IBM Cloud Pak for AIOps 4

Best Practices

FIRST EDITION

Note: Before using this information and the product it supports, read the information in "Notices" located at the end of this document.

Contents

About this publication

The purpose of this document is to be a reference to anyone deploying *IBM Cloud Pak for AIOps 4*, and to help them to implement a solution using standard and recommended methods and techniques.

This document details recommended best practices for when installing and configuring IBM Cloud Pak for AIOps 4. It is not designed to replace official product documentation; instead, it augments the official product documentation and provides recommended standard methods and practices for deployment and configuration.

Installations are always dependent on customer requirements. With that said, the best practices contained in this document should always be adhered to whenever possible.

Note that this document is not intended as a training manual. It is a highly recommended that anyone making use of this document has also completed relevant IBM training.

Intended audience

This publication is intended as essential reading for all technical staff that are responsible for:

- Creating IBM Cloud Pak for AIOps 4 solutions;
- Installing IBM Cloud Pak for AIOps 4;
- Administering IBM Cloud Pak for AIOps 4; advances
- Supporting IBM Cloud Pak for AIOps 4.

What this publication contains

This publication contains the following sections:

- *Chapter 1 Introduction* on page 1:

  This chapter introduces IBM Cloud Pak for AIOps 4 (AIOps) and provides an overview of its many features. It includes a suggested approach to take when deploying AIOps in terms of an order of implementation to get quick value.

- *Chapter 2 Planning* on page 5:

  This chapter provides guidance on how to plan for an AIOps deployment and covers: data sources and rates, Netcool component considerations, deployment architecture, OpenShift sizing and configuration recommendations, and the historic event archive.

- *Chapter 3 Deployment* on page 15:

  This chapter provides guidance on the deployment of AIOps including the setting up of OpenShift, provisioning AIOps, installation of any needed Netcool components, migration from any existing Netcool components, and a note on the importance of stress testing before deploying into production.

- *Chapter 4 Implementation* on page 24:

  This chapter covers where to implement custom functionality, how to get started with events, then topology, then metrics, then logs, then similar ticket analysis and change risk, and finally, the topic of self-monitoring.

Publications

This section lists related publications in the IBM AIOps library. Netcool/OMNIbus and Netcool/Impact are used extensively by customers around the world as part of IBM Cloud Pak for AIOps 4 deployments and so are just as relevant as ever. Moreover, Netcool components will likely be included in any AIOps deployment done, depending on requirements.

Your IBM AIOps related publications library

The following documents are available in the IBM AIOps Best Practices library:

- *IBM Netcool/OMNIbus 8.1 Best Practices Guide*

  This is the primary best practices IBM Netcool/OMNIbus document and includes information on all aspects of IBM Netcool/OMNIbus best practices including: requirements gathering, solution delivery, writing triggers, procedures, Probe rules, implementing automated housekeeping and flood protection mechanisms, and a detailed description of the standard multitier architecture configuration and all its parts.

- *IBM DB2 High Availability for IBM Netcool products - Best Practices*

  Several Netcool products use a DB2 database to store configuration data. This document provides guidance on a best practice method of implementing DB2 high availability using High Availability Disaster Recovery (HADR) thereby providing a resilient database service to the connecting Netcool application.

- *Upgrading IBM Netcool/OMNIbus in Production Environments Best Practices*

  This document provides practical tips for upgrading IBM Netcool/OMNIbus 7.x either *in situ* or where new hardware is deployed. The information contained in this publication is particularly pertinent in a production environment where data preservation is paramount, as is the need to minimize any kind of system outages. Note that although this document was originally written for Netcool/OMNIbus version 7.x, it is also equally applicable to Netcool/OMNIbus version 8.1.

- *Netcool/Impact 7.1 Best Practices Guide*

  This is the primary best practices Netcool/Impact document and includes information on all aspects of Netcool/Impact best practices.

All the publications listed above can be accessed online via the following URL:

http://ibm.biz/nco_bps

Conventions used in this publication

This publication uses several conventions for special terms and actions and operating system-dependent commands and paths.

Typeface conventions

This publication uses the following typeface conventions:

Bold

— Lowercase commands and mixed case commands that are otherwise difficult to distinguish from surrounding text

- Interface controls (check boxes, push buttons, radio buttons, spin buttons, fields, folders, icons, list boxes, items inside list boxes, multicolumn lists, containers, menu choices, menu names, tabs, property sheets), labels (such as **Tip:** and **Operating system considerations:**)
- Keywords and parameters in text

Italic

- Citations (examples: titles of publications, diskettes, and CDs)
- Words defined in text (example: a nonswitched line is called a *point-to-point* line)
- Emphasis of words and letters (words as words example: "Use the word *that* to introduce a restrictive clause."; letters as letters example: "The LUN address must start with the letter *L*.")
- New terms in text (except in a definition list): a *view* is a frame in a workspace that contains data
- Variables and values you must provide: ... where *myname* represents....

`Monospace`

- Examples and code examples
- File names, programming keywords, and other elements that are difficult to distinguish from surrounding text
- Message text and prompts addressed to the user
- Text that the user must type
- Values for arguments or command options

Operating system-dependent variables and paths

This publication uses the UNIX convention for specifying environment variables and for directory notation.

When using the Windows command line, replace $variable with %variable% for environment variables, and replace each forward slash (/) with a backslash (\) in directory paths. For example, on UNIX systems, the $NCHOME environment variable specifies the directory where the IBM Netcool/OMNIbus core components are installed. On Windows systems, the same environment variable is %NCHOME%. The names of environment variables are not always the same in the Windows and UNIX environments. For example, %TEMP% in Windows environments is equivalent to $TMPDIR in UNIX environments.

If you are using the bash shell on a Windows system, you can use the UNIX conventions.

Preface: From on-premise to the Cloud

Netcool has been the market leader in the event, network, and operations management space for more than 20 years. Indeed, most of the world's largest enterprises, telcos, and others still rely on Netcool technologies to keep their environments up and running. With the advent of Cloud, Cloud technologies, and artificial intelligence (AI), the traditional approaches can be dramatically enhanced with new features and capabilities that translate into quicker diagnosis of problems, improved automation, quicker time to repair, reduced costs overall, and ultimately significant business value.

IBM Cloud Pak for AIOps represents the evolution of Netcool technologies and dramatically extends capabilities into realms not possible before. IBM Cloud Pak for AIOps 4 contains several AI and other capabilities that enable the processing of large quantities of different types of data that was not possible before. In addition to event data, IBM Cloud Pak for AIOps 4 can also ingest your topology, metric, ticket, and log data, make sense of it, and provide more insight into your environment from just events alone.

While IBM Cloud Pak for AIOps 4 is a Cloud technology and runs on IBM Red Hat OpenShift, it does not mean it can *only* be installed to a commercial Cloud services provider environment. Deployments can also be done on an OpenShift cluster in your own data centre, into a third-party provided environment, or a combination of both. In short, IBM Red Hat OpenShift is the *cloud platform* on which AIOps runs, but this could be deployed anywhere. This makes AIOps very flexible in terms of deployment options and can cater to different needs in this respect. OpenShift also natively gives a robust local resiliency due to its design.

Chapter 1 Introduction

This document details the best practices that should be employed when deploying IBM Cloud Pak for AIOps 4. This document is designed to help an engineer quickly and efficiently install a working system and ensure that if another engineer must subsequently work on a deployment installed by someone else, they will immediately understand what has been configured and how.

Many of the best practice concepts covered in this document that relate to the Netcool components have remain unchanged for many years and trace their origins to the *Netcool Certified Consultant* programme and previously published best practice guides going back more than 20 years. This document includes the relevant concepts from these sources and introduces new ones based on the new technologies and product features in the latest releases of IBM Cloud Pak for AIOps 4.

Previously, Netcool solutions relied heavily on a pre-canned architecture configuration, called the multi-tier architecture configuration, to enable scaling. While many of the practices used to scale environments out remain unchanged, many have also changed as advancement in technologies have resulted in more effective ways of doing things. You will undoubtedly notice architectural changes to the way things happen as you work your way through this guide.

Concepts and capabilities

IBM Cloud Pak for AIOps 4 (AIOps) has a wealth of new features that provide immense value to operations, as well as dramatically extend any existing traditional on-premise Netcool deployment. This section summarizes these new capabilities.

- Event correlation:

  AIOps has world-leading event correlation capabilities and provides three methods that work simultaneously and collaboratively to correlate events together that are related to the same problem. *Scope-based grouping* is the first one and enables an operations team to apply local knowledge and specify known attributes that events can reasonably be grouped on, such as the same geographic location, or the same business unit. A time window is then applied to the logic to ensure that two entirely different problems with the same scope aren't inadvertently grouped together. Next, AIOps automatically analyzes all events that flow through the system, looking for events that suspiciously always occur together. After a relationship has been identified, the events are grouped together automatically if they are seen occurring again in future. This correlation method is called *temporal grouping*, and is especially useful for identifying events that superficially have no connection, but are in fact related. The third correlation capability leverages the *topology* to correlate events together based on connectedness of the underlying topology. The correlation engine in AIOps runs all three of these correlation methods simultaneously and merges groups together where the groups overlap. This can happen when events are members of more than one group. This is what makes the event correlation capability of AIOps so powerful and enables AIOps users to solve all manner of complex correlation scenarios that one alone could not.

- Event Seasonality:

  An embedded machine learning (ML) algorithm processes all events that pass through AIOps looking at when each event occurs. If there is a perceivable pattern to the timing of when events occur, this is marked up in the event as being *Seasonal*. Identifying events that occur on the same day of the month, the same day of the week, hour of the day, minute of the hour, or a combination of these can provide valuable insights as to why a problem may be occurring. It is also useful for prioritizing maintenance activities.

- Runbook Automation (RBA):

  AIOps comes with a powerful embedded automation engine that enables the automated resolution of large swathes of the event estate. By automating the resolution of large numbers of tedious and repetitive tasks, operators have more time to focus on proactive maintenance activities and issues that need human intervention to resolve.

- Customizable probable-cause analysis:

  After effective event correlation and automation have done their work, the next step is to calculate the most likely probable causes from what's left. AIOps does this by applying a graph analysis algorithm to the underlying topology that relate to the events, then does a keyword analysis of the related events. The probable cause analysis is customizable so that certain keywords can be earmarked for special treatment when they come up.

- Topology:

  AIOps comes with a library of out-of-the-box *Observers* designed to connect to specific sources of topology data, as well as generic Observers designed to read data in from a REST API or from a file. Different topologies can then be stitched together and enhanced with custom icons and tooling to provide visualisation of a connected environment, regardless of where the topology data has come from. Events are then overlaid over the topology providing a visual context to a developing problem. Finally, the topology can be leveraged by the grouping and probable-cause analysis algorithms to infer relatedness amongst events and prioritize actions.

- Metric Anomaly Detection (MAD):

  AIOps ingests your time-series performance metric data and applies several different ML algorithms to learn what "normal" looks like for each one. After a training period of about 2 weeks, it will perform automatic baselining for each metric stream and then continue to track each one. If MAD sees any metric stream stray outside of its normal range, it will generate an alert to notify operations that there may be a problem. These anomaly events, as they're known, are then correlated with other events via the methods described above, and can add additional insights into any developing problem. Indeed MAD anomaly events frequently provide an early warning of a "real" issue before it happens. By following up any detected anomalies, operations can proactively investigate and take steps to avoid outages.

- Change risk:

  There are hundreds of changes that affect a service during its lifecycle. Change risk is an unsupervised learning algorithm that takes historical data from tickets and helps you determine how likely it is that a given change would cause a problem. This determination is based on how successful that similar change was deployed in the past. Using the assessment score provided by change risk, you can determine how safe it is to proceed with the change.

- Log anomaly detection:

  Log anomaly detection is an unsupervised learning algorithm that takes large amounts of log data and trains on it to learn what is normal behaviour for a given component. It uses natural language processing of log messages to find patterns and analyze their frequency. After it has acquired sufficient data for training on, and that training has been completed, AIOps will continue to monitor the log data coming in for anything anomalous. If detected, AIOps will alert the operations team of its findings.

- Similar ticket analysis:

  When an incident occurs, it can be helpful to review details for similar incidents to help determine a resolution. *Similar tickets* is an unsupervised learning algorithm that aggregates information about similar messages, anomalies, and events for a component or a service. It can also extract the steps used to fix previous incidents, if documented. It does this by connecting to your ServiceNow instance and analyzes the existing tickets therein. After training has completed, AIOps will automatically alert users working on an incident to any tickets raised in the past that are similar.

Suggested approach

AIOps has a wide array of capabilities and can provide immense value. Whether you are an existing Netcool customer or not, it can be daunting to know where to start, given the large array of capabilities at your disposal. What you start with will however, to a large extent, depend on the sources of data available. This section provides a recommended high-level approach to which of these many capabilities can provide the quickest value, in a logical and structured order, based on the typical data sources, and capabilities that deliver quick value.

The suggested approach outlined in this section is based on the event sources that you want to integrate into AIOps, and the priority of those sources. The most common primary source of data tends to be events, followed by topology, then metric data, and finally log file data. This order therefore forms the basis for the suggested approach for implementation. More detail regarding the items listed below can be found in *Chapter 4 Implementation*.

EVENTS

The most common source of data is events, and so this is the natural starting point for most. At a high level, the following activities are recommended to help get you off to a quick start:

- Configure all your event integrations;
- Configure any event enrichment;
- Create any right click tooling;
- Create some "quick win" runbooks to automate event resolution;
- Configure scope-based event grouping;
- Enable seasonal event detection and temporal event grouping;
- Configure probable-cause analysis;
- Configure event housekeeping;
- Implement some event storm controls;
- Implement any key integrations with third-party systems such as ticketing.

