



Red Hat Enterprise Linux 9.0

Preparing for disaster recovery with Identity Management

Mitigating the effects of server and data loss scenarios in IdM environments

Red Hat Enterprise Linux 9.0 Preparing for disaster recovery with Identity Management

Mitigating the effects of server and data loss scenarios in IdM environments

Legal Notice

Copyright © 2024 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux[®] is the registered trademark of Linus Torvalds in the United States and other countries.

Java[®] is a registered trademark of Oracle and/or its affiliates.

XFS[®] is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL[®] is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js[®] is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack[®] Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

Server and data loss scenarios, for example due to a hardware failure, are the highest risks in IT environments. In a Red Hat Identity Management (IdM) topology, you can configure replication with other servers, use virtual machine (VM) snapshots, and IdM backups to mitigate the effects of these situations.

Table of Contents

MAKING OPEN SOURCE MORE INCLUSIVE	3
PROVIDING FEEDBACK ON RED HAT DOCUMENTATION	4
CHAPTER 1. DISASTER RECOVERY TOOLS IN IDM	5
CHAPTER 2. DISASTER SCENARIOS IN IDM	6
CHAPTER 3. PREPARING FOR SERVER LOSS WITH REPLICATION	7
3.1. GUIDELINES FOR CONNECTING IDM REPLICAS IN A TOPOLOGY	7
3.2. REPLICA TOPOLOGY EXAMPLES	8
3.3. PROTECTING IDM CA DATA	9
CHAPTER 4. PREPARING FOR DATA LOSS WITH VM SNAPSHOTS	11
CHAPTER 5. PREPARING FOR DATA LOSS WITH IDM BACKUPS	12
5.1. IDM BACKUP TYPES	12
5.2. NAMING CONVENTIONS FOR IDM BACKUP FILES	12
5.3. CONSIDERATIONS WHEN CREATING A BACKUP	13
5.4. CREATING AN IDM BACKUP	13
5.5. CREATING A GPG2-ENCRYPTED IDM BACKUP	15
5.6. CREATING A GPG2 KEY	15
CHAPTER 6. BACKING UP IDM SERVERS USING ANSIBLE PLAYBOOKS	18
6.1. PREPARING YOUR ANSIBLE CONTROL NODE FOR MANAGING IDM	18
6.2. USING ANSIBLE TO CREATE A BACKUP OF AN IDM SERVER	20
6.3. USING ANSIBLE TO CREATE A BACKUP OF AN IDM SERVER ON YOUR ANSIBLE CONTROLLER	21
6.4. USING ANSIBLE TO COPY A BACKUP OF AN IDM SERVER TO YOUR ANSIBLE CONTROLLER	23
6.5. USING ANSIBLE TO COPY A BACKUP OF AN IDM SERVER FROM YOUR ANSIBLE CONTROLLER TO THE IDM SERVER	24
6.6. USING ANSIBLE TO REMOVE A BACKUP FROM AN IDM SERVER	26

MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see [our CTO Chris Wright's message](#).

In Identity Management, planned terminology replacements include:

- ***block list*** replaces *blacklist*
- ***allow list*** replaces *whitelist*
- ***secondary*** replaces *slave*
- The word *master* is being replaced with more precise language, depending on the context:
 - ***IdM server*** replaces *IdM master*
 - ***CA renewal server*** replaces *CA renewal master*
 - ***CRL publisher server*** replaces *CRL master*
 - ***multi-supplier*** replaces *multi-master*

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

We appreciate your feedback on our documentation. Let us know how we can improve it.

Submitting feedback through Jira (account required)

1. Log in to the [Jira](#) website.
2. Click **Create** in the top navigation bar
3. Enter a descriptive title in the **Summary** field.
4. Enter your suggestion for improvement in the **Description** field. Include links to the relevant parts of the documentation.
5. Click **Create** at the bottom of the dialogue.

CHAPTER 1. DISASTER RECOVERY TOOLS IN IDM

A good disaster recovery strategy combines the following tools to recover from a disaster as soon as possible with minimal data loss:

Replication

Replication copies database contents between IdM servers. If an IdM server fails, you can replace the lost server by creating a new replica based on one of the remaining servers.

Virtual machine (VM) snapshots

A snapshot is a view of a VM's operating system and applications on any or all available disks at a given point in time. After taking a VM snapshot, you can use it to return a VM and its IdM data to a previous state.

IdM backups

The **ipa-backup** utility allows you to take a backup of an IdM server's configuration files and its data. You can later use a backup to restore an IdM server to a previous state.

CHAPTER 2. DISASTER SCENARIOS IN IDM

There are two main classes of disaster scenarios: *server loss* and *data loss*.

Table 2.1. Server loss vs. data loss

Disaster type	Example causes	How to prepare
Server loss: The IdM deployment loses one or several servers.	<ul style="list-style-type: none">● Hardware malfunction	<ul style="list-style-type: none">● Preparing for server loss with replication
Data loss: IdM data is unexpectedly modified on a server, and the change is propagated to other servers.	<ul style="list-style-type: none">● A user accidentally deletes data● A software bug modifies data	<ul style="list-style-type: none">● Preparing for data loss with VM snapshots● Preparing for data loss with IdM backups

CHAPTER 3. PREPARING FOR SERVER LOSS WITH REPLICATION

Follow these guidelines to establish a replication topology that will allow you to respond to losing a server.

