

# API Proxy 部署手册

## 1. 适用范围

本文说明如何在管理节点上部署、升级、卸载和排查 API Proxy 服务。发布包支持将 API Proxy 部署在单管理节点和双管理节点上；远程脚本可从任一管理节点作为入口节点发起，必要时会自动探测另一台管理节点并中转执行操作。

## 2. 发布包内容

当前发布包以 `tar.gz` 压缩包形式提供：

- `api-proxy-<version>.tar.gz`

解压后，目录结构主要包含：

```
api-proxy.jar
build-info.properties
README.txt
DEPLOYMENT.zh-CN.pdf
DEPLOYMENT.en-US.pdf
deploy-api-proxy.sh
collect-api-proxy-logs.sh
cleanup-api-proxy-logs.sh
install-api-proxy.sh
deploy/api-proxy-common.sh
```

文件说明：

文件	用途
<code>api-proxy.jar</code>	已处理的可执行服务包。
<code>deploy-api-proxy.sh</code>	远程部署、升级、卸载入口脚本。
<code>install-api-proxy.sh</code>	目标管理节点本地安装、卸载脚本。
<code>collect-api-proxy-logs.sh</code>	收集服务日志、systemd journal、管理节点日志和双管理节点相关信息。
<code>cleanup-api-proxy-logs.sh</code>	清理服务日志和 journal。
<code>deploy/api-proxy-common.sh</code>	部署脚本公共逻辑，包括 SSH、中转和双管理节点拓扑探测。
<code>build-info.properties</code>	版本、构建提交和构建时间。
<code>DEPLOYMENT.zh-CN.pdf</code>	中文部署手册 PDF。
<code>DEPLOYMENT.en-US.pdf</code>	英文部署手册 PDF。
<code>README.txt</code>	发布包快速说明。

### 3. 环境与访问要求

#### 3.1 部署机/执行环境

部署机是可选的，仅在需要远程执行部署、卸载、日志收集或日志清理命令时使用。本质上，它只是一个能够运行这些脚本的 Linux shell 环境，用于对目标管理节点执行安装或运维操作。

- 不需要专门准备一台独立虚拟机。
- 可以是临时 Linux VM、跳板机、运维终端，或者任意一台管理节点。
- 如果你已经把完整发布目录复制到目标管理节点，并准备在目标管理节点本地执行 `install-api-proxy.sh`，就不需要单独准备部署机。

如果使用远程脚本，部署机需要：

- Linux shell 环境。
- `bash`、`ssh`、`sshpass`、`scp`、`curl`。
- `tar`，用于解压发布包。
- 能以 root 用户 SSH 到入口管理节点。

#### 3.2 目标管理节点

目标管理节点需要：

- Java 8 运行时，或通过 `JAVA_BIN` / `JAVA_HOME` 指定可执行文件。
- `systemd` 和 `systemctl`。
- `curl`，用于本地健康检查。
- 预先安装 `qemu-img`，并确保服务启动时可通过 `PATH` 找到；如果安装在非标准路径，请在服务启动前配置 `adapter.backup.qemu-img-path`。
- `mysql` 客户端仅在卸载时清理服务自有表需要；缺失时脚本会跳过数据库清理并继续卸载。
- 端口 `16443/tcp` 对备份客户端或上游调用方可达。

### 4. 默认路径与服务名

项目	默认值
systemd 服务名	<code>api-proxy.service</code>
安装目录	<code>/opt/api-proxy</code>
服务 Jar	<code>/opt/api-proxy/api-proxy.jar</code>
日志目录	<code>/opt/api-proxy/logs</code>
主日志	<code>/opt/api-proxy/logs/api-proxy.log</code>
上传工作目录	<code>/opt/api-proxy/uploads</code>

运行时工作目录	/tmp/api-proxy
TLS 目录	/etc/api-proxy/tls
TLS keystore	/etc/api-proxy/tls/server-keystore.p12
HTTPS 端口	16443
本地健康检查地址	https://127.0.0.1:16443/ovirt-engine/api
证书下载地址	https://<任一管理节点 IP 或 VIP>:16443/ovirt-engine/services/pki-resource

## 5. 解压发布包

当前仅提供 tar.gz 发布包：

```
tar -xzf api-proxy-1.0.0.tar.gz -C /tmp
cd /tmp/api-proxy-1.0.0
```

## 6. 部署、安装、升级与卸载

支持两种方式：

- 远程部署：在部署机上执行 `deploy-api-proxy.sh`，由脚本通过 SSH 完成目标节点安装。
- 本地安装：先把完整发布目录复制到目标管理节点，再在目标节点执行 `install-api-proxy.sh`。

### 6.1 远程部署

推荐使用发布包内的远程部署入口：

```
./deploy-api-proxy.sh --install root@<node-ip>
```

说明：

- 单管理节点部署时，<node-ip> 填该管理节点 IP。
- 双管理节点部署时，<node-ip> 可填写任一管理节点 IP；脚本会自动探测另一台管理节点并完成双管理节点部署。
- 如果未设置 `API_PROXY_SSH_PASSWORD`，脚本会隐藏提示输入 SSH 密码。
- 如果检测到既有部署，脚本会在覆盖前请求确认。
- 安装脚本会在目标节点备份旧 Jar，刷新 systemd unit，然后重启服务。
- 如果检测到 VIP，安装完成后会优先对 VIP 上的健康检查接口发起请求。

非交互方式：

```
export API_PROXY_SSH_PASSWORD='<ssh-password>'
export API_PROXY_ASSUME_YES=1
./deploy-api-proxy.sh --install root@<node-ip>
```

## 6.2 本地安装

如果已经把完整发布目录复制到目标管理节点，也可以直接在该节点执行：

```
sudo ./install-api-proxy.sh --install
```

说明：

- 本地安装必须以 root 用户执行。
- 本地安装不需要单独部署机，也不依赖 `sshpas`。

常用本地覆盖示例：

```
sudo INSTALL_DIR=/data/api-proxy LOG_DIR=/data/api-proxy/logs ./install-api-proxy.sh --install
```

## 6.3 部署后检查

检查接口与证书：

```
curl -k https://<任一管理节点 IP 或 VIP>:16443/ovirt-engine/api  
curl -k https://<任一管理节点 IP 或 VIP>:16443/ovirt-engine/services/pki-resource
```

检查服务状态：

```
systemctl status api-proxy --no-pager -l  
journalctl -u api-proxy -n 100 --no-pager  
tail -n 100 /opt/api-proxy/logs/api-proxy.log
```

## 6.4 卸载

远程卸载：

```
./deploy-api-proxy.sh --uninstall root@<node-ip>
```

本地卸载：

```
sudo ./install-api-proxy.sh --uninstall
```

默认卸载行为：

- 停止并移除 `api-proxy.service`。
- 清理安装目录、日志目录、上传目录和运行时工作目录。
- 删除服务自有数据库表。
- 不删除数据库或 schema 本身。
- 双管理节点部署时，数据库表清理只会在最后一个节点执行一次。

- 如果目标节点缺少 `mysql` 客户端，脚本会跳过数据库清理并继续卸载。

保留部分数据的示例：

```
sudo CLEAN_LOGS=0 CLEAN_UPLOADS=0 ./install-api-proxy.sh --uninstall
```

## 7. Veeam 接入与 Worker 部署

### 7.1 Worker 的作用

在 oVirt / RHV 场景中，Veeam 会通过 Worker 执行数据传输、备份读取和恢复写入等任务。在当前环境中，Veeam 仍按 oVirt 的方式部署和管理 Worker，API Proxy 负责将相关 oVirt API 调用转换为后端管理平台可识别的请求。

从使用方式来看，用户仍然在 Veeam 控制台添加虚拟化平台、部署 Worker、创建备份任务和执行恢复任务；变化仅在于底层对接对象由原生 oVirt 变为后端管理平台与 API Proxy 的组合。

### 7.2 Veeam 连接参数

在 Veeam 中以 oVirt / Red Hat Virtualization 类型添加虚拟化平台时，可参考下表填写参数。

Veeam 配置项	建议填写
平台类型	oVirt / Red Hat Virtualization
地址	任一管理节点 IP 或 VIP；双管理节点部署建议填写 VIP
端口	API Proxy 默认 16443。如果 Veeam 版本只能使用 443，需要将 API Proxy 调整到 443 或做端口转发
用户名	管理平台账号，例如 admin。不要填写 admin@internal，除非环境中的实际账号就是该值
密码	上述管理平台账号的明文密码
证书	使用 API Proxy 自动生成的证书。如果 Veeam 弹出证书信任提示，按界面确认或导入 Root CA

可以用下面的命令验证账号能否通过 API Proxy 换取 oVirt 风格 token：

```
read -r -s MGMT_PASSWORD
curl -k -X POST "https://<任一管理节点 IP 或 VIP>:16443/ovirt-engine/sso/oauth/token" \
-H "Content-Type: application/x-www-form-urlencoded" \
--data-urlencode "grant_type=password" \
--data-urlencode "username=<管理平台账号>" \
--data-urlencode "password=${MGMT_PASSWORD}"
```

### 7.3 关键前提：Worker 必须运行在 Intel 主机上

Veeam Worker 必须运行在 Intel CPU 的计算节点上。如果把 Worker 调度到 AMD 计算节点，可能出现 Worker 虚拟机无法启动、启动后异常退出，或 Veeam 侧一直显示 Worker 不可用等问题。

部署前需要确认目标计算节点 CPU 厂商：

```
lscpu | grep 'Vendor ID'
```

期望输出为：

```
Vendor ID: GenuineIntel
```

如果输出为 AuthenticAMD，不要把该主机作为 Veeam Worker 运行位置。

在混合 CPU 集群中，建议提前准备 Intel-only 的 Worker 运行区域：

- 使用只包含 Intel 主机的集群、主机组或专用资源池。
- 通过调度标签、亲和性策略或运维流程，确保 Worker 不会被调度到 AMD 主机。
- 如果环境中存在自动迁移策略，也要确保 Worker 不会在运行期间迁移到 AMD 主机。