TOPOLOGY

- Consider event sources and monitored services
- Identify relevant sources of topology data;
- Use off-the-shelf Observers where possible and generic ones otherwise;
- Define merge rules to connect topology data sets;
- Create any desired custom icons;
- Define match rules to map incoming event stream to topology resources;

- Define tag rules to create tags that may be useful;
- Create topology Resource group templates.

METRICS

- Identify key systems where time-series performance metric data is available;
- Use off-the-shelf AIOps Connectors where possible;
- Identify conduits on the metric data targets from which to extract the data;
- Implement scripted integrations to pull the metric data from the targets and push into Metric Anomaly Detection's (MAD) ingestion API.

LOG FILES

- Identify key systems where log data is available;
- Use off-the-shelf AIOps Connectors;
- Enable selected log anomaly capabilities as desired.

SIMILAR TICKETS ANALYSIS

- Set up an integration to *ServiceNow*;
- Configure *Similar tickets* in the AI model management view;
- Schedule training and deploy.

CHANGE RISK ANALYSIS

- Configure *Change risk* in the AI model management view;
- Schedule training and deploy.

Chapter 2 Planning

When preparing for a deployment of IBM Cloud Pak for AIOps 4, careful planning must be undertaken to ensure all necessary factors have been considered before provisioning both the software and hardware components, and designing the architecture layout.

This section goes through the main factors that should be considered when planning an IBM Cloud Pak for AIOps 4 solution and gives guidance on how to deploy a sensibly sized solution that will work for your needs.

This chapter covers the following aspects of planning your deployment:

- Compiling a list of the data sources and the estimated quantities;
- Guidance on provisioning of Netcool components;
- Deployment architecture considerations;
- OpenShift sizing and configuration;
- Consideration for the historic event archive;

An initial requirements gathering checklist is provided in *Appendix A* to help with the collation of this information.

Data sources and rates

This section lists the potential sources of input data to consider and collate. This information is used as inputs into the sizing utility to calculate the CPU, memory, and storage requirements, which is covered later in this chapter. These inputs will be fed into the AIOps Sizing Utility which will then suggest a suitable hardware sizing specification.

SOURCES OF INPUT DATA

The following list outlines the initial inputs that should be gathered in relation to input data:

- *Event data sources*: make a list of all the different sources of event data that AIOps will be receiving. Examples include: traps, syslogs, events from an EMS, events from a monitoring system like APM, events from a third-party system via an API, and events received via a webhook. This list of items collated here will define which event receiver components will be needed during the deployment.
- *Estimated event rate*: the expected event rate is a key input used in the sizing of an AIOps deployment. It is defined as the estimated total event rate from all event sources and is expressed in events per second.
- *Estimated maximum number of standing events*: while event rate affects our ability to receive and ingest events into AIOps, another key metric is the estimated maximum number of events AIOps will have at any one time. Events will flow into the system, some will clear, and some will be expired out. Despite the churn, in all cases there will be a "typical" number of events the system will have at any one time. This input should be an estimate of what that number is.
- *Topology data sources*: make a list of all the different sources of topology data that AIOps will be receiving. Also include details of any resources that are common between different topology data sources. This information will be used to merge the topology later.
- *Estimated number of topology resources*: the expected number of topology resources is a key input used in the sizing of an AIOps deployment. It is defined as the estimated total number of nodes, servers, elements, etc. from all topology sources.

- *Metric data sources*: make a list of all the different sources of metric data that AIOps will be receiving. This information will input into the deployment phase when it will be determined how best to get the data in to AIOps.
- *Estimated metric rate*: the expected metric rate is a key input used in the sizing of an AIOps deployment. It is defined as the estimated total number of metric data points received every 5 minutes. Note that a 5-minute interval is optimal for metric data ingestion.
- *Log data sources*: make a list of all the different sources of log data that AIOps will be receiving. This information will input into the deployment phase when it will be determined how best to get the data in to AIOps.
- *Estimated log rate*: the expected log rate is a key input used in the sizing of an AIOps deployment. It is defined as the estimated total number of logs received per second.

CONCURRENT USERS

Estimate the maximum number of concurrent users the AIOps deployment is likely to have. This input is taken into consideration by the sizing utility in the overall hardware calculation.

PROVISION FOR DATA STORMS

It is prudent to add a continency to any maximum estimates to allow the system to be sized to handle data storms. A data storm can happen when there is a major outage, for example a significant weather event that affects a large part of the managed environment, and AIOps receives a surge in events and logs.

For the estimated maximums gathered above, it is suggested to add a 20% uplift.

INTEGRATION WITH THIRD-PARTY SYSTEMS

It is typical that Operations Management Systems have integrations with third-party systems, such as ticketing systems. It is important therefore that any such integrations are captured during the planning phase so that the best integration method can be defined during the implementation phase. The worksheet in *Appendix A* includes a space to enter these.

Netcool component deployment considerations

It is highly likely that an AIOps deployment will incorporate an existing Netcool deployment or will include Netcool components as part of the solution. Some of the inputs gathered in the previous section will be used to specify if Netcool/OMNIbus or Netcool/Impact are needed in the deployment.

Included in the Best Practices series are guides for both Netcool/OMNIbus and Netcool/Impact. These guides should be used to assess the inputs acquired in the previous section and to set out a deployment strategy and sizing for these components.

Note: Where AIOps is to be deployed in conjunction with an existing Netcool deployment, the Netcool components will remain unchanged from an architecture perspective but there will necessarily be some modifications to function and automation. These details will be covered later in a section.

Consideration as whether Netcool components are needed for an AIOps deployment will come down to three main factors:

- What the event sources are;
- What the custom automation and integration requirements are;
- What event handling and performance characteristics are required.

EVENT SOURCES

AIOps provides for several Connectors that enable event collection from a variety of sources. Netcool/OMNIbus however provides a collection of more than 140 Probes that enable the ingestion of events from a myriad of different sources. Due to the diversity of the event estate in most environments, it is highly likely that Netcool Probes will be part of any deployment.

Note: A complete listing of all Netcool Probes is here: https://ibm.biz/netcool_probes

A common requirement for the collection of events is from a third-party API. Where AIOps does not have an off-the-shelf Connector for an event source of this type therefore, Netcool/Impact can be used to retrieve the events.

Note: If Netcool/OMNIbus is also deployed, one option is to insert the events retrieved by Netcool/Impact into the on-premise ObjectServer, and then let the AIOps Netcool Connector pick those events up from there.

CUSTOM AUTOMATION AND INTEGRATION

It is a common requirement to perform custom automations or integrations with third-party applications, for example: implementing event housekeeping policies, performing event enrichment, or creating custom ticketing integrations. In all three cases, Netcool/Impact will likely be needed to meet the requirement.

EVENT HANDLING AND PERFORMANCE

Some AIOps deployments will need to handle extremely high event rates, and most will need to have robust processes in-place to handle event storms. Netcool components have built-in capabilities for doing both and are typically deployed and configured in a manner to do so.

The Netcool/OMNIbus 8.1 Best Practices guide covers these topics in detail, and outlines a strategy for deployment and configuration to both optimize the configuration of a deployment architecture as well as the Probes to ensure optimal throughput.

The following sections cover relevant information for understanding and setting up a Netcool/OMNIbus deployment to cater to high loads and event storm scenarios:

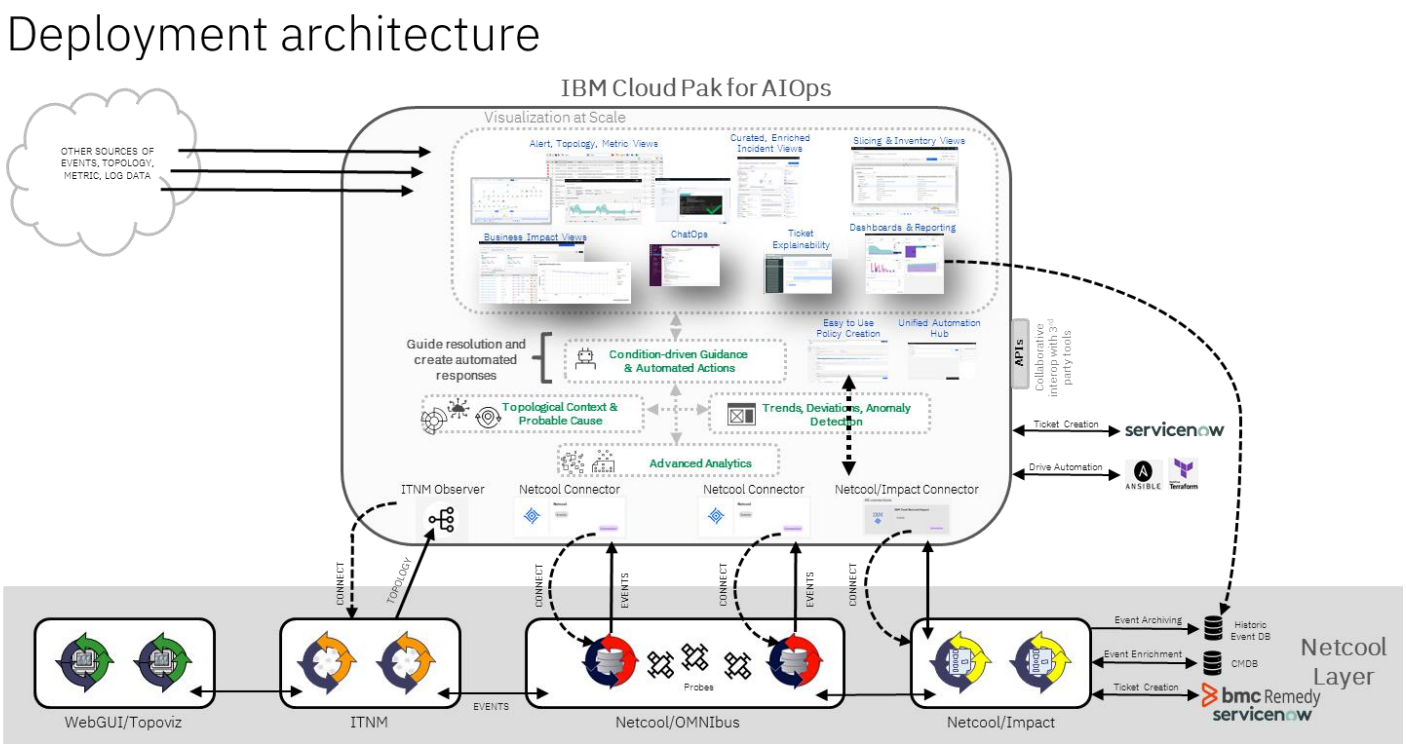
- *Multitiered deployment: the purpose of each tier*
- *Solution sizing and design based on requirements*
- *Architectural considerations*
- *ObjectServer housekeeping*
- *Event flood control*
- *[Probe] Generic Properties*
- *Detecting event floods and anomalous event rates*

Note: The Netcool/OMNIbus 8.1 Best Practices guide and other Netcool Best Practices guides can be downloaded from: https://ibm.biz/nco_bps

Deployment architecture

Most deployments of AIOps will be in an environment where existing Netcool components are already deployed, and some will be so-called "green fields" deployments where no Netcool components are deployed. In any case, the deployment model will be the same: AIOps deployed onto an OpenShift cluster, with potentially Netcool components deployed beneath. The deployment of any Netcool components will depend principally on the data input and automation requirements of the overall solution.

An AIOps deployment will typically look something like the following:



NOTES:

- AIOps is deployed in the primary data centre on OpenShift. The OpenShift platform provides High Availability natively via its architecture;
- Where possible, use a native AIOps Connector to bring data into AIOps. Where one doesn't exist, consider using a Netcool component instead and then connect the Netcool components into AIOps using one of the standard methods;
- If using Netcool/OMNIbus Probes, it is recommended to deploy an Aggregation pair of Netcool/OMNIbus ObjectServers for the Probes to feed their events into. Not only does this result in a cleaner integration from AIOps to Netcool via the Netcool Connector, it also gives many automation options in the ObjectServer, such as event storm handling or other custom functionality possibilities;
- Any needed Netcool components should be deployed in a georedundant fashion, where possible, to ensure the highest level of resiliency possible. Netcool components should always be deployed in multiples at least, to ensure a minimum level of resiliency. See the Netcool Best Practices guides for more information on Netcool deployment: https://ibm.biz/nco_bps;

- A Netcool Connector instance should be set up for each Aggregation ObjectServer pair present. The AIOps schema should be normalized in AIOps via the mapping in the Netcool Connector;
- Event archiving to the REPORTER database can be done either directly from Netcool/OMNIbus, or via AIOps feeding events to Netcool/Impact, which would then insert the events into the event archive directly. Sending events directly to the event archive from Netcool can help to reduce the load on Impact integration and spread the processing load;
- Where IBM Tivoli Network Manager (ITNM) is present, it should be deployed in a resilient manner and connected to a Netcool/OMNIbus ObjectServer. Any events it generates will then flow into AIOps via the Netcool Connector. An instance of DASH and Netcool/OMNIbus WebGUI with Topoviz installed will be needed to access Structure Browser views. Its topology can be brought into AIOps via the ITNM Observer;
- Any custom integrations, automations, or workflows done via Netcool/Impact can either be automatically initiated from the Netcool/OMNIbus Aggregation ObjectServer pair via an Event Reader, or from AIOps directly via the Netcool/Impact Connector;
- Any custom dashboards currently in-use in Netcool/OMNIbus WebGUI will remain in-place and can be continued to be used;

Note: A replacement for DASH custom dashboards and reporting is coming in a future release of AIOps in the form of an optional installable sub-component, namely *IBM Cognos Analytics*.

OpenShift sizing and configuration

This section contains recommendations around the planning of an OpenShift deployment onto servers, to support an AIOps deployment.

