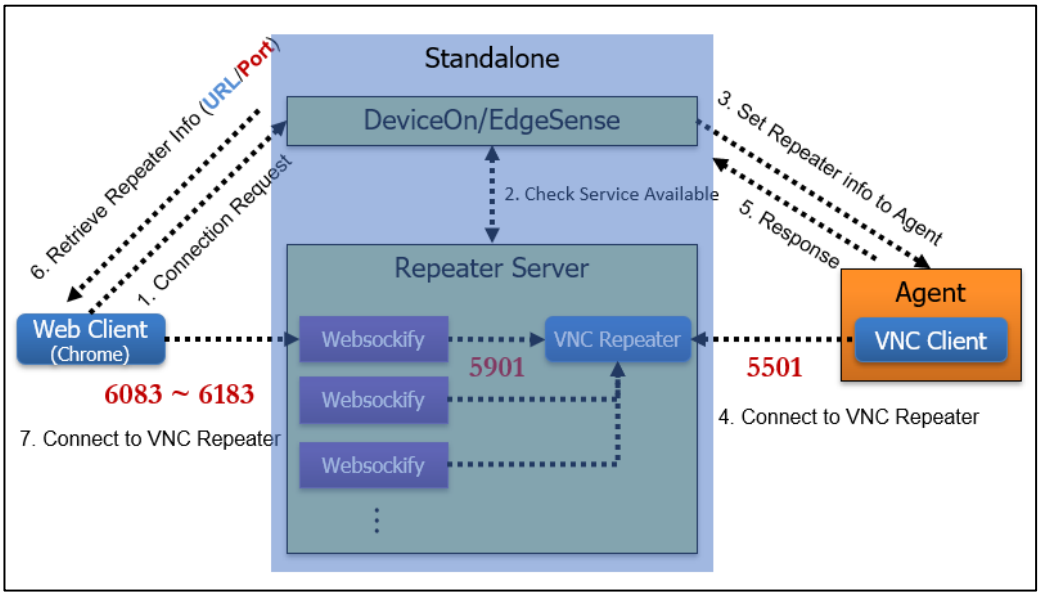
10. **Click or tap OK to save the changes and exit that window. You can also close** down Device Manager.

6.2 Why Cannot Remote Control via KVM (Remote Desktop)

The DevicOn leverage VNC (Virtual Network Computing) technology to achieve remote desktop, to bridge different network between public and private. We build-up a Repeater server on public site for WISE-PaaS/EnSaaS and Azure PaaS. There is a web-client through WebSocket (port: 6083 ~6183) mechanism connect to Repeater and device via 5501 to Repeater, the structure as below. Please help confirm the port available on your browser and device side.

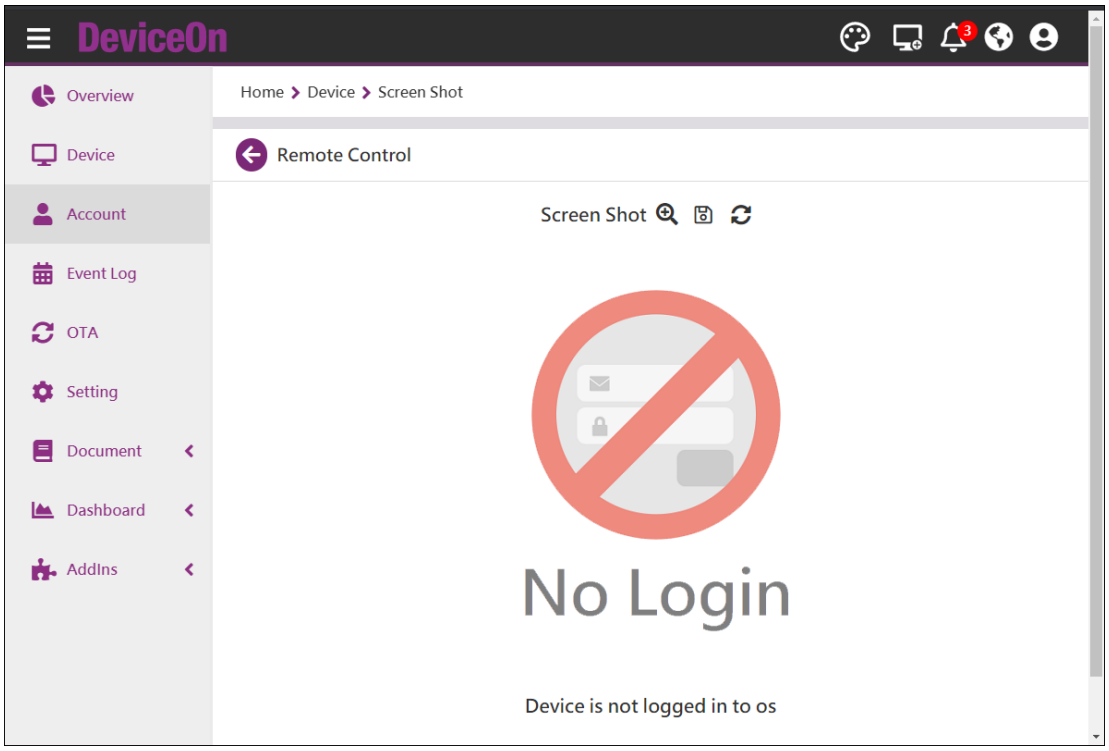


If the DeviceOn running on VM, standalone version, the Repeater also build into same machine, please reference the structure, make sure the VM available for these inbound and outbound ports.

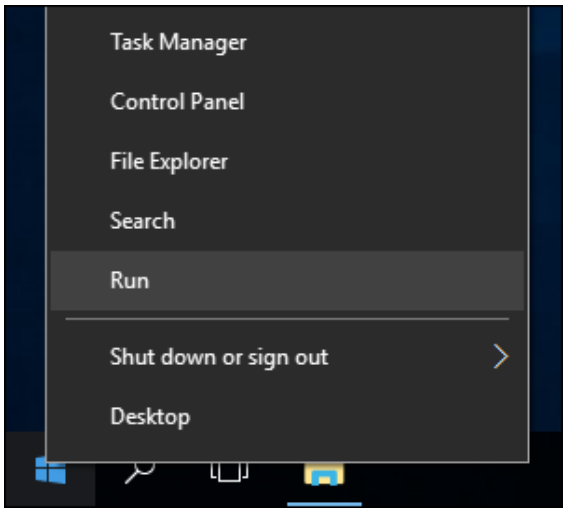


6.3 Why Cannot Screenshot and Always Show Device “No Login”

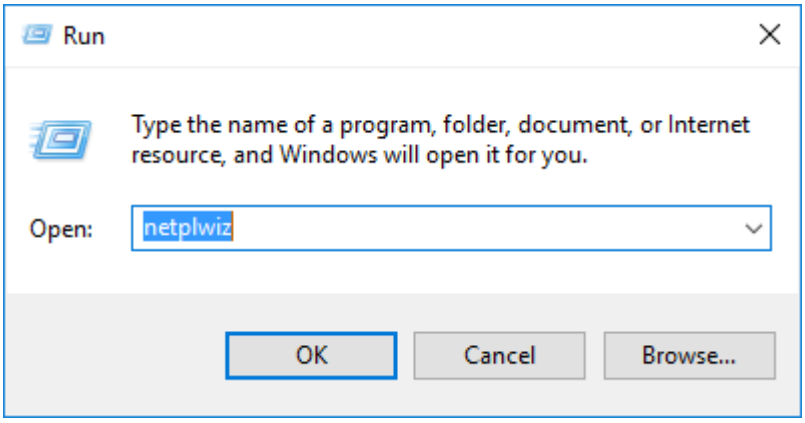
To fix the “No Login” error, you can sign into the system manually, or set the “Automatically Sign in to Windows 10”.



Step 1: Right-click the Start button and select Run from the hidden quick access menu, or use the keyboard shortcut Windows Key + R to bring up the Run dialog.

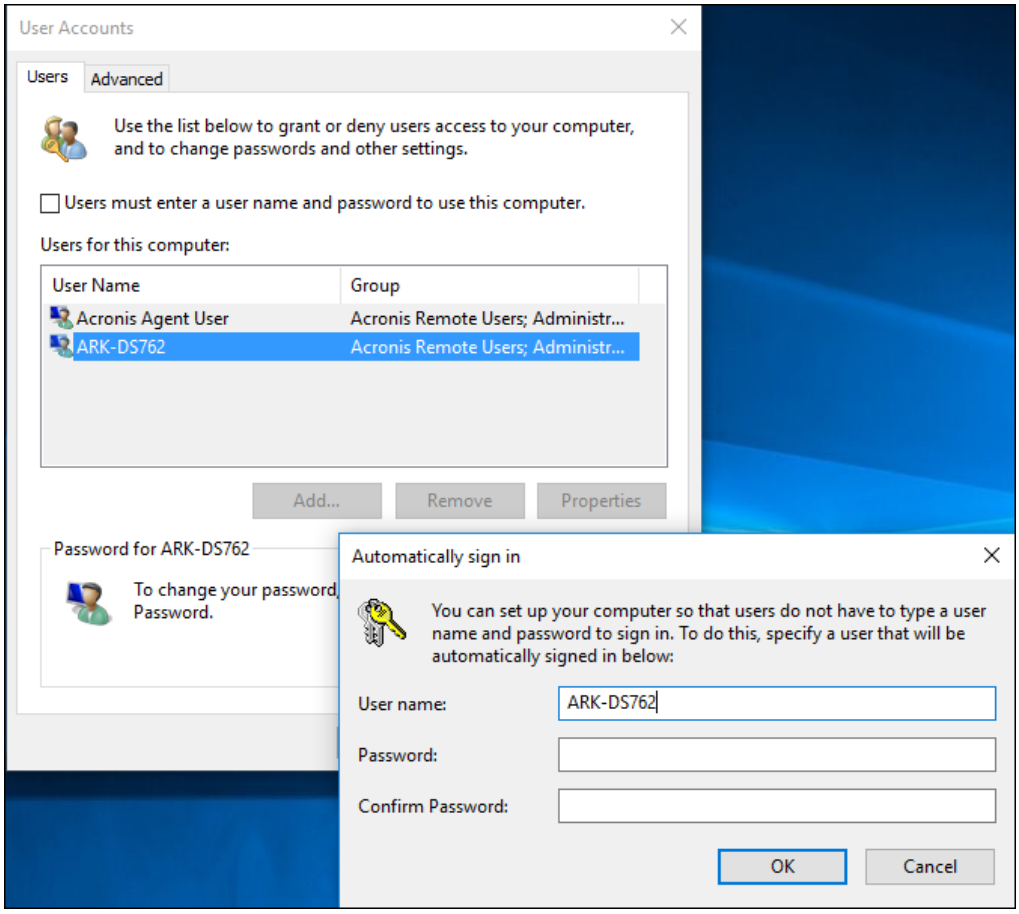


Step 2: Now Then Type: ***netplwiz*** and hit Enter or click OK.



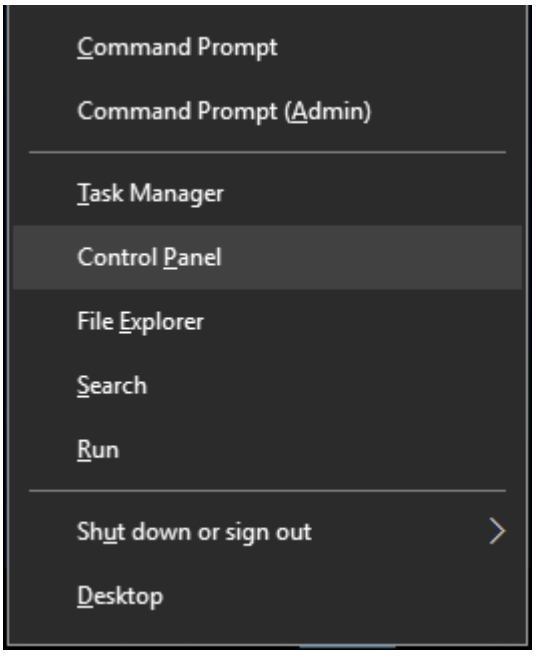
Step 3: Uncheck Users must enter a user name and password to use this computer and click OK.

Step 4: Enter in your user name and the password you use to log into your system twice and click OK.

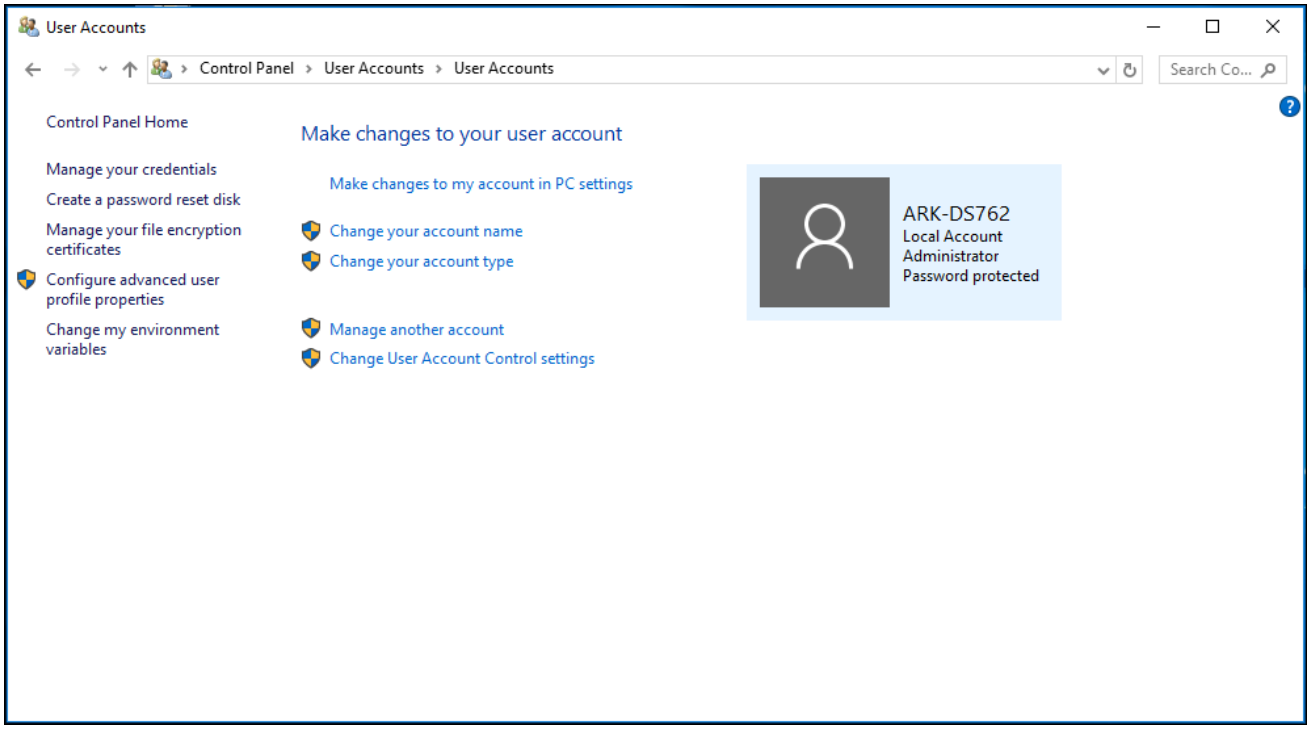


If you still get the “No Login” error or get the “black screen”, then you can try to disable the Windows User Account Control (UAC).