This section covers the following topics:

- [Connecting the replicas in a topology](#)
- [Replica topology examples](#)
- [Protecting IdM CA data](#)

3.1. GUIDELINES FOR CONNECTING IDM REPLICAS IN A TOPOLOGY

Connect each replica to at least two other replicas

Configuring additional replication agreements ensures that information is replicated not just between the initial replica and the first server you installed, but between other replicas as well.

Connect a replica to a maximum of four other replicas (not a hard requirement)

A large number of replication agreements per server does not add significant benefits. A receiving replica can only be updated by one other replica at a time and meanwhile, the other replication agreements are idle. More than four replication agreements per replica typically means a waste of resources.



NOTE

This recommendation applies to both certificate replication and domain replication agreements.

There are two exceptions to the limit of four replication agreements per replica:

- You want failover paths if certain replicas are not online or responding.
- In larger deployments, you want additional direct links between specific nodes.

Configuring a high number of replication agreements can have a negative impact on overall performance: when multiple replication agreements in the topology are sending updates, certain replicas can experience a high contention on the changelog database file between incoming updates and the outgoing updates.

If you decide to use more replication agreements per replica, ensure that you do not experience replication issues and latency. However, note that large distances and high numbers of intermediate nodes can also cause latency problems.

Connect the replicas in a data center with each other

This ensures domain replication within the data center.

Connect each data center to at least two other data centers

This ensures domain replication between data centers.

Connect data centers using at least a pair of replication agreements

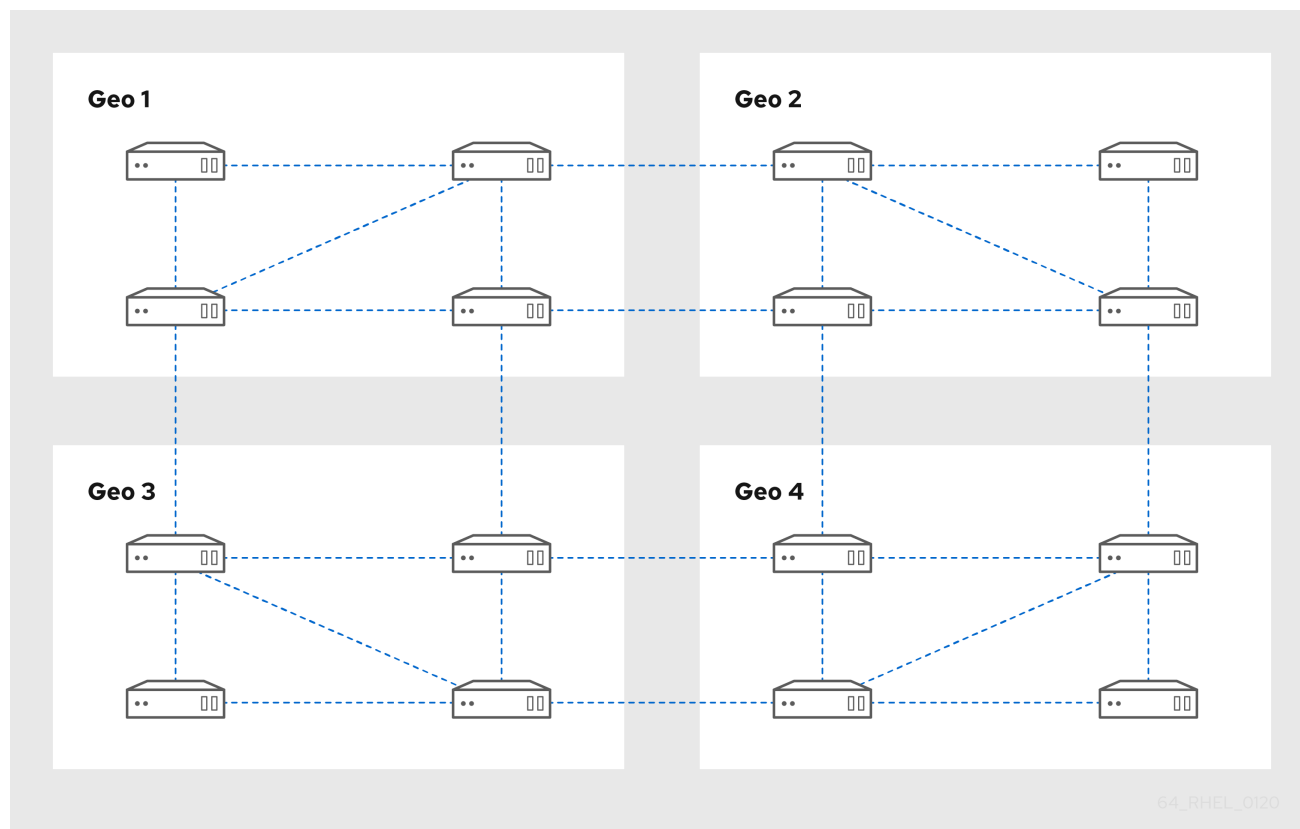
If data centers A and B have a replication agreement from A1 to B1, having a replication agreement from A2 to B2 ensures that if one of the servers is down, the replication can continue between the two data centers.

3.2. REPLICA TOPOLOGY EXAMPLES

The figures below show examples of Identity Management (IdM) topologies based on the guidelines for creating a reliable topology.

