ITOM Visibility security overview
# Table of contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>3</td>
</tr>
<tr>
<td>Understanding ITOM Visibility capabilities</td>
<td>4</td>
</tr>
<tr>
<td>ITOM Visibility Architecture</td>
<td>5</td>
</tr>
<tr>
<td>ServiceNow Instance security</td>
<td>6</td>
</tr>
<tr>
<td>MID Server security and trust</td>
<td>6</td>
</tr>
<tr>
<td>Credential security</td>
<td>7</td>
</tr>
<tr>
<td>- Local credential storage</td>
<td>7</td>
</tr>
<tr>
<td>- External credential storage</td>
<td>8</td>
</tr>
<tr>
<td>Credential-less discovery</td>
<td>9</td>
</tr>
<tr>
<td>Access requirements, protocols, and privileges</td>
<td>9</td>
</tr>
<tr>
<td>- Port scanning</td>
<td>9</td>
</tr>
<tr>
<td>- SNMP-based discovery</td>
<td>9</td>
</tr>
<tr>
<td>- Windows hosts</td>
<td>9</td>
</tr>
<tr>
<td>- Linux/Unix hosts</td>
<td>10</td>
</tr>
<tr>
<td>- Applicative credentials</td>
<td>10</td>
</tr>
<tr>
<td>- Other technology-specific credentials</td>
<td>10</td>
</tr>
<tr>
<td>Cloud Discovery</td>
<td>10</td>
</tr>
<tr>
<td>Additional security controls</td>
<td>11</td>
</tr>
<tr>
<td>- Plain text patterns</td>
<td>11</td>
</tr>
<tr>
<td>- Code signing</td>
<td>11</td>
</tr>
<tr>
<td>- Command Audit Log</td>
<td>11</td>
</tr>
<tr>
<td>- Syslog</td>
<td>11</td>
</tr>
<tr>
<td>- Unauthorized changes</td>
<td>11</td>
</tr>
<tr>
<td>Frequently asked questions</td>
<td>11</td>
</tr>
</tbody>
</table>
Introduction

Visibility is critical in IT environments. Unless you have visibility of your infrastructure and know how this delivers your mission-critical digital services, there’s no easy way to diagnose and fix service outages, resolve performance issues, assess the risk of changes, optimize infrastructure costs, minimize software compliance issues, or respond quickly and effectively to security threats.

ServiceNow® ITOM Visibility creates a comprehensive view of your infrastructure and digital services. It discovers your end-to-end environment, automatically creating a complete, accurate, and up-to-date record in your CMDB. And it’s built to keep pace with dynamic public and private cloud environments while still providing full support for on-premises infrastructure.

To create this view, ITOM Visibility needs to collect information from physical and logical resources in your IT environment, such as servers, databases, applications, storage, virtual machines, cloud resources, containers, Kubernetes clusters, and so on. To do this, it typically requires credentials to log into these resources (although it also supports credentialless zero-trust discovery using embedded agents).

Because credentials provide access to your IT resources, they need to be protected from accidental and malicious data breaches. ITOM Visibility provides robust security mechanisms to safeguard your credentials using state-of-the-art encryption techniques, authentication mechanisms, and security architectures. These mechanisms also minimize the attack surface—for example, by only requiring a single outbound port to be opened in your perimeter firewall, or by limiting privileges on accounts used to discover your infrastructure.

In this paper, we examine these technical security controls in detail. We also look at additional administrative and operational controls that complement these technical mechanisms, providing a comprehensive security strategy.
Understanding ITOM Visibility capabilities

Before we discuss ITOM Visibility’s security architecture, let’s take a quick look at the capabilities ITOM Visibility provides, starting with gathering information about your infrastructure and digital services:

- **Discovery** discovers physical and logical infrastructure in both on-premises and cloud environments, creating corresponding Configuration Items (CIs) in the ServiceNow® CMDB. It also discovers relationships between these CIs, mapping upstream and downstream dependencies to the TCP port and process level. It provides real-time visibility of virtualized and cloud environments by integrating with event-driven configuration interfaces such as the AWS Config API, and it also supports traditional scheduled and on-demand discovery.