Recommended versions

OpenShift leverages Kubernetes, so it brings an uplift in Kubernetes APIs with each successive OpenShift version. This can potentially result in breaking changes to APIs that versioned releases of Cloud Pak of AIOps are developed for.

There are 2 main considerations to keep in mind when planning a deployment therefore:

1. For production, stick with even numbered OpenShift releases to get the maximum support length cycle;
2. Align the version of OpenShift with Cloud Pak of AIOps that you are deploying. For more information, see https://www.ibm.com/docs/en/cloud-paks/cloud-pak-aiops

Storage considerations

Storage for containerized solutions has different characteristics than traditional VM-based application deployments, and should be planned for according. The following tips should be adhered to in any deployment whenever possible.

ENSURE SUPPORT FOR REQUIRED ACCESS MODES

The storage solution needs to provide the required storage and access modes as specified by the application: https://ibm.biz/aiops_storage_reqs

RWX (ReadWriteMany) means a volume can be attached to multiple nodes simultaneously to allow multiple pods to read or write from it at the same time. Not all storage solutions however currently meet RWX capabilities in a uniform manner.

See the following link for more information on storage access modes:

https://kubernetes.io/docs/concepts/storage/persistent-volumes/#access-modes

For example, when using the vSphereVolumes CSI driver, RWX is supported with the caveat that pods are collocated on a node. This may cause day-2 operations issues, especially if anti-affinity settings are needed, as then these two requirements would conflict.

ENSURE MINIMUM IOPS

The product documentation specifies the minimum required IOPS required by AIOps: https://ibm.biz/aiops_iops_reqs

If these minimum IOPS requirements are not met by the underlying infrastructure, it can negatively impact the Cloud Pak for AIOps application performance and result in slower processing and user experience.

USE DEDICATED STORAGE NODES

For production, it is recommended to deploy any SDS storage nodes as dedicated nodes in your cluster, which will run *only* the storage software.

This can significantly improve pod start-up times and other pod lifecycle operations, as well as free up valuable compute node resources to dedicate to application pods for optimum performance.

S3-COMPATIBLE

For the backup and restore function, an AIOps deployment will require an S3-compatible Object Storage solution to store the backups. It is good practice to ensure these backups are stored off-cluster in case a full restore is later required.

AVOID SHARED STORAGE

Using shared underlying storage that services many other applications and services is typically not recommended.

The AIOps analytics pipeline software requires significant and dedicated compute and storage resources with guaranteed performance to be able to process heavy volumes of production data.

SSD RECOMMENDED

Use of SSD with PCI-Passthrough to the underlying NVM devices is the best option for optimum storage performance.

STORAGE NODES

A recommended specification for storage nodes is 16 vCPU with 64GB RAM. This will make expansion of the storage system easier without needing to reconfigure.

SCALE OUT

Scale-out (ie: add more) storage nodes rather than scaling-up (ie: increase the size of) existing storage nodes to protect against performance issues relating to node-level IOPS limits.

This approach is also recommended for day-2 operations, machine sets, and machine pools.

VSAN CHARACTERISTICS

If using vSAN storage, consider its characteristics: is it block backed? What IOPS and latency can we expect? Is it sufficient to run AIOps? Refer to the product documentation for the IOPS requirements: https://ibm.biz/aiops_iops_reqs

DEDICATED STORAGE FOR CONTROL PLANE

Ensure that you provide dedicated, fast storage for Control plane nodes (*etcd* cluster):

https://docs.openshift.com/container-platform/4.12/scalability_and_performance/recommended-performance-scale-practices/recommended-etcd-practices.html

"Avoid NAS or SAN setups and spinning drives. Ceph Rados Block Device (RBD) and other types of network-attached storage can result in unpredictable network latency. To provide fast storage to etcd nodes at scale, use PCI passthrough to pass NVM devices directly to the nodes."

ENABLE VOLUME EXPANSION

It is recommended to enable volume expansion on your persistent storage, or ensure your chosen storage solution supports resizing on-demand.

If certain components use more storage than was originally sized for and volumes become full, filesystems can become inaccessible or go into read-only mode, causing day-2 operations issues and outages. Recovering from these situations requires an advanced Kubernetes/OpenShift skillset and can cause significant operational challenges. Having the ability to resize volumes on-demand in-place however allows you to both monitor available volume capacity and react proactively to add capacity as and when needed, with zero application downtime.

See the following link for further information: https://ibm.biz/aiops_vol_expansion

Networking

Be aware of the following OpenShift default network CIDRs. Network overlaps can cause networking problems when setting up integrations post-install.

- Pod network (also known as Cluster Network):

See table 9: https://docs.openshift.com/container-platform/4.12/installing/installing_vsphere/installing-restricted-networks-installer-provisioned-vsphere.html#installation-configuration-parameters-network_installing-restricted-networks-installer-provisioned-vsphere

- Service network:

See table 9: https://docs.openshift.com/container-platform/4.12/installing/installing_vsphere/installing-restricted-networks-installer-provisioned-vsphere.html#installation-configuration-parameters-network_installing-restricted-networks-installer-provisioned-vsphere

Internal SDN network:

See table 4: https://docs.openshift.com/container-platform/4.12/networking/cluster-network-operator.html#nw-operator-cr-cno-object_cluster-network-operator

It is also recommended to have a full plan network diagram of what will be integrated with AIOps. Use this preparatory information to plan your OpenShift network configurations accordingly.

Note: When updating OpenShift, pay close attention to any changes to default SDN. Changes in SDN CIDR can break network communication for integrations outside the cluster.

Virtual compute resources

It is a common practice, especially when managing virtual servers, to over-commit certain underlying physical resources – eg. CPU or storage capacity. In standard virtual server-based environments, this can optimize resource and cost management.

Given the business-critical nature of the problem domain AIOps is intended for however, and the potential for high data throughput, guaranteed resource commitment from the underlying hardware is essential. When hardware resources are over-committed, virtual servers "believe" they have more resource available than they actually do. This, in turn, causes OpenShift to believe there are more resources per node than there actually are. This can result in degraded processing performance and user experience, especially during a period of high load, such as an event storm.

The following are recommended best practice configurations for running OpenShift on virtual server infrastructure, to ensure AIOps software components always have the required resources they need, at all times.

CPU AND MEMORY

- Ensure there is a 1:1 mapping configured between vCPU and underlying physical core:
  - For VMWare, set *sched.cpu.latencySensitivity* to *high*
    Doc link: https://docs.vmware.com/en/VMware-vSphere/8.0/vsphere-vm-administration/GUID-F5DAC793-7B13-418F-949E-0CD80CEC5D1E.html
- Ensure 100% memory reservation is configured:
  - For VMWare, set *Schedule.mem.min* to *memsize*

STORAGE

Ensure volume expansion is supported by your chosen storage provider and is enabled for your storage classes used for AIOps. These settings can result in significant processing improvements for production systems.

See the following link for further information: https://ibm.biz/aiops_vol_expansion

Node roles

The following OpenShift node roles should be planned for in a production deployment:

- Control plane or master nodes (minimum of 3);
- Compute or worker nodes (to run the AIOps application Pods);
- Storage nodes (minimum of 3);
- Infrastructure nodes (minimum of 3).

NOTES:

- The number of worker nodes is determined by using the Custom Sizing Tool;
- Compute nodes should be *tainted* to prevent non-AIOps workloads from running on them, to ensure that AIOps always has the necessary resources it needs;
- Storage nodes should be *tainted* to prevent other workloads from running on them;

- Infrastructure nodes are used to run the OpenShift layer 7 application load balancer and router component, thereby freeing up compute node resources and network I/O for the AIOps application;
- Infrastructure nodes also allow for day-2 enabling of the OpenShift monitoring stack for better cluster management and visibility;
- Infrastructure nodes should be labelled and *tainted* for infrastructure component use only. The label naming can be user-specified.

Hardware recommendations

AIOps hardware recommendations can be found here: https://www.ibm.com/docs/cloud-paks/cloud-pak-aiops/latest?topic=planning-hardware-requirements

Custom Sizing Tool

The Custom Sizing Tool (CST) is provided by IBM to enable for the suitable sizing of an AIOps deployment. It takes all the relevant data handling inputs and produces a sizing big enough to handle the processing load and storage requirements.

After you have the resulting output, calculate how many virtual machines you'll need. It is recommended to add on an additional two or three worker nodes so that your cluster can comfortably handle the loss of worker nodes, and so there is enough capacity for facilitating upgrades to OpenShift or AIOps.

STORAGE NODES

In addition to the number of worker nodes provisioned for AIOps, it is strongly recommended to deploy three worker nodes and use them exclusively for storage purposes. If ODF is to be used, each of these three storage nodes should be provisioned with at least as much disk that the CST recommends is needed for AIOps. This is because ODF replicates the data across all three nodes for resiliency purposes. Hence the amount of physical storage will be three times the amount specified by the CST. It is advisable to add on a contingency as well. A suggestion is to uplift the storage requirement stated by the CST by 20%.

Prerequisite checker

After standing up the OpenShift cluster, run the prerequisite checker to ensure you have sufficient resources to deploy your AIOps instance. The prerequisite checker can be downloaded from here: https://ibm.biz/aiops_prereq_checker

Historic event archive

Most existing Netcool deployments have a requirement to store historic event data. Normally this is implemented using a Netcool/Gateway for JDBC writing out to a traditional disk-based database system, such as DB2. The schema for the storage of the historic event data and the SQL files needed to build it are provided by IBM in the reporting scripts bundle.

Current Netcool users and new AIOps users alike will likely have the same requirement to store historic event data. As outlined in a previous section, this can be achieved in one of three ways:

1. Connect a Netcool/Gateway for JDBC to your Aggregation ObjectServer pair;
2. Send events from AIOps via an Automation Policy to Netcool/Impact via the Netcool/Impact Connector, and use an Impact policy to write the events to the target database using a Netcool/Impact Data Source Adapter (DSA).

3. A combination of these, ensuring that the events each one is sending don't overlap.

NOTES:

- Using a combination of these two methods will help balance the load over both;
- Note that events that originate directly within AIOps that do not come via the Netcool route will need to be sent to the historic event archive via the Netcool/Impact Connector route, since events in AIOps aren't by default propagated back to the Netcool layer;
- Housekeeping guidance for the historic event archive can be found in the Netcool/OMNIbus 8.1 Best Practices guide, available from: https://ibm.biz/nco_bps.

Chapter 3 Deployment

This chapter covers considerations and recommendations around the deployment of AIOps. Since the deployment of AIOps itself is well documented in the product documentation, this section instead focuses on the deployment and configuration of any peripheral components, such as on-premise Netcool components or any other integrations.

Provisioning AIOps

AIOps installation documentation can be found here: https://ibm.biz/aiops_install

Netcool components

The necessity to include Netcool components in an AIOps deployment will depend on the requirements outlined in *Chapter 2 - Planning*. Where Netcool components are needed, their deployment should be governed by the best practice methods outlined in the *Netcool/OMNIbus 8.1 Best Practices* and *Netcool/Impact 7.1 Best Practices* guides respectively.

Some key points of note:

- All Netcool components should be deployed with the latest fix pack applied;
- All components should be deployed in a resilient manner;
- All components should be configured to auto-start on machine boot.

Note: Netcool Best Practice guides can be found here: https://ibm.biz/nco_bps

Migration from an existing Netcool deployment

Most AIOps deployments will be in the context of migrating from a traditional on-premise Netcool deployment. As such, how to migrate and integrate with the existing deployment is a major consideration. This section contains guidance on how to integrate an existing Netcool deployment to leverage its strengths and build on your existing investment.

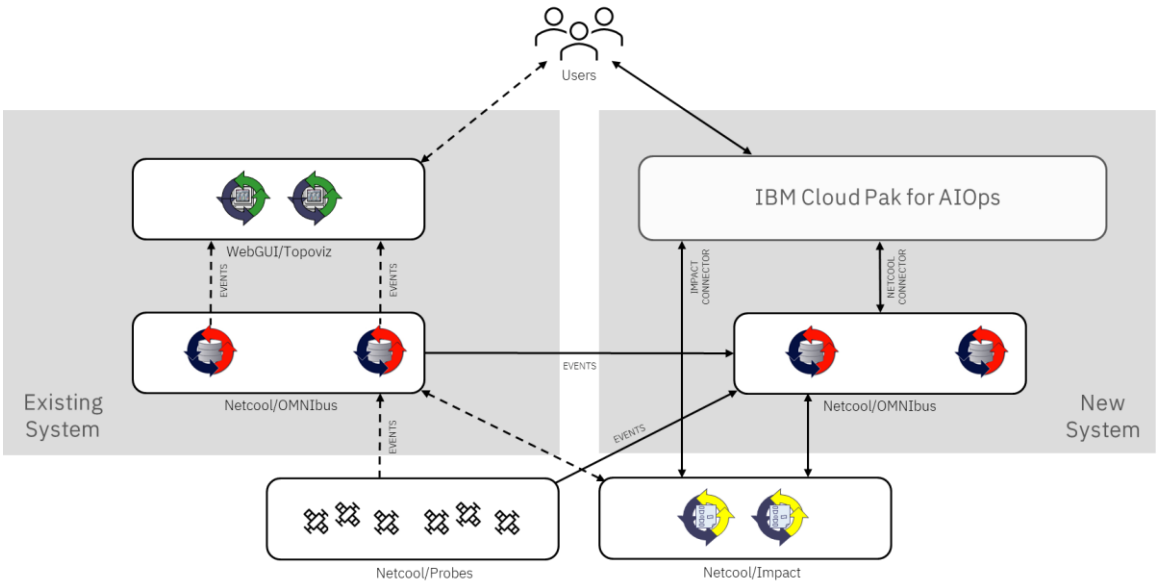
Existing Netcool deployments will likely include Probes, ObjectServers, Gateways, WebGUI, and Impact. These components provide various capabilities, some of which can and should be retained, and others that are superseded by newer AIOps capabilities. How much functionality that is moved up into AIOps is up to the deployment engineer.