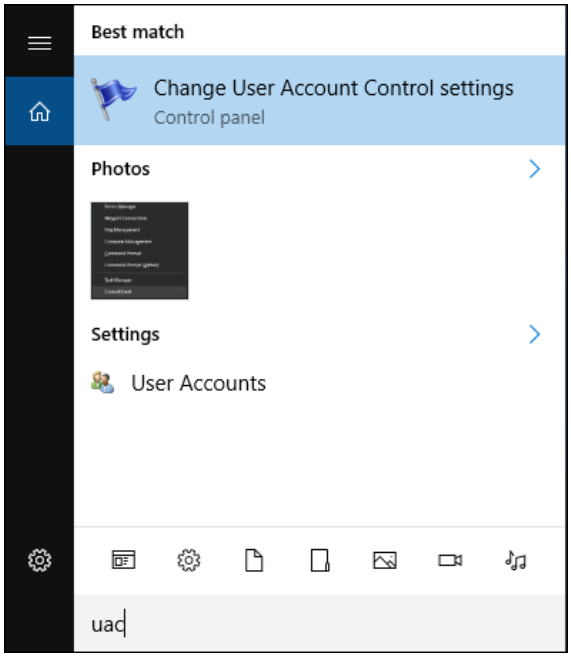
Step 1: Press Windows Key + X hotkeys together on the keyboard and choose the "Control Panel" item.



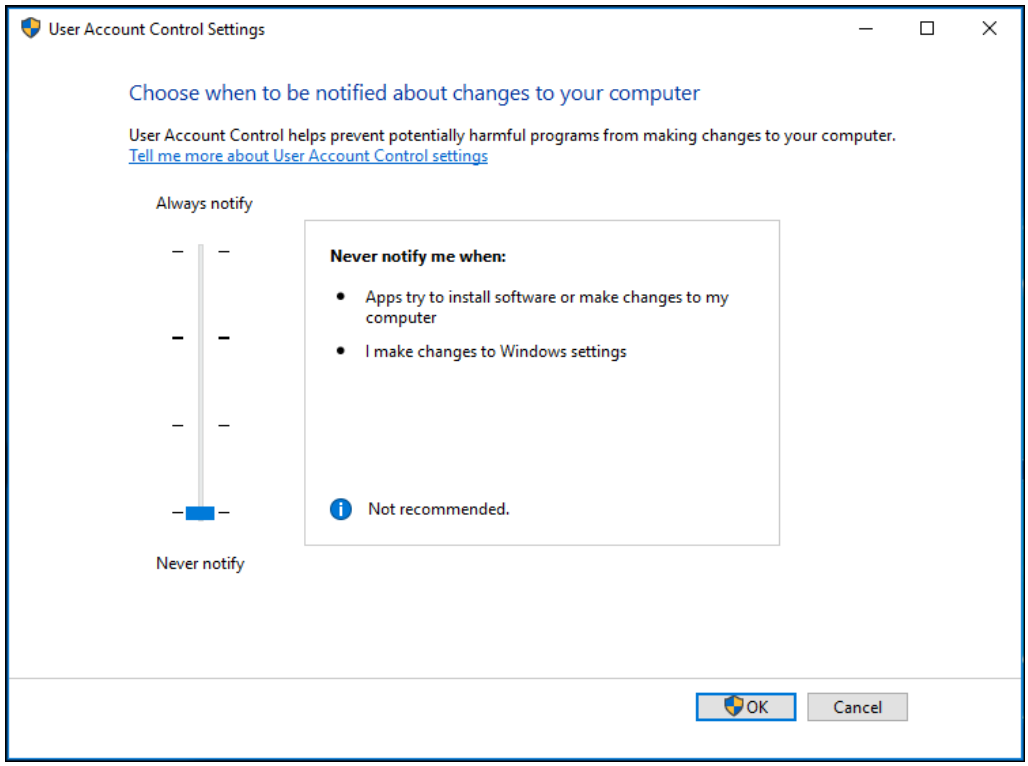
Step 2: Go to the following path: "**Control Panel\User Accounts\User Accounts**" There you will find the Change User Account Control settings link. Click it.



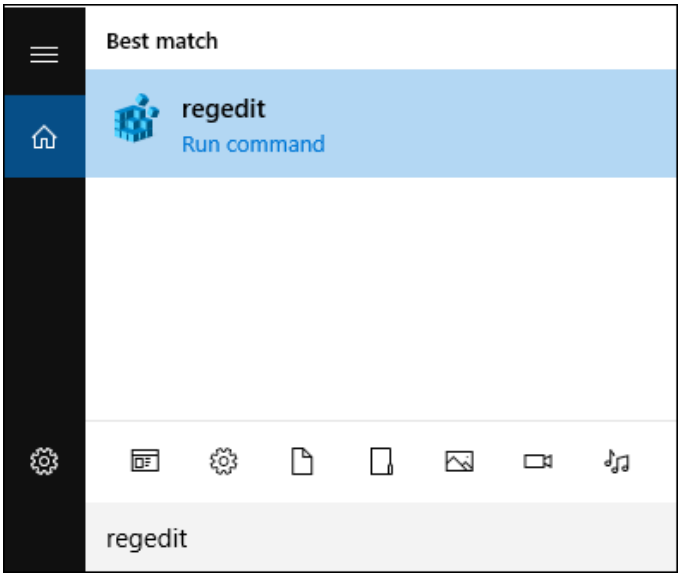
Alternatively, you can enter the "***UAC***" in the Search box to open the User Account Control settings dialog.



Step 3: In the User Account Control settings dialog, move the slider to the bottom (Never Notify).

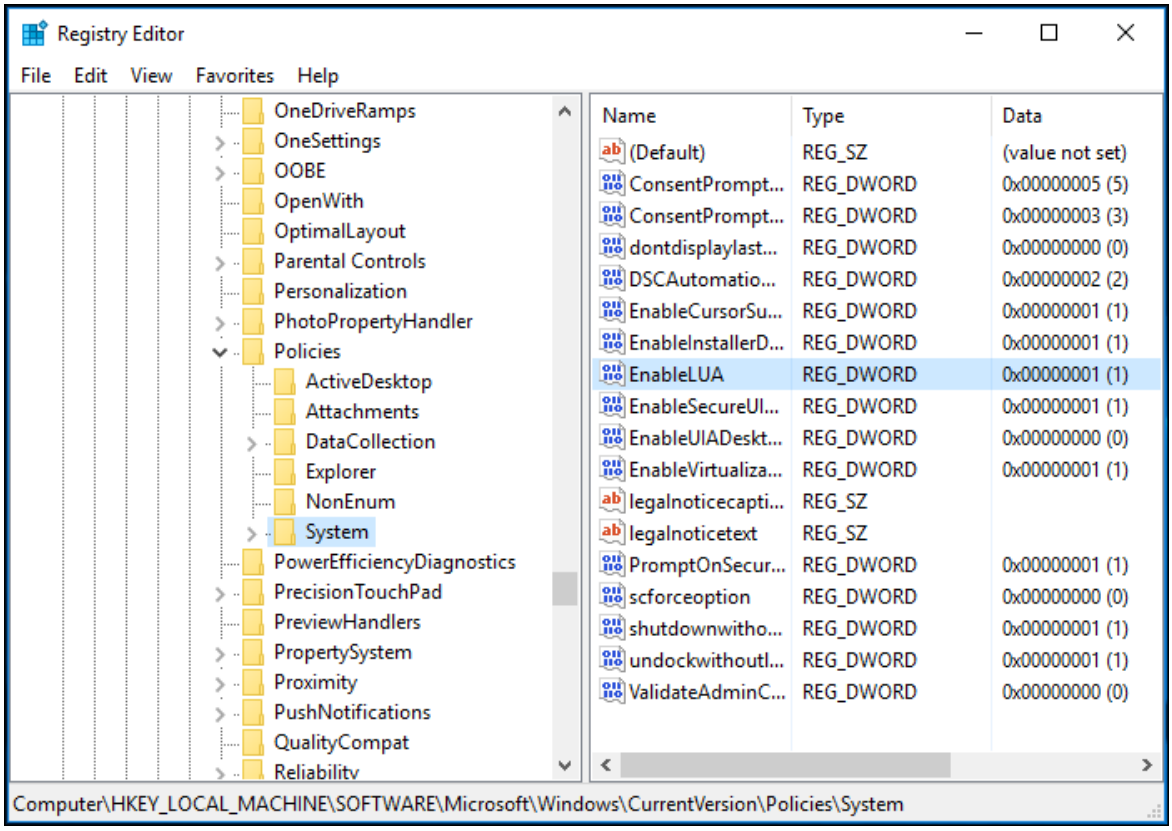


Step 4: Enter the "***regedit***" in the Search box to open the Registry Editor.

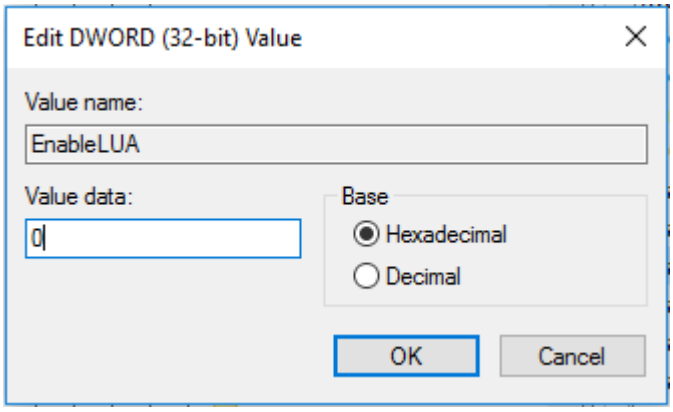


Step 5: Navigate to the following key:

"*HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System*"

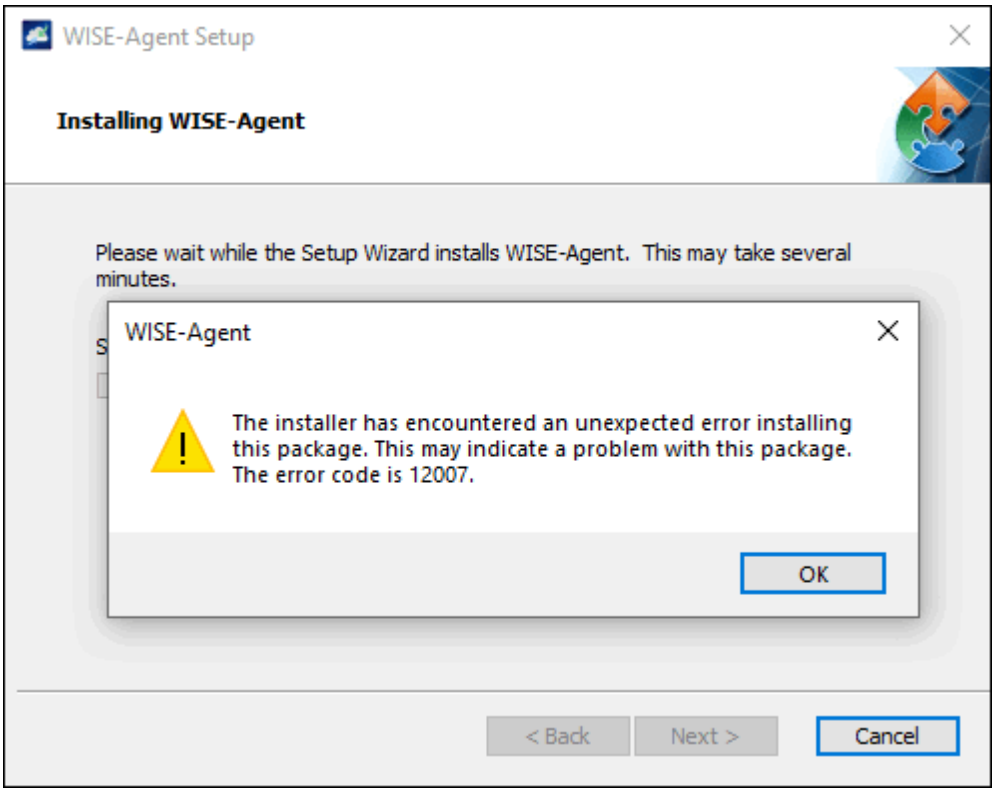


Step 6: In the right pane, modify the value of the **EnableLUA** DWORD value and set it to **0**.



Step 7: Restart your computer.

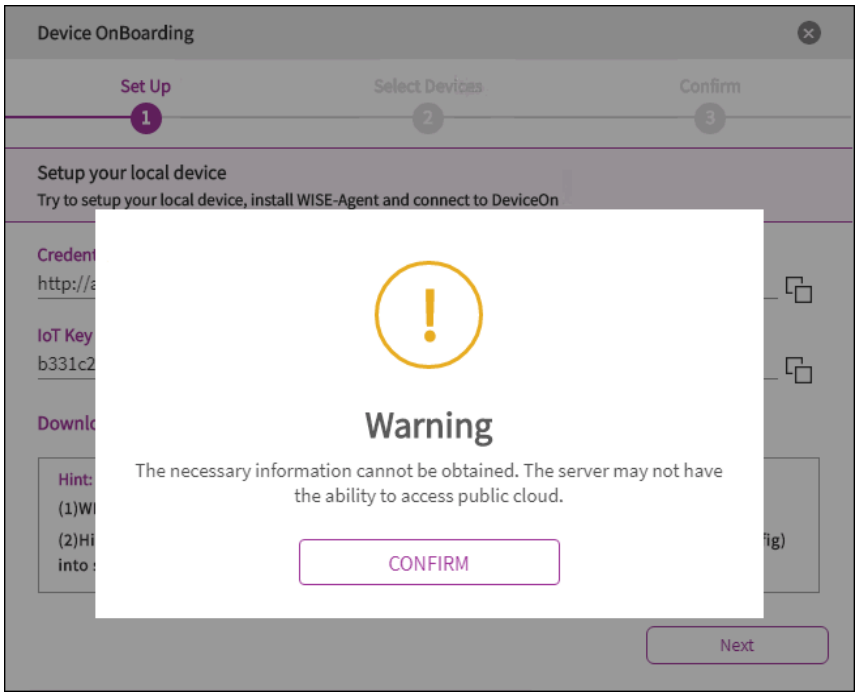
6.4 Why the WISE-Agent Cannot Install? With error code 12007?



WISE-Agent requires the Microsoft Visual C++ Redistributable 2008, 2013, 2015 x86 packages, which will be downloaded from the Internet and set up during the installation process. If you are in an environment with limited or no Internet access, please download the Agent Dependency Package through an Internet connected device and install this package first.

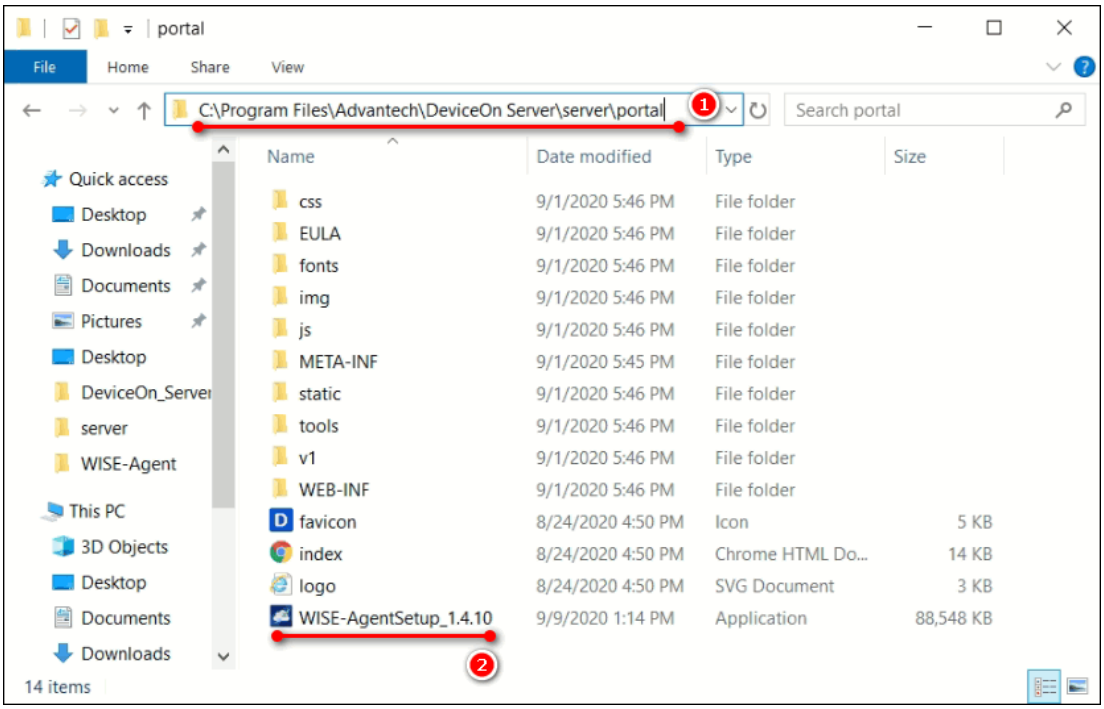
6.5 Why the WISE-Agent Cannot Download from Device Onboarding?