- **Service Mapping** builds on this discovered infrastructure data, automatically creating end-to-end maps of your digital services. It identifies all the CIs that support a service, along with their service-specific relationships. Think of this like a city bus map—Discovery shows you all the roads and junctions (infrastructure) in your city, while Service Mapping shows you the specific route that each bus (service) takes.

In addition, ITOM Visibility provides capabilities that are specifically designed to simplify and strengthen security operations:

- **Certificate Inventory and Management** automatically discovers TLS certificates, creating a comprehensive inventory in the ServiceNow CMDB. It also provides digital workflows for expired and soon-to-be-expired certificates—for example, getting approval to renew a certificate that is about to expire and then automatically renewing it. This reduces certificate management effort and minimizes the risk associated with expired certificates, such as service outages and security breaches.

- **Firewall Inventory and Audit** discovers your firewalls along with their policies, versions, and other attributes, creating a centralized inventory in your CMDB. End users can submit firewall rule change requests through the ServiceNow portal, which are then automatically routed using digital workflows—for instance, to the security team for risk analysis and approval, and then to the network firewall team for fulfillment. All changes are tracked for audit purposes, and administrators can also initiate audits on demand.

“ITOM Visibility provides capabilities that are specifically designed to simplify and strengthen security operations.”
ITOM Visibility Architecture

ITOM Visibility has a distributed architecture built on two main components:

- **The Now Platform.** This is where ITOM Visibility and other ServiceNow applications run, including the ServiceNow CMDB. It is hosted in the ServiceNow Cloud and is maintained by ServiceNow staff. Each ServiceNow customer has one or more instances of the Now Platform running in the ServiceNow Cloud. Each ServiceNow instance has its own database and copy of code, providing strong data separation between customers.

- **The Management Instrumentation and Discovery (MID) Server.** This is a lightweight Java application that runs as a Windows or Linux daemon on a physical or virtual server inside your security perimeter. It executes discovery jobs on behalf of the ServiceNow instance, returning raw discovery data to the ServiceNow instance for further analysis. Examples of these jobs include port scans, SNMP polling, and logging into devices to collect configuration information.

The MID Server is responsible for all interactions with your IT environment. This includes both on-premises and cloud infrastructure. This ensures that there is no direct communication between your IT network and external entities. The only exception to this is configuration events from AWS and Azure, which are sent directly to the ServiceNow instance. These events are used to trigger targeted discovery, which is carried out by the MID Server. Note that configuration events from GCP and VMWare are sent directly to the MID Server, not the ServiceNow instance.
The MID Server communicates with the ServiceNow instance over a single encrypted TLS connection on port 443. It can also be configured to use a proxy server. MID servers initiate all communications with the ServiceNow instance, regularly polling a queue on the instance to look for discovery jobs. This avoids the need to open an inbound port on your perimeter firewall. Also note that ServiceNow does not autonomously configure discovery jobs, giving you complete control over which jobs are executed in your IT environment (see “Plain text patterns” later in this document for more details).

To provide scalability and redundancy, customers can deploy multiple MID Servers. Each MID Server is assigned its own IP address range(s), which can be manually configured, assigned based on IP subnets discovered by ITOM Visibility, or imported from an IPAM tool such as InfoBlox. MID Servers can be deployed to support specific management zones, and it is also recommended to deploy a standalone MID server in each PCI/DMZ zone.

Note that while ServiceNow is responsible for maintaining ServiceNow instances, customers are responsible for deploying and maintaining their MID Servers. This includes activities such as hardening the MID Server host.

**ServiceNow Instance security**

ServiceNow has a comprehensive security program that covers physical, administrative, and logical security. This includes architecture, information lifecycle, physical security, security operations, disaster recovery/business continuity, privacy, compliance, and software development. This program addresses these domains from the perspective of ServiceNow both as a software vendor and as an operator of a large private cloud infrastructure. The details of this security program are beyond the scope of this document, but a comprehensive overview can be found [here](#).

To understand ITOM Visibility security, however, it’s critical to know how encrypted data is stored on the Now Platform. This is because credentials are encrypted and stored in the ServiceNow instance unless you use an external PAM tool such as CyberArk. This is discussed in detail later in this document.