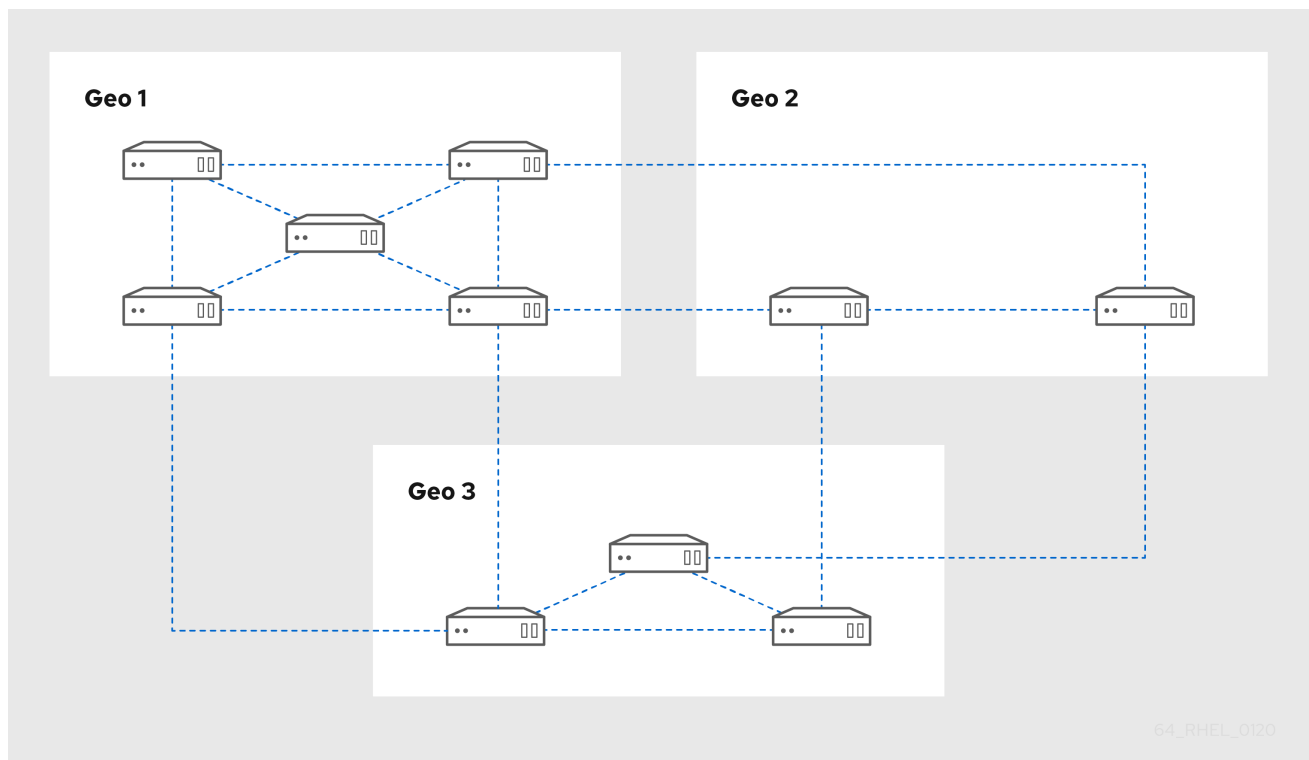
[Replica Topology Example 1](#) shows four data centers, each with four servers. The servers are connected with replication agreements.

Figure 3.1. Replica Topology Example 1



[Replica Topology Example 2](#) shows three data centers, each with a different number of servers. The servers are connected with replication agreements.

Figure 3.2. Replica Topology Example 2



64_RHEL_0120

3.3. PROTECTING IDM CA DATA

If your deployment contains the integrated IdM Certificate Authority (CA), install several CA replicas so you can create additional CA replicas if one is lost.

Procedure

1. Configure three or more replicas to provide CA services.
 - a. To install a new replica with CA services, run **ipa-replica-install** with the **--setup-ca** option.

```
[root@server ~]# ipa-replica-install --setup-ca
```

- b. To install CA services on a preexisting replica, run **ipa-ca-install**.

```
[root@replica ~]# ipa-ca-install
```

2. Create CA replication agreements between your CA replicas.

```
[root@careplica1 ~]# ipa topologysegment-add
Suffix name: ca
Left node: ca-replica1.example.com
Right node: ca-replica2.example.com
Segment name [ca-replica1.example.com-to-ca-replica2.example.com]: new_segment
-----
Added segment "new_segment"
-----
Segment name: new_segment
Left node: ca-replica1.example.com
Right node: ca-replica2.example.com
Connectivity: both
```

**WARNING**

If only one server provides CA services and it is damaged, the entire environment will be lost. If you use the IdM CA, Red Hat **strongly recommends** having three or more replicas with CA services installed, with CA replication agreements between them.

Additional resources

- [Planning your CA services.](#)
- [Installing an IdM replica .](#)
- [Planning the replica topology.](#)

CHAPTER 4. PREPARING FOR DATA LOSS WITH VM SNAPSHOTS

Virtual machine (VM) snapshots are an integral component of a data recovery strategy, since they preserve the full state of an IdM server:

- Operating system software and settings
- IdM software and settings
- IdM customer data

Preparing a VM snapshot of an IdM Certificate Authority (CA) replica allows you to rebuild an entire IdM deployment after a disaster.



WARNING

If your environment uses the integrated CA, a snapshot of a replica *without a CA* will not be sufficient for rebuilding a deployment, because certificate data will not be preserved.

