# Data Center Discovery 2020.1

SCAN ENGINE SECURITY GUIDE

**Copyright notice**
This document is provided strictly as a guide. No guarantees can be provided or expected. This document contains the confidential information and/or proprietary property of Ivanti, Inc. and its affiliates (referred to collectively as "Ivanti") and may not be disclosed or copied without prior written consent of Ivanti.

Ivanti retains the right to make changes to this document or related product specifications and descriptions, at any time, without notice. Ivanti makes no warranty for the use of this document and assumes no responsibility for any errors that can appear in the document nor does it make a commitment to update the information contained herein. For the most current product information, please visit www.ivanti.com.

Copyright © 2020, Ivanti. All rights reserved.
Ivanti and its logos are registered trademarks or trademarks of Ivanti, Inc. and its affiliates in the United States and/or other countries. Other brands and names may be claimed as the property of others.

Protected by patents, see http://www.ivanti.com/patents

Rev 04/20

# Contents

Overview ..... 4
Data Center Discovery scan engine ..... 5
Credential management ..... 6
Permissions required ..... 6
Credential servers - protection ..... 6
Installation ..... 7
Service start-up ..... 7
Addition of a credential ..... 7
Scan job needs a credential ..... 8
Device credentials ..... 9
Ordinary credential creation ..... 9
Credential escalation ..... 10
Windows ..... 11
UNIX ..... 12
Application credentials ..... 13
User interface security ..... 14
Data transfer between the UI and the scan engine database ..... 14
UI to database ..... 14
Database and connection string encryption ..... 14
SQL injection attacks ..... 14
Cross-site scripting attacks ..... 15
User interface access ..... 15
Login management ..... 15
Accounts and roles ..... 16
Page-level permissions ..... 16
Device security considerations ..... 17
Security considerations ..... 18
Security recommendations ..... 20
Data policy ..... 21
Appendix A: Key terms ..... 22

# Overview

The Ivanti Data Center Discovery scan engine was designed with a strong focus on security. This document aims to provide you with important information regarding security while installing and using the product.

Ivanti focuses security concerns across the following areas:

- **Credential management:** This section overs information about managing the necessary credentials to discover and inventory devices during scans.
- **Device credentials:** This section covers information on parent-to-child credentials and instructions on escalating credentials within the user interface (UI).
- **Application credentials:** This section covers the credentials necessary to access applications.
- **UI security:** This section covers security information regarding the user interface and discusses data transfer from the user to the database.
- **Device security:** This section covers the agentless concept of the scan engine, with a focus on Denial of Service.
- **Considerations and recommendations:** These sections provide useful checklists.

# Data Center Discovery scan engine

The Data Center Discovery scan engine is a highly scalable and secure network inventory tool. It uses a number of protocols to access multiple platforms and applications to securely gather hardware and software information.

The data gathered is typically used for:

- Software license management
- IT asset management
- IT security
- IT operations/operational management
- IT support
- Configuration Management Database (CMDB) population

Data Center Discovery consists of the following components:

- **Scan engine**: The scan engine scans your estate to discover all devices. It uses a variety of discovery methods further defined in the *Data Center Discovery—Scan Engine User Guide*. Discovered devices are then scanned again for information on what is installed to inform more educated licensing decisions.
- **Scan database**: The database in which the gathered data is stored.

For more detailed information, refer to the *Data Center Discovery—Scan Engine User Guide* and the *Data Center Discovery—Scan Engine Prerequisites Guide*.

# Credential management

The Data Center Discovery scan engine can **discover** devices based upon IP address. However, to **inventory** discovered devices, it must be configured with credentials that can log into the target device. These credentials enable the scan engine to scan and report on detailed information for devices, operating systems, and applications that reside on the network. The required rights and permissions to inventory Windows, UNIX/Linux, databases, and virtual hosts are described below.

## Permissions required

The required credentials are generally **read-only** access to device and application configuration files, and permission to execute commands on the target device.

> **Note:** An exception to the access level requirement exists for Windows remote process scanning, which is detailed in the "Windows remote process data retrieval" section of this document.

Examples of file types that the scan engine requires access to:

- Oracle database configuration files (e.g., tnsnames.ora and oratab)
- Veritas Cluster Server configuration file (main.cf)
- WebLogic configuration files
- Internet Authentication Service configuration files
- SQL Server configuration files
- Windows, UNIX, and Linux configuration files

## Credential servers - protection

The scan engine stores credentials within its own database in a secure manner.

The credentials required to access operating systems, applications, and devices on the network are entered on the **Locations menu > Credentials tab** of the UI.