## 7.4 Worker 部署前检查

部署 Worker 前建议确认：

- Veeam Backup & Replication Server 能访问 API Proxy 地址和端口。
- Worker 所在网络能访问 Veeam Server、备份仓库，以及任一管理节点 IP 或 VIP。
- 必须为 Worker 手动分配静态 IP 地址，不要依赖 DHCP 自动分配。
- Worker 所在网络能访问备份/恢复过程中使用的镜像传输地址。
- 目标主存储有足够空间放置 Worker 虚机和临时磁盘。
- 管理平台账号具备查询 VM、创建恢复 VM、创建或挂载磁盘、镜像传输等必要权限。实施阶段建议先使用管理员账号完成验证。
- Veeam Server、管理节点和计算节点时间同步正常。

## 7.5 在 Veeam 中添加 API Proxy 并部署 Worker

以下步骤基于 Veeam Backup & Replication 13.0.1.180。不同版本的界面名称可能略有差异，整体可按 oVirt / Red Hat Virtualization 的对应入口操作。

### 7.5.1 添加虚拟化平台

1. 在 Veeam 控制台进入虚拟化基础设施管理页面。

Veeam Backup and Replication

Home Server

Add Server Edit Server Remove Server Manage Server Rescan Create Veeam Deployment Kit Tools

**Backup Infrastructure**

- Backup Repositories
- External Repositories
- Scale-out Repositories
- WAN Accelerators
- Service Providers
- SureBackup
- Application Groups
- Virtual Labs
- Managed Servers**
- Microsoft Windows
- oVirt KVM

Home Inventory Backup Infrastructure History

3 servers

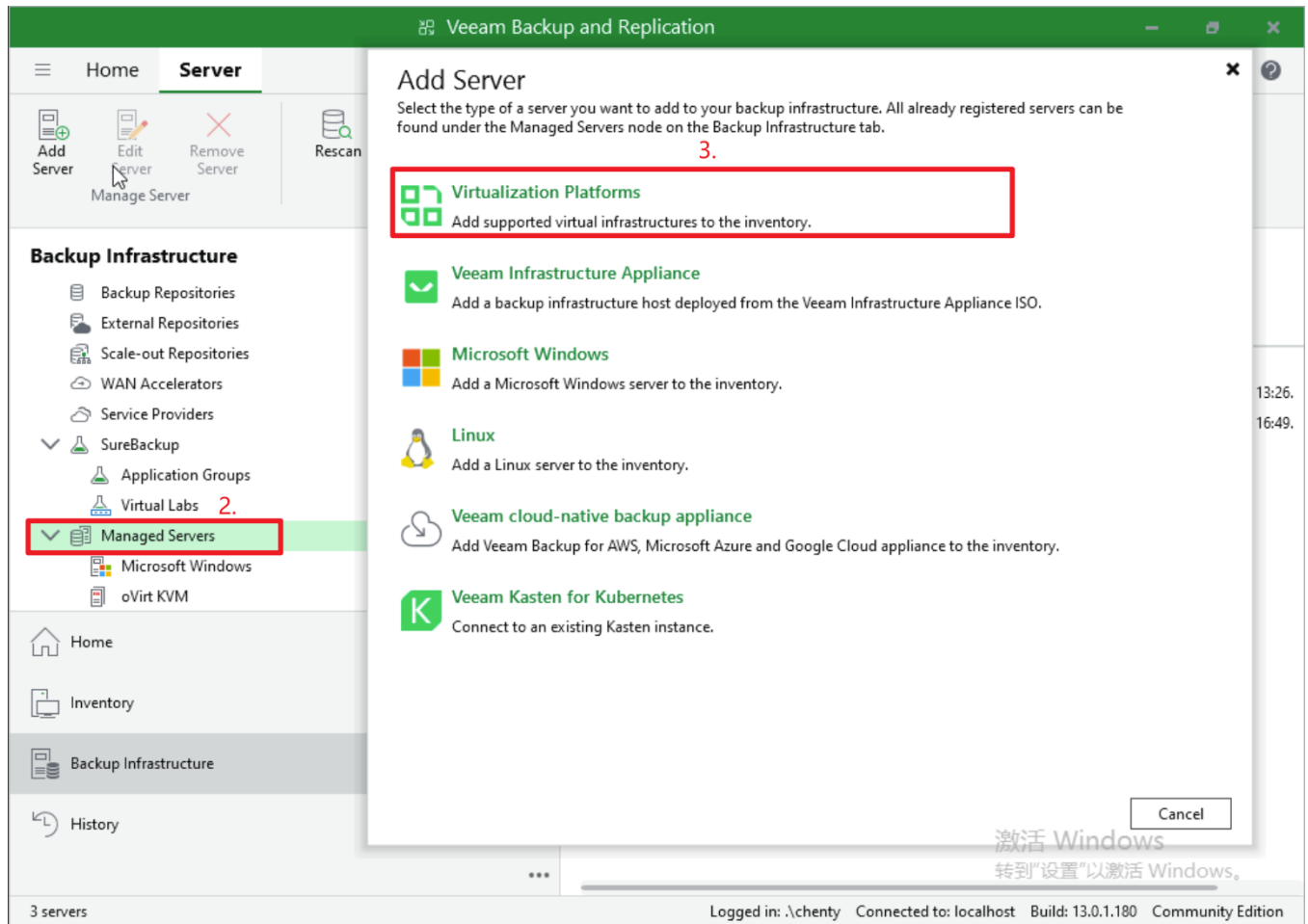
Type in an object name to search for

Name	Type	Description
172-20-19-233	Microsoft Windows server (defau...	Backup server
172.20.19.233	Microsoft Windows server	Created by .\chentay at 2026/3/31 13:26.
172.26.50.9:5443	oVirt KVM Manager	Created by .\chentay at 2026/4/25 16:49.

激活 Windows  
转到“设置”以激活 Windows。

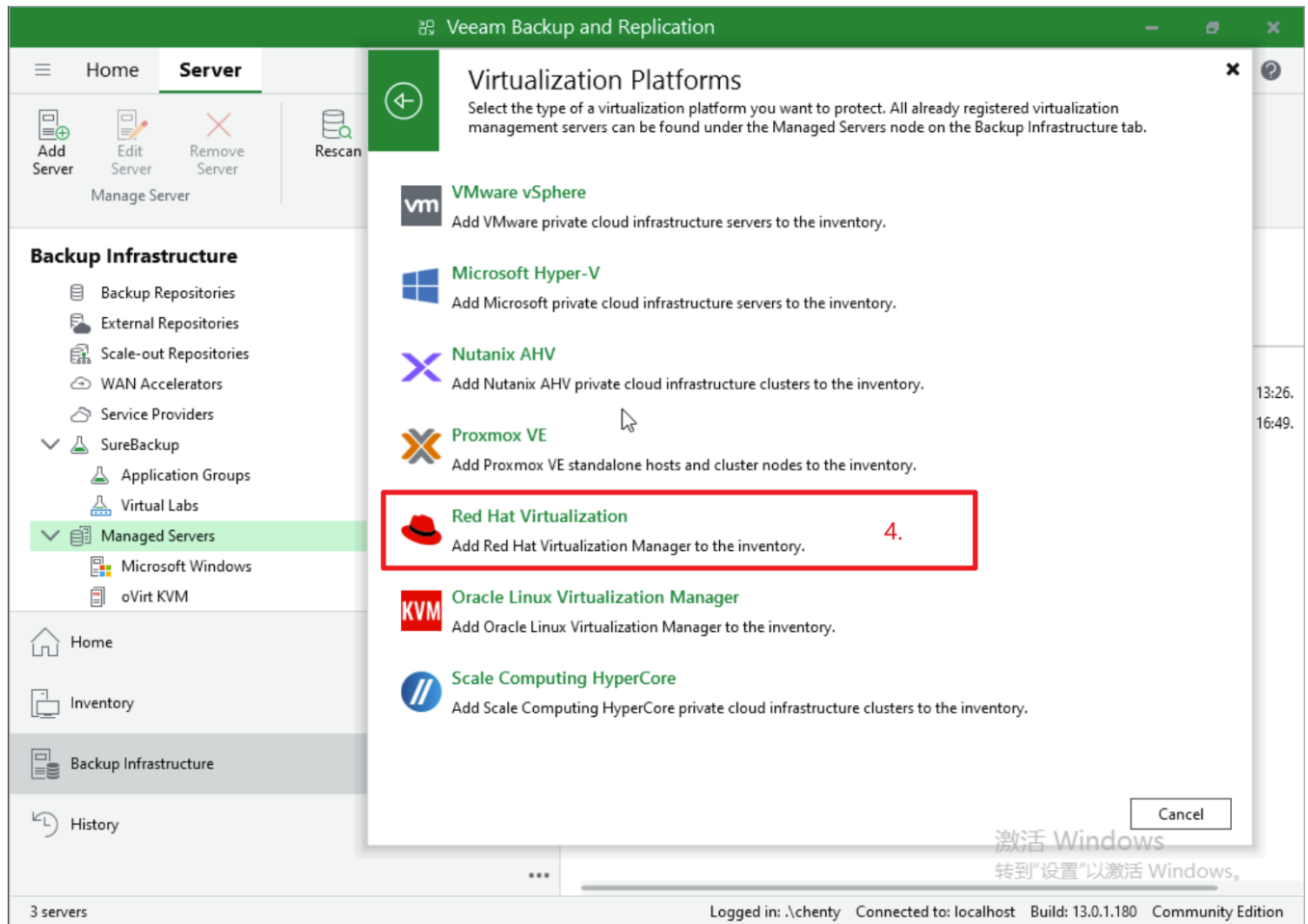
Logged in: .\chentay Connected to: localhost Build: 13.0.1.180 Community Edition