This section contains a suggested general approach to migration from an existing Netcool deployment to one that includes AIOps. The suggested approach is based on existing methodologies used previously to upgrade production Netcool environments. This document provides two different approaches to migration: one where an upgrade to the ObjectServer hardware at the same time is desired, and one where the existing Netcool components should remain as-is (*in situ*).

Note: It is assumed that any setup and configuration items in the new system, and the migration process, are tested thoroughly in a pre-production environment prior to performing any such migrations in production.

This section also goes through the Netcool component types and provides additional considerations that may apply.

MIGRATION WITH NEW NETCOOL/OMNIBUS OBJECTSERVER HARDWARE



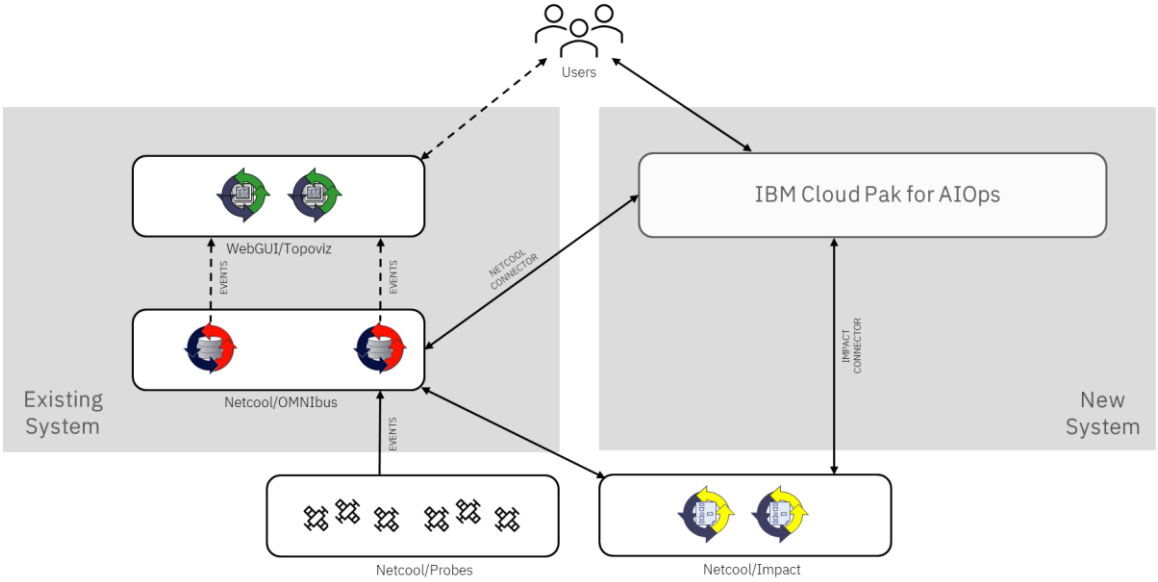
A suggested high-level approach to production migration is as follows:

- Deploy AIOps on OpenShift;
- Configure any authentication settings within AIOps – eg. corporate LDAP;
- Deploy a new on-prem Aggregation ObjectServer pair (using multitier configuration) corresponding to each Aggregation ObjectServer pair in the existing system;
- Recreate any needed custom fields in the new ObjectServer pair as well as any automation such as event storm protection, event housekeeping, or enrichment, that will stay within the Netcool layer. Note that this excludes any correlation type functions which will need to be moved to the AIOps layer;
- Create and configure a new AIOps Netcool/Impact Connector instance, as well as new AIOps Netcool Connector instances for each of the new Aggregation ObjectServer pairs, ensuring to map any additional fields in the ObjectServer to AIOps attributes[1];
- Set up a unidirectional Netcool/OMNIbus ObjectServer Gateway for each new Aggregation ObjectServer pair, to feed it events from the corresponding Aggregation ObjectServer pair in the existing system. Filter out any correlation fields in the mapping, so that the new system does not inherit any correlation from the old system;
- Configure all correlation functions, UI tooling, and new automation in AIOps;
- Configure any needed integrations with external systems in the new system;
- Perform checks to ensure the new system is ready for use and functioning as expected;
- Move users over to new AIOps UI on the new system;
- Edit Netcool/Impact configuration to point to the new ObjectServer pair;
- Reconfigure the Probes to instead connect to the new ObjectServers;
- Reconfigure any other items to connect to the new ObjectServers instead;

[1] Additional ObjectServer fields should be mapped to sub-attributes of *resource* in AIOps – eg. *resource.myfield*.

- Decommission the existing system ObjectServers and WebGUI servers.

MIGRATION IN SITU



A suggested high-level approach to production migration is as follows:

- Deploy AIOps on OpenShift;
- Configure any authentication settings within AIOps – eg. corporate LDAP;
- Disable any correlation type automation at the OMNIbus layer. Any automation such as event storm protection, event housekeeping, or enrichment, can remain within the Netcool layer however;
- Create and configure a new AIOps Netcool/Impact Connector instance, as well as new AIOps Netcool Connector instances for each of the Aggregation ObjectServer pairs, ensuring to map any additional fields in the ObjectServer to AIOps attributes[2];
- Configure all correlation functions, UI tooling, and new automation in AIOps;
- Move any integrations to AIOps, where desirable;
- Perform checks to ensure the system is ready for use and functioning as expected;
- Move users over to new AIOps UI;
- Decommission the existing system WebGUI servers.

Netcool/OMNIbus Probes

There are several event integrations provided by AIOps that are set up and maintained via the AIOps GUI. Where a built-in target integration is required, it is suggested to use the AIOps integration. However, where the integration has special custom configuration needs, or where an AIOps integration does not exist, a Netcool/Probe will likely be needed. Custom configuration may include enrichment via lookup tables, event storm protection functions, or custom logic with respect to event handling.

[2] Additional ObjectServer fields should be mapped to sub-attributes of *resource* in AIOps – eg. *resource.myfield*.

Note that if the existing Probe integration is working well, and there is no advantage to switching to a native AIOps integration, there is no need to do so. Netcool Probes will be supported indefinitely by IBM and provide a convenient, robust, and efficient method for ingesting events. In general, Probes can be left where they are, and continue to operate as they have previously.

AIOps contains an embedded containerized ObjectServer which is there to accept in-bound connections from Netcool Probes or ObjectServer Gateways for the purpose of event ingestion. It is an option therefore to connect Probes directly to AIOps, however, there may be other processes that require the presence of an on-premise ObjectServer pair anyway. It is better in such cases to leave the Probes connected to the on-premise ObjectServer pair, as this provides the opportunity to retain many additional functions that can't be achieved without an on-premise Aggregation ObjectServer pair.

Summary information for migration to AIOps with respect to Probes:

- Netcool/Probes can remain and continue to provide service as before;
- Probe functions may be replaced by AIOps integrations where available, but certain custom functionality options may be lost – for example, custom data token processing in the Probe rules file is not possible to same extent in some AIOps integrations.

Netcool/OMNIbus ObjectServers

The Netcool/OMNIbus ObjectServer is the centre of any traditional Netcool deployment and is the primary repository where events are received and stored, and where event automation is generally configured.

Where ObjectServers are present in an existing Netcool deployment, one or more instances of the AIOps Netcool Connector should be used to bring that event set into AIOps. Netcool Connector instances should be connected to Aggregation ObjectServer instances. Display ObjectServers can be removed unless there is a particular reason why they are needed.

The AIOps Netcool Connector maintains synchronization of events between an ObjectServer pair and AIOps. The connection is principally one-way however from OMNIbus to AIOps: event inserts and updates are propagated from OMNIbus to AIOps but not the other way around. Clears and deletes however are the exception and are propagated in both directions: when events are cleared or deleted in OMNIbus, they are cleared and deleted in AIOps, and vice versa.

The ObjectServer is where integrations with external systems usually originate, whether that be via a Gateway or an integration with a script or Netcool/Impact. The ObjectServer also contains triggers which provide both IBM-provided and custom automation.

In addition to providing a deduplication function, ObjectServers are also routinely configured to provide event storm protection and event housekeeping functions, and a means to control the flow of events to user consoles.

The event storm function from an ObjectServer perspective is normally implemented at the Collection layer, to shield the Aggregation layer from sudden influxes of events. If these functions are already in-place at the Collection layer, they can remain and continue to provide that capability.

Event housekeeping functions are normally implemented at the Aggregation layer. A common housekeeping automation is one that enacts an event expiry policy as set by the business. In an AIOps deployment where ObjectServers are connected to via the Netcool Connector, these functions can remain, since the ObjectServer retains the primary copy of that event set which is replicated to the AIOps layer. When the events are expired out of the ObjectServer, they are also removed from AIOps.

AIOps provides three ways to correlate events: temporal-based, scope-based, and topology-based. Event groups are also further merged where events are members of more than one of these groups. Since the AIOps layer is the primary point where events are correlated, any existing event correlation automations implemented in the ObjectServer should be removed or deactivated, and the correlation implemented within AIOps instead. This includes any custom or IBM-provided automations.

Any custom automation that deals with updating individual events according to a set of criteria or logic can generally remain as it is, since any changes to the event at the ObjectServer layer will be propagated to the AIOps copy of the event via the Netcool Connector.

Outbound Gateways from ObjectServers can generally be retained however it must be considered that AIOps will likely have a bigger set of events that what an ObjectServer instance has, and most logic or updates made to events at the AIOps layer is not propagated back to the ObjectServer. Hence it should be considered in each case if the ObjectServer is still the best integration point.

Netcool/Impact is typically present in any Netcool deployment. Generally speaking, any current integration between OMNIbus with Impact can remain, such as event enrichment, or a generic event manipulation automation. Any event correlation type automation however, where synthetic parent events are created cannot easily be replicated up into AIOps layer, since AIOps does not create synthetic parent events in the same way that OMNIbus does. Such automations should be disabled therefore, and those functions re-implemented in the AIOps layer. Additionally, since ticketing is preferably based on groups of events rather than individual events, it usually also makes sense to move any ticketing integration to the AIOps layer. Note that Netcool/Impact may still be the way the ticket integration is implemented, however the initiation will be moved to the AIOps layer and executed via the Netcool/Impact Connector instead.

Summary information for migration to AIOps with respect to ObjectServers:

- Netcool/OMNIbus ObjectServer stacks can remain in-place and continue to be a source and repository of events for AIOps;
- An AIOps Netcool Connector instance should be created for each Netcool/OMNIbus ObjectServer stack, connecting to the Aggregation pair;
- Collection ObjectServers can continue to be used as they are, however, Display ObjectServers can be removed, except in specific circumstances[3];
- Custom and IBM-provided automations may remain except for ones that correlate events. This function has moved to the AIOps layer and must be reimplemented there;
- Integrations with external systems may remain however each one should be considered as to whether it makes sense to keep it at the Netcool layer or move it to the AIOps layer.

Netcool/OMNIbus Gateways

Netcool/Gateways are conduits by which events flow out of Netcool/OMNIbus ObjectServers. When deploying AIOps, each Gateway integration should be assessed as to whether it is the best integration point in the context of AIOps.

Unidirectional ObjectServer Gateways that forward events from the Aggregation layer to some other ObjectServer target can continue to be used. Other than the failover Gateway that connects an Aggregation pair, it is not recommended to connect any bidirectional

[3] See the Netcool/OMNIbus 8.1 Best Practices guide for more information about ObjectServer tiers.

ObjectServer Gateways to the Aggregation pair. Note that this is a best practice that predates AIOps.

Event archiving Gateways (JDBC) are used to forward a historic record of the event estate to a disk-based database such as DB2 for long-term storage and reporting. While such Gateways are an efficient way to copy the events to the archive, since correlation and many other automation functions are done at the AIOps layer, those updates will not be saved in the archive, as the Netcool Connector does not propagate event updates from AIOps back to the ObjectServer. It may be however that only a subset of the event estate is sent up to AIOps, and the remainder is only kept for archiving purposes. Such events can and should be sent for archiving via the existing JDBC Gateway. The control of which events get sent to either target can be controlled with flag fields.

Netcool ticketing Gateways can continue to be used. Logic to initiate the creation of a ticket for an event from the AIOps layer can be flagged in the ObjectServer's copy of the event via Netcool/Impact and the Netcool/Impact Connector. The ticket number can then be passed back to AIOps via the Netcool Connector.

Other Gateways like the Message Bus Gateway or the Socket Gateway can continue to be used.

Summary information for migration to AIOps with respect to Gateways:

- Each Gateway integration should be assessed as to whether it is still the best integration point in the context of AIOps;
- Unidirectional ObjectServer Gateways to other ObjectServers can continue to be used;
- Bidirectional ObjectServer Gateways should not be connected to Aggregation pairs, except for the failover Gateway that connects a primary Aggregation ObjectServer to a backup;
- Event archiving Gateways can continue to be used in certain circumstances;
- Ticketing Gateways can continue to be used however consideration should be given as to where the creation is initiated;
- Other outbound Gateways like Socket or Message Bus can continue to be used.

Netcool/OMNIbus WebGUI

Netcool WebGUI is the traditional UI for Netcool deployments and needs to be retained if any parts are being used for which there is not an AIOps replacement, such as the ITNM Structure Browser.

Since correlation processes are moved to the AIOps layer and those correlations are not by default propagated back to Netcool/OMNIbus, any event grouping will not be seen at the Netcool layer. As such, the *Event Viewer* should not generally be used, and users should instead use the newer *Alerts viewer* in AIOps.

RIGHT-CLICK TOOLS

Netcool/OMNIbus WebGUI supports the following tool types: CGI, SQL, Command-tools, and JavaScript scripts. AIOps by comparison, supports the following tool types:

- HTTP: send an HTTP request to a web service;
- Client-side: launch a web page from the browser;
- SSH: use the configured SSH provider to run a script remotely.

