



U Y U N I

Uyuni 2024.08

클라이언트 구성 가이드

2024년08월 9일



차례

클라이언트 구성 안내서 개요	1
1. 지원되는 클라이언트 및 기능	2
1.1. 지원되는 클라이언트 시스템	2
1.2. 지원되는 도구 패키지	3
1.3. 지원되는 SLES 및 openSUSE 기능	3
1.4. Supported SLE Micro Features	5
1.5. 지원되는 SL Micro 클라이언트 기능	7
1.6. openSUSE Leap Micro 클라이언트 기능	9
1.7. 지원하는 Alibaba Cloud Linux 기능	11
1.8. 지원하는 AlmaLinux 기능	13
1.9. 지원되는 Amazon Linux 기능	15
1.10. 지원하는 CentOS 기능	17
1.11. 지원하는 Debian 기능	18
1.12. 지원되는 openEuler 기능	20
1.13. 지원하는 Oracle 기능	22
1.14. 지원되는 Raspberry Pi OS 기능	24
1.15. 지원하는 Red Hat Enterprise Linux 기능	26
1.16. 지원하는 Rocky Linux 기능	28
1.17. 지원하는 Ubuntu 기능	30
2. 구성 기본	33
2.1. 소프트웨어 채널	33
2.1.1. SUSE Package Hub가 제공하는 패키지	33
2.1.2. AppStream이 제공하는 패키지	34
2.1.3. EPEL이 제공하는 패키지	34
2.1.4. SUSE Linux Enterprise 클라이언트의 Unified Installer 업데이트 채널	34
2.1.5. 소프트웨어 리포지토리	34
2.1.6. 소프트웨어 제품	36
2.1.7. AppStream 모듈	36
2.2. 부트스트랩 리포지토리	36
2.2.1. 부트스트랩 리포지토리 생성 준비	36
2.2.2. 자동 모드를 위한 옵션	37
2.2.3. 부트스트랩 리포지토리를 수동으로 생성	38
2.2.4. 부트스트랩 및 사용자 지정 채널	39
2.3. 활성화 키	39
2.3.1. 활성화 키	41
2.3.2. 활성화 키 모범 사례	42
2.4. GPG 키	43
2.4.1. 클라이언트의 GPG 키 신뢰	43
3. 클라이언트 연락 방법	46
3.1. 기본 연락 방법(Salt)	46
3.1.1. 온보딩 정보	47
3.2. SSH Push 연락 방법	47
3.2.1. 사용 가능한 파라미터	49
3.2.2. 작업 실행	50
3.2.3. 향후 출시될 기능	50
3.3. SSH Push(터널 사용) 연락 방법	50
3.4. Salt Bundle	53
3.4.1. Salt Bundle이란 무엇입니까?	53
3.4.2. Salt Bundle을 Minion으로 클라이언트 등록	53
3.4.3. Salt 번들을 사용한 SSH Push	54
3.4.4. pip를 사용하여 Python 패키지와 함께 Salt Bundle 확장	55
4. 클라이언트 등록	56

4.1. 등록 방법	56
4.1.1. Web UI로 클라이언트 등록	56
4.1.2. 부트스트랩 스크립트로 클라이언트 등록	58
4.1.3. 명령줄에서 클라이언트 등록	61
4.2. SUSE 클라이언트 등록	64
4.2.1. SUSE Linux Enterprise 클라이언트 등록	64
4.2.2. Registering SLE Micro Clients	68
4.2.3. SL Micro 클라이언트 등록	72
4.3. openSUSE 클라이언트 등록	75
4.3.1. openSUSE Leap 클라이언트 등록	75
4.3.2. openSUSE Leap Micro 클라이언트 등록	77
4.4. Alibaba Cloud Linux 클라이언트 등록	79
4.4.1. Alibaba Cloud Linux 클라이언트 등록	79
4.5. AlmaLinux 클라이언트 등록	81
4.5.1. AlmaLinux 클라이언트 등록	81
4.6. Amazon Linux 클라이언트 등록	84
4.6.1. Amazon Linux 클라이언트 등록	84
4.7. CentOS 클라이언트 등록	86
4.7.1. CentOS 클라이언트 등록	87
4.8. Debian 클라이언트 등록	91
4.8.1. Debian 클라이언트 등록	91
4.9. OpenEuler 클라이언트 등록	94
4.9.1. openEuler 클라이언트 등록	95
4.10. Oracle 클라이언트 등록	98
4.10.1. Oracle Linux 클라이언트 등록	98
4.11. Raspberry Pi OS 클라이언트 등록	101
4.11.1. Raspberry Pi OS 클라이언트 등록	101
4.12. Red Hat 클라이언트 등록	105
4.12.1. Red Hat Enterprise Linux 클라이언트를 CDN으로 등록	105
4.12.2. RHUI를 사용하여 Red Hat Enterprise Linux 클라이언트 등록	112
4.13. Rocky Linux 클라이언트 등록	117
4.13.1. Rocky Linux 클라이언트 등록	118
4.14. Ubuntu 클라이언트 등록	120
4.14.1. Ubuntu 20.04 및 22.04 클라이언트 등록	120
4.15. 프록시에 클라이언트 등록	125
4.15.1. 프록시 간 클라이언트 이동	125
4.15.2. 프록시에서 서버로 클라이언트 이동	126
4.15.3. Web UI로 클라이언트를 프록시에 등록	126
4.15.4. 명령줄에서 클라이언트 등록	127
4.16. 공용 클라우드에서 클라이언트 등록	127
4.16.1. 제품 추가 및 리포지토리 동기화	128
4.16.2. 온디맨드 이미지 준비	128
4.16.3. 클라이언트 등록	128
4.16.4. 활성화 키	129
4.16.5. Terraform에서 생성한 클라이언트의 자동 등록	129
5. 클라이언트 업그레이드	131
5.1. 주 버전 업그레이드	131
5.1.1. 마이그레이션 준비	131
5.1.2. 마이그레이션	133
5.2. 컨텐트 라이프사이클 관리자를 사용하여 업그레이드	134
5.2.1. 업그레이드 준비	134
5.2.2. 업그레이드	135
5.3. 제품 마이그레이션	136
5.3.1. 단일 시스템 마이그레이션	137
5.3.2. 제품 대량 마이그레이션	137
5.4. Uyuni 클라이언트 업그레이드	140