2. 选择添加虚拟化平台。

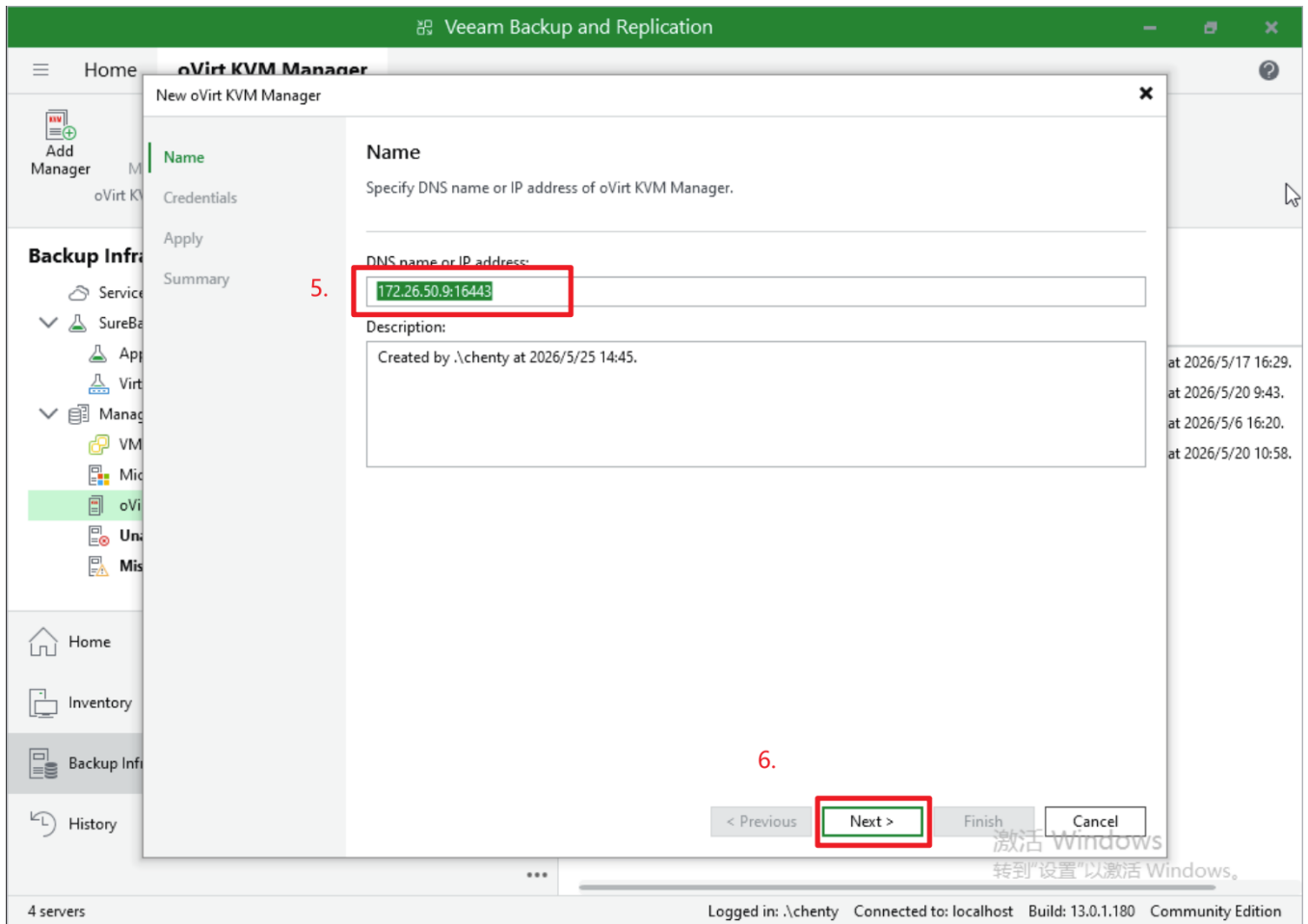


3. 选择 Red Hat Virtualization / oVirt 类型。

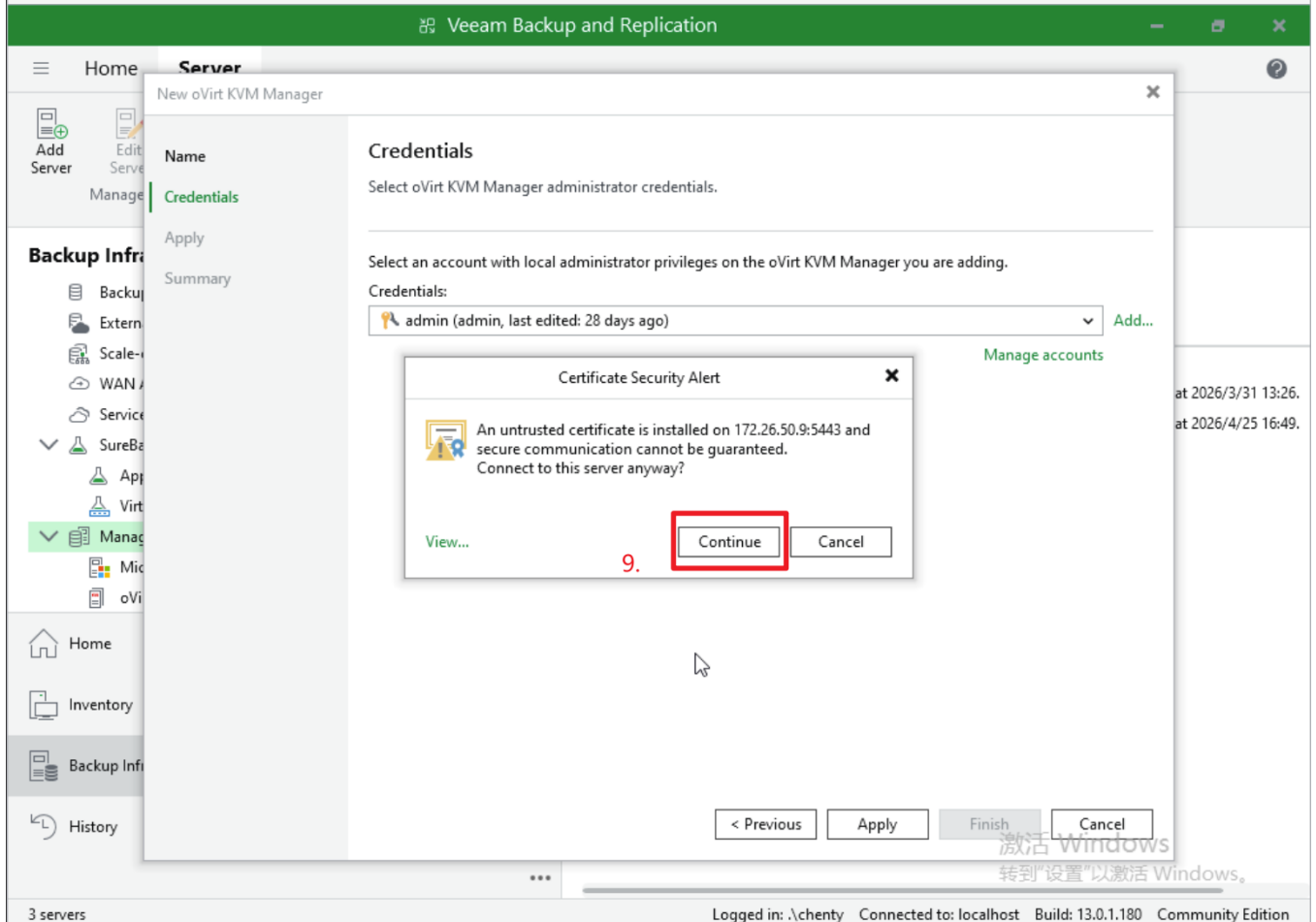
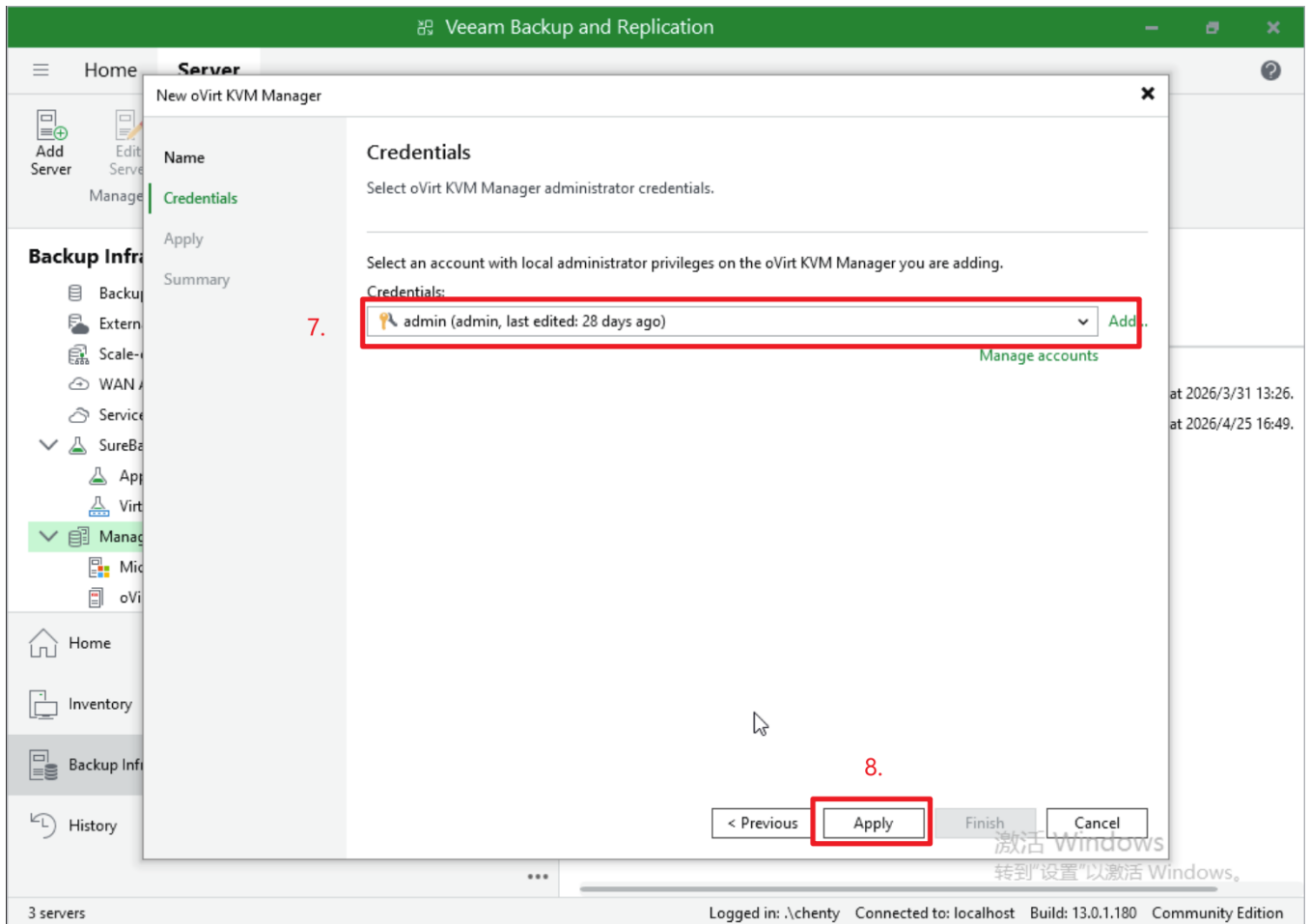