Since the tool types are different, there will necessarily need to be some reimplementation of the way right-click tools do their tasks. Also, since there is no migration tool to translate Netcool tools into AIOps tools, they will all have to be recreated in any case.

FILTERS AND VIEWS

The filters and views in AIOps provide equivalent functionality to those in Netcool WebGUI systems. These all need to be recreated in AIOps during a migration.

CUSTOM DASHBOARDS

IBM Cognos Analytics is coming to AIOps in Q1 2024 and will provide a rich dashboarding and reporting capability. Existing dashboards must be recreated in AIOps and will benefit from a new, modern look and feel, with new features not available in DASH.

Netcool/Impact

Netcool/Impact is typically present in any Netcool deployment and is used for a myriad of different purposes from event enrichment to integrations with third-party systems. In an AIOps context, it can continue to be used to provide automation to Netcool/OMNIbus ObjectServers or to AIOps directly via the Netcool/Impact Connector.

There are two primary ways that Impact policies are executed: either via an *Event Reader Service* or via a *Policy Activator Service*.

EVENT READER SERVICES

An Event Reader Service is designed to continuously read data from a source (usually an ObjectServer) and perform some action on it. This might be a one-time action, or it might be a process that is carried out multiple times. The Event Reader works by connecting to a source of events and pulling events from that source either based on its unique serial number (process once) or its last updated timestamp (process multiple times).

In an AIOps context, the Event Reader Service will only change if the policies to be processed are to be executed from AIOps directly via the Netcool/Impact Connector. In brief, the changes are as follows:

- References to an event's fields will change: ObjectServer field names will need to be changed to AIOps event attributes instead;
- An Event Reader service that calls multiple policies will have to call a single policy instead that calls those multiple policies. Any chaining logic will have to be encoded in the higher-level policy;
- The syntax for writing the back event updates to AIOps will be slightly different.

There are benefits to executing policies directly from AIOps as it uses a push model instead of a pull model. AIOps automations are configured to push events to Impact for execution against a policy instead of Impact querying the ObjectServer. This makes for more timely and efficient execution of such automation.

POLICY ACTIVATOR SERVICES

A Policy Activator Service is designed to execute an Impact policy every defined number of seconds. If the Impact policy being executed via the Policy Activator Service is performing some sort of correlation in the ObjectServer, then this must be disabled and reimplemented at the AIOps layer. In most other scenarios, no modifications to the policy or the Service are necessary, and they can remain performing their respective tasks against events in the ObjectServer.

IBM Tivoli Network Manager (ITNM)

Network Manager is a powerful network discovery and monitoring tool that has been and continues to be used widely around the world, for the discovery and monitoring of IP networks. Where it is part of an existing Netcool deployment that is to be carried over to an AIOps context, the following points should be considered:

- The configuration and setup of ITNM can remain the same when AIOps is added to the environment;
- The Aggregation ObjectServer pair ITNM connects to needs to remain in-place, and the ITNM Gateway will continue to connect to it, read and analyze events, and generate polling events. This ObjectServer pair will link in to AIOps via a Netcool Connector connection instance;
- A DASH-based WebGUI server setup will need to remain to provide ITNM Structure Browser views. The WebGUI instance will continue to use the existing Aggregation ObjectServer pair for its primary event source;
- ITNM RCA correlation views will only be visible in the WebGUI Event Viewer as these relationships are not visible in the AIOps views. Such views can be imported into AIOps via the AIOps dashboarding toolbox, however.

Migration from Netcool Operations Insight

This section outlines additional steps required when a migration from an existing Netcool Operations Insight (NOI) deployment is needed. There are two main versions of NOI to be considered:

- The fully on-premise version where only on-premise Netcool components exist, and event analytics capabilities are provided via Netcool/Impact;
- The fully or partially containerized version where certain components run on OpenShift.

Legacy on-premise NOI

This version of NOI has only traditional on-premise Netcool components (Netcool/OMNIbus and Netcool/Impact). In addition to the tasks outlined in the section entitled: *Migration from an existing Netcool deployment*, the following tasks need to be completed:

- Uninstall Netcool Operations Insight extensions on your WebGUI servers;
- Uninstall Netcool Operations Insight extensions on Netcool/Impact.

Note: There is no migration mechanism to copy event analytics data from a legacy NOI system to AIOps. Instead, training should be re-run on the AIOps system to rediscover any Seasonal events and temporal groupings.

Further information regarding uninstalling legacy NOI components can be found here:

https://www.ibm.com/docs/en/noi/latest?topic=analytics-uninstalling-event

Containerized NOI

If you need to migrate from NOI to AIOps, you will need to set up AIOps on a new OpenShift cluster and the configurations manually recreated on the new platform. In this case, the following non-exhaustive set of tasks would need to be completed manually:

- Event Analytics in AIOps will re-train either from scratch or from historic data;
- Topology Observer jobs need to be reconfigured and rerun;
- Topology rules need to be recreated;
- Topology tooling needs to be recreated;
- Topology custom icons need to be recreated;
- Resource and relationship types need to be reconfigured;
- Topology business criticality needs to be redefined;
- Topology group template definitions need to be recreated;
- Scope-based grouping policies need to be recreated;
- Runbooks and all associated configurations need to be recreated;
- Event Viewer filters need to be recreated;
- Event Viewer views need to be recreated.

Stress testing prior to deployment into production

It is a longstanding best practice to stress test any environment prior to deploying into production. The following are some summary notes on this topic:

- Provision more hardware than you will need so that you can tolerate both the failure of at least one worker node as well as an event storm scenario;
- Stress test with an event/metric/log file rate of at least 20% more than the highest rate anticipated;
- Stress test with a concurrent user load of the maximum anticipated.

Chapter 4 Implementation

This chapter leads on from a successful deployment of AIOps and covers topics associated with implementation and use of the system post-deployment. It outlines a strategy for which features and capabilities to target in a suggested order to gain quick value and get off to a fast start. It starts off however by providing some notes on how to integrate an existing Netcool environment to the new AIOps platform, and answering the question of where best to implement the needed custom functionality.

In terms of product features and capabilities, the suggested implementation approach begins by considering events and the features related to event use-cases, as this tends to be the central focus of most existing operations environments. Next, topology and topology-related capabilities are brought in to provide additional context and capabilities. After this, metric data is considered, then finally, log data. Note that not all these data types will be present in all environments.

Finally, this chapter covers the topic of self-monitoring within the AIOps Cloud Pak and outlines some guidance of what key parts of the deployment should be monitored and how best to do it.

Where to implement custom functionality?

Before planning to implement custom functionality, consideration must be given as to where best to implement it. It is best practice to use any out of the box functionality to achieve the desired goal, where possible. For example, any event correlation scenarios should be implemented using scope-based or topology-based correlation methods.

The following table provides guidance on where to implement custom functionality:

Implementation point	**Notes**
AIOps Automations	AIOps comes with options for automation actions including: • Suppress alerts; • Group alerts based on a common attribute (scope); • Assign a runbook to an alert; • Promote alerts to an Incident; • Invoke a Netcool/Impact policy. If the custom functionality can be implemented using an AIOps automation, this is a very convenient option and should be considered first.
Runbook Automation	AIOps comes with an embedded Runbook Automation capability. There are many advantages to using Runbook Automation to implement custom functionality including: • Runbook Automation can integrate directly into Ansible Tower and execute any playbooks contained therein; • Runbook Automation can efficiently execute scripts on target systems, either directly or via a jump host; • Runbook Automation provides a convenient ecosystem in the UI that operations teams can access directly without involvement from the tooling or platform administration teams; • Runbooks can be authored by authorized operations team members and trialled by operators before being set to run fully automatically. Runbook Automation is an ideal place to implement a custom function where the function is or can be easily created in Ansible Tower or a script.

Implementation point	Notes
Netcool/OMNIbus Probe rules file	Many AIOps deployments will include Netcool/OMNIbus Probes as part of the implementation. The Probe rules file is where the incoming token stream is parsed and assigned to ObjectServer fields but gives opportunity to customize the contents of the alert or other functions, such as event storm protection, or the implementation of an event housekeeping policy. It is a best practice to implement as much functionality as possible at the Probe rules file level before events are sent to an ObjectServer or onwards into AIOps. This helps distribute the processing load down to the Probe endpoints. An example of this is where events that aren't needed can be dropped by the Probe directly. Note however that discarding events in at the Probe level is not always possible if, for example, a record of every event must be archived for legal reasons. In such scenarios, events can be sent to an ObjectServer, forwarded to a historical archive, and then deleted, without propagating up to the AIOps layer. Alternatively, these "archive only" events can also be rerouted from the Probe directly to a separate, dedicated ObjectServer pair, so as not to affect the performance of the main ObjectServer pair. The solution that is selected will depend on the event numbers and subsequent ObjectServer loading. Lookup files are useful at the Probe level for basic event enrichment so long as the data set is static. If the data in the lookup table is dynamic, it is better to store it in a database table (eg. ObjectServer or other external database) and then perform event enrichment via an ObjectServer trigger or via IBM Netcool/Impact. Note however that creating large custom tables may cause ObjectServer performance to suffer. Custom tables and triggers, like any other custom functionality, should be thoroughly tested before being put into production.
Netcool/OMNIbus ObjectServer database trigger	The ObjectServer database trigger is used for implementing actions based on changes to the event data set — for example: an insert of, reinsert of (deduplication), update to, or deletion of a row in an ObjectServer table. For example: a custom reinsert database trigger might be created on the *alerts.status* table to specify how a number of specified custom fields should be updated on deduplication. Care must be taken not to overload the actions of a database trigger as it may negatively impact on the ObjectServer's performance. Thorough testing and monitoring of database triggers in a test environment should be done to validate their performance however before deploying into production.
Netcool/OMNIbus ObjectServer temporal triggers	The temporal trigger is used for implementing actions on a regular, timed basis. Common tasks of temporal triggers include event housekeeping, event flood detection and mitigation, and general event life cycle actions. Note that any sort of event correlation where parent events are created and then linked to child events should be avoided, since the replication of such relationships to AIOps is difficult. It is better to keep all correlation activities at the AIOps layer. Thorough testing and monitoring of temporal triggers in a test environment should be done to validate their performance before deploying into production.

Implementation point	Notes
Netcool/Impact	Netcool/Impact is an extremely powerful event processing engine and can be used for a great number of purposes including: • Event enrichment: typically, where the target data resides on an external data source, such as a DB2 database or accessed via an API; • Any sort of data correlation or data processing (event driven or on a timed basis) on data contained either in the ObjectServer or another source — for example: a file or a RDBMS; • Interaction with external databases; • Interaction with web services; • Interaction with external APIs; • Invocation of an automatic process or interaction with a system that requires timed delays (via the *Hibernation* function). Where custom functionality cannot be implemented by a Probe rules file or ObjectServer trigger due to the target system being external, consideration should be given to use Netcool/Impact. Netcool/Impact can access events in an ObjectServer or can be pushed individual events from AIOps via the Impact Connector. The approach used will depend on the specifics of the use-case.

Getting started with events

Events are typically the primary source of data of any existing Netcool deployment or indeed any other kind of operations environment, hence it is a logical starting point for implementing a AIOps solution. This section contains a series of topics that pave the way to get started with events in AIOps, providing a logical order to follow. Note that some of the focus areas outlined can be done concurrently.

Event sources

The first activity in getting started with events is to connect all the event sources. There are typically multiple ways to integrate an event source however ideally each one will be done using the following options in order of precedence.

AIOPS EVENT INTEGRATIONS

AIOps comes with several built-in event integrations. It is recommended to use these wherever possible as a first option for any event integration. For example, if there is not an off-the-shelf option for an event integration, however the source of the events is able to send to a webhook, the generic webhook integration can be used to ingest the events.

A full list of incoming integrations is published here:
https://ibm.biz/aiops_incoming_integrations

NETCOOL PROBES

Where a built-in integration does not exist, the Netcool Probe library contains more than 140 Probes to retrieve or receive events from very many different sources. For example, any trap event sources can be consumed by an SNMP Probe.

A full list of Probes is published here: https://ibm.biz/netcool_probes

NETCOOL/IMPACT INTEGRATIONS

In some cases, an event source might have an API through which to retrieve the events. In this case, Netcool/Impact can be employed to retrieve those events. Netcool/Impact is also

commonly used to retrieve data from third-party systems for the purposes of event enrichment. The Netcool/Impact policy function *GetHTTP* is a key function for interacting with APIs.

A full list of Netcool/Impact functions is published here:

https://ibm.biz/netcool_impact_functions

Right-click tooling

AIOps has the capability to create context-sensitive tooling for the Alerts list view. Right-click tooling enables a user to take actions on alerts in their view. The following tool types are available in AIOps:

- HTTP: Send an HTTP request to a web service;
- Client-side: Launch a web page from the browser;
- SSH: Use the configured SSH provider to run a script remotely.

If there are any opportunities to provide users with tooling that will enable them to better investigate or resolve underlying issues, this task can deliver quick value to the business.

Note: Right-click tooling can be used to launch Netcool/Impact policies via Impact's APIs. See the Netcool/Impact documentation for the different API options available via the following link: https://ibm.biz/nci_apis

---

More details of how to work events in AIOps, including the creation of context-sensitive tooling options, is published here: https://ibm.biz/aiops_working_alerts

Runbook Automation

Runbook Automation (RBA) is a powerful capability in AIOps that allows for the linkage of the incoming event stream to a runbook with a view to attempting to correct the underlying problem, or perhaps to gather more information about the fault.

After the various event integrations have been set up, and events are flowing into AIOps, start to look at which alerts are candidates for runbook-enabled resolution.