In order to allow users to obtain the latest and stable WISE-Agent, the DeviceOn team will place the latest version on the cloud. When this message appears, it means that your server network cannot access the cloud or does not have network connectivity.



You could download WISE-Agent through your mobile device or laptop and put it in the following path. The file name must be "**WISE-AgentSetup_x.y.z**".

Installation Path\DeviceOn Server\server\portal\



6.6 Why the Acronis and McAfee failed to install?

Since the installer package require .Net Framework 3.5 dependency, please help to confirm is .Net Framework 3.5 installed on your devices.

6.7 Why Your SMTP Server Cannot Send a Mail?

Case I: Your DeviceOn service is deployed on Azure cloud and your SMTP server adopt port 25.
Starting on November 15, 2017, outbound email messages that are sent directly to external domains (such as outlook.com and gmail.com) from a virtual machine (VM) are made available only to certain subscription types in Microsoft Azure. Outbound SMTP connections that use TCP port 25 were blocked. (Port 25 is primarily used for unauthenticated email delivery.)

This change in behavior applies only to new subscriptions and new deployments since November 15, 2017. Referenced site>

Case II: Alway authentication failed through your Gmail account.
Step 1: Less secure apps & your Google Account.
Please enter to the page with your Google account and set it to **Enable**.
Step 2: Unlocking Google's Gmail CAPTCHA
Please enter to the page with your Google account and click **Continue**.

6.8 How the Device Data Flow and Debug from Edge to Cloud

The **WISE-DeviceOn** offers a general solution of gathering device, equipment and sensor information from the edge device via WISE-Agent. This document will walk you through how the data will be transmitted to **WISE-PaaS/EnSaaS** and the data flow over our architecture. Besides, it also covers how to clarify the issues while using **WISE-DeviceOn**.

For those advanced system integrators, below figure illustrates an overview of data flow among our software components. You also can access several documents (links of these docs…) that will help you to dive deeper of each item. As an example of document item 1, how to retrieve credential for device to connect to **WISE-PaaS/EnSaaS** IoTHub. We implemented a virtual host mechanism in IoThub to process messages independently for different space in case data confusion of messages in same topic. Each key can only be used for accessing data for one specific space as below figure. Key No. 1 can neither publish nor subscribe data for space B, but only key No. 4 is available to access IoThub for space B.

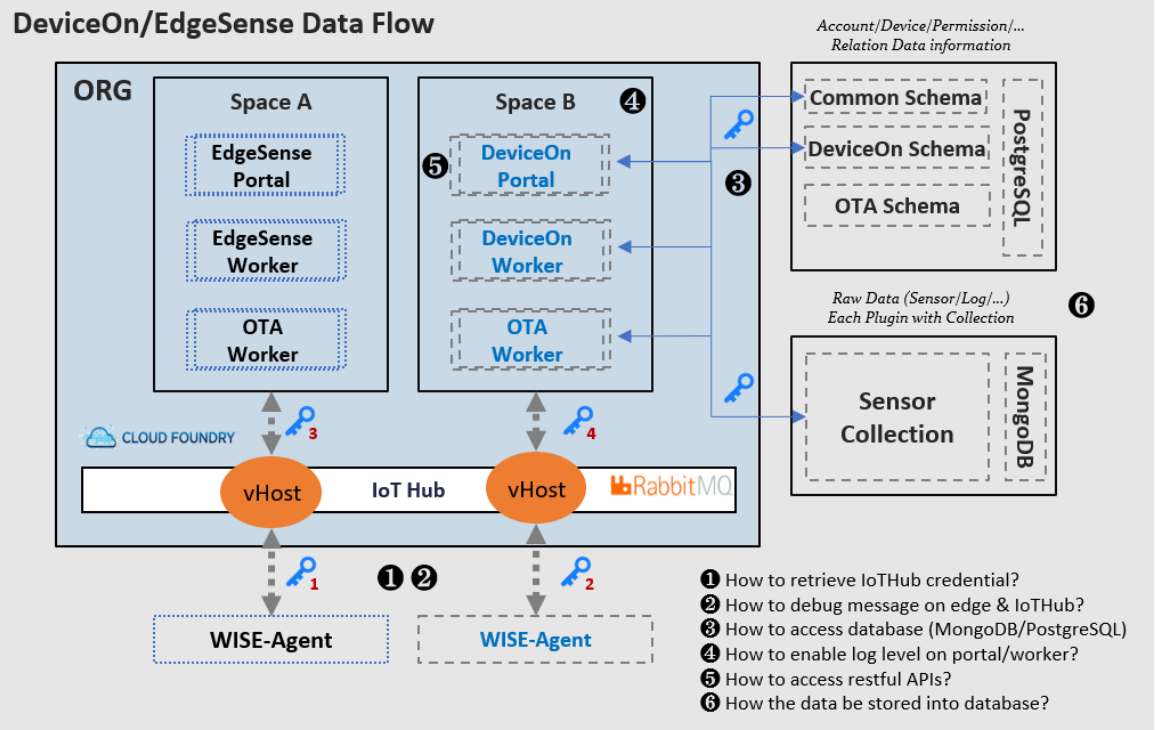
Furthermore, one 3rd party tool **node-RED** can help you to check if all the data are transmitted to IoTHub properly. It's an open source project has been uploaded on the GitLab, as document item2 will walk you through how to publish node-RED application on our **WISE-PaaS/EnSaaS** and how to monitor messages on edge and IoThub.

Once the data has been published to target space of **WISE-PaaS/EnSaaS**, the worker will process it and store it into corresponding databases. There are two databases we adopted; one is relational

database **PostgreSQL** for storing relational data. For instance, MAC address, device name, platform name, OS information…etc. The other is NoSQL database **MongoDB** for storing sensor raw data. Hence, document as item 3 will walk you through how to access these databases to make sure all the data are stored in databases properly.

In **WISE-DevieOn**, we implemented a log system to record each operation. As document item 4 will walk you through how to enable log message for advanced error tracking. Besides, as document item 5, it will also walk you through how to accessing data from database directly by using RESTful APIs for data visualization or application development.

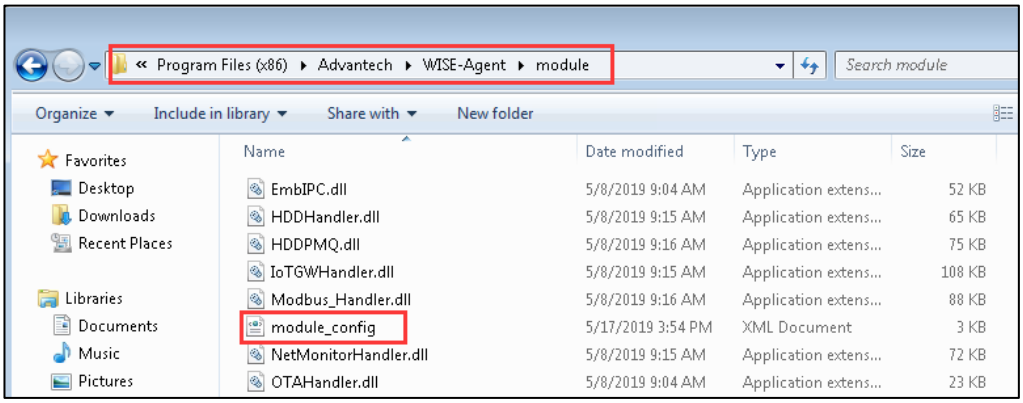
Finally, document **item 6** gives an example to show you what kind of data will be stored in **PostgreSQL** and **MongoDB** respectively once edge devices are connected.



6.9 How to Enable/Disable plugins on WISE-Agent

Step 1: Adjust configuration file on WISE-Agent

Open **module_config.xml** on Installation path\module\



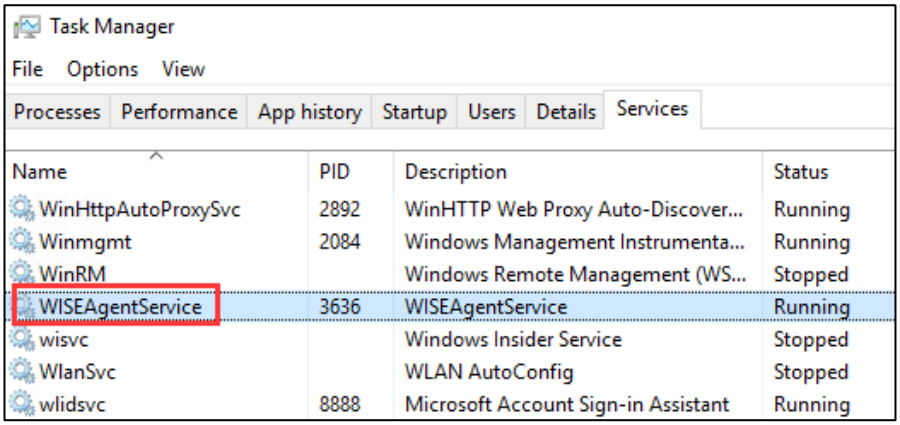
Adjust “ModuleEnable” to TRUE/FALSE to enable and disable.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<XMLConfigSettings>
  <BaseSettings>
    <ModuleNum>15</ModuleNum>
    <ModuleName1>HDDHandler</ModuleName1>
    <ModulePath1>\module\HDDHandler.dll</ModulePath1>
    <ModuleEnable1>TRUE</ModuleEnable1>
    <ModuleName2>PowerOnOffHandler</ModuleName2>
    <ModulePath2>\module\PowerOnOffHandler.dll</ModulePath2>
    <ModuleEnable2>TRUE</ModuleEnable2>
    <ModuleName3>ScreenshotHandler</ModuleName3>
    <ModulePath3>\module\ScreenshotHandler.dll</ModulePath3>
    <ModuleEnable3>TRUE</ModuleEnable3>
    <ModuleName4>NetMonitorHandler</ModuleName4>
    <ModulePath4>\module\NetMonitorHandler.dll</ModulePath4>
    <ModuleEnable4>TRUE</ModuleEnable4>
    <ModuleName5>ProcessMonitorHandler</ModuleName5>
    <ModulePath5>\module\ProcessMonitorHandler.dll</ModulePath5>
    <ModuleEnable5>TRUE</ModuleEnable5>
```

HDD Monitoring

Step 2: Restart WISE-Agent

Open “Task Manager” and switch to “Services”



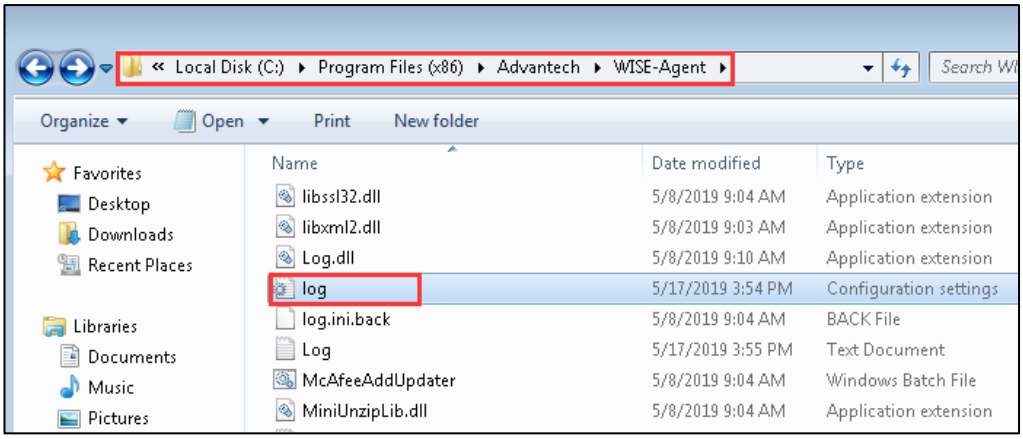
Restart “WISEAgentService” to connect to DeviceOn

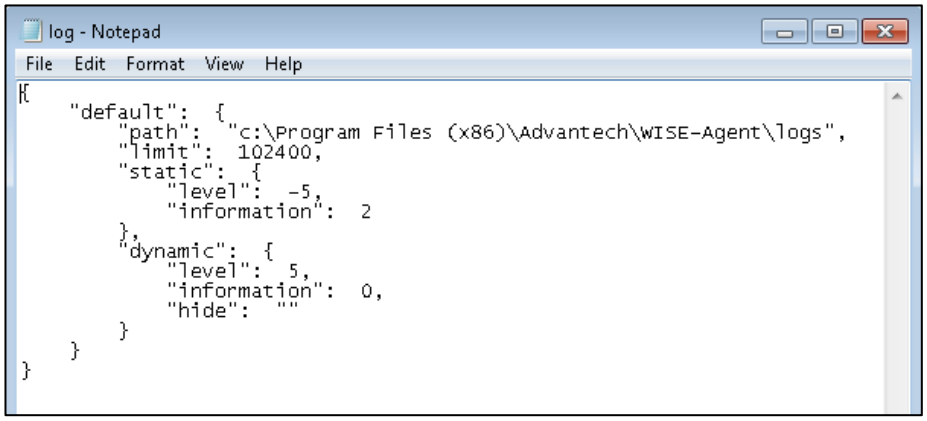
6.10 How to Enable and Adjust WISE-Agent Log Levels

[WISE-Agent v-1.3.x & v-1.2.x]

Step 1: Adjust configuration file on WISE-Agent

Open **log.ini** on Installation path

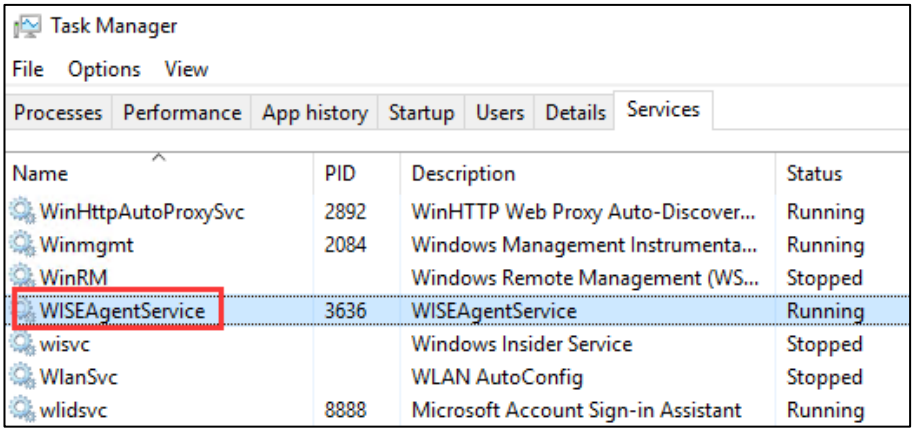




Adjust level 5 to 7, minus stand for HTML format.

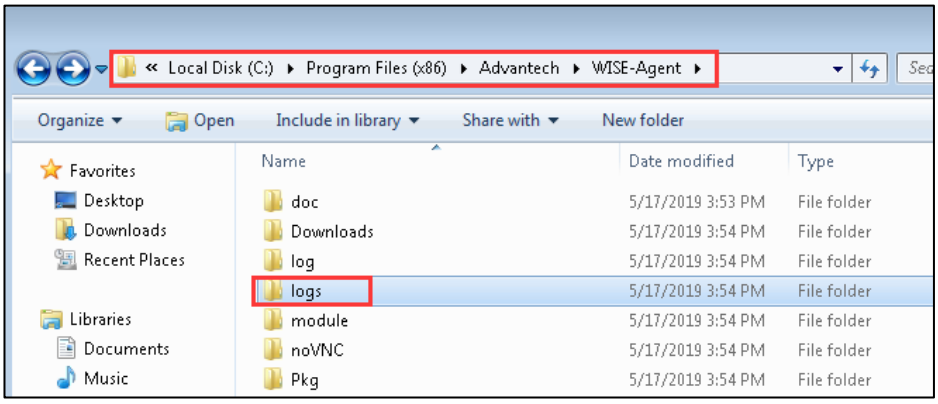
Step 2: Restart WISE-Agent

Open “Task Manager” and switch to “Services”, and restart “WISEAgentService”



Step 3: Retrieve log files from WISE-Agent

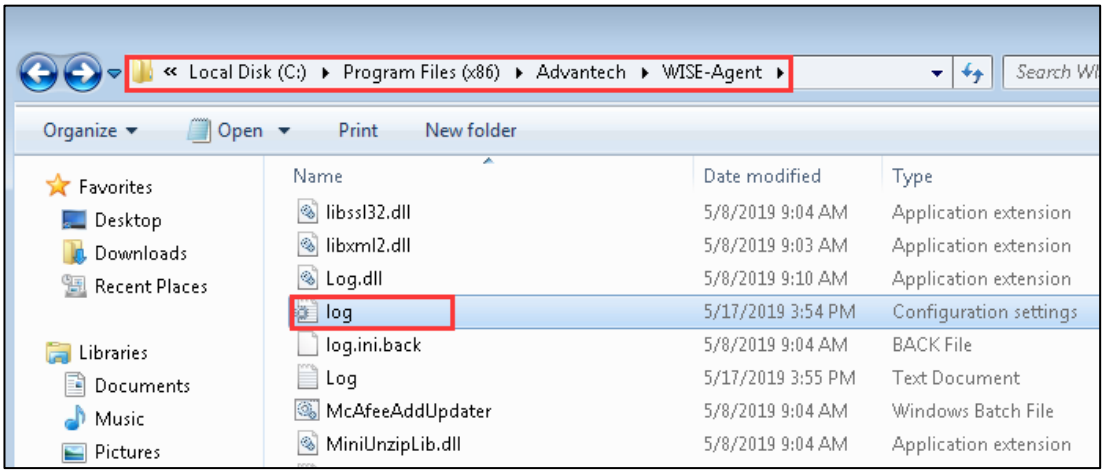
The log files under the Installation path\logs



[WISE-Agent v-1.4.x and above]

Step 1: Adjust configuration file on WISE-Agent

Open **log.ini** on Installation path\module\

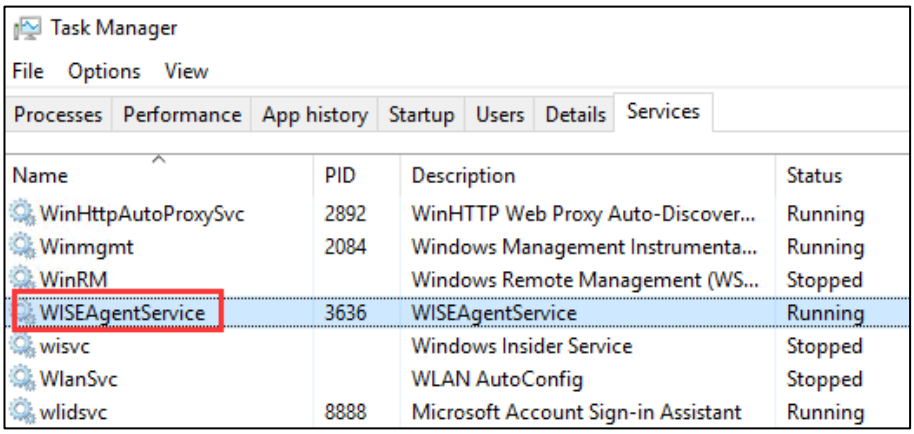