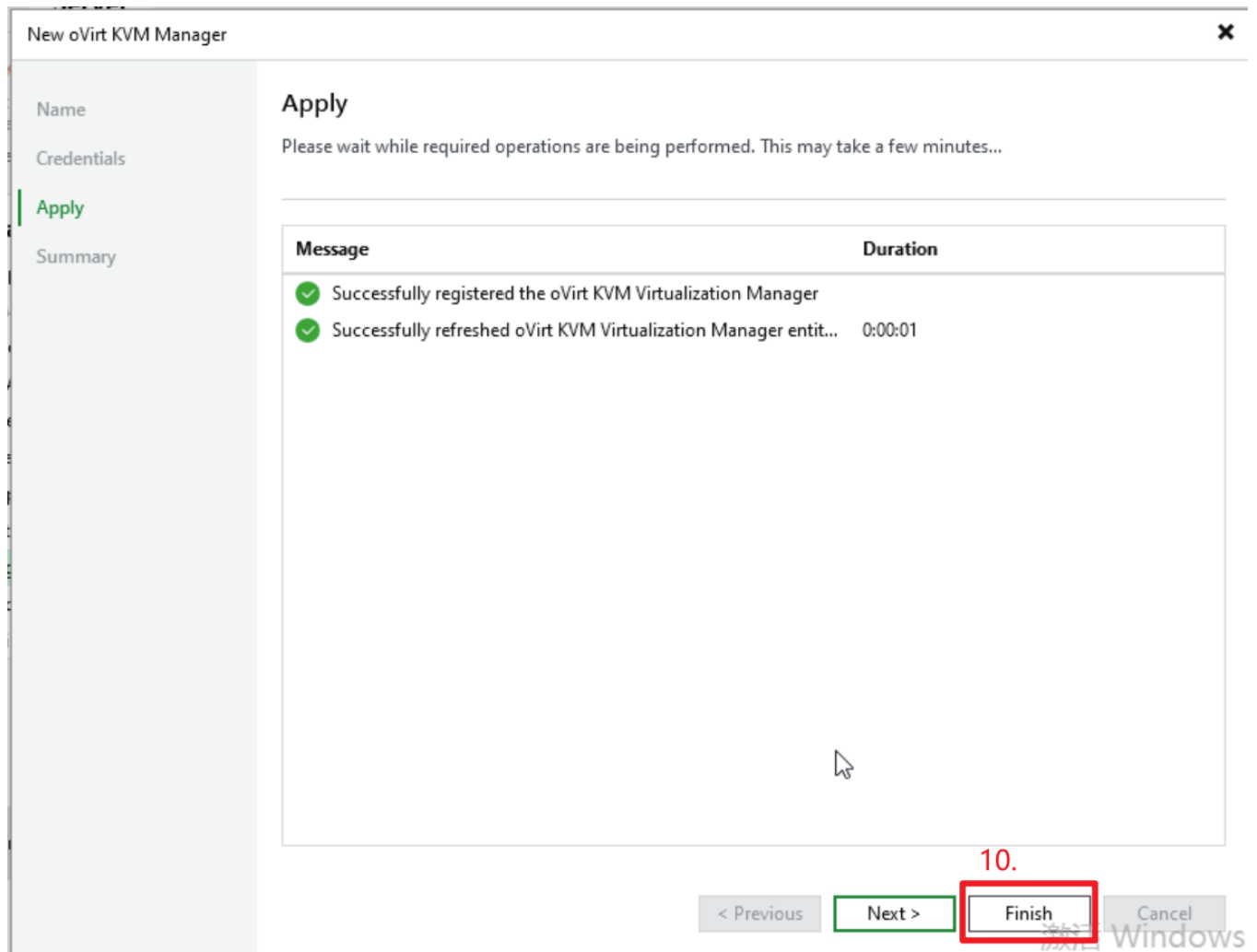
4. 新建 oVirt KVM Manager , 填写 API Proxy 的地址和端口号。



5. 填写管理平台账号和密码，并信任 API Proxy 证书。



6. 选择 **Finish** 后，完成 oVirt KVM Manager 创建。



### 7.5.2 创建 Worker

1. 在 Veeam 控制台进入虚拟化基础设施管理页面。

Veeam Backup and Replication

Home Server

Add Server Edit Server Remove Server Manage Server Rescan Create Veeam Deployment Kit Tools

**Backup Infrastructure**

- Backup Repositories
- External Repositories
- Scale-out Repositories
- WAN Accelerators
- Service Providers
- SureBackup
- Application Groups
- Virtual Labs
- Managed Servers**
- Microsoft Windows
- oVirt KVM

Home Inventory Backup Infrastructure History

3 servers

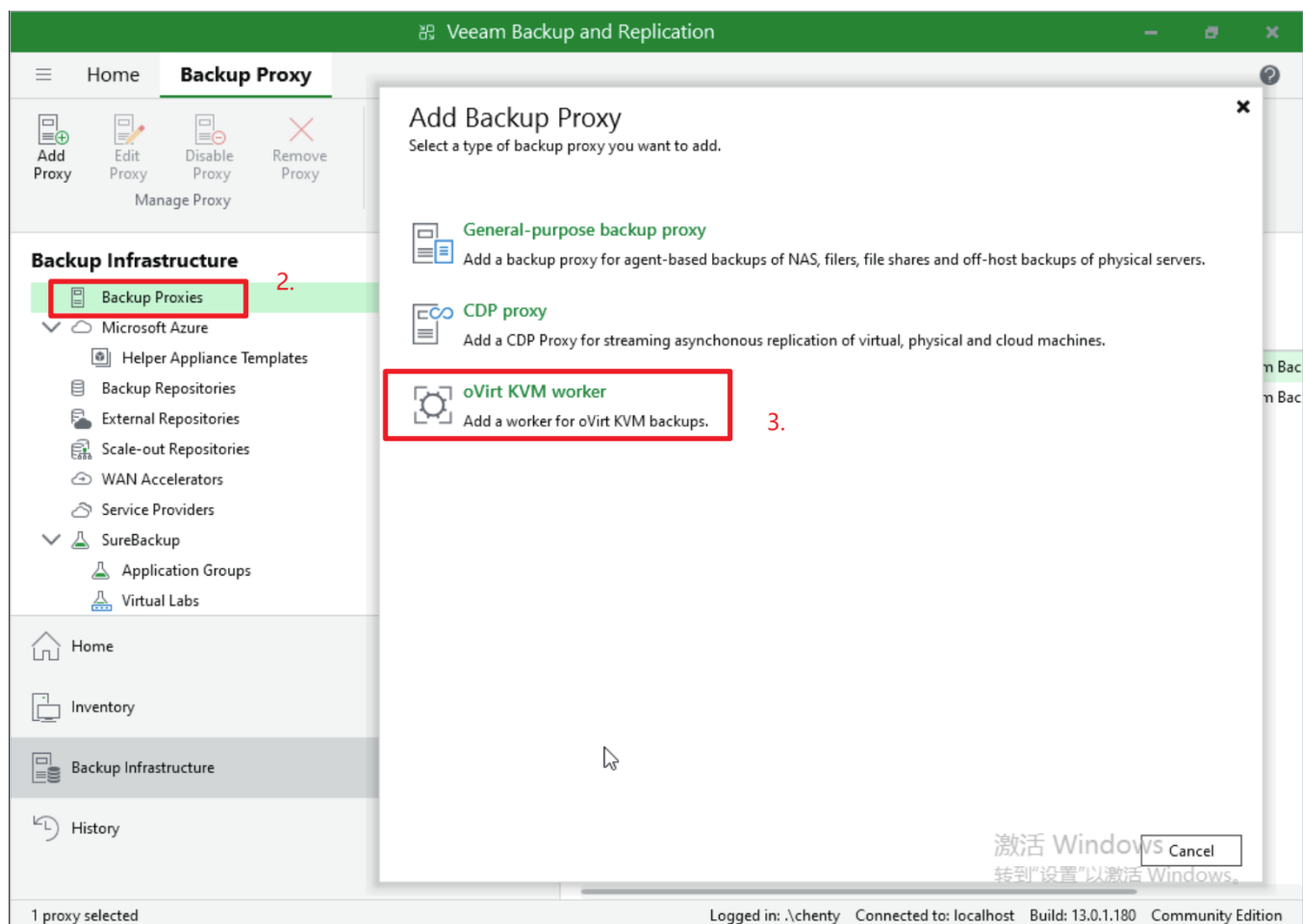
Type in an object name to search for

Name	Type	Description
172-20-19-233	Microsoft Windows server (defau...	Backup server
172.20.19.233	Microsoft Windows server	Created by .\chenthy at 2026/3/31 13:26.
172.26.50.9:5443	oVirt KVM Manager	Created by .\chenthy at 2026/4/25 16:49.