Runbook Automation allows you to create runbooks that contain manual steps or automated ones. If a runbook contains any manual steps, then they will always have to be initiated and actioned by a person. If it contains only automated steps, then it has the potential to become full automated with no human intervention. It's these latter types of runbooks therefore that are of most value, as it allows for the evolution towards a more self-healing environment.

More information about the kinds of runbooks that can be created can be found in the product documentation: https://ibm.biz/aiops_runbook_automation

Scope-based event grouping

AIOps uses three different approaches to event correlation that work together to gather events together that related to the same problem. One of these methods is called scope-based grouping and works by grouping together events that share a common attribute over a defined time window. For example, a scope-based grouping policy may be to group events together that occur at the same physical geographical location within a 5-minute window. The "scope" could be any attribute that makes sense in the context of the correlation: the same application group, business unit, or physical building.

Scope-based grouping is defined by the user and is a way of inputting local knowledge and expertise about how events can and should be grouped based on various attributes. In practice, scope-based grouping often yields a lot of value. Most customers see a reduction in events due to correlation upwards of 50% so the case to leverage scope-based grouping is clear.

AIOps includes two scope-based grouping policies out-of-the-box:

- Group alerts that have the same resource name (host name), and use a 900 second rolling window to do so;
- Group alerts that are coming from a Netcool environment where the ScopeID field is set, based on its value, and use a 900 second rolling window to do so.

Note: Unlike Netcool environments, alerts in AIOps can have multiple scopes and be members of more than one scope-based group at once. Where different groups have common events, those groups are automatically merged in AIOps into "super groups".

The process for implementing scope-based grouping in AIOps generally follows:

- Identify alert attributes that link related events together;
- Specify a time window to apply – either fixed or rolling;
- Create an AIOps automation policy for each use-case: "Group alerts based on scope"

EXISTING NETCOOL ENVIRONMENTS

Where scope-based grouping is used in an existing Netcool environment, the *correlation_triggers* group should be disabled in the ObjectServer infrastructure, and the correlation done at the AIOps layer instead. The *ScopeID* field from Netcool/OMNIbus should be mapped to AIOps via the Netcool Connector connection to the *alert.resource.scopeId* attribute which is then acted on by the policy: *Default Netcool scope-based grouping*.

Note: This default policy cannot be modified however it can be disabled and an alternative set of criteria implemented if the default is not appropriate for your environment.

Alert seasonality detection

An event is considered seasonal if it recurs with chronological predictability. Seasonal events are marked as such in AIOps in the Alerts viewer. No configuration is required to activate alert seasonality detection; the model training simply needs to be configured and run.

To enable alert seasonality detection, log into the AIOps UI, go to *AI model management*, go into *Alert seasonality detection*, and click on *Train model and create policies*. AIOps will then attempt to train and create models based on the available data. After a successful training run, alert seasonality policies may be generated, and will start to mark any related events as being seasonal.

Note: When scheduling each of the trainable AI algorithms, it is strongly recommended to schedule training to happen during quieter periods of the day, and to stagger them out – for example, schedule them to do their training runs at least one hour apart from each other. This will help to spread the analytical processing load out.

Temporal grouping

The second of the event correlation mechanisms in AIOps is temporal grouping. Just like for alert seasonality detection, no configuration is required to activate temporal grouping; the model training simply needs to be configured and run.

To enable alert temporal grouping, log into the AIOps UI, go to *AI model management*, go into *Temporal grouping*, and click on *Train model and create policies*. AIOps will then attempt to train and create models based on the available data. After a successful training run, Temporal grouping policies may be generated, and will start to group any related events accordingly.

Probable-cause analysis

Probable cause analysis or grouping is an unsupervised learning algorithm that analyzes event and topology data to understand how alerts are related to each other. It then uses that understanding to try and determine the root cause of a problem. This is enabled by default, and enables AIOps to identify and highlight the most likely probable cause of any grouping of events. It then uses elements of the most likely probable cause event to populate any parent events or incident headlines.

Probable cause analysis can be customized to allow a user to input local knowledge about certain events that should be ranked more highly than others based on keyword analysis. Probable cause can also be customized in terms of what significance severity has to the calculation. More information on this is here: https://ibm.biz/aiops_probable_cause

Event housekeeping

Event housekeeping automation is an essential function in any AIOps deployment as it ensures that the system does not get overwhelmed or overloaded with event data. There are different ways to approach event housekeeping: ensuring all events are set to expire, gradually reducing event severity as events age, monitoring the event numbers, and performing automated actions when the row counts become too big.

expirySeconds

In a traditional Netcool environment, a typical technique to keep event numbers under control is via the built-in event expiry automation. It is implemented by setting the *ExpireTime* field to a non-null value, either in the Probe rules, or an ObjectServer or Netcool/Impact automation. A default trigger called *expire* then acts on events where *ExpireTime* is set and clears events where the first occurrence timestamp of each event is more than is *ExpireTime* number of seconds ago.

An equivalent function exists in AIOps by setting the *expirySeconds* attribute in the same manner. A suggested way to do this is to simply send every newly inserted event to a custom Netcool/Impact policy and set *expirySeconds* where not set, according to the defined event expiry policy. The following Netcool/Impact JavaScript policy provides an example event expiry housekeeping policy based on the severity of the incoming event:

```
// ONLY SET expirySeconds FOR EVENTS WHERE IT IS NOT SET

if (String(EventContainer.alert.expirySeconds) == "undefined") {

// KEEP CRITICAL ALERTS FOR SEVEN DAYS

  if (Int(EventContainer.alert.severity) == 6) {
aiopsUtils.patchAlertNoWait(EventContainer.alert.id,{expirySeconds:604800});}

// KEEP MAJOR ALERTS FOR FIVE DAYS

  else if (Int(EventContainer.alert.severity) == 5) {
aiopsUtils.patchAlertNoWait(EventContainer.alert.id,{expirySeconds:432000}); }
```

```
// KEEP MINOR ALERTS FOR THREE DAYS

  else if (Int(EventContainer.alert.severity) == 4) {
aiopsUtils.patchAlertNoWait(EventContainer.alert.id,{expirySeconds:259200}); }

// KEEP WARNING ALERTS FOR ONE DAY

  else if (Int(EventContainer.alert.severity) == 3) {
aiopsUtils.patchAlertNoWait(EventContainer.alert.id,{expirySeconds:86400}); }

// KEEP INFORMATIONAL EVENTS FOR TWELVE HOURS

  else if (Int(EventContainer.alert.severity) == 2) {
aiopsUtils.patchAlertNoWait(EventContainer.alert.id,{expirySeconds:43200}); }

// KEEP INDETERMINATE EVENTS FOR SIX HOURS

  else if (Int(EventContainer.alert.severity) == 1) {
aiopsUtils.patchAlertNoWait(EventContainer.alert.id,{expirySeconds:21600}); }

}
```

EXISTING NETCOOL DEPLOYMENTS

In the scenario where AIOps is being adopted on top of an existing Netcool deployment, it is probable that an event expiry policy will already be in-place. In this case, the *ExpireTime* field can simply be mapped to the *expirySeconds* attribute in AIOps to replicate the same behaviour. Note that this will only set *expirySeconds* for events that originate in Netcool. If other event integrations exist in the AIOps environment, then something like the Netcool/Impact policy outlined above will need to be in-place to handle those other events. Note that in the example above, the Netcool/Impact policy only sets the *expirySeconds* attribute when it is not already set.

Event storm control

Event storms (also known as event floods) in the context of this document are defined as a sudden, large influx of events into the AIOps infrastructure. Event storm control is about configuring controls and automated mechanisms into an AIOps deployment that will protect it against the potentially overwhelming effects of an event storm.

AIOps can handle event bursts of up to 1,000 events per second however large environments can generate much higher event rates than this during major outages. Consideration should be given therefore to the kinds of event rates typically seen during such events and planning in architectural features and automations that will provide protection to the core AIOps system.

NETCOOL EVENT STORM CAPABILITIES

Netcool has provided event storm protection capabilities for many years and these capabilities can be employed in the same way with an AIOps deployment. Most AIOps deployments will invariably include Netcool Probes for event collection. Although Netcool Probes can be connected directly to AIOps, it is recommended to deploy at least one failover pair of Netcool/OMNIbus Aggregation ObjectServers and instead connect Probes to these in a resilient manner.

Employing Netcool/OMNIbus ObjectServers as a connection point for your Netcool Probe estate provides several benefits including:

- Providing a deduplication function to the event stream;
- Allowing you to filter which events propagate up to AIOps;
- Giving you the ability to implement event storm detection and contingency automation;

- Providing a cleaner and simpler implementation where AIOps connects to each ObjectServer pair separately via a Netcool Connector instance.

The planning of the deployment of Netcool/OMNIbus Aggregation ObjectServers should be done in the normal way, considering event rates and volumes. These factors will determine how many ObjectServer pairs are required. Any ObjectServers deployed should be done using the standard multitier architecture configuration that ships with Netcool/OMNIbus. In addition to Aggregation ObjectServers, there may also be a need to deploy Collection layer ObjectServers.

Further information and guidance of deployment considerations in relation to Netcool/OMNIbus components and event storm control and considerations can be found in the *Netcool/OMNIbus 8.1 Best Practices Guide*, available here: https://ibm.biz/nco_bps

Note: Netcool components can be used to implement event storm controls for any data coming through them. Note however that AIOps includes other event integrations via its Connector suite that will need further consideration since those events won't be subject to Netcool automation or controls.

NON-NETCOOL EVENT STORM CONTROLS

Events flowing into AIOps via non-Netcool routes will need their own controls in-place to control event numbers and surges. Part of event storm contingency planning includes good housekeeping practices, which has been covered earlier. Ensuring all events expire in a reasonable timeframe, for example, is an important part of this. If further monitoring of event numbers is required, Netcool/Impact can be used to both monitor and take remedial action if event numbers creep up beyond defined thresholds.

Key event integrations

Most deployments will include requirements to integrate with third-party systems, such as a source of information for event enrichment, or for ticketing purposes. AIOps comes with several off-the-shelf Connectors that can be used for some kinds of integrations. Other integrations, or ones that have specific or complex needs, can be implemented via Netcool/Impact. There are two main ways of doing this.

EVENTS IN THE OBJECTSERVER

If the nature of the integration is a simple or a one-time action, events can be picked up by Netcool/Impact directly from the Netcool/OMNIbus ObjectServer. Any updates to the event, such as custom field enrichment or a ticket number, can be written back to the event. These updates will then propagate up to AIOps via the Netcool Connector.

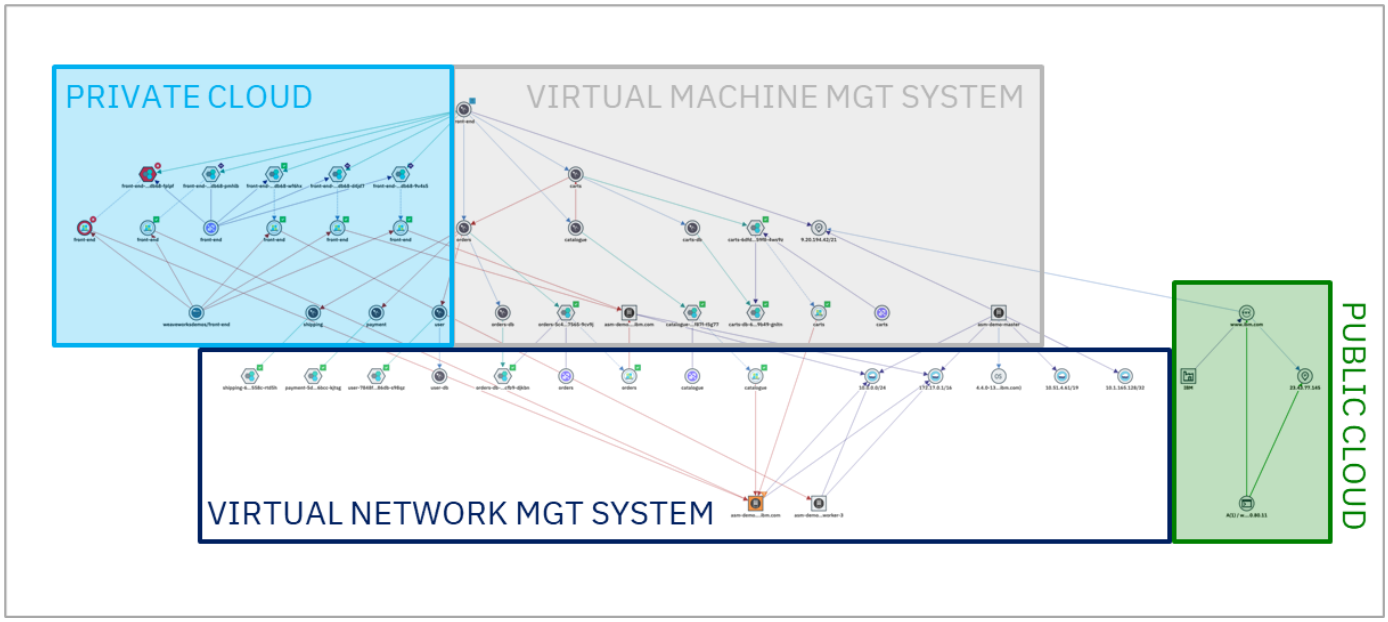
ALERTS IN AIOPS

An alternative approach is to create an AIOps Automation policy that sends the event to Netcool/Impact via the Netcool/Impact Connector under the desired criteria. The Netcool/Impact policy can then either write any updates or enrichments back to the Netcool/OMNIbus ObjectServer or back to AIOps directly.

Getting started with topology

After events, the processing of topology data is typically the second data type that is considered during deployments. Adding topology provides invaluable context to triaging and troubleshooting as well as enabling key functionality in AIOps such as topology-based event correlation and a topological dimension of probable-cause analysis.