Similarly, if your environment uses the IdM Key Recovery Authority (KRA), make sure you create snapshots of a KRA replica, or you may lose the storage key.

Red Hat recommends creating snapshots of a VM that has all of the IdM server roles installed which are in use in your deployment: CA, KRA, DNS.

Prerequisites

- A hypervisor capable of hosting RHEL VMs.

Procedure

1. Configure at least one **CA replica** in the deployment to run inside a VM.
 - a. If IdM DNS or KRA are used in your environment, consider installing DNS and KRA services on this replica as well.
 - b. Optionally, configure this VM replica as a [hidden replica](#).
2. Periodically shutdown this VM, take a full snapshot of it, and bring it back online so it continues to receive replication updates. If the VM is a hidden replica, IdM Clients will not be disrupted during this procedure.

Additional resources

- [Which hypervisors are certified to run Red Hat Enterprise Linux?](#)
- [The hidden replica mode.](#)

CHAPTER 5. PREPARING FOR DATA LOSS WITH IDM BACKUPS

IdM provides the **ipa-backup** utility to backup IdM data, and the **ipa-restore** utility to restore servers and data from those backups.

This section covers the following topics:

- [IdM backup types](#)
- [Naming conventions for IdM backup files](#)
- [Considerations when creating a backup](#)
- [Creating an IdM backup](#)
- [Creating a GPG2-encrypted IdM backup](#)
- [Creating a GPG2 key](#)



NOTE

Red Hat recommends running backups as often as necessary on a *hidden replica* with all server roles installed, especially the Certificate Authority (CA) role if the environment uses the integrated IdM CA. See [Installing an IdM hidden replica](#).

5.1. IDM BACKUP TYPES

With the **ipa-backup** utility, you can create two types of backups:

Full-server backup

- **Contains** all server configuration files related to IdM, and LDAP data in LDAP Data Interchange Format (LDIF) files
- IdM services must be **offline**.
- **Suitable for** rebuilding an IdM deployment from scratch.

Data-only backup

- **Contains** LDAP data in LDIF files and the replication changelog
- IdM services can be **online or offline**.
- **Suitable for** restoring IdM data to a state in the past

5.2. NAMING CONVENTIONS FOR IDM BACKUP FILES

By default, IdM stores backups as **.tar** archives in subdirectories of the **/var/lib/ipa/backup/** directory.

The archives and subdirectories follow these naming conventions:

Full-server backup

An archive named **ipa-full.tar** in a directory named **ipa-full-*<YEAR-MM-DD-HH-MM-SS>***, with the time specified in GMT time.

```
[root@server ~]# ll /var/lib/ipa/backup/ipa-full-2021-01-29-12-11-46
total 3056
-rw-r--r--. 1 root root    158 Jan 29 12:11 header
-rw-r--r--. 1 root root 3121511 Jan 29 12:11 ipa-full.tar
```

Data-only backup

An archive named **ipa-data.tar** in a directory named **ipa-data-*<YEAR-MM-DD-HH-MM-SS>***, with the time specified in GMT time.

```
[root@server ~]# ll /var/lib/ipa/backup/ipa-data-2021-01-29-12-14-23
total 1072
-rw-r--r--. 1 root root    158 Jan 29 12:14 header
-rw-r--r--. 1 root root 1090388 Jan 29 12:14 ipa-data.tar
```



NOTE

Uninstalling an IdM server does not automatically remove any backup files.

5.3. CONSIDERATIONS WHEN CREATING A BACKUP

The important behaviors and limitations of the **ipa-backup** command include the following:

- By default, the **ipa-backup** utility runs in offline mode, which stops all IdM services. The utility automatically restarts IdM services after the backup is finished.
- A full-server backup must **always** run with IdM services offline, but a data-only backup may be performed with services online.
- By default, the **ipa-backup** utility creates backups on the file system containing the **/var/lib/ipa/backup/** directory. Red Hat recommends creating backups regularly on a file system separate from the production filesystem used by IdM, and archiving the backups to a fixed medium, such as tape or optical storage.
- Consider performing backups on [hidden replicas](#). IdM services can be shut down on hidden replicas without affecting IdM clients.
- The **ipa-backup** utility checks if all of the services used in your IdM cluster, such as a Certificate Authority (CA), Domain Name System (DNS), and Key Recovery Agent (KRA), are installed on the server where you are running the backup. If the server does not have all these services installed, the **ipa-backup** utility exits with a warning, because backups taken on that host would not be sufficient for a full cluster restoration.
For example, if your IdM deployment uses an integrated Certificate Authority (CA), a backup run on a non-CA replica will not capture CA data. Red Hat recommends verifying that the replica where you perform an **ipa-backup** has all of the IdM services used in the cluster installed.

You can bypass the IdM server role check with the **ipa-backup --disable-role-check** command, but the resulting backup will not contain all the data necessary to restore IdM fully.

5.4. CREATING AN IDM BACKUP

Follow this procedure to create a full-server and data-only backup in offline and online modes using the **ipa-backup** command.

Prerequisites

- You must have **root** privileges to run the **ipa-backup** utility.