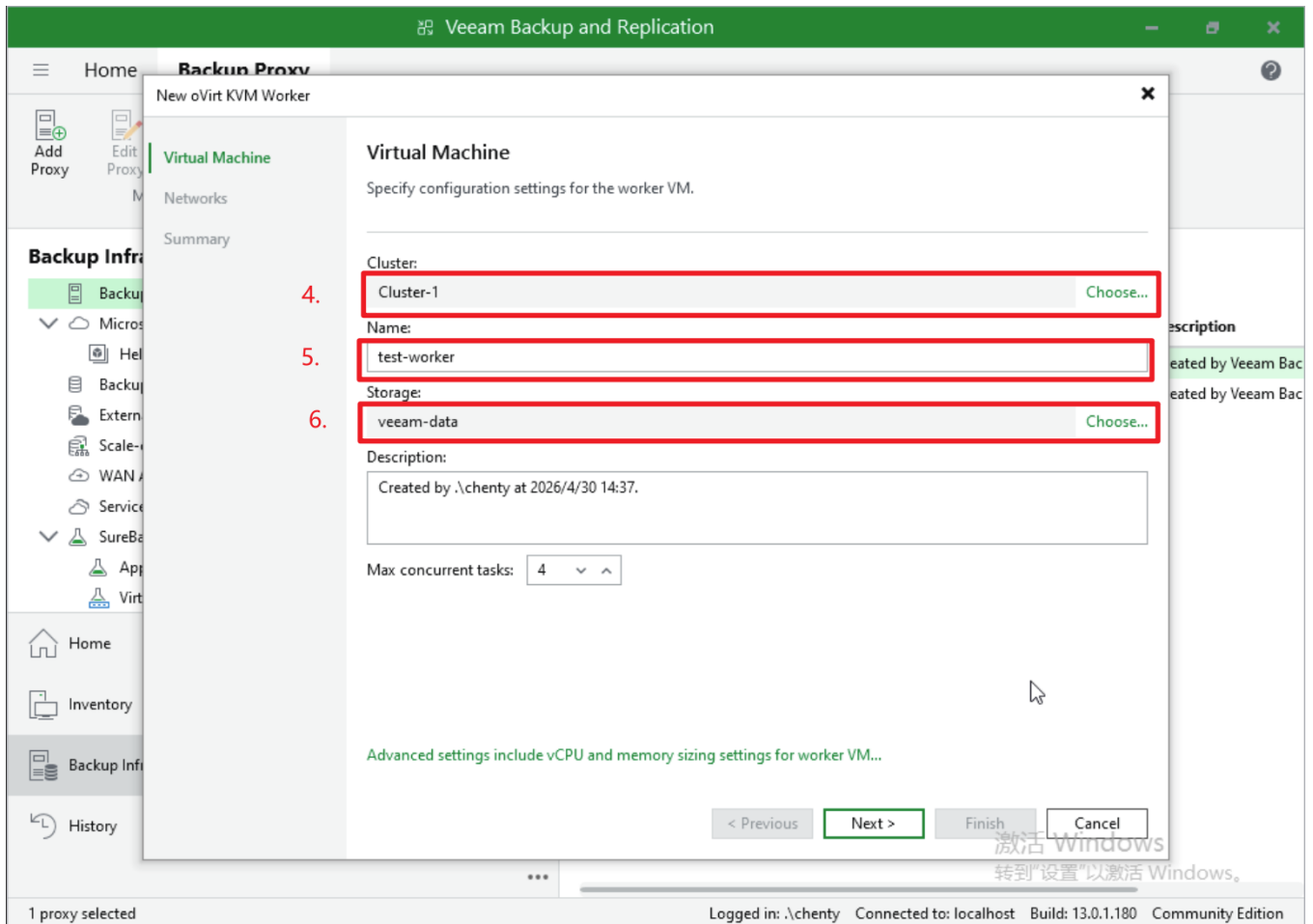
激活 Windows  
转到“设置”以激活 Windows。

Logged in: .\chenthy Connected to: localhost Build: 13.0.1.180 Community Edition

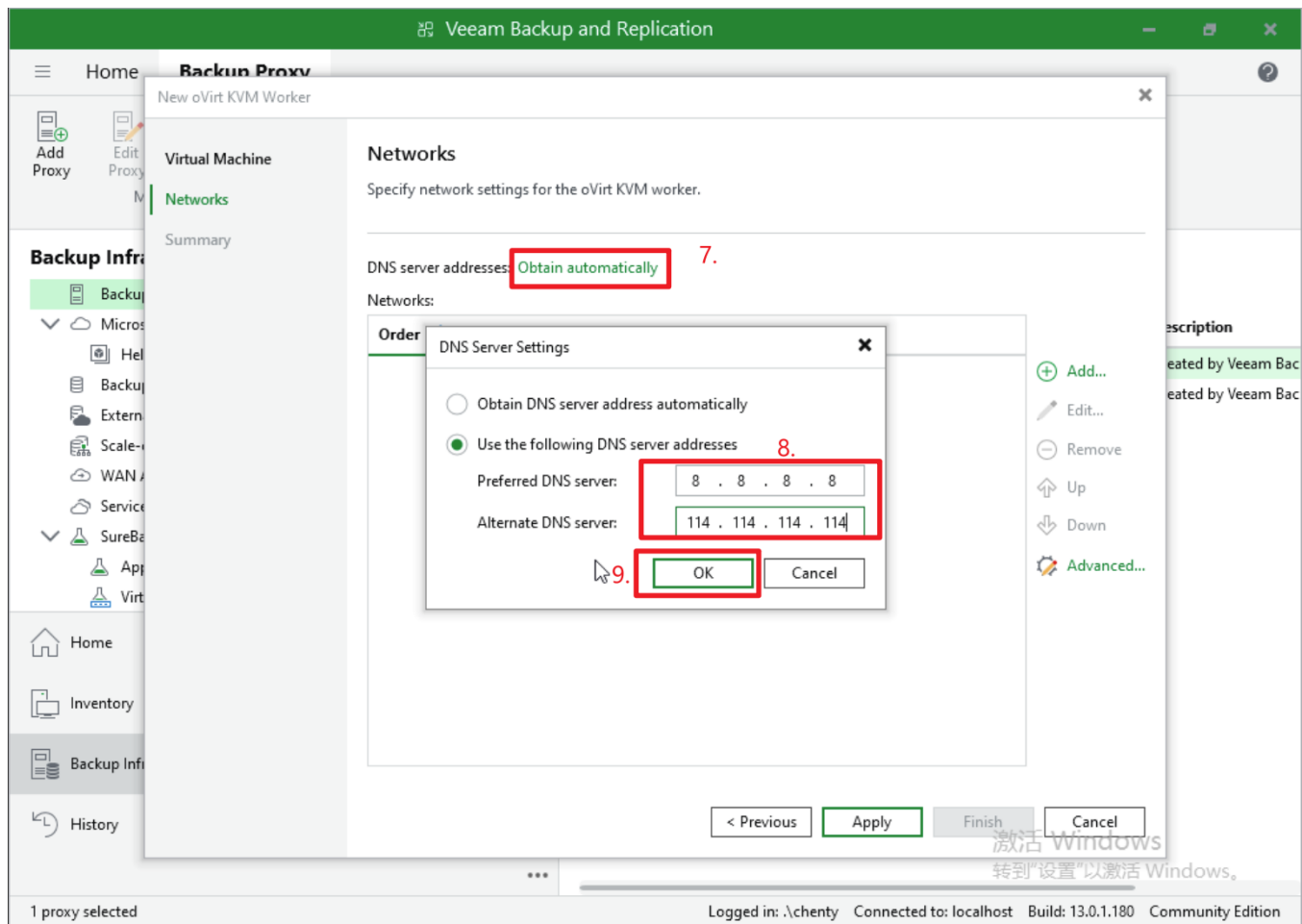
2. 在 Backup Proxy 中选择添加 oVirt KVM Worker。



3. 指定 Worker 相关配置，包括所在集群、Worker 名称、存储池等。选择 Worker 所在集群时，必须选择 Intel 主机、Intel-only 集群，或已通过调度策略限制到 Intel 主机的资源范围。CPU、内存和并发任务数可先使用 Veeam 推荐值，稳定后再根据备份窗口调整。