```
[LogClient]
#log_level=4, LOG_FATAL(0), LOG_ALARM(1), LOG_ERROR(2), LOG_WARNING(2), LOG_NORMAL(4), LOG_DEBUG(5
log_level=5
#to_stderr=1, 1: print to stderr, 0: doesn't print stderr
#logd_ip=127.0.0.1, ip of logd
#logd_port=9278
```

Adjust **log_level** from 4 to 5.

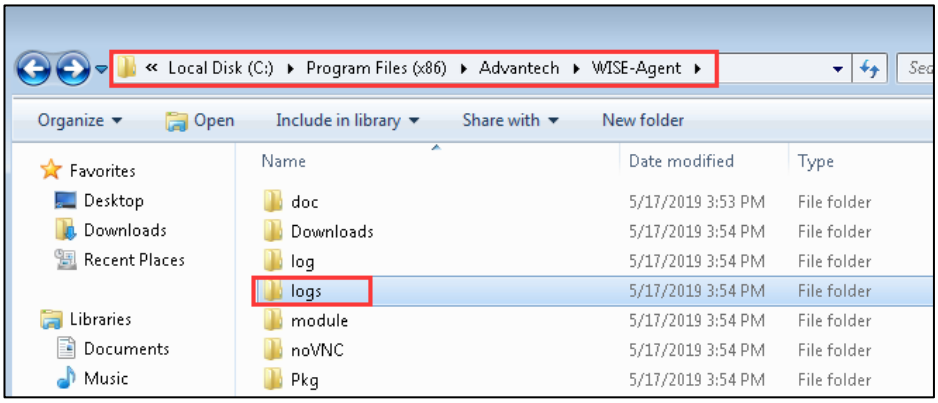
Step 2: Restart WISE-Agent

Open “Task Manager” and switch to “Services”, and restart “WISEAgentService”



Step 3: Retrieve log files from WISE-Agent

The log files under the Installation path\logs

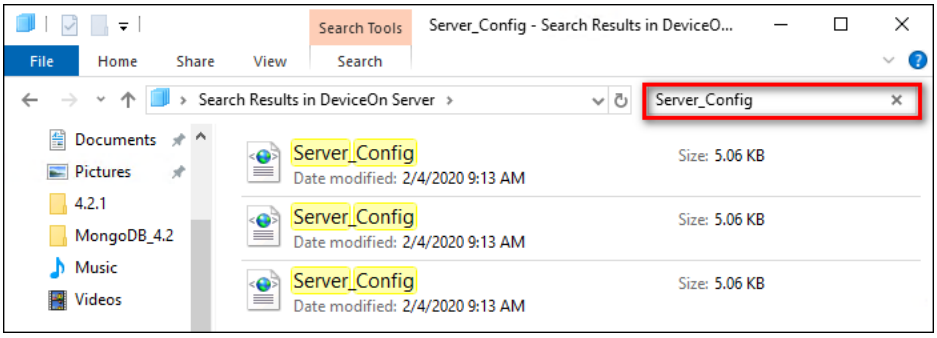


6.11 How to Change DeviceOn Server Address (Standalone)

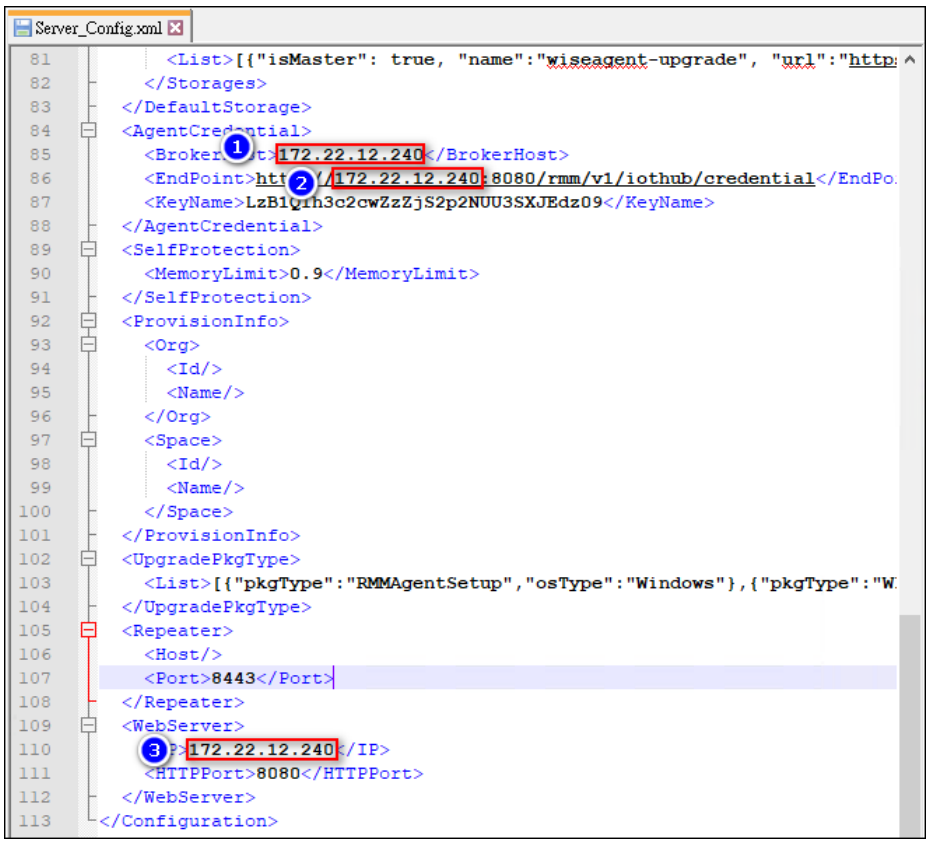
If your DeviceOn Server (Standalone) running on public cloud or on premise environment, and then you would like to update DeviceOn Server address, due to machine/VM IP changed. Here are few steps to update server setting.

Step 1: Search **Server_config.xml** on installation path,

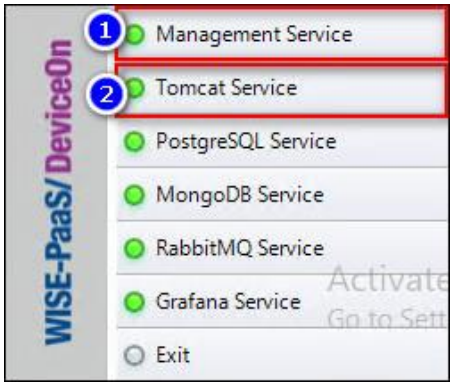
(example, C:\Program Files\Advantech\DeviceOn Server\)



Step 2: Open these files with notepad or other txt editor, and then update host IP address to below path.



Step 3: Restart the **Tomcat** and **Management Services** through DeviceOn Server Control.



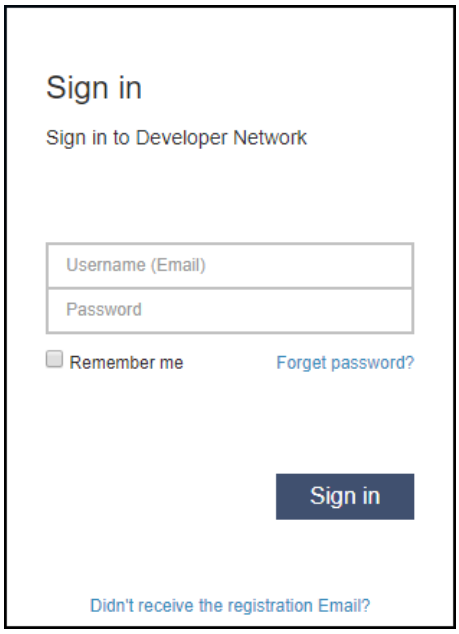
6.12 How to Migrate/Transfer EdgeSense Database to DeviceOn (WISE-PaaS/EnSaaS)

Actually, the DeviceOn is a new product for IoT device management and the backend cores, database structure is based on EdgeSense to develop. In the section, we give a few steps to migrate, transfer database from EdgeSense to DeviceOn. Before the steps, you should prepare the database tool, download and install the program.

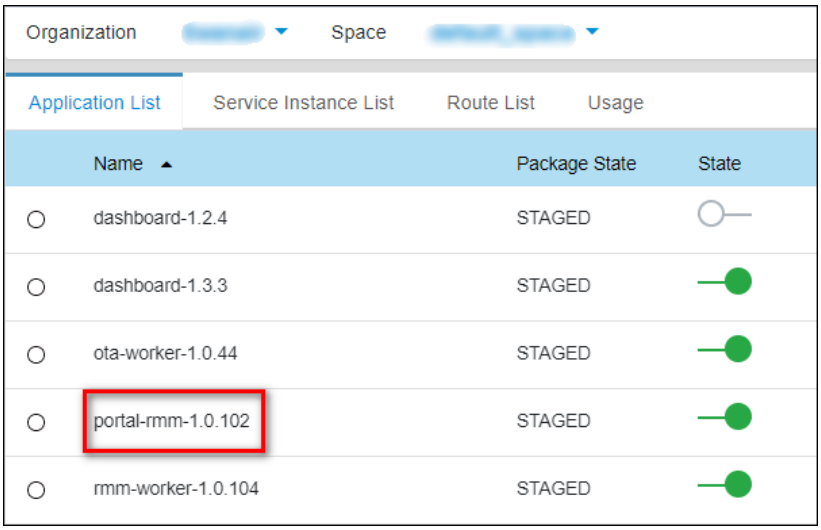
PostgreSQL: pg_dump, psql

MongoDB: mongodump, mongorestore

Step 1: Sign in to your WISE-PaaSEnSaaS Management portal



Step 2: Enter to your organization, space and listing your applications.



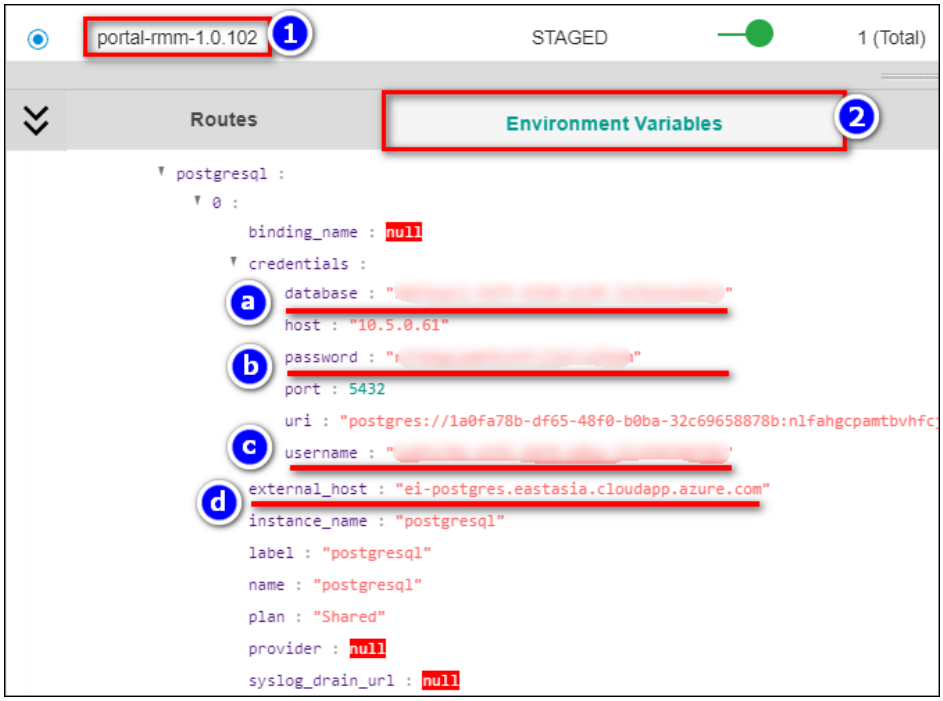
Step 3: Retrieve PostgreSQL information via Application (“portal-rmm-1.0.x”) environment, click on the application.

a. DATABASE_NAME

b. DATABASE_PASSWORD

c. DATABASE_USERNAME

d. DATABASE_HOST

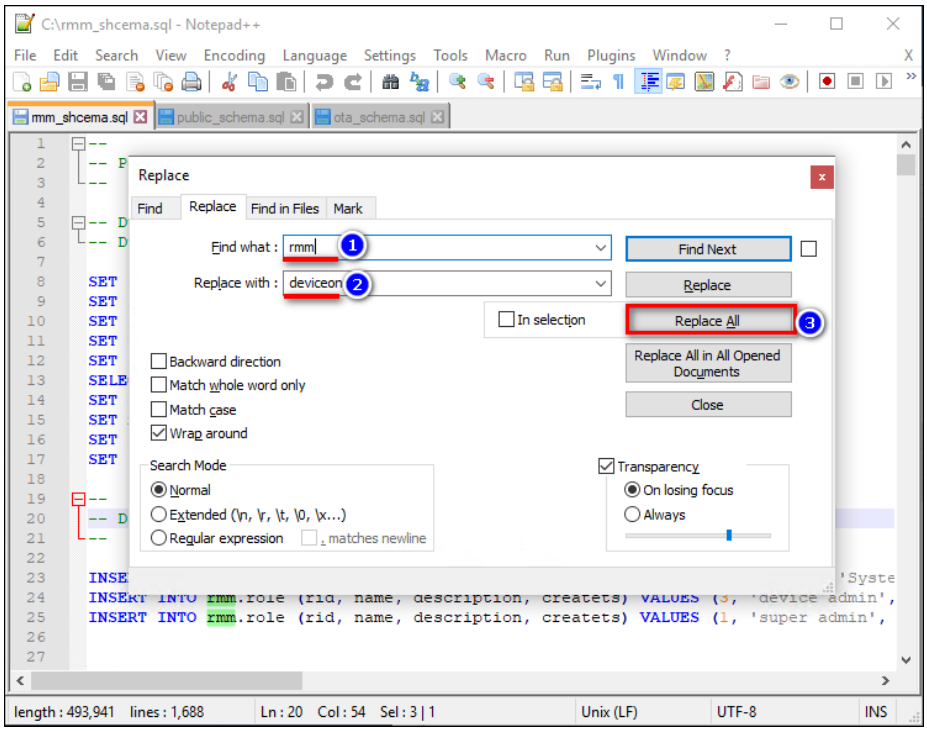


Step 4: Start to backup PostgreSQL data, open the terminal and enter to your PostgreSQL tool path, for example, **<INSTALLATION_PATH>\PostgreSQL\11\bin**

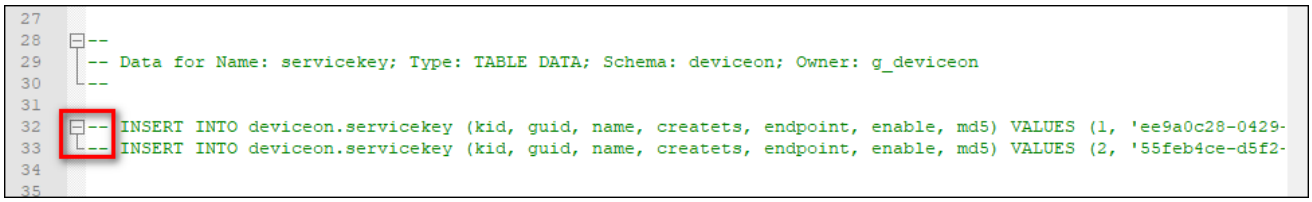
Run the following commands and give a password to backup 3 schema data only.