There are usually many potential sources of topology data available in modern environments. Indeed, most modern environments span many different platforms and are comprised of many different technologies. Moreover, modern environments are typically highly dynamic, and some have automated orchestration based on load. The need for a dynamic topology capability is clear, therefore.



Consider event sources and monitored services

Ahead of compiling a list of topology sources, it is recommended to first consider what the sources of events are or will be, as well as the services that need to be monitored, as this will define what topology is needed.

While it is not essential to have topology coverage for the whole event estate, it is highly beneficial: it will enable visualisation of affected resources, and it will provide opportunities for topology-based event correlation. Try to identify therefore the best sources of topology data that will span as much as possible of your event estate.

It is also important to consider what topology data exists for any key monitored services and their dependent parts. Bringing this topology data into AIOps will be key to enable the visualisation of these key services.

These activities can be helpful in highlighting any gaps in the monitoring or the topology data. Ideally all parts of the managed environment will be covered from a monitoring perspective, as well as from a topology perspective.

Use off-the-shelf Observers where possible

After compiling a list of your topology data sources, review the current Observer list and map these together. If an off-the-shelf Observer exists for a source of topology data, it is of course recommended to use that as a first option. Where there is no off-the-shelf Observer, then a generic integration option must be used instead. The options are either the File Observer or the REST Observer.

A full list of Observers is available here: https://ibm.biz/aiops_observers

REST Observer versus File Observer

The File Observer is a very convenient testing and development tool to enable for quick and easy ingestion into AIOps. It involves creating a file containing lines of topology data, one resource per line, and then loading that file into AIOps via the GUI or the command-line. The REST Observer takes a feed of data in the same format however instead of being loaded as a static file, it instead provides a URL endpoint through which the data stream can be fed. A REST Observer job is likely a better choice for a production environment as then an external data processing engine can retrieve and post topology data more easily on-the-fly.

Both types of generic Observer use the following format:

```
V:{"_operation": "InsertReplace", "name": "switch1", "uniqueId": " switch1",
"_references": [], "entityTypes": ["switch"], "Attribute1": "ThisValue"}

V:{"_operation": "InsertReplace", "name": "myserver1", "uniqueId": " myserver1",
"_references": [{"_toUniqueId": "switch1", "_edgeType": "connectedTo"}],
"entityTypes": ["host"], "Attribute1": "ThisValue"}

V:{"_operation": "InsertReplace", "name": "myserver2", "uniqueId": " myserver2",
"_references": [{"_toUniqueId": "switch1", "_edgeType": "connectedTo"}],
"entityTypes": ["host"], "Attribute1": "ThisValue"}
```

The above example will create three resources: a switch called *"switch1"* and two hosts *"myserver1"* and *"myserver2"*. The two servers will be connected to the switch via a *"connectedTo"* relationship.

NOTES:

- *"V"* stands for Vertex and means the same thing as a resource in the topology. Other options are possible too, and are documented here: https://ibm.biz/aiops_file_observer
- The *"InsertReplace"* operation directive means that if the resource is already present in the topology, the incoming data will act like an update rather than generating an error. It is very useful for updating existing resources with updated attribute and connectivity data.
- The *"name"* attribute is the standard label applied to resources in the topology.
- The *"uniqueId"* attribute is what uniquely identifies this resource in the topology. It is equivalent to a primary key field in a database.
- The *"_references"* attribute is an array and can include several sub-attributes. Key ones of these are the *"_toUniqueId"* and *"_fromUniqueId"* which are used to define connections between this resource and other resources. The values of these attributes should be the *"uniqueId"* values of other resources. The addition of these attributes results in relationship lines connecting resources together. The *"_edgeType"* attribute defines the type of connection, and forms the label for the connection in the GUI.
- The *"entityTypes"* attribute defined the type of resource being represented. An icon can then be associated to this resource in the *Relationship types* window in the GUI.

- Any other name value pairs can also be added to the entry. These additional attributes will be visible via the GUI when a resource's details are examined.

Merging topology data sets

After all the topology data sets have been loaded into AIOps, the next step is to connect them together in a meaningful way. This can be done by creating connections between resources across the data sets, or more commonly, identifying resources that are common across topology data sets and merging them together. AIOps merges topology data sets in a non-destructive manner; that is, you define a merge rule for each relationship you want to define.

Note: AIOps will not automatically merge any topology data sets together unless you specify a merge rule to explicitly do so. This is intentionally so, so that AIOps does not inadvertently merge resources that shouldn't be.

EXAMPLE

Bruce has created a *VMware vCenter* Observer job to connect to his *VMware vCenter* instance to pull the topology data. Similarly, he has created an *ITNM* Observer job to connect to *ITNM* and pull discovered network topology data. He is looking for a way to join the two sets of data together, specifically where virtual machine resources in the *vCenter* topology represent node resources in the *ITNM* topology. He notes that "*virtualmachine*" resources in the *vCenter* topology dataset have an attribute called "*IPaddress*" that corresponds to the "*networkaddress*" attribute of "*host*" resources in the *ITNM* topology dataset.

He therefore creates a Merge rule in AIOps with the following properties:

- He populates the Rule name field: *vCenterITNMhosts*;
- He adds two Tokens which are the attributes that contain the values that are to be compared during the merge process: *IPaddress* and *networkaddress*;
- He adds the *VMware vCenter* and *ITNM* to the list of Observer types this rule should apply to;
- He adds the two specific Observer jobs that created the two topology data sets to the list of Providers that this rule should apply to;
- He selects the two Resource types this rule should apply to: *virtualmachine* and *host*;
- He navigates back to the Observer jobs screen and re-runs the *VMware vCenter* and ITNM jobs to generate Merge tokens for the two topology data sets;
- The two jobs run, Merge tokens are generated, and the topology data sets are merged;
- Bruce goes the *Resource management* view, searches for a known merged resource, clicks on *More details*, verifies the merge tokens have been created on the *Properties* tab, and verifies that the *Data origin* tab shows both Observer jobs from whence the resource originated. He further notes that the *Properties* tab also now contains a superset of the attributes brought in from both the Observer jobs, thereby providing a fuller context to the resource.

Note: Tying the rule to a specific Observer type, Observer job, or Resource type isn't mandatory but care must be taken to not apply this rule to more resources than is intended.

Matching events to topology resources

Correlating events to topology resources involves creating one or more Match token rules for each topology dataset in your environment. Each Match token rule specifies which topology resource attribute should be used to try to match to an event's *resource.name* attribute.

EXAMPLE

Bruce has a *VMware vCenter* Observer job pulling topology data from his *VMware vCenter* instance. Separately, he is receiving alerts into AIOps via a *Netcool/Probe for Syslogd* that relate to the virtual machines deployed within *VMware vCenter*. He wants to create a Match tokens rule so that the incoming events will be matched to the relevant topology resources. The resources' "name" attribute matches the *Node* field in the Probe and similarly the *resource.name* attribute in AIOps.

He therefore creates a Match tokens rule in AIOps with the following properties:

- He populates the Rule name field: *vCenterITNMhosts*;
- He adds "*name*" to the Tokens section to specify the attribute that should be used to try to match against the events' *resource.name* attribute;
- He adds the *VMware vCenter* to the list of Observer types this rule should apply to;
- He adds the specific Observer jobs that created the topology dataset to the list of Providers that this rule should apply to;
- He selects the Resource type this rule should apply to: *virtualmachine*;
- He navigates back to the Observer jobs screen and re-runs the *VMware vCenter* job to generate Match tokens for the topology dataset;
- The job runs, and Match tokens are generated for the resources within the job;
- Bruce goes the *Alerts viewer* view, and looks for a newly inserted event that relates to a *VMware* topology asset where the Topology column is marked with a dot;
- Bruce opens the *Alert details* view and clicks on the *Topology* tab to view the topology.

Warning: If an event can potentially match more than one resource, it will result in matching no resources due to the conflict. Care must be taken therefore to ensure that a Match tokens rule will only match exactly one resource.

Using tags

Observer jobs create resources in the topology that contain attributes. Any of these attributes can be add to each resource's tags list. Tags can then be used for a variety of purposes including being searched on from the Resource management view, or for event correlation. The process for specifying which attributes should be added to a resource's tags list is done within the *Tags rules* section under *Topology configuration*.

EXAMPLE

Bruce has a *VMware vCenter* Observer job pulling topology data from his *VMware vCenter* instance. All the virtual machine instances in his topology include an "*owner*" attribute which is populated with the business unit that is responsible for the virtual machine. Bruce wants to create a topology view for each of the six main business units.

He therefore creates a Tags rule in AIOps with the following properties:

- He populates the Rule name field: *vCenterITNMownerTag*;

- He adds *"owner"* to the Tokens section to specify the attribute that should be added to the tags list of the resource;
- He adds the *VMware vCenter* to the list of Observer types this rule should apply to;
- He adds the specific Observer jobs that created the topology dataset to the list of Providers that this rule should apply to;
- He selects the Resource type this rule should apply to: *virtualmachine*;
- He navigates back to the Observer jobs screen and re-runs the *VMware vCenter* job to generate tags for the topology dataset;
- The job runs, and tags are generated for the resources within the job;
- Bruce goes the *Alerts viewer* view, and looks for a newly inserted event that relates to a VMware topology asset where the Topology column is marked with a dot;
- Bruce goes the *Resource management* view, searches for a known virtual machine, clicks on *More details*, toggles the *"Show JSON"* switch, and verifies that tags have been created;
- Bruce then goes on to create a one tag-based Resource group template for each business unit, to group the underlying resources together based on business unit.

Creating topology Resource group templates

Topology Resource group templates are used to create groups of resources in the topology for both the visualisation and correlation purposes. For example, you may want to visualize all the assets owned by a business unit, or you may want to correlate underlying events together that relate to a set of underlying dependent resources. In both cases, you need to create a Resource group template.

There are four types of Resource group template available:

- Dynamic template: this template allows you to specify a pattern that identifies a group of resources based on the example you give it. It is very powerful as it means you can specify an example once, and AIOps will automatically find all other examples that match your pattern and create a group for each.
- Tag based template: this template allows you to create a single resource group containing the resources that have a specified set of tags.
- Token template: this template defines a set of rules which use the properties of your resources to automatically create one or more groups which contain those resources.
- Exact template: this template creates and updates a single group based on the specified seed resource and the subsequent criteria you specify.

Note: In all cases, the resulting group or groups will be created, but you can opt to toggle on or off the option to correlate any underlying events for any groups that are created.

Getting started with metric data

Ingesting metric data into AIOps is an extremely powerful way of monitoring key performance metric data streams, since it requires no manual baselining; it automatically selects one or more suitable metric analysis algorithms to process each data stream and automatically set dynamic thresholds based on a period of normal operation. After AIOps completes its training period, it will then continue to monitor each metric stream and alert you if any start behaving anomalously. You can also examine any of the metric streams via the GUI over a date range of your choosing.

Identify key systems to be monitored

The first step is to make a list of all the key metric streams that are important to the business. Ideally you won't just monitor everything, but take a targeted approach to the ones that matter. For example, a KPI for a core application may be response or transaction time. For a key set of backbone network interfaces, it may be the transmit and receive packet quantities.

Use off-the-shelf Observers where possible

Using off-the-shelf Observers is always preferable to creating a custom integration, as it simplifies the deployment. At the time of writing, there are 6 AIOps Connectors that will automatically pull and process metrics from target systems:

- *Appdynamics*
- *AWS CloudWatch* connections
- *Dynatrace*
- *Instana*
- *New Relic*
- *Zabbix*

For a full list of off-the-shelf Connectors, go to: https://ibm.biz/aiops_connection_types

Implement metric data integrations

Where an off-the-shelf Connector is not available for a source of metric data, you will need to retrieve the metric data, format it, and then inject it into the AIOps metric data API. A convenient way of doing this is to create a Netcool/Impact policy to access the target API, pull and format the data, then insert it into the AIOps Metric Anomaly Detection API.

Note: The optimal polling period for metric data is 5-minute intervals.

The schema that the Metric API expects is as follows:

```
{
    "groups": [
        {
            "timestamp": <timestamp>,
            "resourceID": "<resource ID>",
            "metrics": {
                "<kpi-1>": <value-1>,
                "<kpi-2>": <value-2>,
                ...
                "<kpi-n>": <value-n>
            },
            "attributes": {
                "group": "<metric namespace>",
                "node": "<top-level resource>"
            }
        }
    ]
}
```

An example Netcool/Impact policy to retrieve metric data from one API and push it to the Metric Anomaly Detection API is provided in *Appendix B*.

Further information on how to access the Metric data API is provided here: https://ibm.biz/aiops_metric_api

Getting started with log data

The Cloud Pak for AIOps can leverage infrastructure and application logs to detect system anomalies. These anomalies are represented as log anomaly alerts that are correlated with other alerts coming from other sources. Log anomaly detection only makes sense when you have good quality logs that represent a healthy system. Once AIOps has trained on what a healthy system looks like, it can then spot an anomalous problem.

There are different data requirements required by each log anomaly algorithm. If the infrastructure or application logs are an accurate real-time representation of the health of the system, then log anomaly alerts could help to minimize the mean time to error detection, including the early warning signs of potentially bigger problems happening later. Since log anomaly detection is extremely resource intensive, it is important to limit log analysis to infrastructure and applications that are considered business critical, to ensure hardware requirements do not spiral upwards.

Note: The *Custom Sizing Tool* discussed on page 12 can help you estimate hardware requirements based on the amount of input data.

AIOps has several integrations with log aggregators such as *ELK*, *Falcon LogScale*, *LogDNA*, and *Splunk*, and comes with a REST service for custom integrations. These integrations connect to the source and pull the log data. There is also a *Kafka* based integration for cases

where the customer does not have a log aggregator for which AIOps has an off-the-shelf Connector. The Kafka-based integration also supports log processing where historic logs are fed in offline. Note that more time needs to be allocated for planning and development in this scenario since custom logic may be needed for parsing and transforming the logs into a log format expected by AIOps.