The credentials are protected by public key encryption procedures. Each scanning server is associated with a customer-specific private key generated by the scan engine installation process. Each of these keys is signed with a key associated with Ivanti.

The use of two keys ensures that there is no overlap of known information between Ivanti customers.

Once entered, the credentials are stored in the scan engine database in their encrypted form. They cannot be read or retrieved from the database. It's recommended that you additionally configure the UI to use HTTPS, which ensures credential information is not captured while in transit between the browser and IIS web server.

On the scanning server, the credentials are retrieved from the database in their encrypted form when a list of scanning targets is requested. They're decrypted (in RAM) on the scanning server and used to

provide the scanning server with access to the scan targets. Once access has been provided, the credential is cleared from the scan engine.

The components involved by this design are:

- The installer
- The web-based user interface
- The database
- The scan service

In this scheme, Ivanti maintains a master key pair called **K-master-public** and **K-master-private.** This key pair is used to sign site keys **K-site-public** and **K-site-private**. The site keys are used to protect the credentials.

## Installation

During installation of the scan engine, you're prompted to select one of two options:

- Generate a new site key.
- Use a site key that was previously generated.

With option 1, the installer creates a new key pair K-site-public and K-site-private. It signs the site key pair with K-master-private and writes the signed key pair to the service configuration file protected by Windows Data Protection API (DPAPI).

The installer then encrypts the site key pair with K-master-public and presents that cipher text to you. You're prompted to retain this cypher text for use in future installations (i.e., if option 2 is selected in future installations).

With option 2, the installer requests the text provided from option 1 on a previous install. The installer will decrypt the site key pair, check the signature, and write the signed key pair to the service configuration file protected by Windows DPAPI (i.e., the site pair is now shared between more than one installation of scan engine).

## Service start-up

After installation, each scanning server is required to register itself with the scan engine database. This happens on first execution of the service. At that point, the service also writes to the database the site public key **K-site-public,** signed with the master private key **K-master-private** that it will use.

## Addition of a credential

When you enter a credential in the UI, it reads each of the signed site public keys **K-site-public** from the database and tests the signature. It pads the password with 128 bits of random content (a salt). It then writes to the database the credential encrypted using each site public key.

## Scan job needs a credential

When a job is provided to the scanning service, the service receives the credential encrypted using the site public key. It decrypts that credential using the site private key and removes the 128 bits of salt before sending it to the target.

The table below shows which keys are visible to the four components impacted by this design.

| Key Visibility | Installer | Service | UI | Database |
|---|---|---|---|---|
| K_master_private | x | | | |
| K_master_public | x | x | x | |
| K_site_private | x | x | | |
| K_site_public | x | x | x | x |

# Device credentials

Device credentials are used to log in/access a remote device. Basic credentials are assigned to an associated protocol type and are used to establish the protocol connection to the remote device.

There are a number of cases when a two-stage process is required to access a remote device. Access of this type is typically through a proxy server and a two-stage credential setup is required. The child credential is used to access the proxy server, and the parent credential is used to issue a new command to access the remote device via the proxy server.

> **Note: Credential** escalation is different from the **command** escalation. Credential escalation uses an interim device to gain access to a remote target device. Command escalation is promoting a logged-in user's ability to execute a command on a remote device (e.g., by using the SUDO command).

## Ordinary credential creation

Complete the following procedure to access credentials from the UI. This is covered in more depth in the *Data Center Discovery—Scan Engine User Guide*.

**To access credentials**

1. Select the location in which the credential is to be assigned.
2. Click the **Credentials** tab.
3. Click the **Create** button.
4. Select the credential type and add the additional login information to access the remote device.

5. Click the **Save** button.

> **Note:** If the locations associated with the scan engine have more than one level, the newly created credential will be available to all the levels below the current level.

## Credential escalation

The normal use of a credential is a direct login into a device. The scanning server takes the credential, creates a suitable connection, and passes the credential over the connection to access the remote device.

Credential escalation is required when the above use of a direct-credentials login isn't possible. The connectivity from the scan engine to the target does not allow a direct login into the remote device.

**Parent credential**: Provides the login user and password for a device or a group of devices. This credential should be created as if a direct connection to the device(s) was available. The only modification is that the connection type should be set to **SSHProxy**.

**Child credential**: Provides a login user and password to access the proxy. The child credential allows the specification of the ongoing connection method and allows credential information provided in the parent credential to be used to remotely access the device through the proxy.

The parent/child credential set allows this two-stage process to be created.