```
1. pg_dump.exe -h DATABASE_HOST -U DATABASE_USERNAME --column-inserts --data-only --
   schema=rmm --dbname=DATABASE_NAME --file=d:\rmm_schema.sql
2. pg_dump.exe -h DATABASE_HOST -U DATABASE_USERNAME --column-inserts --data-only --
   schema=public --dbname=DATABASE_NAME --file=d:\public_schema.sql
3. pg_dump.exe -h DATABASE_HOST -U DATABASE_USERNAME --column-inserts --data-only --
   schema=ota --dbname=<DATABASE_NAME> --file=d:\ota_schema.sql
```

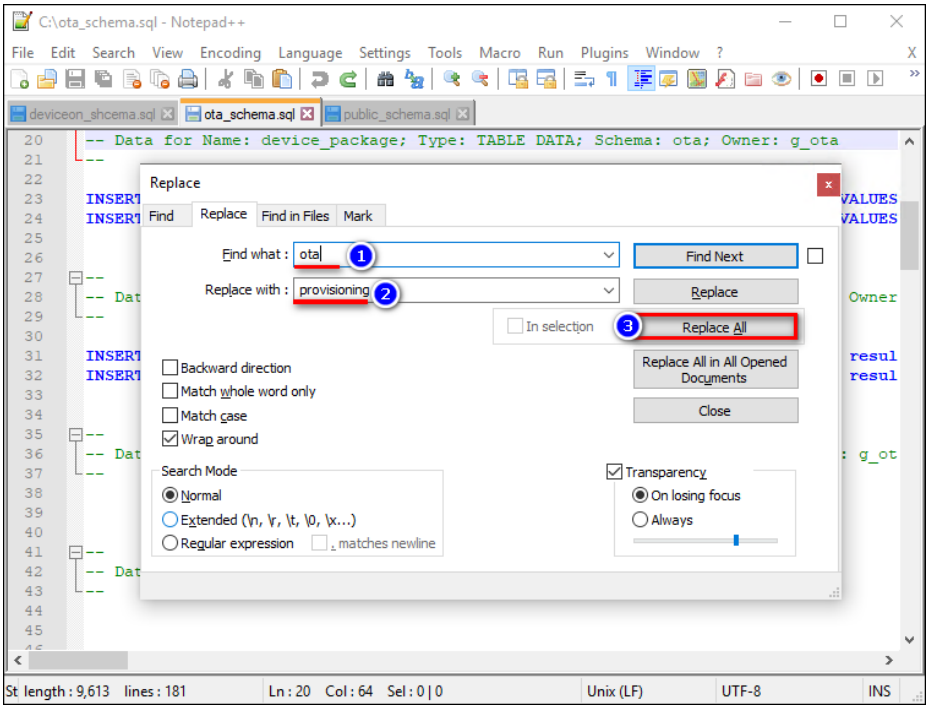
Step 5: Open **rmm_schema.sql** on text editor tool, replace “**rmm**” word to “**deviceon**”.



Then, remove or mark the data on “servicekey”, save as another file (**deviceon_schema.sql**)



Step 6: Open **ota_schema.sql** on text editor tool, replace “**ota**” word to “**provisioning**”, and save as another file (**provisioning_schema.sql**)



Step 7: Before to restore database to **DeviceOn**, please retrieve related information on Management portal, such as Database name, user name, password and host. On WISE-PaaS 3.0, the steps similar to previous, click on the application (portal-deviceon-1.1.x) and get the information via environment.

Step 8: Start to restore PostgreSQL data, open the terminal and enter to your PostgreSQL tool path, for example, **<INSTALLATION_PATH>\PostgreSQL\11\bin**

Run the following commands with the SQL that adjusted and give a password to restore 3 schema data only.

```
1. psql.exe -h DATABASE_HOST -U DATABASE_USERNAME -d DATABASE_NAME -f d:\public_schema.sql
2. psql.exe -h DATABASE_HOST -U DATABASE_USERNAME -d DATABASE_NAME -f d:\deviceon_schema.sql
3. psql.exe -h DATABASE_HOST -U DATABASE_USERNAME -d DATABASE_NAME -
   f d:\provisioning_schema.sql
```

Step 9: For MongoDB backup and restore, you could get the credential on application's environment, and start to run below command to dump collection.

```
1. mongodump.exe --host DATABASE_HOST --db DATABASE_NAME --collection COLLECTION_NAME --
   out d:\mongodb --username DATABASE_USERNAME --password DATABASE_PASSWORD
```

Run the following commands to restore collection to new database.

```
1. mongorestore.exe --host DATABASE_HOST --db DATABASE_NAME --
   collection COLLECTION_NAME D:\mongodb\COLLECTION_NAME.bson --username DATABASE_USERNAME --
   password DATABASE_PASSWORD
```

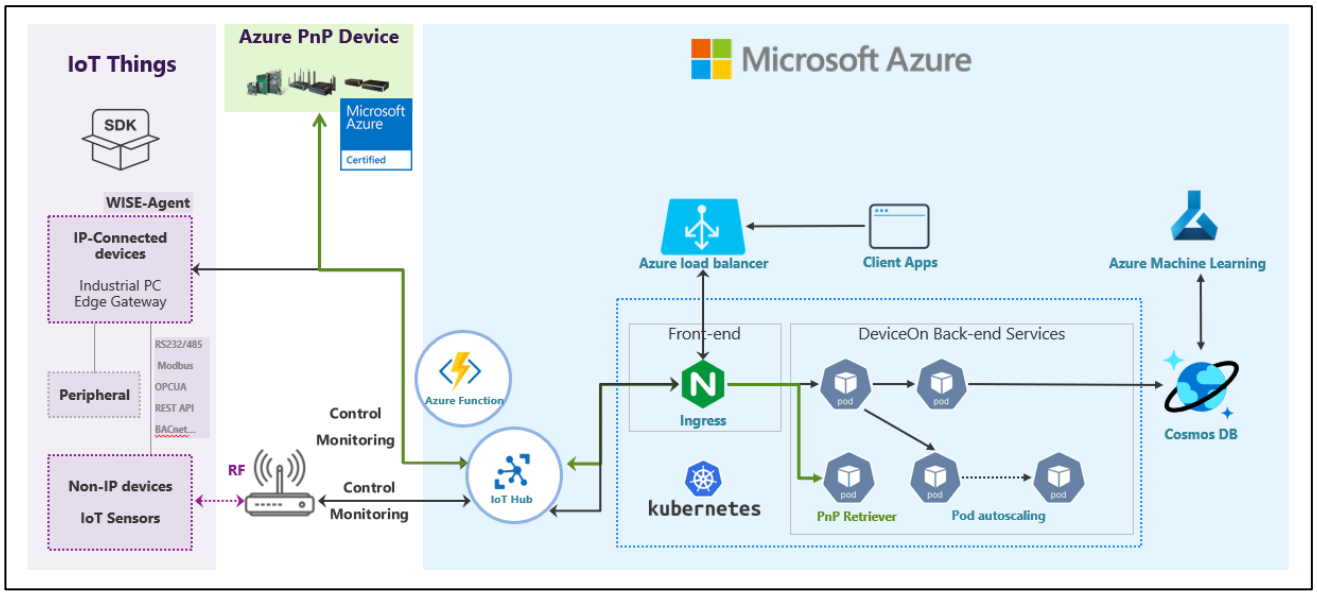
6.13 How Does DeviceOn Interact with AI and Machine Learning

Before to realize the value of data, to export a precise model on your field side, you must collect these raw data from edge side through WISE-Agent. The WISE-Agent not only IPC management but data acquisition for various wire/wireless protocols. DeviceOn could deploy on Azure Kubernetes to leverage Azure PaaS resource, such as Azure Function, IoTHub, Cosmos DB, meanwhile, much easier to start training via Azure Machine Learning.

Leverage Azure Machine Learning, automated ML is the process of automating the time consuming, iterative tasks of machine learning model development. It allows data scientists, analysts, and developers to build ML models with high scale, efficiency, and productivity all while sustaining model quality. Automated ML is based on a breakthrough from our Microsoft Research division.

Traditional machine learning model development is resource-intensive, requiring significant domain knowledge and time to produce and compare dozens of models. Apply automated ML when you want Azure Machine Learning to train and tune a model for you using the target metric you specify. The service then iterates through ML algorithms paired with feature selections, where each iteration produces a model with a training score. The higher the score, the better the model is considered to "fit" your data.

With automated machine learning, you'll accelerate the time it takes to get production-ready ML models with great ease and efficiency.



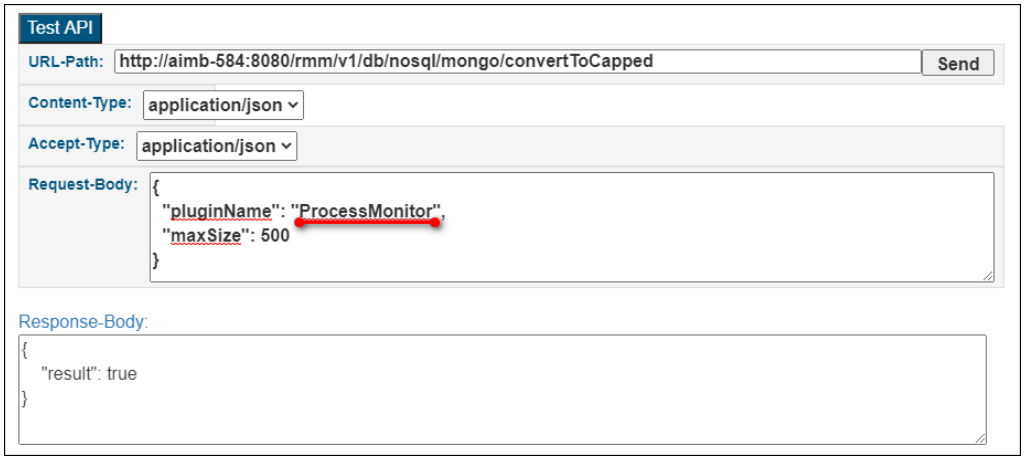
6.14 How to Enable Data Retention on DeviceOn

The device's raw data, such as hardware information, voltage, FAN, network or the wireless sensor data are stored into MongoDB. If ran out of disk storage, the MongoDB service would be stopped. To avoid this situation, you could set up the retention size of each collection on MongoDB via the API or MongoDB command, if the collection existed. Second method, you can adjust the configuration (Server_Config.xml) to enable retention for all collections after the DeviceOn server installed. However, the second method will affect newly created collections only.

- Collection existed:

Please refer to API document to convert collection to capped, the command takes a time (depend on your collection size) to process in the background.

`/rmm/v1/db/nosql/mongo/convertToCapped`



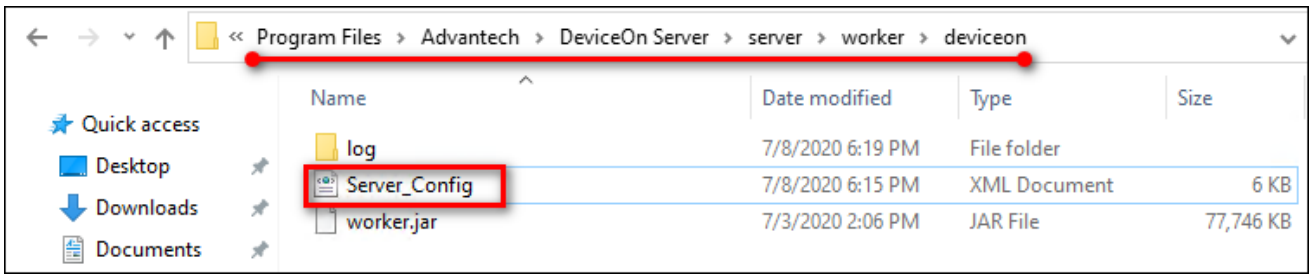
MongoDB Command line:

```
1. db.runCommand({"convertToCapped": "common_ProcessMonitor", size: 524288000});
```

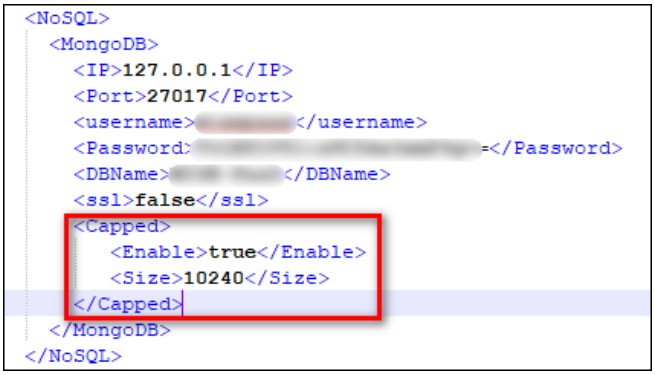
- Collection non-existed (Apply to newly created):

Please add the “Capped” item into Server_config.xml that locate on:

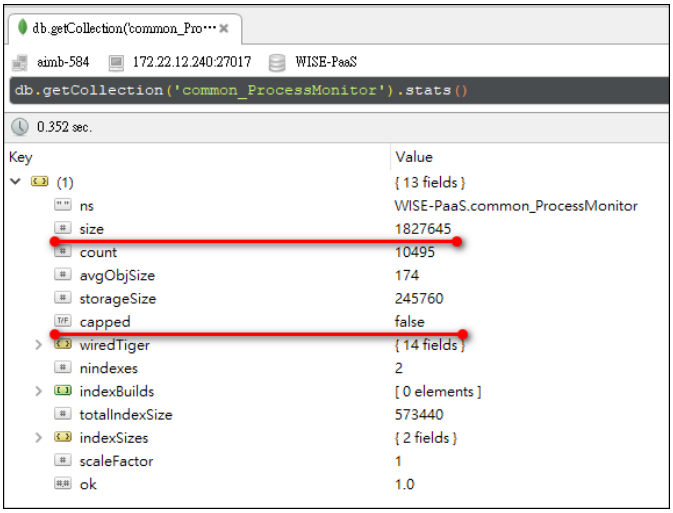
\DeviceOn Server\server\worker\deviceon

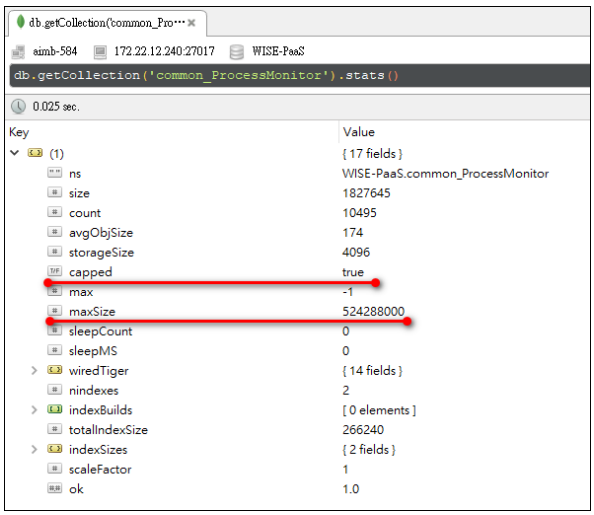


The unit of size is MB, that’s mean the maximum size (uncompressed) is limited, and then restart the Management service.



After that, you could check and confirm the collection applied through third party tool (Rob 3T).





6.15 How to Enable HTTPS on DeviceOn Web Service

Generate Let's Encrypt certificate using Certbot for DeviceOn.

- Let's Encrypt is a new free, automated, and open source, Certificate Authority.
- Certbot is a console based certificate generation tool for Let's Encrypt.

In this recipe, we will generate a Let's Encypt certificate using Certbot. This certificate will then be deployed for use in the DeviceOn server.

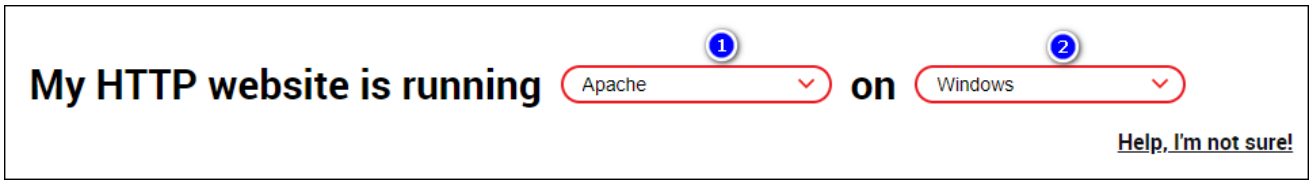
Dependancies:

- Port 443 for https needs to be open and available at time of executing certbot.
- Certbot needs root access while executing because only root is allowed to bind to any port below 1024.
- We will be using our own domain myminio.com as an example in this recipe. Replace with your own domain under your setup.

Step 1: Install Certbot

Install Certbot by following the documentation at https://certbot.eff.org/

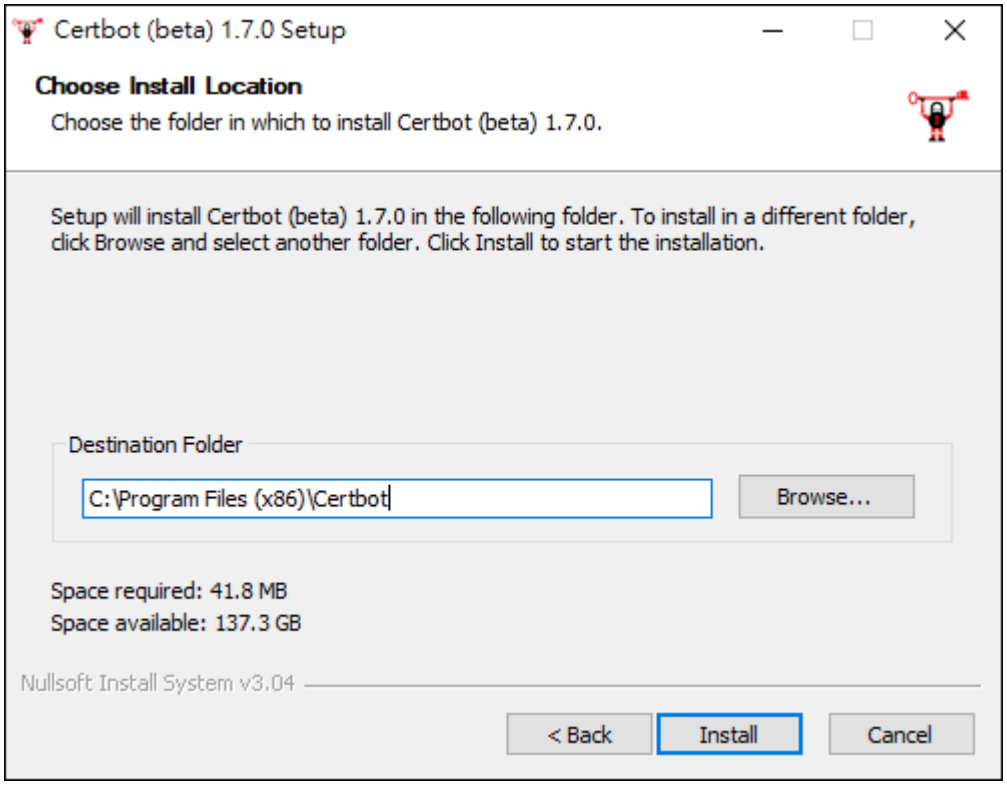
Since the DeviceOn Web service is running on Apache Tomcat, please select to "Apache" and "Windows" to donwnload Certbot installer.



Scrolling down the instruction, you may get the installer package on Step 4.

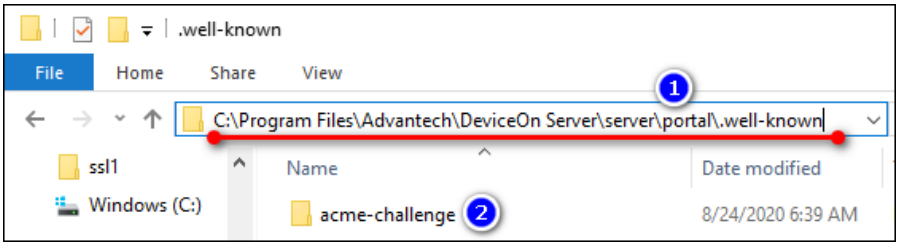
https://dl.eff.org/certbot-beta-installer-win32.exe

Run the installer and follow the wizard. The installer will propose a default installation directory, C:\Program Files(x86), that can be customized.)



Step 2: Create the folder to authenticate

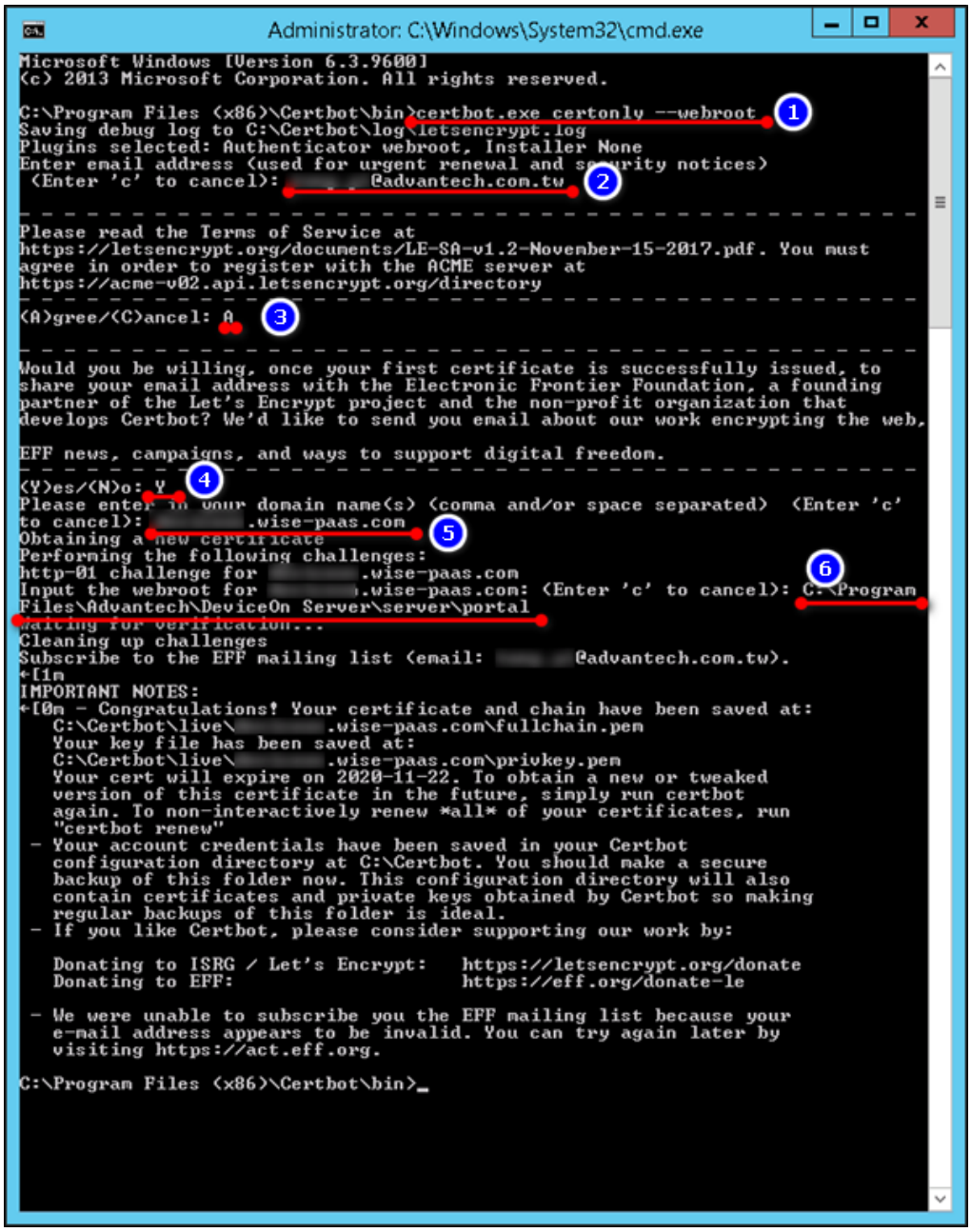
Create the folder named **acme-challenge** under **<DeviceOn Folder>\server\portal\.well-known**. If the folder "**.well-known**" is not exist, please create it manually.



Step 3: Choose how you'd like to run Certbot

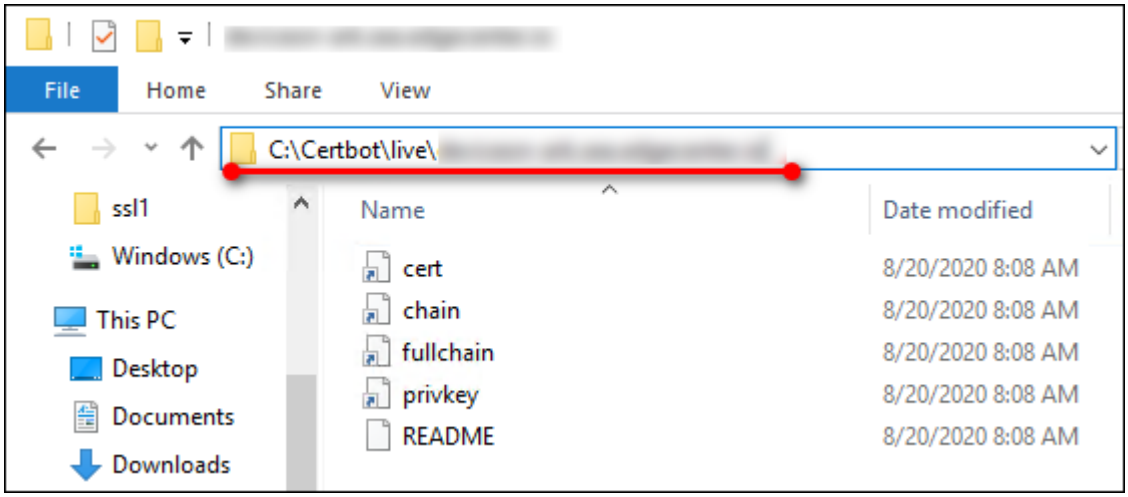
Run the following command to create credential files and enter your website information. The domain name(s) should input yours and the webroot to (**\DeviceOn Path\server\portal**)

```
1. certbot.exe certonly --webroot
```



Step 4: Install your certificate

You'll need to install your new certificate in the configuration file or interface for your webserver. Certificates are located in `C:\Certbot\live\[certificate_name]`, where `[certificate_name]` is the name of your certificate (usually the first domain if the `--cert-name` flag has not been used on the certonly command)



Step 5: Install your certificate on DeviceOn Web Services

- Open the **server.xml** on text editor tool that located in the **\DeviceOn Path\tomcat\conf**
- Append the following XML attribute into **Service** tag and give your certification path (Line, 26 to 28) that generated on Step 4.

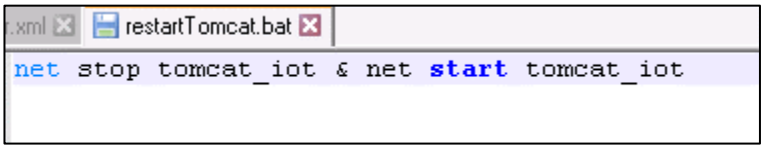
```
1.  <Connector port="443"
2.      protocol="org.apache.coyote.http11.Http11AprProtocol"
3.      connectionTimeout="20000"
4.      useSendfile="false"
5.      compression="on"
6.      compressionMinSize="2048"
7.      noCompressionUserAgents="gozilla, traviata"
8.      compressableMimeType="text/html,text/xml,text/plain,text/css,text/javascript,applicatio
    n/javascript,application/xml,application/json"
9.      redirectPort="8443"
10.     maxThreads="150"
11.     scheme="https"
12.     secure="true"
13.     SSLEnabled="true">
14.     <UpgradeProtocol className="org.apache.coyote.http2.Http2Protocol"
15.         overheadWindowUpdateThreshold="-1"
16.         overheadDataThreshold="-1"
17.         writeTimeout="-1"
18.         streamWriteTimeout="-1"
19.         streamReadTimeout="-1"
20.         maxHeaderSize="8192"
21.         maxConcurrentStreams="300"
```