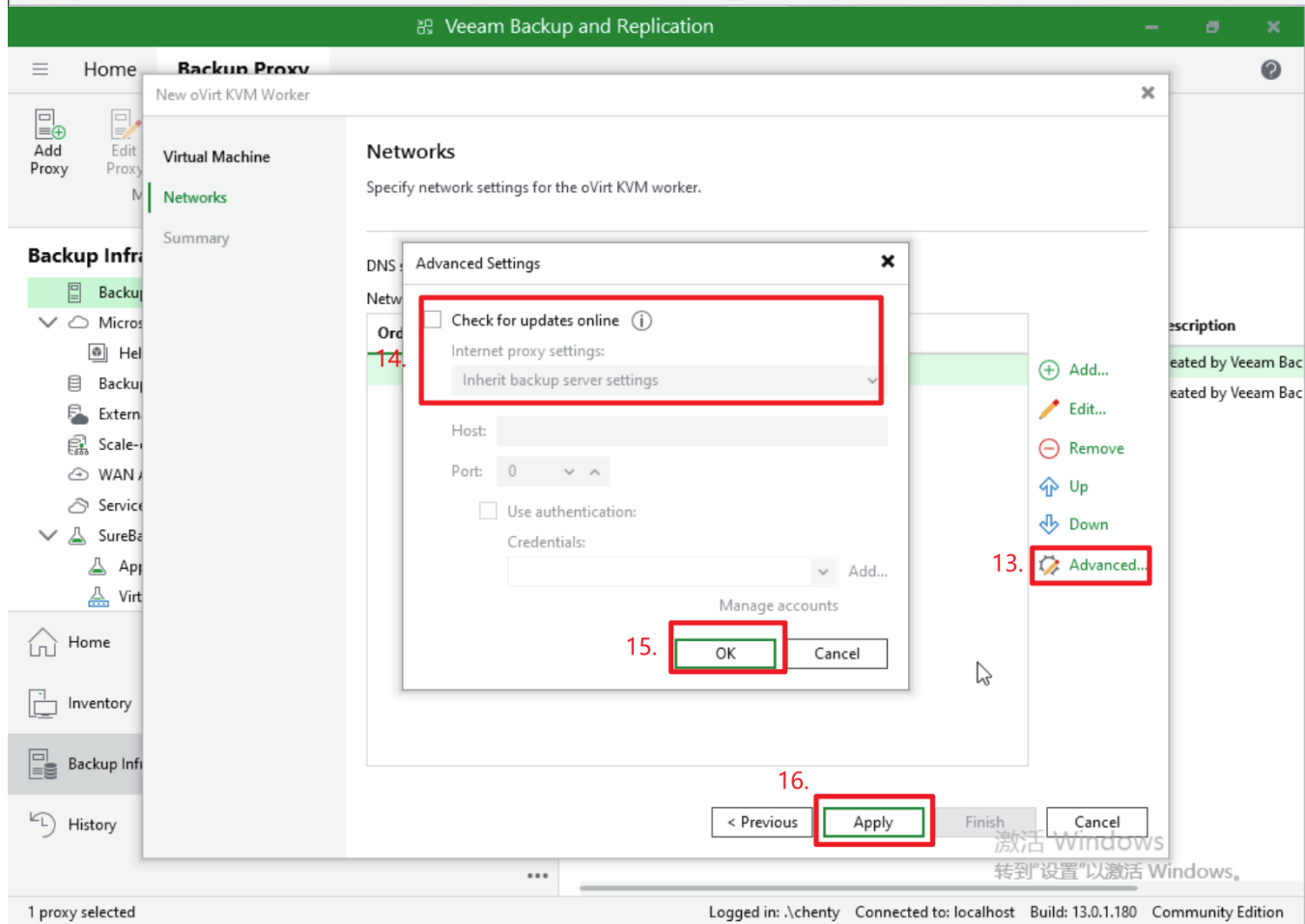
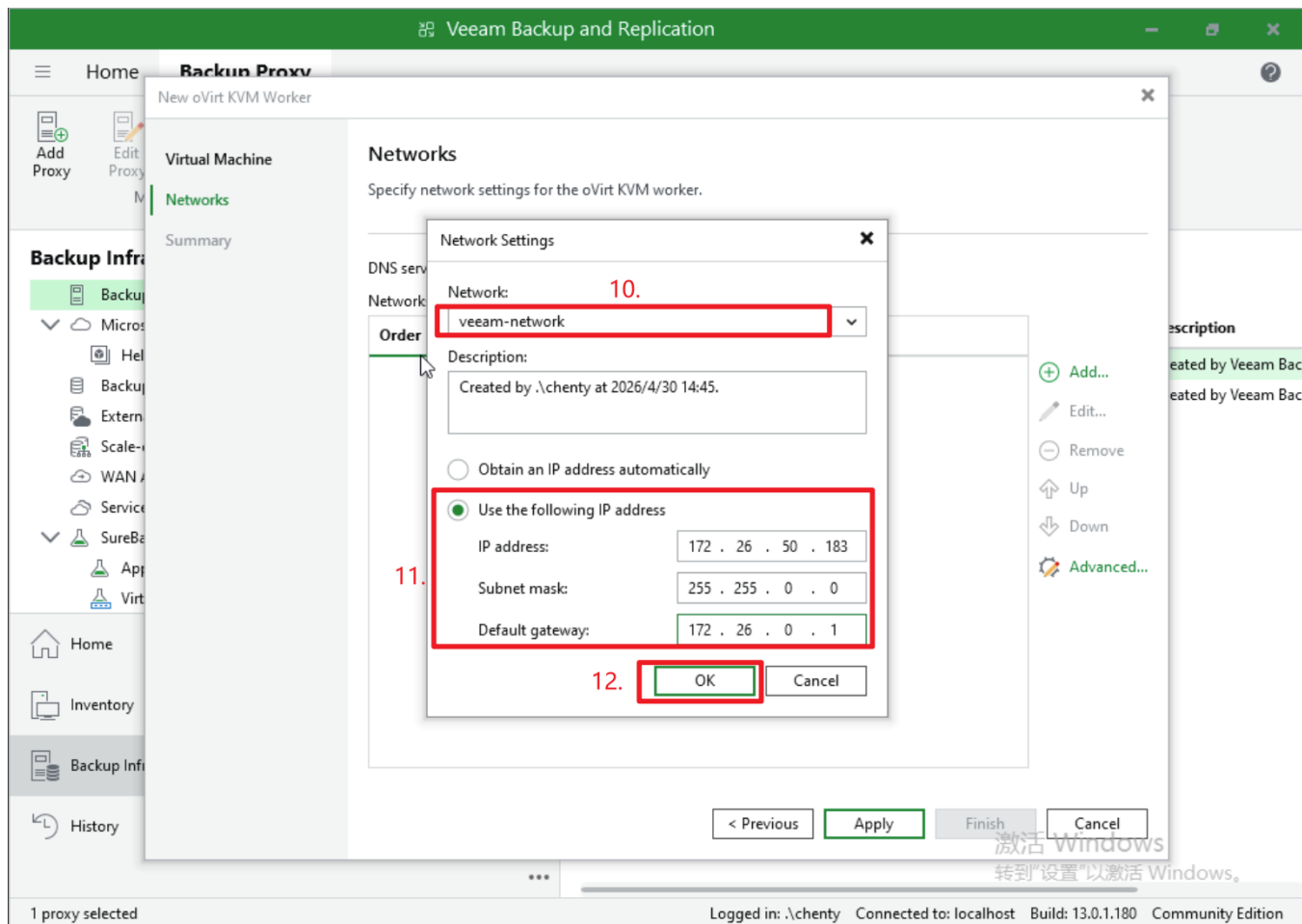