Procedure

- To create a full-server backup in offline mode, use the **ipa-backup** utility without additional options.

```
[root@server ~]# ipa-backup
Preparing backup on server.example.com
Stopping IPA services
Backing up ipaca in EXAMPLE-COM to LDIF
Backing up userRoot in EXAMPLE-COM to LDIF
Backing up EXAMPLE-COM
Backing up files
Starting IPA service
Backed up to /var/lib/ipa/backup/ipa-full-2020-01-14-11-26-06
The ipa-backup command was successful
```

- To create an offline data-only backup, specify the **--data** option.

```
[root@server ~]# ipa-backup --data
```

- To create a full-server backup that includes IdM log files, use the **--logs** option.

```
[root@server ~]# ipa-backup --logs
```

- To create a data-only backup while IdM services are running, specify both **--data** and **--online** options.

```
[root@server ~]# ipa-backup --data --online
```

NOTE

If the backup fails due to insufficient space in the **/tmp** directory, use the **TMPDIR** environment variable to change the destination for temporary files created by the backup process:

```
[root@server ~]# TMPDIR=/new/location ipa-backup
```

For more details, see [ipa-backup Command Fails to Finish](#).

Verification Steps

- The backup directory contains an archive with the backup.

```
[root@server ~]# ls /var/lib/ipa/backup/ipa-full-2020-01-14-11-26-06
header ipa-full.tar
```

5.5. CREATING A GPG2-ENCRYPTED IDM BACKUP

You can create encrypted backups using GNU Privacy Guard (GPG) encryption. The following procedure creates an IdM backup and encrypts it using a GPG2 key.

Prerequisites

- You have created a GPG2 key. See [Creating a GPG2 key](#).

Procedure

- Create a GPG-encrypted backup by specifying the **--gpg** option.

```
[root@server ~]# ipa-backup --gpg
Preparing backup on server.example.com
Stopping IPA services
Backing up ipaca in EXAMPLE-COM to LDIF
Backing up userRoot in EXAMPLE-COM to LDIF
Backing up EXAMPLE-COM
Backing up files
Starting IPA service
Encrypting /var/lib/ipa/backup/ipa-full-2020-01-13-14-38-00/ipa-full.tar
Backed up to /var/lib/ipa/backup/ipa-full-2020-01-13-14-38-00
The ipa-backup command was successful
```

Verification Steps

- Ensure that the backup directory contains an encrypted archive with a **.gpg** file extension.

```
[root@server ~]# ls /var/lib/ipa/backup/ipa-full-2020-01-13-14-38-00
header ipa-full.tar.gpg
```

Additional resources

- [Creating a backup](#).

5.6. CREATING A GPG2 KEY

The following procedure describes how to generate a GPG2 key to use with encryption utilities.

Prerequisites

- You need **root** privileges.

Procedure

1. Install and configure the **pinentry** utility.

```
[root@server ~]# dnf install pinentry
[root@server ~]# mkdir ~/.gnupg -m 700
[root@server ~]# echo "pinentry-program /usr/bin/pinentry-curses" >> ~/.gnupg/gpg-agent.conf
```

2. Create a **key-input** file used for generating a GPG keypair with your preferred details. For example:

```
[root@server ~]# cat >key-input <<EOF
%echo Generating a standard key
Key-Type: RSA
Key-Length: 2048
Name-Real: GPG User
Name-Comment: first key
Name-Email: root@example.com
Expire-Date: 0
%commit
%echo Finished creating standard key
EOF
```

3. (Optional) By default, GPG2 stores its keyring in the `~/.gnupg` file. To use a custom keyring location, set the **GNUPGHOME** environment variable to a directory that is only accessible by root.

```
[root@server ~]# export GNUPGHOME=/root/backup

[root@server ~]# mkdir -p $GNUPGHOME -m 700
```

4. Generate a new GPG2 key based on the contents of the **key-input** file.

```
[root@server ~]# gpg2 --batch --gen-key key-input
```

5. Enter a passphrase to protect the GPG2 key. You use this passphrase to access the private key for decryption.

```

Please enter the passphrase to
protect your new key

Passphrase: <passphrase>

<OK>          <Cancel>
```

6. Confirm the correct passphrase by entering it again.

```

Please re-enter this passphrase

Passphrase: <passphrase>

<OK>          <Cancel>
```

7. Verify that the new GPG2 key was created successfully.

```
gpg: keybox '/root/backup/pubring.kbx' created
gpg: Generating a standard key
gpg: /root/backup/trustdb.gpg: trustdb created
```

```
gpg: key BF28FFA302EF4557 marked as ultimately trusted
gpg: directory '/root/backup/openpgp-revocs.d' created
gpg: revocation certificate stored as '/root/backup/openpgp-
revocs.d/8F6FCF10C80359D5A05AED67BF28FFA302EF4557.rev'
gpg: Finished creating standard key
```

Verification Steps

- List the GPG keys on the server.

```
[root@server ~]# gpg2 --list-secret-keys
gpg: checking the trustdb
gpg: marginals needed: 3 completes needed: 1 trust model: pgp
gpg: depth: 0 valid: 1 signed: 0 trust: 0-, 0q, 0n, 0m, 0f, 1u
/root/backup/pubring.kbx
-----
sec  rsa2048 2020-01-13 [SCEA]
      8F6FCF10C80359D5A05AED67BF28FFA302EF4557
uid      [ultimate] GPG User (first key) <root@example.com>
```

Additional resources

- [GNU Privacy Guard](#)

CHAPTER 6. BACKING UP IDM SERVERS USING ANSIBLE PLAYBOOKS

Using the **ipabackup** Ansible role, you can automate backing up an IdM server and transferring backup files between servers and your Ansible controller.