```
        readTimeout="-1"
        compressibleMimeType="text/html,text/xml,text/plain,text/css,text/javascript,application/javascript,application/json"
        compression="on" compressionMinSize="1024"/>
    <SSLHostConfig>
        <Certificate certificateKeyFile="C:\Certbot\live\<DNS>\privkey.pem"
                     certificateFile="C:\Certbot\live\<DNS>\cert.pem"
                     certificateChainFile="C:\Certbot\live\<DNS>\fullchain.pem"
                     type="RSA" />
    </SSLHostConfig>
</Connector>
```

Step 6: Restart DeviceOn web services (Tomcat_IoT) to reload the configuration

Step 7: Enable to automatic renewal

- Create a batch file named restartTomcat.bat which content as below.

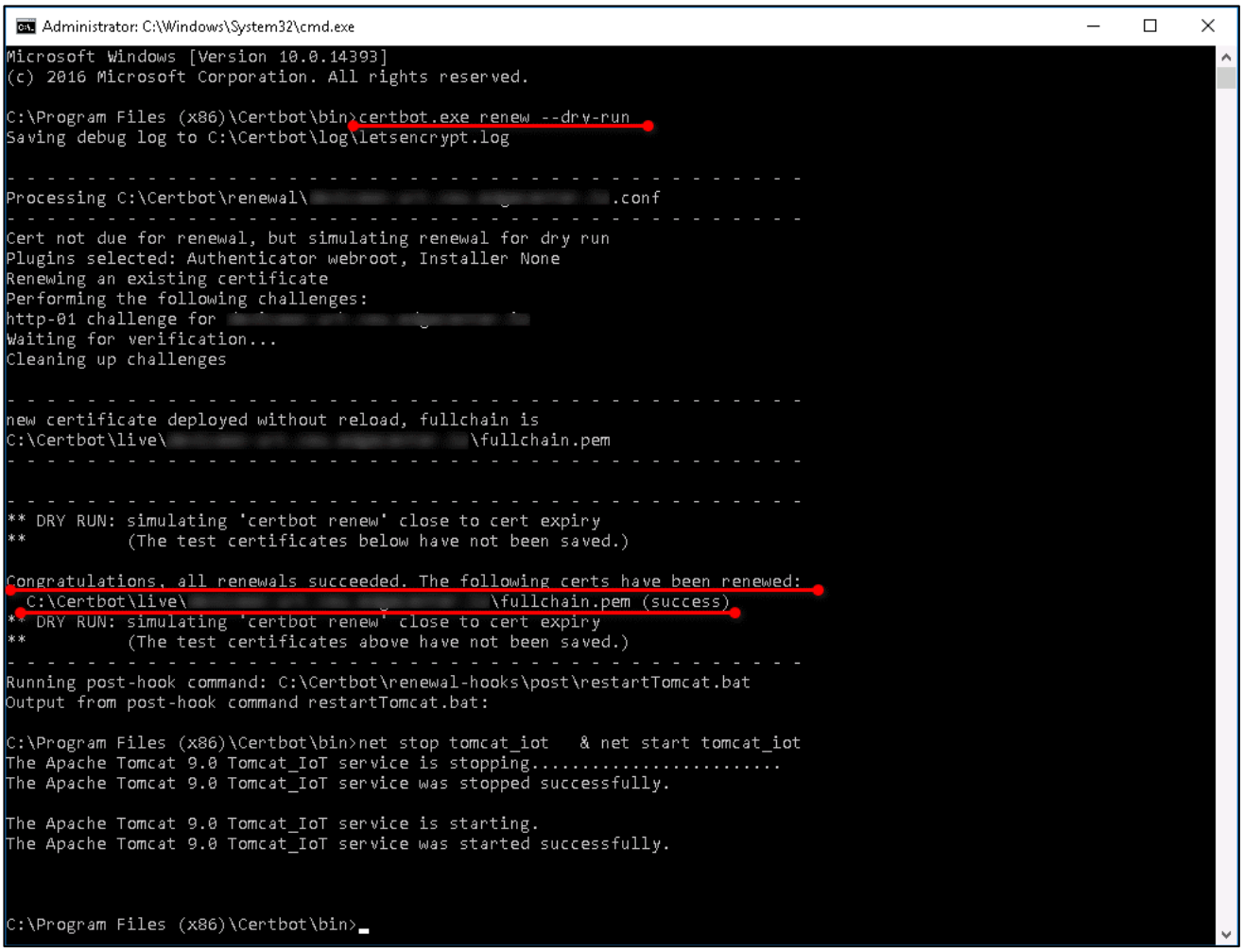


- Copy the batch file into C:\Certbot\renewal-hooks\post\

Step 8 (Optional), Test automatic renewal, please run the following command