5.4.1. 업그레이드 준비	140
5.4.2. 업그레이드	141
6. 클라이언트 삭제	142
6.1. Web UI를 사용한 클라이언트 삭제	142
6.2. 명령 줄에서 클라이언트 삭제(API 호출 사용)	142
6.3. 명령 줄에서 클라이언트 삭제	143
7. 클라이언트 작업	144
7.1. 패키지 관리	144
7.1.1. Compare Packages Using Profiles	144
7.2. 패치 관리	145
7.2.1. 패치 생성	145
7.2.2. 패치를 클라이언트에 적용	146
7.3. 시스템 잠금	147
7.3.1. 클라이언트의 시스템 잠금	147
7.3.2. 패키지 잠금	148
7.4. 구성 관리	149
7.4.1. 구성 채널 생성	149
7.4.2. 구성 파일, 디렉토리 또는 심볼 링크 추가	150
7.4.3. 클라이언트를 구성 채널에 가입	150
7.4.4. 구성 파일 비교	151
7.4.5. 클라이언트의 Jinja 템플릿	151
7.5. 전원 관리	152
7.5.1. 전원 관리 및 Cobbler	152
7.6. 사용자 정의 시스템 정보	153
7.7. 시스템 세트 관리자	153
7.7.1. SSM에서 기본 채널 변경하기	155
7.8. 시스템 그룹	155
7.8.1. 그룹 생성	156
7.8.2. 클라이언트를 그룹에 추가	156
7.8.3. 그룹 작업	156
7.9. 시스템 유형	157
8. 운영 체제 설치	158
8.1. 등록된 시스템 재설치	159
8.2. CD-ROM 또는 USB 키를 통한 설치	159
8.2.1. Cobbler를 사용한 ISO 이미지 빌드	159
8.2.2. KIWI를 사용한 SUSE ISO 이미지 빌드	160
8.2.3. Cobbler를 사용한 Red Hat ISO 이미지 빌드	160
8.3. 자동 설치 가능한 배포판	160
8.3.1. ISO 이미지 기반 배포	161
8.3.2. RPM 패키지 기반 배포	161
8.3.3. 자동 설치 가능한 배포판 선언	162
8.4. 자동 설치 프로파일	163
8.4.1. 프로파일 선언	163
8.4.2. AutoYaST 프로파일	164
8.4.3. 커스텀 프로파일	165
8.4.4. 템플릿 구문	165
8.5. 무인 프로비저닝	168
8.6. 자체 GPG 키 사용	168
9. 시작화	170
9.1. 가상화 호스트 관리	170
9.2. 가상 게스트 생성	170
9.3. Xen 및 KVM을 이용한 가상화	171
9.3.1. 호스트 설정	171
9.3.2. VM 게스트 자동 설치	172
9.3.3. VM 게스트 관리	175

10. 가상 호스트 관리자	177
10.1. 가상 호스트 관리자 및 Amazon Web Services	177
10.1.1. Amazon EC2 VHM 생성	177
10.1.2. 가상 호스트 관리자에 대한 AWS 권한	178
10.2. 가상 호스트 관리자 및 Azure	179
10.2.1. 전제 조건	179
10.2.2. Azure VHM 생성	179
10.2.3. 권한 할당	179
10.2.4. Azure UUID	180
10.3. 가상 호스트 관리자 및 Google Compute Engine	180
10.3.1. 전제 조건	180
10.3.2. GCE VHM 생성	181
10.3.3. 권한 할당	181
10.3.4. GCE UUID	182
10.4. Nutanix를 이용한 가상화	182
10.4.1. VHM 설정	182
10.5. VMWare를 이용한 가상화	183
10.5.1. VHM 설정	183
10.5.2. VMWare에서 SSL 오류 문제 해결	184
10.6. 다른 타사 공급자를 통한 가상화	185
11. GNU Free Documentation License	188

클라이언트 구성 안내서 개요

업데이트 날짜: 2024-08-09

Uyuni 설치 후 수행하는 첫 단계는 클라이언트를 등록하는 일입니다. Uyuni에서 등록 클라이언트를 관리하는 작업에 대부분의 시간을 소요합니다.