4. 选择 Worker 使用的管理网络，保证它能访问 Veeam Server、备份仓库和 API Proxy 地址。

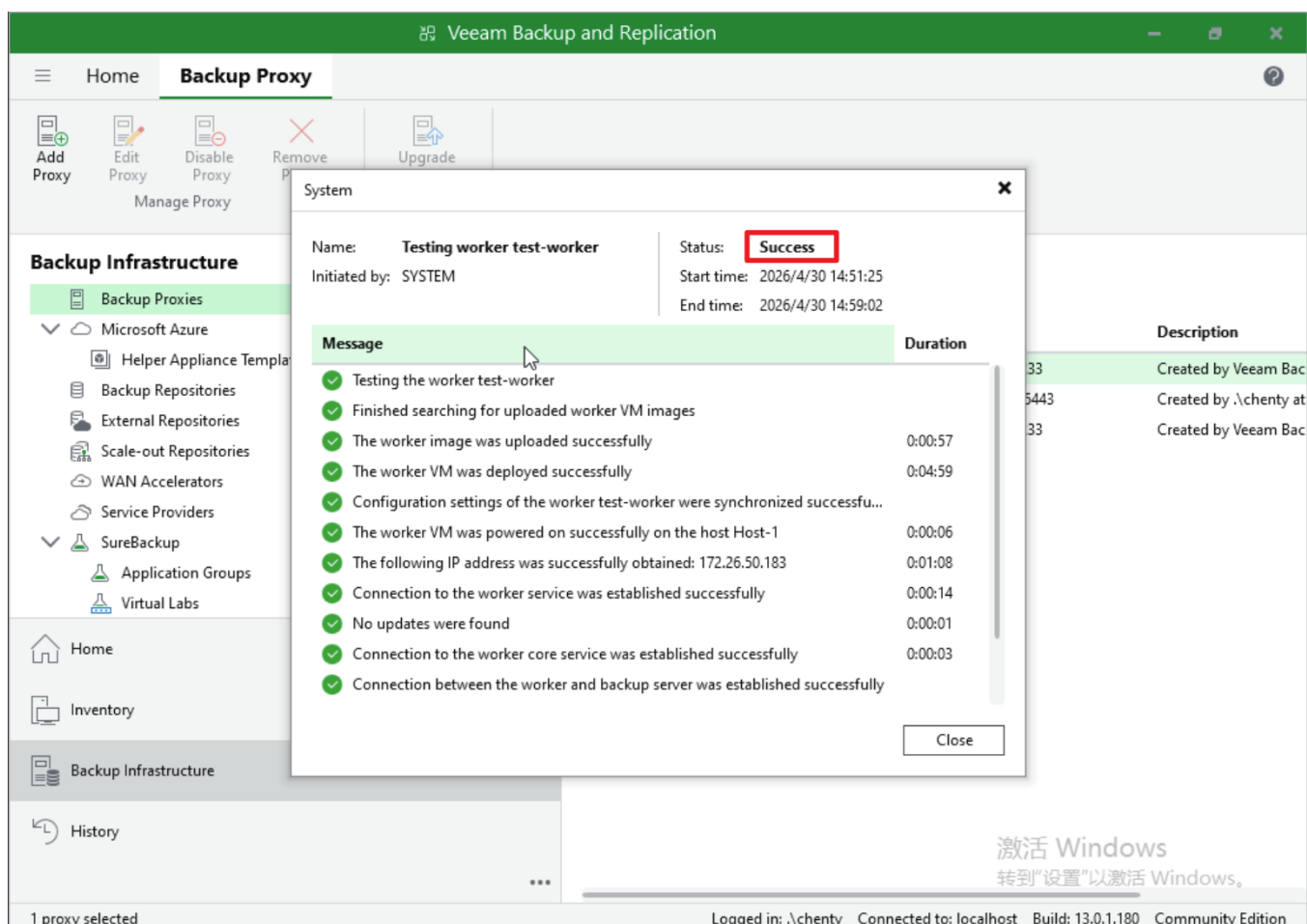
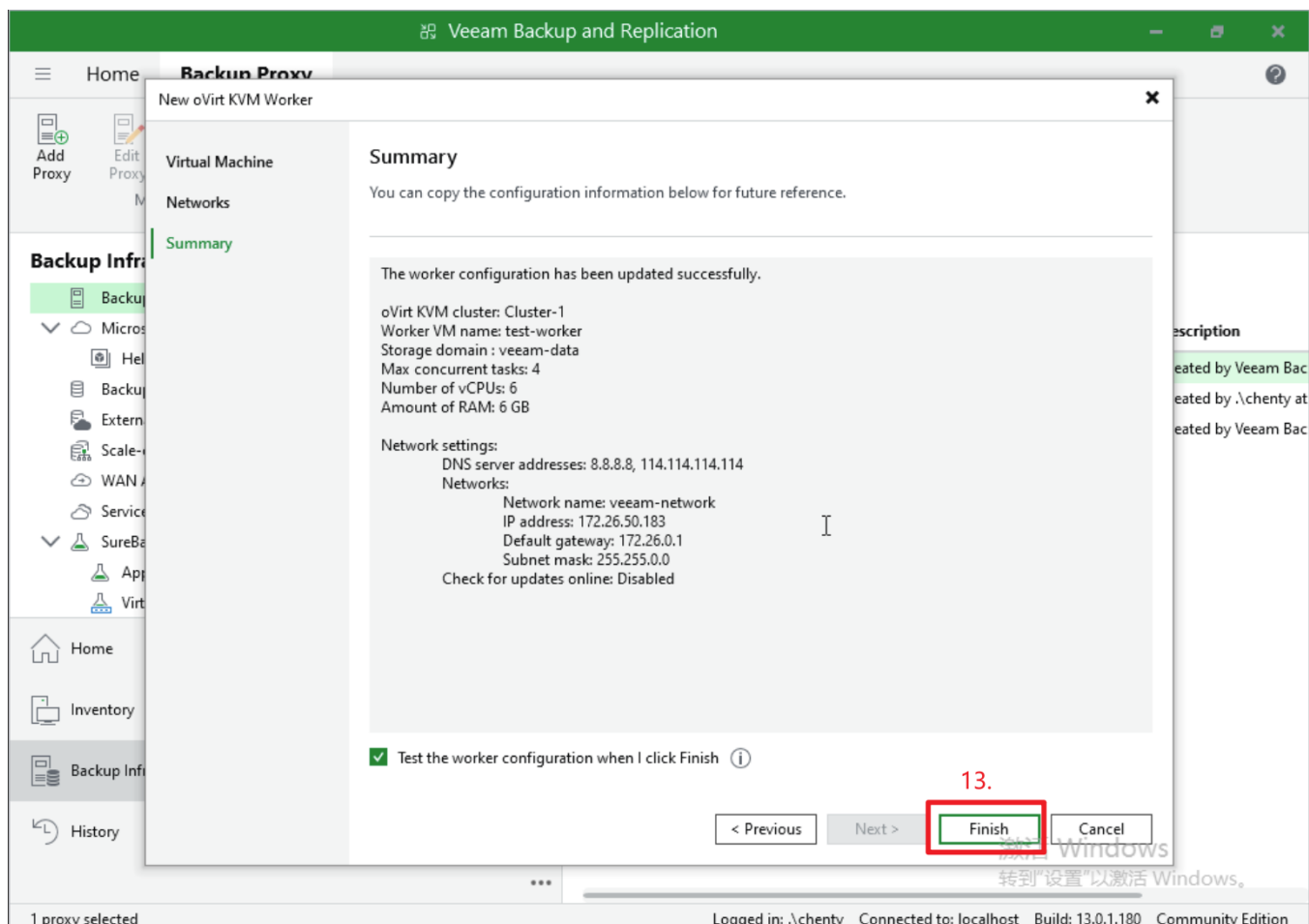


5. 为 Worker 手动分配静态 IP 地址，并确保该地址在管理网络中可达，且不会被 DHCP 或其他系统重复占用。

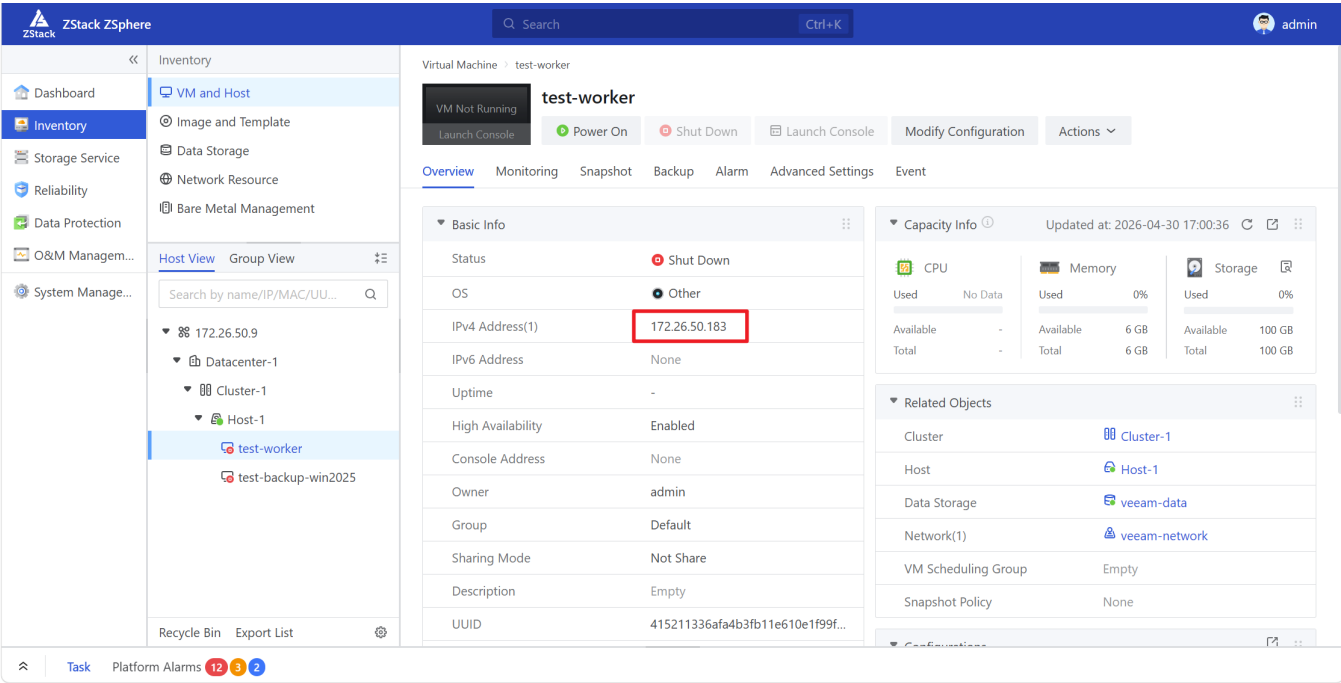




6. 完成向导后，在管理平台控制台确认 Worker 虚机创建成功、IP 与预期静态地址一致，并且处于运行状态。



7. 回到 Veeam 控制台确认 Worker 状态为可用。



7.6 Worker 部署失败排查

如果 Worker 部署或启动失败，优先检查以下项：

现象	检查项	处理建议
Worker 虚机无法启动	是否被调度到 AMD 主机	删除失败的 Worker，重新部署到 Intel 主机或 Intel-only 集群
Worker 已创建但 Veeam 无法稳定连接	是否为 Worker 手动分配了正确的静态 IP	为 Worker 重新配置固定静态 IP，并确认 Veeam Server、API Proxy、备份仓库都能访问该地址
Veeam 显示 Worker 不可用	Worker 到 Veeam Server / API Proxy / 仓库网络不通	检查安全组、防火墙、路由和 DNS
Veeam 无法添加虚拟化平台	API Proxy 地址、端口、证书或账号错误	先用 curl -k https://<地址>:16443/ovirt-engine/api 和 token 接口验证
备份任务启动后失败	Worker 网络或存储访问异常	检查 Worker 所在网络、API Proxy 日志和 Veeam 任务日志
恢复任务创建 VM 失败	管理平台账号权限不足，或目标集群、网络、存储选择不正确	使用管理员账号复测，并确认恢复目标资源可用

API Proxy 侧排障建议收集：

```
./collect-api-proxy-logs.sh root@<入口节点IP>
```

Veeam 侧同时导出对应任务日志，便于把 Veeam 请求时间点和 API Proxy 日志对齐。

## 7.7 备份与恢复使用建议

- 在 Veeam 中创建备份任务时，按 oVirt / RHV 虚拟机选择流程选择通过 API Proxy 展示的虚拟机。
- 第一次接入建议先选择一台测试虚拟机执行 Active Full 备份，确认 Worker、网络和存储链路全部正常。
- 恢复时按 oVirt / RHV 恢复流程选择恢复点、目标集群、目标存储和目标网络。
- 双管理节点部署时，建议在 Veeam 中使用 API Proxy VIP 作为连接地址，避免管理节点切换后仍连接到非当前服务入口。
- 如果 API Proxy VIP、管理网络或证书 SAN 发生变化，需要重新检查证书和 Veeam 连接配置；必要时重新生成 API Proxy TLS 证书。
- 生产环境建议为 Veeam 单独创建管理平台账号，并只授予备份恢复所需权限；联调阶段可以先用管理员账号缩短排障路径。