The child credential (proxy credential) is likely to be shared between a number of parent credentials. For this reason, once the child credential is attached to a parent credential, the child is no longer editable from the UI. This is to ensure that the proxy credential is modified in a controlled manner.

## Windows

We recommend using **local admin** rights on the Windows device being scanned to gather important system information via the Windows Registry and Windows Management Instrumentation (WMI). Local admin enables the scan engine to limit effects to individual devices while still maintaining control for optimal performance.

> **Note:** The scan engine defaults to local admin rights. However, this is not a requirement, and the system can be configured to accommodate other roles with privileges sufficient to scan.

It's recommended that you create a domain account, which is added to the local administrator group across the domain.

While it's possible to configure WMI and the Registry to use credentials other than local admin, local admin ensures that all domain-joined devices can be properly inventoried. Domain-joined Windows servers and computers use a Microsoft implementation of the Kerberos security protocol for secure login.

Use **Active Directory** policies and/or other mechanisms to restrict the account being used for scanning to the minimum required permission set. Such restrictions include:

- **Disabling** the account when scanning is not active.
- **Limiting** access to the remote device for this credential from the scanning server.
- **Setting** the account to be a non-interactive login (sometimes called service-only account).

- **Using** a one-way single trust relationship, so that authentication requests can only be passed from the domain in which the device to be scanned is in, to the domain in which the scanning server is in.
- **Using** a firewall to block non-needed traffic.

### Windows remote process data retrieval

The scan engine gathers the process information using the WMI protocol. This process creates a temporary file on the target device. The credential used in scan operations must have the appropriate privileges to create this temporary file.

## UNIX

When accessing UNIX and Linux systems, you're provided the option of either using Telnet or Secure Shell (SSH). The use of **SSH** is recommended to ensure the security of authentication credentials.

SSH proxy communication is possible and credential considerations are covered in the "Credential escalation" section of this document.

The scan engine uses a variety of commands to retrieve as much relevant inventory data as possible from UNIX and Linux devices. The commands used are listed in the "Appendix B: Privileged command access" section of the *Data Center Discovery—Scan Engine Prerequisites Guide*. The credentials provided to the scan engine must have the authorization to run these commands.

# Application credentials

Refer to the *Data Center Discovery—Scan Engine Prerequisites Guide* for details about the permissions required for scanning the following environments:

**Oracle:** To gather detailed information required for an audit of Oracle, the scan engine needs to log on to the database and read certain tables. The user account used to access Oracle requires specific access privileges.

**MS SQL user:** To gather detailed information required for an audit of Microsoft SQL Server, the scan engine needs to log on to the database with specific access privileges.

**Informix user:** To gather detailed information required for an audit of Informix, the scan engine needs to log on to the database with specific access privileges.

**ESX Access:** To enable scanning of VMware ESX versions, the scan engine requires access to the VMware vSphere Web Services.

**IBM PowerVM:** To enable scanning of IBM PowerVM Virtualization environments, the scan engine requires SSH or Telnet access to the Hardware Management Console(s) (HMC) used to connect to the Managed Frames.

# User interface security

The UI has undergone penetration testing (PENTEST) by an external consultancy company.

## Data transfer between the UI and the scan engine database

### UI to database

The scan engine submits data to the database via an authenticated SQL client server connection. **Windows Server Internet Protocol security (IPsec)** can be used to provide an additional layer of authentication and encryption for all data exchanged between the scan engine servers.

### Secure Sockets Layer (SSL)

All cookie information is encrypted and secured when accessing the UI to prevent any monitoring of traffic between the server and any client devices. The UI is configured to use the SSL, provided a certificate and https binding exists for the default website before an installation of the scan engine.

### Database and connection string encryption

All information is stored in a SQL Server database. Manual queries and adding or removing information from the database requires a **connection string** and an associated set of login credentials.

This connection string is encrypted and includes:

- The **server name**
- The **database name**
- The **security credentials** required to access the database.

You can also add parameters to the connection string to encrypt the connection.

### SQL injection attacks

SQL injections insert SQL commands into an SQL statement via webpage input. SQL injections input by malicious users alter SQL statements and compromise the security of a web application.

The scan engine uses parameterized queries and stored procedures to mitigate SQL injection attacks.

- **Parameterized** queries use bound, typed parameters that keep the query and data separate using placeholders known as **bound** parameters.
- **Stored** procedures help to create a layer of abstraction that limits the amount of information supplied by the webpage when connecting to the database via a web-based query.

The common language runtime (CLR) functions in the scan engine operate with permissions set to **safe**, which means the managed code executing in the construct is unable to access external resources.

### Cross-site scripting attacks

To prevent cross-site scripting (XSS) attacks, the scan engine uses a variety of security techniques on both input and output data.