```
certbot.exe renew -dry-run
```

Step 9 (Optional), if you get all renewals succeeded, it means your configuration is correct.



7. Reference

7.1 User Permission

Item	Action	Description	Root	System Admin	Device Admin
Account Management	Create	Create Account	✔ (Not Include Self)	✔ (Only Device Admin)	
	Edit	Edit Account Basic Information	✔	✔ (Only Self & Device Admin)	✔ (Only Self)
	Edit	Edit Account Role	✔ (Not Include Self)		
	Edit	Disable Account	✔ (Not Include	✔ (Only Device Admin)	

			Self)		
	View	View Account Information	✔	✔ (Only Self & Device Admin)	✔ (Only Self)
Device Group Management	Create	Create Device Group	✔	✔ (Only Self & Device Admin)	✔ (Only Self)
	Edit	Edit Device Group Information	✔	✔ (Only Self & Device Admin)	✔ (Only Self)
	View	View Device Group Information	✔	✔ (Only Self & Device Admin)	✔ (Only Self)
	Delete	Delete Device Group	✔	✔ (Only Self & Device Admin)	✔ (Only Self)
Device Control & Management	Add	Add Unmanaged Device	✔	✔	✔
	Edit	Edit Device Information	✔	✔ (Only Self & Device Admin)	✔ (Only Self-Managed Devices)
	View	View Device Information	✔	✔ (Only Self & Device Admin)	✔ (Only Self-Managed Devices)
	Edit	Remove Device	✔	✔ (Only Self & Device Admin)	✔ (Only Self-Managed Devices)
	View	Search Unmanaged Devices	✔	✔	✔
	Control	Power, Remote Desktop, Terminal, Screenshot, Backup/Recovery, Protection, Windows Lockdown Actions…	✔	✔ (Only Self & Device Admin)	✔ (Only Self-Managed Devices)
Event Log Management	View	View and Export Device Event	✔	✔ (Only Self-Managed &	✔ (Only Self-Managed

				Device Admin Devices)	Devices)
	View	View and Export System Event	✔	✔	
	View	View and Export Operation Event	✔	✔ (Only Self-Managed & Device Admin Devices)	✔ (Only Self-Managed Devices)
	View	Long-polling, Web-Socket	✔	✔	✔
OTA Management	Create	Create Storage Repository	✔	✔	
	Edit	Edit Storage Repository	✔	✔	
	View	View Storage Repository	✔	✔	✔
	Delete	Delete Storage Repository	✔	✔	
	Upload	Upload OTA Package	✔ (Only Self)	✔ (Only Self)	✔ (Only Self)
	View	View OTA Package	✔ (Only Self)	✔ (Only Self)	✔ (Only Self)
	Delete	Delete OTA Package	✔ (Only Self)	✔ (Only Self)	✔ (Only Self)
	Deploy	Deploy OTA Package	✔	✔ (Only Self-Managed & Device Admin Devices)	✔ (Only Self-Managed Devices)
	Edit	Edit OTA Deploy Configuration	✔	✔	
System Setting Management	Create	Create an Action	✔ (All Groups) on Self Account	✔ (Only Self-Groups & Device Admin Groups) on Self Account	✔ (Only Self-Groups)
	Edit	Update an Action	✔ (All	✔ (Only Self-	✔ (Only Self-

			Groups) on Self Account	Groups & Device Admin Groups) on Self Account	Groups)
	View	View Action	✔ Self Account	✔ Self Account	✔ Self Account
	Delete	Delete Action	✔ Self Account	✔ Self Account	✔ Self Account
	Provisioning	Power Management	✔	✔ (Only Self-Managed & Device Admin Devices)	✔ (Only Self-Managed Devices)
		Backup/Recovery	✔	✔ (Only Self-Managed & Device Admin Devices)	✔ (Only Self-Managed Devices)
		Protection	✔	✔ (Only Self-Managed & Device Admin Devices)	✔ (Only Self-Managed Devices)
	Edit	Edit Event Alert Setting	✔ (Only Self)	✔ (Only Self)	✔ (Only Self)
	Edit	Configure Alert Service	✔	✔	
	Create	Create Rule Engine	✔	✔ (Only Self & Device Admin)	✔ (Only Self-Managed Devices)
	Update	Edit Rule Engine	✔	✔ (Only Self & Device Admin)	✔ (Only Self-Managed Devices)
	View	View Rule Engine	✔	✔ (Only Self & Device Admin)	✔ (Only Self-Managed Devices)
	Delete	Delete Rule Engine	✔	✔ (Only Self & Device Admin)	✔ (Only Self-Managed Devices)
	Edit/View	Edit/View System UI	✔	✔	
	Edit	Activate	✔	✔	

		DeviceOn License (Perpetual Only)			

7.2 Retrieve My Azure Account Information

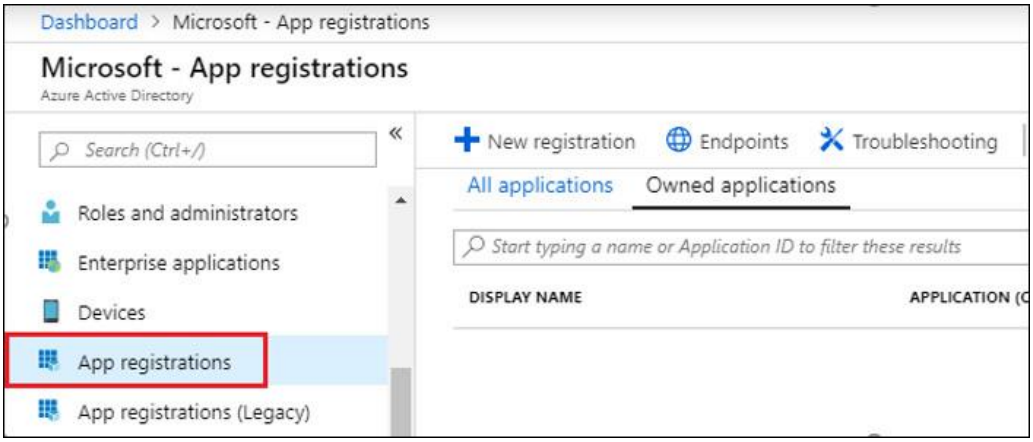
7.2.1 Method 1 – Create & Get Information on Azure Portal

Step 1: Create Your Application

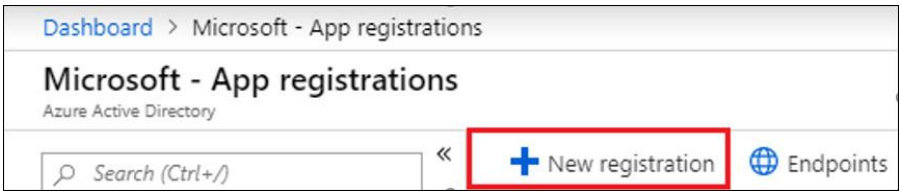
1.1. Log into your Azure Portal

1.2. Select [**Azure Active Directory**]

1.3. Select **[App registrations**]



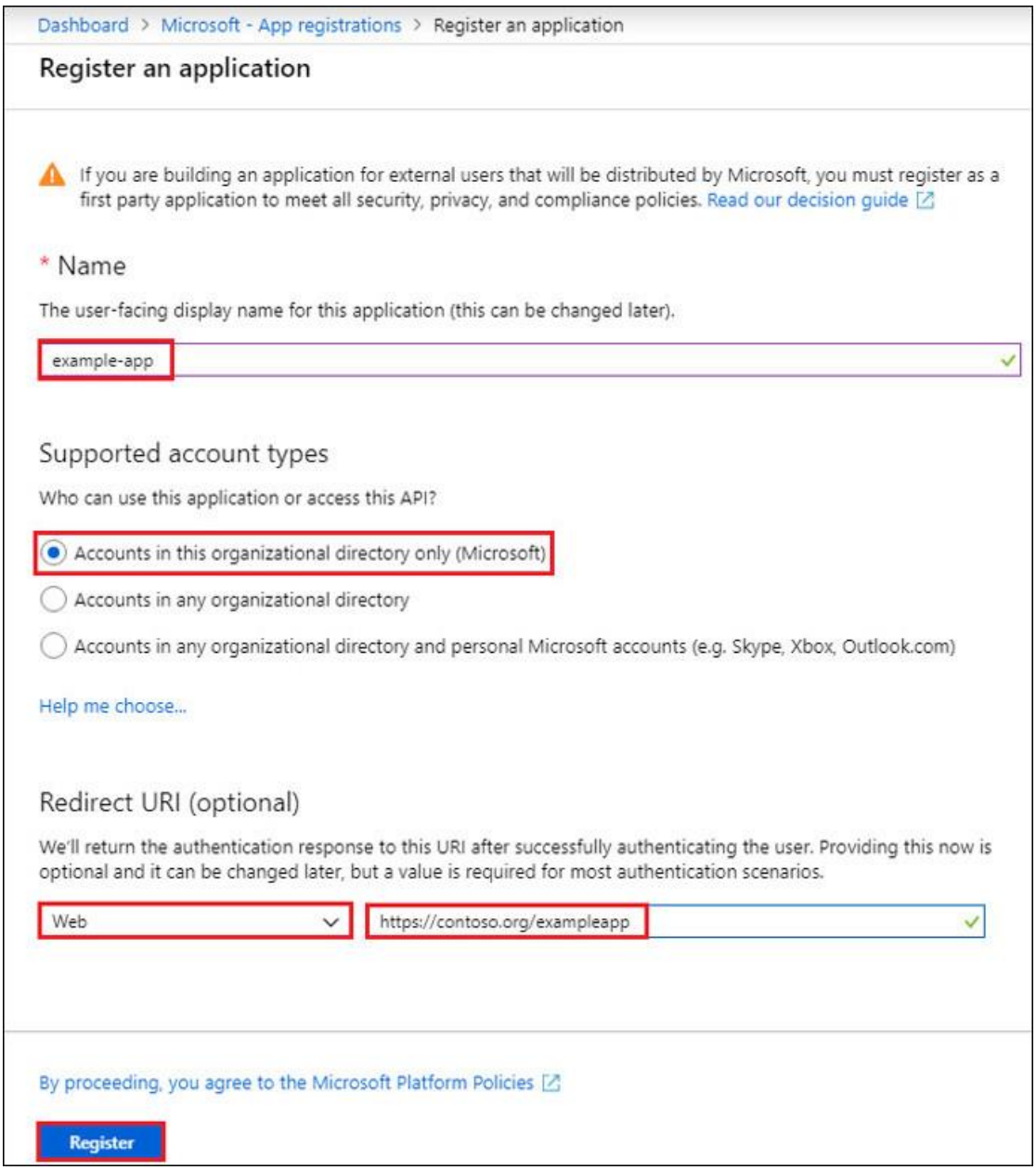
1.4. Add [**New Registration**]



1.5. Setup your **Application Name** then click [**Register**].

- Enter your Application display name in **Name** filed.
- Setup **Supported account types** by selecting the respective account type for this API.
- Under **Redirect URI**, select Web for the type of application you want to create. Enter the URI where the access token is sent to.

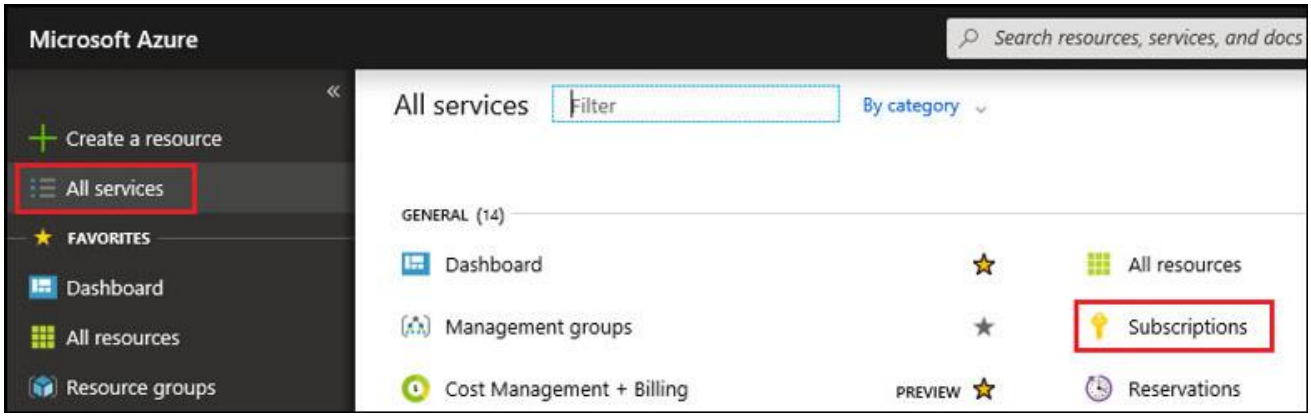
  Note: You cannot create a Native application credential nor use the type for an automated application.



Step 2: Get Subscription ID

To access resources in your subscription, you must assign a role to the Application. You can pick between Subscription, Resource Group or Resource. Permissions are inherited to lower scope levels. [For more details, see RBAC: Built in Roles]

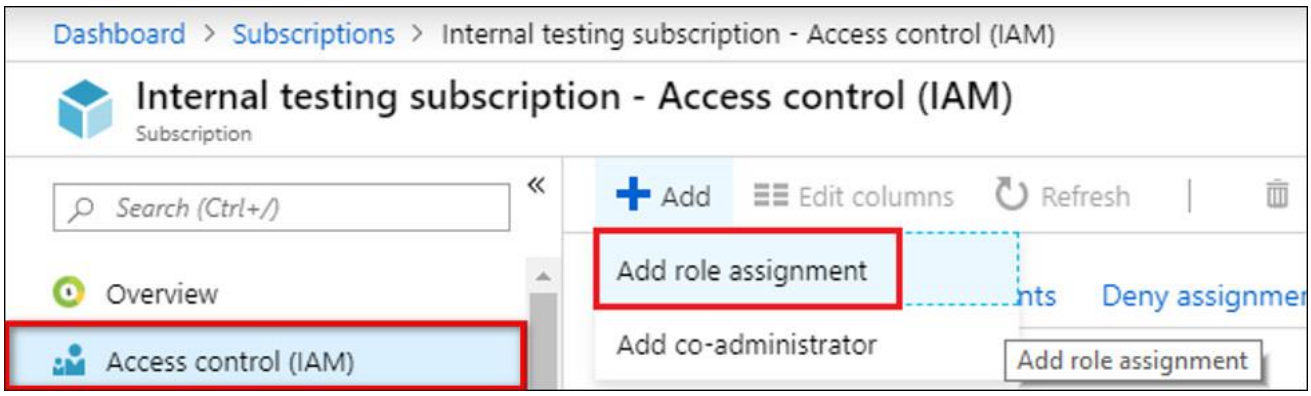
2.1. Select **All services** then select **Subscriptions** to set up the level of scope you wish to assign this application.



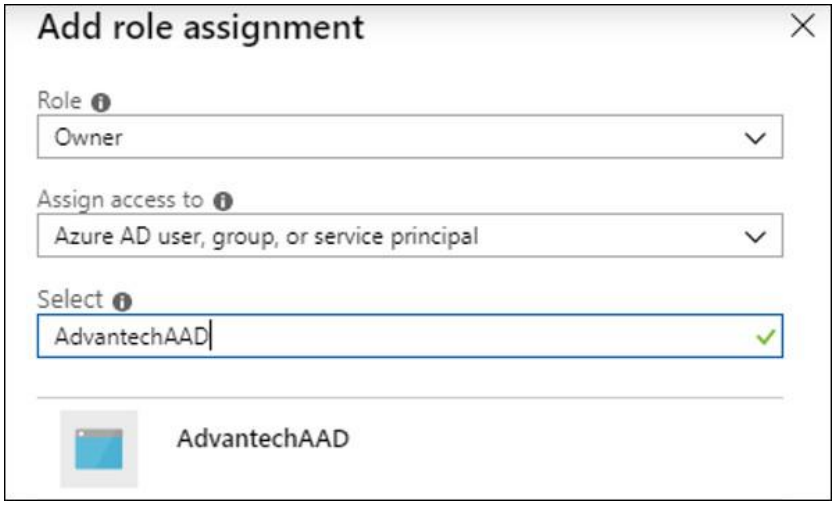
2.2. Find the Subscription you would like to assign to the Application created in the Step 1. Copy the **Subscription ID**, as this is one of the Azure data fields required on the WISE-PaaS Marketplace later. (**Ref: Marketplace field #A)**

！Troubleshoot: If you do not see the subscription you're looking for, select global subscriptions filter. Make sure the subscription you want is selected for the portal.

2.3. Select **Access control (IAM)** then **Add role assignment**



2.4. Select the **Owner** role. By default, Azure AD applications are not displayed in the available options. To find your application, search for the name.

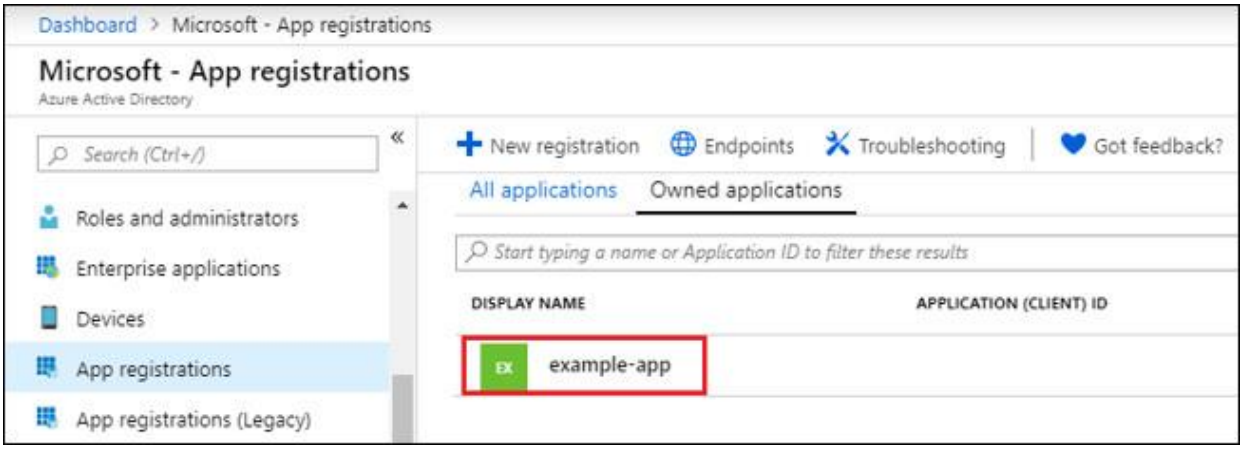


2.5. Click **Save** to finish assigning the role. You will be able to see your application in the list of users assigned to a role for that scope.

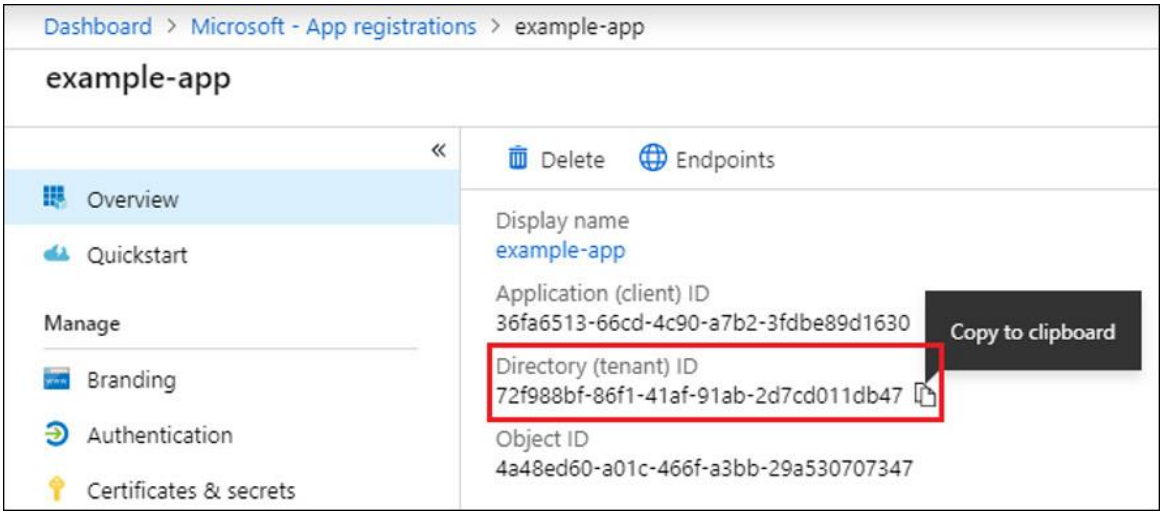
Step 3: Get Application & Tenant ID

3.1. Select **Azure Active Directory**

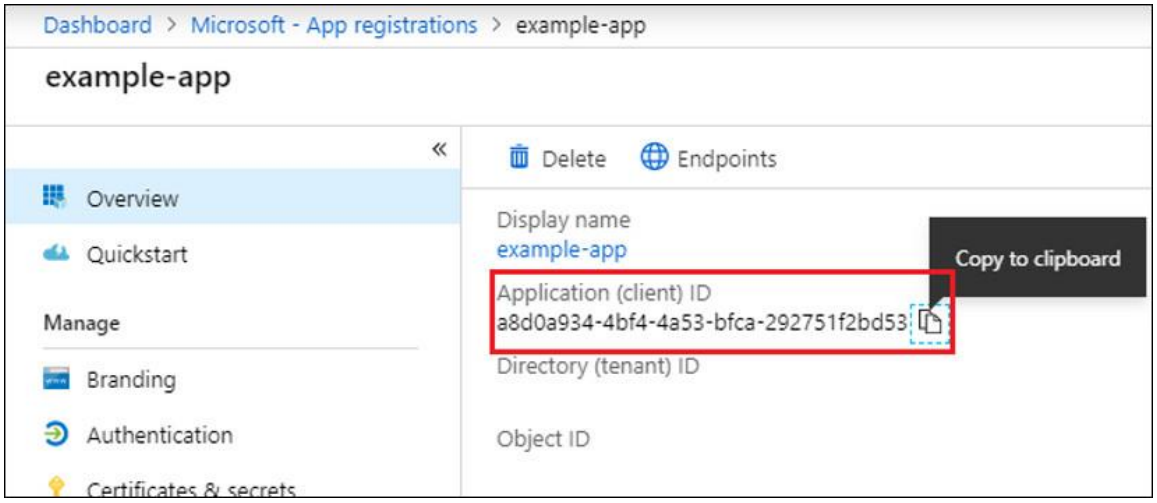
3.2. From **App registrations** in Azure AD, select your application



3.3. Copy the **Directory (tenant) ID** as another piece of Azure information that will be required on the WISE-PaaS Marketplace later. (**Ref: Marketplace field #C)**



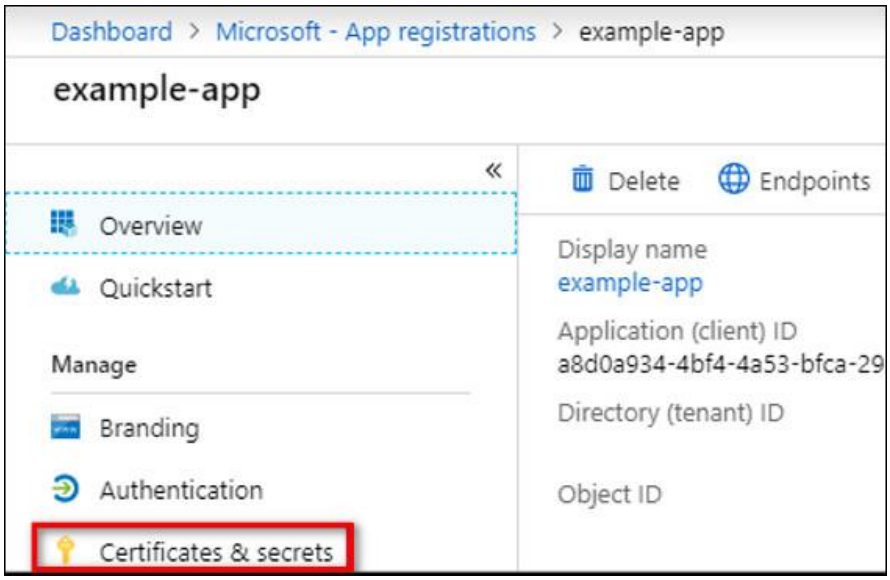
3.4. Copy the **Application (client) ID** as part of Azure information that will be required on the WISE-PaaS Marketplace later. (**Ref: Marketplace field #B)**



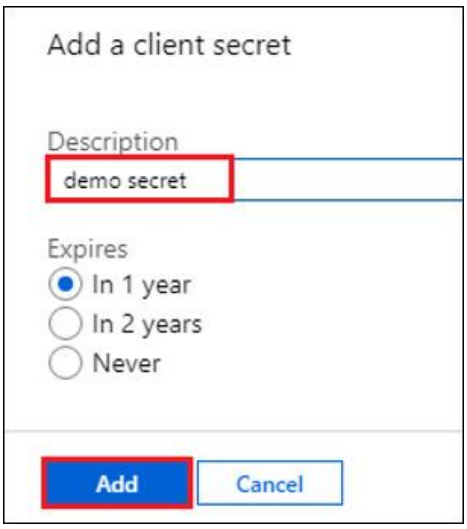
Step 4: Add & Get Client Secret

4.1. Select [**Certificates & secrets**]

4.2. Select **Client secrets** then **New client secret**



4.3. Provide a description for the new client secret, set up the expiration period. Then Click [**Add**]



Copy Client Secret (**Ref: Marketplace field #D**)

7.2.2 **Method 2 – Create via Azure CLI (Command-line Tool)**

Step 1: Install Azure CLI

For details, please view this step by step guide

Step 2: Sign in to the Azure Account

```
1. C:\>az login
```

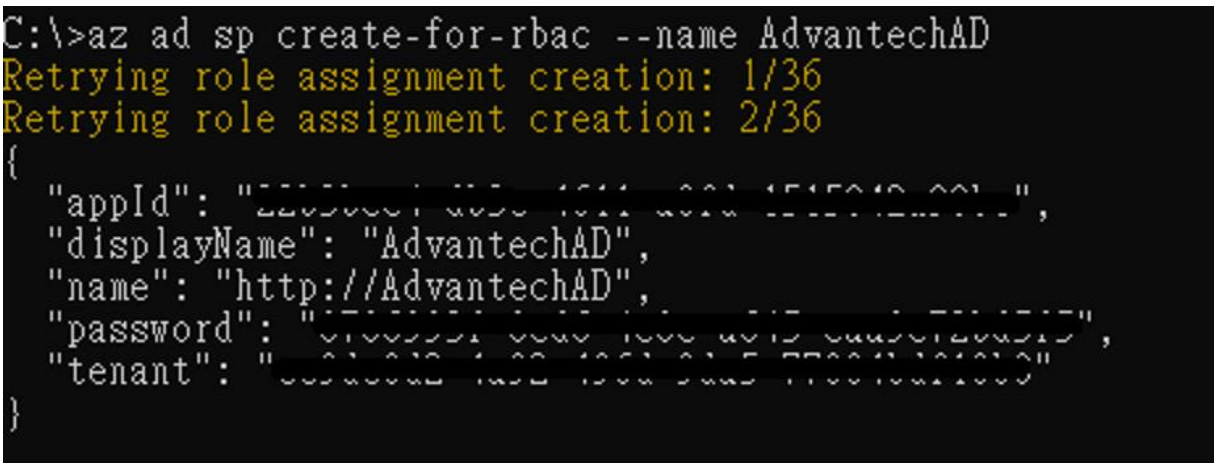
Note: If the CLI can open your default browser, it will do so and load a sign-in page. Otherwise, you need to open a browser page and follow the instructions on the command line to enter an authorization code after navigating to https://aka.ms/devicelogin in your browser. Sign in with your account credentials in the browser.

Step 3: Get Subscription ID & Copy Output

```
2. C:\>az account show --query id
```

Step 4: Create service principal and get Application ID, Tenant ID and Client Secret

```
3. C:\>az ad sp create-for-rbac --name ServicePrincipalName
```



Reference: [Create an Azure service principal with Azure CLI >]