This section covers the following topics:

- [Preparing your Ansible control node for managing IdM](#)
- [Using Ansible to create a backup of an IdM server](#)
- [Using Ansible to create a backup of an IdM server on your Ansible controller](#)
- [Using Ansible to copy a backup of an IdM server to your Ansible controller](#)
- [Using Ansible to copy a backup of an IdM server from your Ansible controller to the IdM server](#)
- [Using Ansible to remove a backup from an IdM server](#)

6.1. PREPARING YOUR ANSIBLE CONTROL NODE FOR MANAGING IDM

As a system administrator managing Identity Management (IdM), when working with Red Hat Ansible Engine, it is good practice to do the following:

- Create a subdirectory dedicated to Ansible playbooks in your home directory, for example **~/MyPlaybooks**.
- Copy and adapt sample Ansible playbooks from the **/usr/share/doc/ansible-freeipa/*** and **/usr/share/doc/rhel-system-roles/*** directories and subdirectories into your **~/MyPlaybooks** directory.
- Include your inventory file in your **~/MyPlaybooks** directory.

By following this practice, you can find all your playbooks in one place and you can run your playbooks without invoking root privileges.



NOTE

You only need **root** privileges on the managed nodes to execute the **ipaserver**, **ipareplica**, **ipaclient**, **ipabackup**, **ipasmartcard_server** and **ipasmartcard_client** **ansible-freeipa** roles. These roles require privileged access to directories and the **dnf** software package manager.

Follow this procedure to create the **~/MyPlaybooks** directory and configure it so that you can use it to store and run Ansible playbooks.

Prerequisites

- You have installed an IdM server on your managed nodes, **server.idm.example.com** and **replica.idm.example.com**.

- You have configured DNS and networking so you can log in to the managed nodes, *server.idm.example.com* and *replica.idm.example.com*, directly from the control node.
- You know the IdM **admin** password.

Procedure

1. Create a directory for your Ansible configuration and playbooks in your home directory:

```
$ mkdir ~/MyPlaybooks/
```

2. Change into the *~/MyPlaybooks/* directory:

```
$ cd ~/MyPlaybooks
```

3. Create the *~/MyPlaybooks/ansible.cfg* file with the following content:

```
[defaults]
inventory = /home/your_username/MyPlaybooks/inventory

[privilege_escalation]
become=True
```

4. Create the *~/MyPlaybooks/inventory* file with the following content:

```
[ipaserver]
server.idm.example.com

[ipareplicas]
replica1.idm.example.com
replica2.idm.example.com

[ipacluster:children]
ipaserver
ipareplicas

[ipacluster:vars]
ipaadmin_password=SomeADMINpassword

[ipaclients]
ipaclient1.example.com
ipaclient2.example.com

[ipaclients:vars]
ipaadmin_password=SomeADMINpassword
```

This configuration defines two host groups, **eu** and **us**, for hosts in these locations. Additionally, this configuration defines the **ipaserver** host group, which contains all hosts from the **eu** and **us** groups.

5. [Optional] Create an SSH public and private key. To simplify access in your test environment, do not set a password on the private key:

```
$ ssh-keygen
```

6. Copy the SSH public key to the IdM **admin** account on each managed node:

```
$ ssh-copy-id admin@server.idm.example.com
$ ssh-copy-id admin@replica.idm.example.com
```

You must enter the IdM **admin** password when you enter these commands.

Additional resources

- [Installing an Identity Management server using an Ansible playbook](#) .
- [How to build your inventory](#) .

6.2. USING ANSIBLE TO CREATE A BACKUP OF AN IDM SERVER

The following procedure describes how to use the `ipabackup` role in an Ansible playbook to create a backup of an IdM server and store it on the IdM server.

Prerequisites

- You have configured your Ansible control node to meet the following requirements:
 - You are using Ansible version 2.14 or later.
 - You have installed the [ansible-freeipa](#) package on the Ansible controller.
 - The example assumes that in the `~/MyPlaybooks/` directory, you have created an [Ansible inventory file](#) with the fully-qualified domain name (FQDN) of the IdM server.
 - The example assumes that the `secret.yml` Ansible vault stores your `ipadmin_password`.

Procedure

1. Navigate to the `~/MyPlaybooks/` directory:

```
$ cd ~/MyPlaybooks/
```

2. Make a copy of the `backup-server.yml` file located in the `/usr/share/doc/ansible-freeipa/playbooks` directory:

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/backup-server.yml backup-my-server.yml
```

3. Open the `backup-my-server.yml` Ansible playbook file for editing.
4. Adapt the file by setting the `hosts` variable to a host group from your inventory file. In this example, set it to the `ipaserver` host group:

```
---
- name: Playbook to backup IPA server
  hosts: ipaserver
  become: true
```

```
roles:
- role: ipabackup
  state: present
```

5. Save the file.
6. Run the Ansible playbook, specifying the inventory file and the playbook file:

```
$ ansible-playbook --vault-password-file=password_file -v -i ~/MyPlaybooks/inventory
backup-my-server.yml
```

Verification steps

1. Log into the IdM server that you have backed up.
2. Verify that the backup is in the **/var/lib/ipa/backup** directory.