- **Input data:** The scan engine validates and constrains input data to ensure submitted content is free of dangerous client-side scripts.
- **Output data:** The scan engine implements encoded output and encoded output URLs. Encoded output converts characters with unique HTML meaning into harmless HTML characters.

## User interface access

The UI displays data and allows for scans and services to be started and stopped.

A user interface which has been compromised, using a technique such as session hijacking, could allow an attacker unauthorized access to the scan-engine functions. A malicious user with access to these functions could affect the scan engine by stopping the service from running. It could also allow a malicious user to reconfigure the system and alter credentials. These risks are mitigated by the actions suggested in the “Security recommendations” section in this document.

### Login management

All users are required to log in before gaining the ability to access other UI pages. This is the first line of defense to prevent attacks.

See the table below for information on login security.

| UI access | Restrictions | Security action |
|---|---|---|
| Login attempts | 3 login attempts | 30-minute account suspension |
| Further login attempts (post 3rd attempt) | 1 attempt | 30-minute account suspension |
| Inactivity timeout | 160 minutes | Automatic logout |
| Closed window | n/a | Automatic logout |

> **Note:** These are the default security settings. You can configure these settings to suit the unique security needs of your organization.

## Accounts and roles

The scan engine uses an administrative account with a default username and password for demos and company use. However, you should replace the default credentials with usernames and passwords that comply with your organization's security policy for a client installation.

The scan engine allows for multiple user accounts to be created and all accounts can fall under customer-specific roles. The scan engine comes with the default role of **admin**. When a user is assigned this role, they can access all functions of the scan engine and oversee the application. This role is for use by administrative and support users who perform scans and have total access to the scan operations within the estate.

## Page-level permissions

Page-level permissions within the UI allow control of which roles can access which pages.

# Device security considerations

**Agentless:** The scan engine is agentless and makes no changes to the servers scanned in any way. No temporary files are created, and no scripts are copied to the target device or application being scanned. See the "Windows remote process data retrieval" section for the single exception to this rule.

The set of commands issued by the scan engine are standard commands, available as standard on the target OS. It's generally not possible to get the scan engine to run additional commands without using encrypted extension capabilities.

**Denial of Service (DoS):** The scan-engine architecture is self-throttling and will only attempt to scan a device or application when the available resources are available. Additionally, the scan engine will only attempt to scan a particular IP address a maximum number of times (specified through the UI). This design prevents the scanning server from causing DoS-type situations due to unintentional or intentional misconfiguration.

**Ports:** The ports used for **discovery scans** are defined within the UI, and a typical installation of the scan engine would be behind the organization's firewall. The scan engine does not mandate the modification of firewall rules or exceptions.

# Security considerations

**Binaries and code:** All scan engine binaries are built upon .NET and take advantage of the provided security framework. Several measures are built-in for security, including specialized .NET encryption to ensure the integrity of each binary in the release. Any screening of these binaries will show the author as **Ivanti**.

Ivanti can generate a checksum of the deliverable release installer and product binaries within the installer. These checksums can be verified by the person installing the software on the designated server to ensure no infection has occurred in transit.

Strict software revision tracking is employed internally by Ivanti, logging all changes to files across each release. Physical access to the source repositories servers is restricted, as is remote server access.

**Access to the database and security:** The scan engine database contains all the critical credentials and configuration data required to run. The security credentials required to access the database are protected and stored in an encrypted config file. The config file cannot be used on another device, because the encryption mechanisms required to encrypt and unencrypt the config file are device specific. Use NTFS permissions to provide an additional layer of security to the config file. This prevents unauthorized users from being able to see or access the encrypted file.

Without the connection string to the database, the only way to gain access to the Data Center Discovery scan engine would be to acquire local logon administrative rights to the server that houses the scan engine database.

Securing physical access is beyond the scope of this document.

**Internet connectivity and updates:** The operation of the scan engine doesn't require external internet access, but access to the internet can be used as part of normal operations. By default, the scanning server makes a single license request once every 24 hours outside the network to a licensing server, but this is optional and is not required. During the install, if a lack of internet access blocks the automatic retrieval of license information, you have the option of manually registering the license through the UI.

Data Center Discovery doesn't download any configuration information or update information from the internet. All software updates must be installed by your organization.

**Signature checks (certificate chain checking):** Several measures are built-in for security, including specialized .NET signatures, to ensure the integrity of each binary/DLL in the release. When the scan engine is started, the cryptographic signatures associated with the component DLLs are checked to ensure that a trust-chain can be established from the signing certificate to the root CA.