## 8. 单管理节点和双管理节点的入口 IP 规则

部署、卸载、日志收集和日志清理均支持单管理节点和双管理节点部署。

<node-ip> 表示脚本执行时使用的入口节点，不一定等同于客户端最终访问 API Proxy 的地址。

场景	命令中填写的 <node-ip>	客户端访问 API Proxy 时建议使用
单管理节点	唯一管理节点 IP	同一管理节点 IP
双管理节点	任一管理节点 IP；脚本会以其作为入口节点，自动探测另一台管理节点并执行对应操作	VIP

- 入口节点可以是任一管理节点。
- 脚本会通过 keepalived 配置探测另一台管理节点和 VIP。
- 远程操作时，脚本会先连接入口节点，再由入口节点中转到另一台管理节点。
- 安装完成后，如果检测到 VIP，会优先检查 VIP 上的健康检查接口。
- 日志清理不会停止 keepalived，也不会主动漂移 VIP。
- 日志清理如果发现 api-proxy 正在运行，会先短暂停止服务，清理完成后再启动并做健康检查。

## 9. TLS

服务默认使用 HTTPS：

- 监听端口：16443
- keystore：/etc/api-proxy/tls/server-keystore.p12
- keystore 类型：PKCS12

如果 keystore 不存在，服务启动阶段可通过内置初始化逻辑生成自签名证书。生产环境如需使用正式证书，可提前放置 keystore，并通过启动参数覆盖路径或密码。

示例：

```
sudo APP_OPTS="--server.ssl.key-store=file:/etc/api-proxy/tls/server-keystore.p12 --server.ssl.key-store-password=<password>" ./install-api-proxy.sh --install
```

## 10. 日志相关

### 10.1 启用 API 请求/响应日志

默认只记录常规运行日志，不记录完整 API 请求体和响应体。排查协议交互时可以临时开启：

```
./deploy-api-proxy.sh --install --enable-api-logging root@<node-ip>
```

本地安装时可使用对应命令：

```
sudo ./install-api-proxy.sh --install --enable-api-logging
```

该参数会把 `--adapter.api-logging.enabled=true` 写入 `systemd` 启动参数。后续重新部署或重新安装时如果不带该参数，安装脚本会刷新 `unit` 并恢复为默认关闭。

### 10.2 日志收集

收集单管理节点或双管理节点部署的全量日志：

```
./collect-api-proxy-logs.sh root@<node-ip>
```

指定输出目录：

```
./collect-api-proxy-logs.sh --output-dir /tmp/api-proxy-logs root@<node-ip>
```

常用参数：

```
./collect-api-proxy-logs.sh --keepalived root@<node-ip>
./collect-api-proxy-logs.sh --skip-keepalived root@<node-ip>
```

说明：

- 双管理节点部署时默认会收集 `keepalived` 配置和 `journal`。
- 单管理节点部署默认不收集 `keepalived` 信息，除非显式指定 `--keepalived`。
- 输出目录默认形如 `api-proxy-logs-YYYYMMDD-HHMMSS`。
- 顶层目录会额外生成 `collection-summary.txt`，汇总入口节点、目标节点、VIP 和采集模式等信息。

每个节点目录通常包含：

文件	内容

summary.txt	节点状态、服务状态、VIP 状态、日志目录概览
api-proxy-logs.tar.gz	API Proxy 文件日志
journal-api-proxy.log	API Proxy systemd journal
mn-logs.tar.gz	管理节点文件日志
journal-mn.log	管理节点服务 journal
keepalived.conf	双管理节点配置，双管理节点部署时默认收集
journal-keepalived.log	keepalived journal，双管理节点部署时默认收集

## 10.3 日志清理

默认会截断当前服务日志、删除滚动压缩日志，并按 7 天保留 systemd journal：

```
./cleanup-api-proxy-logs.sh root@<node-ip>
```

常用参数：

```
./cleanup-api-proxy-logs.sh --vacuum-time 7d root@<node-ip>
./cleanup-api-proxy-logs.sh --vacuum-size 1G root@<node-ip>
./cleanup-api-proxy-logs.sh --skip-journal-vacuum root@<node-ip>
./cleanup-api-proxy-logs.sh --delete-log-dir root@<node-ip>
```

说明：

- 如果服务在清理前处于运行状态，脚本会短暂停止 api-proxy，清理完成后重新启动并做健康检查。
- 脚本不会停止 keepalived，也不会主动移动 VIP。
- --delete-log-dir 会删除轮转后的应用日志，并截断当前活动日志。
- --skip-failover 和 --force-active-cleanup 仍被兼容接受，但现在只会给出提示，不再触发任何 VIP 漂移或 keepalived 操作。

## 11. 常用环境变量

变量	默认值	说明
API_PROXY_SSH_PASSWORD	空	远程脚本使用的 SSH 密码；为空时交互输入
API_PROXY_ASSUME_YES	空	设置为 1 时跳过覆盖确认
API_PROXY_ENABLE_API_LOGGING	0	远程安装时设置为 1 等同于 --enable-api-logging
SERVICE_NAME	api-proxy	systemd 服务名
INSTALL_DIR	/opt/api-proxy	安装目录
LOG_DIR	/opt/api-proxy/logs	日志目录

UPLOAD_DIR	/opt/api-proxy/uploads	上传工作目录
RUNTIME_WORK_DIR	/tmp/api-proxy	运行时工作目录
JAVA_BIN	自动发现	Java 可执行文件路径
JAVA_OPTS	-Xms256m -Xmx2g	JVM 参数
APP_OPTS	空	服务启动参数
AUTO_CREATE_UNIT	1	systemd unit 缺失时自动创建
SYSTEMD_UNIT_PATH	/etc/systemd/system/api-proxy.service	systemd unit 路径
HEALTH_CHECK_URL	https://127.0.0.1:16443/ovirt-engine/api	本地健康检查地址
BACKEND_CONFIG_PATH	自动发现	后端配置文件路径；安装脚本会优先读取该变量
MYSQL_BIN	mysql	卸载数据库清理时使用的 mysql 客户端
API_PROXY_DB_NAME	自动匹配	卸载时清理服务自有表所用数据库名；默认跟随后端当前数据库
API_PROXY_DB_HOST	自动匹配	卸载时清理服务自有表所用数据库地址
API_PROXY_DB_PORT	自动匹配	卸载时清理服务自有表所用数据库端口
API_PROXY_DB_USERNAME	自动匹配	卸载时清理服务自有表所用数据库用户
API_PROXY_DB_PASSWORD	自动匹配	卸载时清理服务自有表所用数据库密码
CLEAN_INSTALL_DIR	1	卸载时清理安装目录
CLEAN_LOGS	1	卸载时清理日志目录
CLEAN_UPLOADS	1	卸载时清理上传目录
CLEAN_RUNTIME_WORK_DIR	1	卸载时清理运行时工作目录
CLEAN_DATABASE_TABLES	1	卸载时清理服务自有数据库表

## 12. 健康检查与排障

服务状态：

```
systemctl status api-proxy --no-pager -l
systemctl is-active api-proxy
```

查看日志：

```
journalctl -u api-proxy --no-pager -n 200
less /opt/api-proxy/logs/api-proxy.log
```



检查端口与接口：

```
ss -lntp | grep ':16443'
curl -k -i https://127.0.0.1:16443/ovirt-engine/api
```

常见问题：

现象	检查项
SSH 连接失败	root 登录权限、密码、网络连通性、防火墙
提示缺少 Java	安装 Java 8，或设置 JAVA_BIN / JAVA_HOME
服务启动失败，日志中出现 qemu-img not found at startup	在目标节点安装 qemu-img，确保其在 PATH 中，或通过 adapter.backup.qemu-img-path 指定路径
本地健康检查超时	查看 journalctl -u api-proxy 和主日志，确认端口、TLS、数据库连接
卸载时数据库清理跳过	确认 mysql 客户端是否存在，或通过 API_PROXY_DB_* 指定连接信息
VIP 健康检查失败	确认 VIP 是否在任一节点、端口是否开放、keepalived 是否正常
Veeam Worker 不可用	检查 Worker 是否运行在 Intel 主机、静态 IP 是否正确，以及到 API Proxy 和仓库网络是否可达

### 13. 最小验收清单

上线前建议至少完成以下检查：

- deploy-api-proxy.sh --install 执行成功。
- systemctl status api-proxy 显示服务运行正常。
- curl -k https://<任一管理节点 IP 或 VIP>:16443/ovirt-engine/api 返回成功。
- Veeam 能以 oVirt / RHV 类型成功添加 API Proxy。
- Veeam 控制台能看到通过 API Proxy 展示的集群、主机、存储和虚拟机。
- Veeam Worker 部署在 Intel 主机上，并在 Veeam 中显示可用。
- 至少完成一台测试虚拟机的备份。
- 至少完成一次测试恢复，并确认恢复后的虚拟机可启动、网络和磁盘符合预期。

### 14. 安全建议

- 不建议长期启用 API 请求/响应日志；问题排查完成后重新部署并移除 --enable-api-logging。
- 日志包可能包含请求路径、资源 ID、错误栈和环境信息，传递前应按内部流程处理。
- SSH 密码建议通过短生命周期的环境变量传入，不要写入脚本或手册。
- 生产证书请使用受信任 CA 签发的 keystore，并妥善管理 keystore 密码。