```
[root@server ~]# ls /var/lib/ipa/backup/
ipa-full-2021-04-30-13-12-00
```

Additional resources

- For more sample Ansible playbooks that use the **ipabackup** role, see:
 - The **README.md** file in the **/usr/share/doc/ansible-freeipa/roles/ipabackup** directory.
 - The **/usr/share/doc/ansible-freeipa/playbooks/** directory.

6.3. USING ANSIBLE TO CREATE A BACKUP OF AN IDM SERVER ON YOUR ANSIBLE CONTROLLER

The following procedure describes how to use the **ipabackup** role in an Ansible playbook to create a backup of an IdM server and automatically transfer it on your Ansible controller. Your backup file name begins with the host name of the IdM server.

Prerequisites

- You have configured your Ansible control node to meet the following requirements:
 - You are using Ansible version 2.14 or later.
 - You have installed the **ansible-freeipa** package on the Ansible controller.
 - The example assumes that in the **~/MyPlaybooks/** directory, you have created an [Ansible inventory file](#) with the fully-qualified domain name (FQDN) of the IdM server.
 - The example assumes that the **secret.yml** Ansible vault stores your **ipadmin_password**.

Procedure

1. To store the backups, create a subdirectory in your home directory on the Ansible controller.

```
$ mkdir ~/ipabackups
```

2. Navigate to the **~/MyPlaybooks/** directory:

```
$ cd ~/MyPlaybooks/
```

3. Make a copy of the **backup-server-to-controller.yml** file located in the **/usr/share/doc/ansible-freeipa/playbooks** directory:

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/backup-server-to-controller.yml backup-my-server-to-my-controller.yml
```

4. Open the **backup-my-server-to-my-controller.yml** file for editing.

5. Adapt the file by setting the following variables:

- a. Set the **hosts** variable to a host group from your inventory file. In this example, set it to the **ipaserver** host group.
- b. (Optional) To maintain a copy of the backup on the IdM server, uncomment the following line:

```
# ipabackup_keep_on_server: yes
```

6. By default, backups are stored in the present working directory of the Ansible controller. To specify the backup directory you created in Step 1, add the **ipabackup_controller_path** variable and set it to the **/home/user/ipabackups** directory.

```
---
- name: Playbook to backup IPA server to controller
  hosts: ipaserver
  become: true
  vars:
    ipabackup_to_controller: yes
    # ipabackup_keep_on_server: yes
    ipabackup_controller_path: /home/user/ipabackups

  roles:
    - role: ipabackup
      state: present
```

7. Save the file.
8. Run the Ansible playbook, specifying the inventory file and the playbook file:

```
$ ansible-playbook --vault-password-file=password_file -v -i ~/MyPlaybooks/inventory backup-my-server-to-my-controller.yml
```

Verification steps

- Verify that the backup is in the **/home/user/ipabackups** directory of your Ansible controller:

```
[user@controller ~]$ ls /home/user/ipabackups
server.idm.example.com_ipa-full-2021-04-30-13-12-00
```

Additional resources

- For more sample Ansible playbooks that use the **ipabackup** role, see:
 - The **README.md** file in the `/usr/share/doc/ansible-freeipa/roles/ipabackup` directory.
 - The `/usr/share/doc/ansible-freeipa/playbooks/` directory.

6.4. USING ANSIBLE TO COPY A BACKUP OF AN IDM SERVER TO YOUR ANSIBLE CONTROLLER

The following procedure describes how to use an Ansible playbook to copy a backup of an IdM server from the IdM server to your Ansible controller.

Prerequisites

- You have configured your Ansible control node to meet the following requirements:
 - You are using Ansible version 2.14 or later.
 - You have installed the **ansible-freeipa** package on the Ansible controller.
 - The example assumes that in the `~/MyPlaybooks/` directory, you have created an [Ansible inventory file](#) with the fully-qualified domain name (FQDN) of the IdM server.
 - The example assumes that the **secret.yml** Ansible vault stores your **ipaadmin_password**.

Procedure

1. To store the backups, create a subdirectory in your home directory on the Ansible controller.

```
$ mkdir ~/ipabackups
```

2. Navigate to the `~/MyPlaybooks/` directory:

```
$ cd ~/MyPlaybooks/
```

3. Make a copy of the **copy-backup-from-server.yml** file located in the `/usr/share/doc/ansible-freeipa/playbooks` directory:

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/copy-backup-from-server.yml copy-backup-from-my-server-to-my-controller.yml
```

4. Open the **copy-my-backup-from-my-server-to-my-controller.yml** file for editing.
5. Adapt the file by setting the following variables:
 - a. Set the **hosts** variable to a host group from your inventory file. In this example, set it to the **ipaserver** host group.
 - b. Set the **ipabackup_name** variable to the name of the **ipabackup** on your IdM server to copy to your Ansible controller.
 - c. By default, backups are stored in the present working directory of the Ansible controller. To specify the directory you created in Step 1, add the **ipabackup_controller_path** variable and set it to the `/home/user/ipabackups` directory.