All encrypted data (not just credentials) on a ServiceNow instance are AES-128 encrypted using a ServiceNow Instance encryption key. This key is unique to the instance and is not shared by any other instance. Instance keys are stored in ServiceNow’s secure SafeNet key management appliance. ServiceNow has implemented strict access control for this appliance and has established clear separation of duties. Only four people in ServiceNow security operations have access to the appliance, and these employees do not have access to any ServiceNow customer instance. ServiceNow administrators and engineers who have access to customer instances do not have access to the SafeNet appliance.

**MID Server security and trust**

When running on Windows, the MID Server runs by default on the Local System account. A domain or local service account can also be used, as can a non-privileged account (as of the Paris release). Where applicable, standard Windows authentication is used. Accounts and passwords are never shared with ServiceNow. Note that if a domain account is used, this allows the MID Server to discover and map services on every Windows server in the domain.

On Linux, the MID Server can be run as a non-root user (as of the Paris release).
When a MID Server powers up, it generates an RSA-2048 key pair that is subsequently used to provide message-level encryption of credentials between the MID Server and the ServiceNow instance. These keys are held in memory and are never written to disk or any other file store.

The MID Server then establishes a secure TLS connection on port 443 to the ServiceNow instance. While the MID Server can use basic TLS authentication, it also supports mutual authentication, which is a stronger authentication mechanism that provides more trust between system components and better safeguards against man in the middle attacks. This mutual authentication is underpinned by a unified key store where customers can import their own key pairs.

Once the TLS connection is established, the MID Server logs into a dedicated MID Server account on the ServiceNow instance. The SOAP authentication parameters for the account are stored locally in the MID Server’s configuration file. The authentication parameters are initially entered into the file using plain text, but these are automatically encrypted on first use using symmetric AES-128 encryption. At this point, the MID Server sends its public encryption key to these ServiceNow instance.

Once the MID server has connected and logged in to the ServiceNow instance, it is placed in an untrusted state. While in this state, the ServiceNow instance will not download credentials to the MID Server, preventing malicious actors from logging into the ServiceNow instance using the MID Server account and accessing credentials. An authorized ServiceNow administrator is then required to manually verify the MID Server and place it in a trusted state.

**Credential security**

Credentials are stored in the ServiceNow instance, which then distributes them to MID servers. Alternatively, credentials can be stored in an external credential store such as CyberArk or HashiCorp Vault, in which case the MID Server retrieves the credentials directly from the external store.

**Local credential storage**

To store credentials locally, a ServiceNow administrator enters the credentials using the ServiceNow web interface. The credentials are sent over a secure TLS connection to the ServiceNow instance. The ServiceNow instance encrypts the credentials in memory immediately upon receipt using its ServiceNow instance key and stores the encrypted credentials in the ServiceNow database. Note that once a credential has been entered, it cannot be redisplayed.

When a MID Server needs a credential, it sends a request over its TLS connection to the ServiceNow instance. The ServiceNow instance retrieves the credential from the ServiceNow database and decrypts it using the ServiceNow instance key. It then immediately re-encrypts the credential using the MID Server’s RSA-2048 public encryption key and sends the encrypted key to the MID Server over the TLS connection. The MID Server keeps the credential encrypted in memory, decoding it only at point of use. Once again, the MID server never writes these credentials to disk or any other file store.
External credential storage

In this configuration, the MID Server is directly integrated with an external credential store by installing appropriate CyberArk or HashiCorp JAR files and APIs on the MID Server. This allows the MID Server to retrieve credentials from the external store using a credential ID.

The flow for external credentials is similar to that for local credential storage, with the ServiceNow instance providing credential IDs instead of encrypted credentials. A ServiceNow administrator enters credential IDs which are stored in the ServiceNow instance database. When the MID server needs a credential, it sends a request over the TLS connection to the ServiceNow instance, which retrieves and returns the corresponding credential ID. The MID server then uses this credential ID to retrieve the credential from the external credential store.
Credential-less discovery

In addition to credential-based discovery, ITOM Visibility also supports credential-less discovery using embedded agents installed on discovery targets, providing robust zero-trust discovery. This is done using the ServiceNow® Agent Client Collector (ACC), an extensible agent-based technology that provides both discovery and monitoring capabilities. With ACC, agents initiate all communications with the MID Server, sending discovery information over TLS-encrypted WebSocket connections. This means there is no need to store credentials on the MID Server or open up internal firewall ports.