Another consideration is the language used in the logs. As of version 4.2, only the following language combinations are supported: English & Spanish, English & German, English & French, English & Italian, and English & Korean.

Change risk

There are hundreds of changes that affect a service during its lifecycle. *Change risk* is an unsupervised learning algorithm that takes historical data from tickets and helps you determine how likely it is that a given change would cause a problem. This determination is based on how successful that similar change was deployed in the past. Using the assessment score provided by change risk, you can determine how safe it is to proceed with the change.

HOW WILL IT HELP ME?

Training this AI algorithm will help you ensure that risky changes for a service are assessed before deployment. Change risk assessments are sent to your ChatOps interface when a change request ticket is opened.

WHAT KIND OF DATA DO I NEED?

To train this AI algorithm you need change request and incident ticket data. If you enabled the collection of historical data when you connected to *ServiceNow*, data for the time frame you selected will be available to train on.

To provide the best data for change risk, a mix of successful and failed requests is needed. Ideally, you'll want around 10,000 change tickets with 2% that were failures, but models will be created with less.

Log anomaly detection - golden signals

Log anomaly detection - golden signals and *Metric anomaly detection* work together to continually model normal log message activity, and generate alerts when the current activity differs from normal activity.

Model training continually discovers patterns within large volumes of log messages. Each pattern represents a general type of log message, and includes a mix of constants such as “Enterprise application”, and variables such as “DATE”. Each pattern is stored in a template, and domain knowledge is used to assign each template one of the following golden signal types: Error, Latency, Saturation, Traffic, Availability, Exception, or Information. The templates, except those assigned the golden signal type Information, are used when modelling and tracking log message activity.

Each alert includes the golden signal type and ID of each template that is involved. In some cases, the golden signal information triggers alerts to be promoted to incidents. It is also used when determining the probable causes of incidents.

HOW WILL IT HELP ME?

IT operation teams can use alerts generated to identify the log patterns that deviated from normal behaviour in the period in which the abnormality was observed. This information will help with the root cause analysis and reduce the time required to resolve IT incidents.

WHAT KIND OF DATA DO I NEED?

To begin training this algorithm, you need to add at least one source of log data via the *Data and tool connections* window in AIOps.

Note: All enabled log data connections must be set to *Live data* for initial training mode. This rule applies to log data connections created both before and after training setup.

Alerts will be generated after:

- At least 100,000 log messages have been processed (to ensure template quality);
- The logs span at least 3.5 days and metric anomaly detection training has been run on that data.

Note: Before enabling training of *Log anomaly detection - golden signals*, you need to set up metric anomaly detection so that it includes regular, scheduled training; otherwise, log anomalies will not be detected.

Log anomaly detection – natural language

Log anomaly detection – natural language is an unsupervised learning algorithm that takes large amounts of log data and trains on it to learn what is normal behaviour for a given resource. It uses natural language processing of log messages to find patterns and analyzes their frequency.

HOW WILL IT HELP ME?

Training this AI algorithm will help AIOps automatically detect unusual patterns in logs and notify you when they occur. Anomalies are raised to your ChatOps interface. For information on an anomaly, you can view related events to see the patterns and associated log messages.

WHAT KIND OF DATA DO I NEED?

To train *Log anomaly detection – natural language*, you need log data that shows normal operational behaviour. If your logs include anomalous situations, such as a critical incident, technical glitch, or some other change that deviates from the norm, you can filter that data out before training the algorithm.

When providing log data, select a date range that includes at least 10,000 lines of messages for each resource that you want insights from. An ideal amount would be around 100,000 lines. When selecting longer date ranges, consider that it might take more time to process the data and could have system impacts.

Similar tickets

When an incident occurs, it can be helpful to review details for similar incidents to help determine a resolution. *Similar tickets* is an unsupervised learning algorithm that aggregates information about similar messages, anomalies, and events for a component or a service. It can also extract the steps used to fix previous incidents, if documented.

HOW WILL IT HELP ME?

Training this AI algorithm will help you discover historical incidents to aid in the remediation of current problems. Information about similar past resolution tickets is also included in the incident details you can access from your ChatOps interface.

WHAT KIND OF DATA DO I NEED?

To train this AI algorithm, you need to provide incident ticket information. There is no set minimum for how much data to provide; the AI will train on whatever amount is available.

The more tickets that can be consumed however, the better. Any data collected for historical training will be available to use as data. You can also collect data from the live connection and as future tickets are created, they will be processed automatically.

Log anomaly detection – statistical baseline

Log anomaly detection – statistical baseline is an unsupervised learning algorithm that automatically detects unusual patterns in logs and notifies you when they occur. Data that is used for analysis is updated every 30 minutes, so this algorithm provides quick value.

HOW WILL IT HELP ME?

Enabling this algorithm helps AIOps understand normal behaviour so when an anomalous situation arises, you will be notified. Anomalies are raised to your ChatOps interface. For information on an anomaly, you can view related alerts to see the patterns and associated log messages.

Note: This algorithm is enabled by default. If you disable it, you will not see any details about statistical baseline anomalies in your *Alert viewer*.

Log Anomaly tips and recommendations

This section contains several tips and recommendations to help you use the various log analysis capabilities and maximize the value gained from them.

IBM MQ OR IBM WEBSPHERE LOGS

AIOps provides domain-specific log anomaly detection for *IBM MQ* and *IBM WebSphere* and uses the same algorithm as statistical baseline log anomaly detection. In this case, the statistical baseline model already has a baseline for domain-specific logs. Thus, the model can detect potential errors straight away without the need for training.

BE MINDFUL OF LOG PROCESSING CAPACITY

Log Anomaly Detection is a relatively high-cost process in terms of hardware requirements, it is therefore recommended to limit log processing to mission critical applications only that have high quality logs. Note however that the golden signals algorithm its optimized for lower resources therefore it can process larger quantities of logs.

SCALE LOG ANOMALY DETECTION PODS AS NEEDED

Depending on how much log data you need to process, you may need to scale pod replicas to handle processing for additional components. Make sure to follow the recommendations in the documentation, which includes a very helpful troubleshooting AI-models section.

OFFLINE LOG ANOMALY TRAINING

You can use the Kafka Connector to enable training with offline, historic log data. Make sure to follow the recommendations and steps described in the documentation. As of AIOps version 4.2, offline log processing is only supported with the natural language algorithm.

WHICH LOG ANOMALY ALGORITHM SHOULD I USE?

If the ChatOps interface (ie. *Slack* or *Microsoft Teams*) is to be used, the natural language algorithm has the richest user experience.

If you want to get quick value from log anomaly analysis, the statistical baseline algorithm only requires 30 minutes of log file data to baseline itself.

The golden signals algorithm has demonstrated better performance compared with the other algorithms in several areas. It only requires two to three days of logs, and it has some unique qualities compared to the others, such as:

- Golden signals tagging uses keyword matching techniques to identify what is important in each log message which provides additional context information to facilitate root cause analysis. The golden signal types are Latency, Error, Availability, Exception, Traffic, Saturation, and Information.
- Logs considered to be informational only are initially filtered out and not processed resulting in lower overall hardware resource requirements and faster processing.
- Each *log pattern template* is treated as a separate metric. Metric Anomaly Detection is used to train models to find log anomalies. The same metric graphs are used to show the progression of anomalous logs over time and to show actual sample log lines related to the alert, which aids faster root cause analysis and Explainability.
- The new *Templates Web View* provides the most recent log anomaly alerts for the last 24 hours, as well as around two weeks' worth of related log lines for each discovered template.

Appendix A. Initial requirements gathering checklist

This checklist should be completed during the planning phase of an IBM Cloud Pak for AIOps deployment to capture the inputs required to size the deployment.

Sources of events: …… …… …… …… …… …… …… ……	Rate (eps): …… …… …… …… …… …… …… ……
Estimated overall event rate:	……
Estimated number of standing events:	……
Sources of topology data: …… …… …… …… …… …… …… ……	Est. # of resources: …… …… …… …… …… …… …… ……
Estimated number of topology resources:	……
Sources of metric data: …… …… …… …… …… …… …… ……	Est. # of streams: …… …… …… …… …… …… …… ……
Estimated number of metric data items per 5-minute interval:	……
Sources of log data: …… …… …… …… …… ……	Est. # of logs per sec …… …… …… …… …… ……

Estimated total number of logs per second:	……
Estimated maximum number of concurrent users of the system:	……
Continency allowance for data storms (suggested 20%):	……%

List of external systems that AIOps will integrate with:

……
……
……
……
……
……
……
……

LDAP integration details:

……
……
……
……
……
……
……
……

Historic event archive requirements:

……
……
……
……
……
……
……
……

High level outline of the event expiry policy that will be implemented:

……
……
……
……
……
……
……
……
……
……
……

List of all custom functionality required with the deployment (pseudocode) — page …… of …… :

Appendix B. Sample Metric Anomaly Detection policy

The following is an example Netcool/Impact policy designed to pull metric data from a target API and push it into the Metric Anomaly Detection API in AIOps.

```
////////////////////////////////////////////////////////////////////////
// GetMetricData
// Created by: N. Etcool 30/11/2023
// The purpose of this policy is to pull metric data from SystemX and
// insert it into the Metric Anomaly Detection API.
// This policy is designed to run off a PolicyActivator Service every 5 minutes.
////////////////////////////////////////////////////////////////////////

Log("METRIC: Running policy AIOPS_GetMetric...");

// DEFINE SYSTEMX API CONNECTION PARAMETERS
HTTPHost = "systemx.widget.com";
HTTPPort = 443;
Protocol = "https";
ChannelKey = "";
FilesToSend = null;
FormParameters = null;
Method = "GET";
AuthHandlerActionTreeName = null;
HttpProperties = NewObject();
HttpProperties.AlwaysSendAuth=true;
HttpProperties.UserId="zane@widget.com";
HttpProperties.Password="mySecretPassword123";
HttpProperties.TrustCertificate=true;
Path = "/metric/path/api";

// DEFINE METRIC ANOMALY DETECTION API CONNECTION PARAMETERS
MADHTTPHost = " cpd-aiops.apps.aiops.cp.widget.com";
MADHTTPPort = 443;
MADProtocol = "https";
MADChannelKey = "";
MADFilesToSend = null;
MADFormParameters = null;
MADMethod = "POST";
MADAuthHandlerActionTreeName = null;
MADHeadersToSend = NewObject();
MADHeadersToSend['Accept'] = "application/json";
MADHeadersToSend['Content-Type'] = "application/json";
MADHeadersToSend['X-TenantID'] = "cfd95b7e-3bc7-4006-a4a8-a73a79c71255";
MADHttpProperties = NewObject();
```

```
MADHttpProperties.AlwaysSendAuth = true;
MADHttpProperties.UserId = "cfd95b7e-3bc7-4006-a4a8-a73a79c71255/nilpifroswiq";
MADHttpProperties.Password = "myOtherSecretPassword123";
MADHttpProperties.TrustCertificate = true;
MADPath = " /aiops/api/app/metric-api/v1/metrics";


// STEP 1 OF 2: GET METRIC FROM SYSTEMX API
x = GetHTTP(HTTPHost, HTTPPort, Protocol, Path, ChannelKey, Method,
AuthHandlerActionTreeName, FormParameters, FilesToSend, HeadersToSend,
HttpProperties);
Log("METRIC: " + ErrorReason);
Log("METRIC: " + ResultCode);


// IF RECORD RETRIEVAL WAS SUCCESSFUL
if (ResultCode == 200 OR ResultCode == 201 OR ResultCode == 202) {


  mymetric = ParseJSON(x);


  // FORMAT PAYLOAD FOR INJECTION INTO METRIC ANOMALY DETECTION API
  Payload = '{"groups": [ {"timestamp": "' + mymetric.data[0].timestamp +
        '", "resourceID": "myserver01:' + mymetric.data[0].interfaceName +
        '", "metrics": {"Transmit": ' + mymetric.data[0]. data[0][1] +
                    ', "Receive": ' + mymetric.data[0]. data[0][2] +
        '}, "attributes": {"group": "group01", "node": "myserver01"}}]}';


  MADHttpProperties.Content = Payload;


// STEP 2 OF 2: SEND METRIC PAYLOAD TO METRIC ANOMALY DETECTION API
  y = GetHTTP(MADHTTPHost, MADHTTPPort, MADProtocol, MADPath, MADChannelKey,
MADMethod, MADAuthHandlerActionTreeName, MADFormParameters, MADFilesToSend,
MADHeadersToSend, MADHttpProperties);
  Log("METRIC: " + ErrorReason);
  Log("METRIC: " + ResultCode);
}
```

Notices

This information was developed for products and services offered in the U.S.A.

IBM® may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing

IBM Corporation

North Castle Drive

Armonk, NY 10504-1785

U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation

Licensing

2-31 Roppongi 3-chome, Minato-ku

Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions; therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation

958/NH04

IBM Centre, St Leonards

601 Pacific Hwy

St Leonards, NSW, 2069

Australia

IBM Corporation

896471/H128B

76 Upper Ground

London

SE1 9PZ

United Kingdom

IBM Corporation

JBF1/SOM1 294

Route 100

Somers, NY, 10589-0100

United States of America

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements, or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. These names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing, or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

If you are viewing this information softcopy, the photographs and colour illustrations may not appear.

Trademarks

These terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

- *AIX*
- *developerWorks*
- *IBM*
- *Lotus*
- *Netcool*
- *Tivoli*
- *WebSphere*

Adobe, *Acrobat*, *Portable Document Format (PDF)*, *PostScript*, and all Adobe-based trademarks are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, *Windows*, *Windows NT*, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Intel is a registered trademark of Intel Corporation, Inc. in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.