If all certificates are published to the local certificate store, then access to the internet is not required for certificate retrieval operations. The only occasion when additional information is required is when checking for Certificate Revocation Lists (CRLs). CRLs are typically retrieved through internet queries to the associated CA registration point (provided by the certificate).

The default operation for the scan engine is to check certificates against online revocation lists. Failure to retrieve the CRL causes the signature check to fail. This means that the DLLs will not be loaded and preclude normal operation.

This operation is controlled by a configuration key in the scan engine configuration file. The key is called **Revocation Mode** and the available values are:

- **Online**: Build and test the certificate chain and test the certificate against the online CRLs.
- **Offline**: Build and test the certificate chain and test the certificate against locally stored CRLs (if available) but ignore any errors due to the lack of CRLs being available.
- **None**: Build and test the certificate chain, but don't test for revocation.

If an install is made on a server that has no access to the internet, then either **offline** or **none** would be suitable values for this setting prior to starting the scan engine.

**Personal credentials:** The scan engine **does not** collect any user's credentials when scanning devices in a network.

Information that falls within the Personally Identifiable Information (PII) domain **can** be collected. Customer definitions of PII will vary. Typical information that may fall into this domain are login names (and associated e-mail addresses), information associated with Oracle, and SQL Server application usage.

Data such as credit card information is **not** collected.

# Security recommendations

Ivanti's aim is to ensure optimal security for clients. Refer to the following list to see recommendations for a "defense in depth" security approach:

- **Configure** HTTPS on the server hosting the user interface.
- **Set** IIS to only allow traffic from recognized IP addresses corresponding to administrative traffic.
- **Configure** the network routing of the scanning and database servers to allow only inbound and outbound traffic from **recognized** IP addresses.
- **Deploy** the scan-engine service configuration files to a Windows Server operating system using the NTFS file system; **configure** permissions on each service configuration file to restrict file access to the scan user and the appropriate administrative account.
- **Provision** a separate SQL Server instance to store the scan data on the database server. Do not place this instance on a shared virtualization infrastructure unless all other applications deployed run in the same security context.
- **Restrict** the SQL Server account used by the scan engine to connect only to the database on the instance.
- **Assign** a separate SQL Server user login for administrative staff that is independent of the scanning service account; this user login should have **column-level permissions** added to **deny read/write** on the password column of the credential table within the scan engine database.
- **Disable** the scan engine database logins when scanning and reporting functions are inactive.
- **House** the server hosting the scan engine and its associated storage in a secure computing environment.

# Data policy

Ivanti follows careful processes in relation to client data management before, during, and after engagements.

Before any confidential data is reviewed or exchanged between Ivanti and a client, a Non-Disclosure Agreement (NDA) is put in place. The terms of the NDA govern the subsequent sharing and management of data.

All data gathered by the scan engine remains on the client's server under the control of the client's administration team.

When data must be exchanged outside of the client's network, the client's preferred secure mechanism for document sharing such as an extranet or document management system can be used.

Ivanti's default policy is to **retain** client data so long as the contract is valid. Upon request, Ivanti can dispose of client data immediately after the end of a contract or retain the information internally for future use.

# Appendix A: Key terms

| Term | Definition |
|---|---|
| checksum | A count of the number of bits in a transmission unit that is included with the unit so that the receiver can check to see whether the same number of bits arrived. |
| CLR (common language runtime) | A .NET run-time environment that runs the code and provides services that make the development process easier. |
| connection string | A string that specifies information about the data source and the means of connecting to that data. |
| Cross-site scripting (XSS) | When hackers inject client-side script into web pages: persistent and non-persistent attacks. |
| defense in depth | An IT security model in which multiple layers of security controls are implemented throughout the system life cycle. |
| IIS | Internet Information Services |
| Non-disclosure agreement (NDA) | A non-disclosure agreement (NDA) is a legal contract between at least two parties that outlines confidential material, knowledge, or information that the parties wish to share with one another for certain purposes, but wish to restrict access to or by third parties. It is a contract through which the parties agree not to disclose information covered by the agreement. |
| NTFS | New Technology File System |
| page-level permissions | Differing access levels for different people on different pages. |
| parameterized queries | A query in which placeholders are used for parameters and the parameter values are supplied at the time of execution. |
| session hijacking | The exploitation of a computer session to gain unauthorized access to information or services. |
| SQL injection | A code-injection technique that is used to attack data-driven applications, in which malicious statements are inserted for execution. |
| stored procedures | Prepared SQL code that can be reused. |
| target device | The device that the scan engine attempts to discover and scan. |