ACC agents run as non-root users on the target and prevent other non-root accounts from reading or modifying sensitive agent configuration files. Agents also locally encrypt sensitive information in memory, such as MID Server passwords and sensitive command line parameters. Sensitive command line parameters are also obfuscated when displayed.

This push-based discovery mechanism is ideal for micro-segmented architectures, since it avoids the need to identify and periodically pull discovery information from large numbers of discovery targets. ACC also provides out-of-the-box support for a wide range of infrastructure components, including operating systems (Linux, Windows), web servers (Tomcat, IIS, and Apache), application servers (JBoss, WebSphere, and WebLogic), and databases (Oracle, MySQL, and MS SQL).

Note: ITOM Visibility also supports credential-less discovery using Nmap. However, this should only be used to provide an initial basic inventory of CIs in the ServiceNow CMDB prior to full discovery. It should not be regarded as a standalone discovery solution, since it does not provide the depth of detail needed by operations and security teams and does not discover relationships between CIs.

Access requirements, protocols, and privileges

The following only applies to credential-based discovery. It does not apply when using embedded ACC agents.

Port scanning

To identify discovery targets, each MID Server carries out regular port scans across its configured IP ranges. This means that the internal ports which the MID Server scans need to be open. A full list of these ports and associated protocols can be found [here](#).

Note that there is no need to open these ports in the perimeter firewall.

SNMP-based discovery

To enable SNMP-based discovery, each MID Server needs to be added to the appropriate ACLs. The MID Server also requires read-only access to SNMP community strings.

Windows hosts

MID Servers discover Windows hosts using PowerShell remoting over WinRM or WMI. Both protocols require PowerShell to be installed on the remote Windows host. ServiceNow recommends WinRM, since this only uses ports 5895 and 5896, whereas WMI uses a wide range of dynamic ports from 1024 to 65535. We also recommend using PowerShell 5.
The MID Server also needs read-write access to the host’s $admin share. This allows it to retrieve configuration files and output files from the commands it executes. If the $admin share is turned off on a hardened server, the MID Server can fall back to using HTTP from the remote host to the MID Server on port 8585.

Note: The MID Server supports Microsoft’s Just Enough Administration (JEA). This is similar to `sudo` in Linux. This allows the MID Server to discover a Windows host using a non-admin account with delegated commands and privileges.

**Linux/Unix hosts**

MID Servers discover Linux and UNIX hosts using SSH. They require elevated privileges on specific commands, such as `ss`, `lsof`, and `ps`. This allows the MID Server to discover all of the connections, running processes, and other information on the server without being limited to a single user’s context (similar to Windows above). These commands can be delegated using `sudo` rather than setting up a root account. Similar to Windows hosts, the MID Server also requires read-only access to configuration files on Linux and Unix hosts.

**Applicative credentials**

To discover some applications, ITOM Visibility needs application-specific credentials in addition to host credentials. ServiceNow refers to these as applicative credentials. Examples include IBM WebSphere MQ, Microsoft IIS Virtual Directory, Microsoft SharePoint, Nginx, Oracle Database, ABAP SAP Central Services, and others.

**Other technology-specific credentials**

In addition to Windows, Linux/Unix, SNMP, and applicative credentials, ITOM Visibility also requires credentials to discover specific types of technology. Examples include CIM credentials to discover storage devices and read-only vCenter credentials to discover VMware ESX machines, virtual machines, and resource pools.

**Cloud Discovery**

ITOM Visibility provides comprehensive discovery of cloud resources and associated metadata such as tags. To do this, it uses cloud-vendor APIs such as the AWS Config API. This provides fast, centralized discovery, including real-time event-driven discovery. To discover cloud resources, ITOM Visibility needs a read-only account (for example, an IAM user account for AWS). In addition to AWS, it also supports Microsoft Azure, Google GCP, IBM Cloud, and Oracle Cloud out of the box.

Note that if you install a MID Server in an AWS EC2 instance, it can discover cloud resources across multiple member accounts without needing to retrieve credentials from the ServiceNow instance or an external credential store. This is more convenient and secure than using large numbers of member account credentials, especially in large AWS environments. To do this, the MID Server obtains temporary credentials for each member account using the AssumeRole action in the AWS Security Token Service API. These credentials are managed by AWS and rotated automatically.