```

---
- name: Playbook to copy backup from IPA server
  hosts: ipaserver
  become: true
  vars:
    ipabackup_name: ipa-full-2021-04-30-13-12-00
    ipabackup_to_controller: yes
    ipabackup_controller_path: /home/user/ipabackups

  roles:
    - role: ipabackup
      state: present

```

6. Save the file.

7. Run the Ansible playbook, specifying the inventory file and the playbook file:

```
$ ansible-playbook --vault-password-file=password_file -v -i ~/MyPlaybooks/inventory copy-backup-from-my-server-to-my-controller.yml
```

NOTE

To copy **all** IdM backups to your controller, set the **ipabackup_name** variable in the Ansible playbook to **all**:

```

vars:
  ipabackup_name: all
  ipabackup_to_controller: yes

```

For an example, see the **copy-all-backups-from-server.yml** Ansible playbook in the **/usr/share/doc/ansible-freeipa/playbooks** directory.

Verification steps

- Verify your backup is in the **/home/user/ipabackups** directory on your Ansible controller:

```
[user@controller ~]$ ls /home/user/ipabackups
server.idm.example.com_ipa-full-2021-04-30-13-12-00
```

Additional resources

- The **README.md** file in the **/usr/share/doc/ansible-freeipa/roles/ipabackup** directory.
- The **/usr/share/doc/ansible-freeipa/playbooks/** directory.

6.5. USING ANSIBLE TO COPY A BACKUP OF AN IDM SERVER FROM YOUR ANSIBLE CONTROLLER TO THE IDM SERVER

The following procedure describes how to use an Ansible playbook to copy a backup of an IdM server from your Ansible controller to the IdM server.

Prerequisites

- You have configured your Ansible control node to meet the following requirements:
 - You are using Ansible version 2.14 or later.
 - You have installed the [ansible-freeipa](#) package on the Ansible controller.
 - The example assumes that in the `~/MyPlaybooks/` directory, you have created an [Ansible inventory file](#) with the fully-qualified domain name (FQDN) of the IdM server.
 - The example assumes that the `secret.yml` Ansible vault stores your `ipadmin_password`.

Procedure

1. Navigate to the `~/MyPlaybooks/` directory:

```
$ cd ~/MyPlaybooks/
```

2. Make a copy of the `copy-backup-from-controller.yml` file located in the `/usr/share/doc/ansible-freeipa/playbooks` directory:

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/copy-backup-from-controller.yml copy-backup-from-my-controller-to-my-server.yml
```

3. Open the `copy-my-backup-from-my-controller-to-my-server.yml` file for editing.
4. Adapt the file by setting the following variables:
 - a. Set the `hosts` variable to a host group from your inventory file. In this example, set it to the `ipaserver` host group.
 - b. Set the `ipabackup_name` variable to the name of the `ipabackup` on your Ansible controller to copy to the IdM server.

```
---
- name: Playbook to copy a backup from controller to the IPA server
  hosts: ipaserver
  become: true

  vars:
    ipabackup_name: server.idm.example.com_ipa-full-2021-04-30-13-12-00
    ipabackup_from_controller: yes

  roles:
    - role: ipabackup
      state: copied
```

5. Save the file.
6. Run the Ansible playbook, specifying the inventory file and the playbook file:

```
$ ansible-playbook --vault-password-file=password_file -v -i ~/MyPlaybooks/inventory copy-backup-from-my-controller-to-my-server.yml
```

Additional resources

- The **README.md** file in the `/usr/share/doc/ansible-freeipa/roles/ipabackup` directory.
- The `/usr/share/doc/ansible-freeipa/playbooks/` directory.

6.6. USING ANSIBLE TO REMOVE A BACKUP FROM AN IDM SERVER

The following procedure describes how to use an Ansible playbook to remove a backup from an IdM server.

Prerequisites

- You have configured your Ansible control node to meet the following requirements:
 - You are using Ansible version 2.14 or later.
 - You have installed the [ansible-freeipa](#) package on the Ansible controller.
 - The example assumes that in the `~/MyPlaybooks/` directory, you have created an [Ansible inventory file](#) with the fully-qualified domain name (FQDN) of the IdM server.
 - The example assumes that the `secret.yml` Ansible vault stores your `ipadmin_password`.

Procedure

1. Navigate to the `~/MyPlaybooks/` directory:

```
$ cd ~/MyPlaybooks/
```

2. Make a copy of the **remove-backup-from-server.yml** file located in the `/usr/share/doc/ansible-freeipa/playbooks` directory:

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/remove-backup-from-server.yml remove-backup-from-my-server.yml
```

3. Open the **remove-backup-from-my-server.yml** file for editing.
4. Adapt the file by setting the following variables:
 - a. Set the **hosts** variable to a host group from your inventory file. In this example, set it to the **ipaserver** host group.
 - b. Set the **ipabackup_name** variable to the name of the **ipabackup** to remove from your IdM server.

```
---
- name: Playbook to remove backup from IPA server
  hosts: ipaserver
  become: true

  vars:
    ipabackup_name: ipa-full-2021-04-30-13-12-00

  roles:
    - role: ipabackup
      state: absent
```

5. Save the file.
6. Run the Ansible playbook, specifying the inventory file and the playbook file:

```
$ ansible-playbook --vault-password-file=password_file -v -i ~/MyPlaybooks/inventory  
remove-backup-from-my-server.yml
```



NOTE

To remove **all** IdM backups from the IdM server, set the **ipabackup_name** variable in the Ansible playbook to **all**:

```
vars:  
  ipabackup_name: all
```

For an example, see the **remove-all-backups-from-server.yml** Ansible playbook in the **/usr/share/doc/ansible-freeipa/playbooks** directory.

Additional resources

- The **README.md** file in the **/usr/share/doc/ansible-freeipa/roles/ipabackup** directory.
- The **/usr/share/doc/ansible-freeipa/playbooks/** directory.