To discover complete technology stacks in cloud environments, we recommend using cloud discovery in conjunction with Agent Client Collector. As stated previously, ACC’s credential-less, push-based discovery approach is well-suited for the micro-segmented architectures found in cloud environments.
Additional security controls

In addition to the technical controls already described in this document, ITOM Visibility also provides a number of other controls designed to enhance security.

Plain text patterns

ITOM Visibility patterns contain instructions for discovering infrastructure components and mapping digital services. ServiceNow releases new patterns every month, but customers decide when and if these patterns are put into production. Because patterns are in plain text, customers can inspect exactly what patterns do, including the commands executed by the MID Server. This provides complete visibility and control. By using ServiceNow access control capabilities, customers can also prevent users from modifying patterns or adding them to discovery schedules—and even generate an alert if a user tries to do this without authorization.

Code signing

All MID Server code is signed using SHA512withRSA. This includes jar files, patterns, scripts, and other artifacts. Code is evaluated for integrity and authenticity, and it is not executed if it is modified or improperly signed.

Command Audit Log

All PowerShell and SSH commands executed by MID Servers are sent back to the ServiceNow instance and stored in a Command Audit Log for inspection. A hash is calculated for each record in the log to verify that the log has not been modified.

Syslog

All actions on the ServiceNow instance are logged. This includes abnormal ITOM Visibility events, such as a failed login attempt to a MID Server account. These logs can be sent to an on-premises or cloud-based log server via the MID Server.

Unauthorized changes

When ITOM Visibility detects a change in an application CI, it looks for a matching ITSM change request. If there is no matching request, ITOM Visibility flags the unauthorized change and automatically creates an emergency change request to resolve the issue. This is optional functionality and requires a licensed ITSM deployment on the ServiceNow instance.

Frequently asked questions

What level of access is needed to discover public and private clouds?

Read-only access is needed for all public cloud vendors. This allows ITOM Visibility to discover all CIs at the virtualization layer. The same is true for on-premises environments such as VMware.

When does ITOM Visibility need authorization to execute commands?

Authorization is needed when doing deep discovery of servers and other computers. For example, ITOM Visibility needs to run ss with local admin rights to discover all of the incoming and outgoing TCP connections on a Windows server.

Why are elevated privileges needed?

This is due to operating system design. For example, if ss is run as a read-only user on Windows, it only returns the subset of TCP connections that are within the user’s context.
What can be done with just read-only access?

Modern, structured APIs generally support read-only access for auditing. Public and private cloud vendors support this, as do many other devices and applications (for example, Palo Alto Panorama firewall managers). ITOM Visibility can use these APIs when they are available.

Can we audit commands before they are run?

ITOM Visibility uses patterns for command-based discovery. These patterns are in clear text, allowing you to audit commands before you put them into production. Note that ServiceNow releases new patterns on a regular basis but does not automatically put them into production, giving you complete control over which patterns are executed.

How do we prevent unauthorized users from changing how discovery operates?

By using ServiceNow access control capabilities, you can prevent users from modifying patterns or adding them to discovery schedules. We also provide a log of commands executed by MID Servers, as well as a log of commands executed on the main ServiceNow instance (this can be sent to an on-premises or cloud-based syslog server).

How do you ensure the integrity of the MID Server?

MID Server code is signed and is not executed if it is modified or the signature is changed. The MID Server also has file and folder level access control.

How is the MID Server validated by the ServiceNow instance?

The MID Server establishes a secure TLS connection to the ServiceNow instance. Once the TLS connection is established, the MID Server logs into a dedicated MID Server account on the ServiceNow instance. At this point, the MID Server is an untrusted state and cannot download credentials. An authorized ServiceNow administrator is then required to manually verify the MID Server and place it in a trusted state.

How do you store credentials?

Credentials can be stored in the ServiceNow instance or in an on-premises credential vault such as CyberArk. When credentials are stored in the ServiceNow instance, they are AES-128 encrypted using a unique instance encryption key. Instance keys are stored in ServiceNow’s secure SafeNet key management appliance. ServiceNow has implemented strict access control for this appliance and has also established clear separation of duties.

How do you limit the blast radius of the MID Server?

You do not need to use a single service account. For example, you can have a separate MID server for each segment. Since individual MID Servers only have access to credentials for the devices they are discovering, no MID Server has universal credential access. You can also turn off MID Servers locally or remotely from the ServiceNow instance.