Uyuni은(는) SUSE Linux Enterprise 또는 다른 Linux 운영 체제 및 다양한 하드웨어 옵션과 호환됩니다.

지원되는 클라이언트 및 기능의 전체 목록은 [Client-configuration > Supported-features](#)를 참조하십시오.

이 안내서에서는 다양한 클라이언트를 수동 및 자동으로 등록하고 구성하는 방법에 대해 설명합니다.

Chapter 1. 지원되는 클라이언트 및 기능

Uyuni은(는) 다양한 클라이언트 기술과 호환됩니다. SUSE Linux Enterprise 또는 다른 Linux 운영 체제로 구동되는 Salt 클라이언트를 다양한 하드웨어 옵션과 함께 설치할 수 있습니다.

이 섹션에는 지원되는 클라이언트 시스템에 관한 요약이 포함되어 있습니다. 각 클라이언트에서 사용할 수 있는 기능의 상세 목록은 다음 페이지를 참조하십시오.

1.1. 지원되는 클라이언트 시스템

이 표에는 Salt 클라이언트에서 지원되는 운영 체제가 나열되어 있습니다.

이 표의 아이콘에는 다음과 같은 의미가 있습니다.

- ✓ 이 운영 체제를 실행하는 클라이언트를 SUSE에서 지원합니다.
- ✗ 이 운영 체제를 실행하는 클라이언트를 SUSE에서 지원하지 않습니다.
- ? 이 클라이언트는 현재 지원 고려 중이며, 이후에 지원될 수도 있고 지원되지 않을 수도 있습니다.



클라이언트 운영 체제 버전 및 SP 수준에는 Uyuni로 지원될 일반 지원(일반 또는 LTSS)이 적용되어야 합니다. 지원되는 제품 버전에 대한 상세 정보는 <https://www.suse.com/lifecycle>에서 참조하십시오.



클라이언트에서 실행되는 운영 체제는 운영 체제를 제공하는 조직이 지원합니다.

표 1. 지원되는 클라이언트 시스템 및 아키텍처

Operating System	x86-64	ppc64le	IBM Z	aarch64	arm64 / armhf
SUSE Linux Enterprise 15, 12	✓	✓	✓	✓	✗
SUSE Linux Enterprise Server for SAP 15, 12	✓	✓	✗	✗	✗
SLE Micro	✓	✗	✗	✓	✗
SL Micro	✓	✗	✗	✓	✗
openSUSE Leap Micro	✓	✗	✗	✓	✗
openSUSE Leap 15	✓	✗	✗	✓	✗
Alibaba Cloud Linux 2	✓	✗	✗	✓	✗
AlmaLinux 9, 8	✓	✗	✗	✓	✗
Amazon Linux 2003, 2	✓	✗	✗	✓	✗
CentOS 7	✓	✓	✗	✓	✗
Debian 12, 11	✓	✗	✗	✗	✗
openEuler 22.03	✓	✗	✗	✗	✗
Oracle Linux 9, 8, 7	✓	✗	✗	✓	✗

Operating System	x86-64	ppc64le	IBM Z	aarch64	arm64 / armhf
Raspberry Pi OS 12	✗	✗	✗	✗	✓
Red Hat Enterprise Linux 9, 8, 7	✓	✗	✗	✗	✗
Rocky Linux 9, 8	✓	✗	✗	✓	✗
Ubuntu 22.04, 20.04	✓	✗	✗	✗	✗



(*) Debian 및 Ubuntu에서는 x86-64 아키텍처가 amd64로 표시됩니다.

배포의 수명이 만료하면 3개월의 유예 기간이 시작되며 지원은 중단한 것으로 간주됩니다. 이 기간이 경과하면 해당 제품은 지원하지 않는 것으로 간주됩니다. 모든 지원은 최선의 노력으로 제공합니다.

수명 종료에 대한 자세한 설명은 <https://endoflife.software/operating-systems>를 참조하십시오.

1.2. 지원되는 도구 패키지

spacewalk-utils 및 spacewalk-utils-extras 패키지를 통해 추가 서비스 및 기능을 제공할 수 있습니다.

표 2. Spacewalk 유틸리티

도구 이름	설명	지원 여부
spacewalk-common-channels	SUSE Customer Center 가 제공하지 않는 채널 추가	✓
spacewalk-hostname-rename	Uyuni의 호스트 이름 변경	✓
spacewalk-clone-by-date	특정 날짜까지 채널 복제	✓
spacewalk-sync-setup	ISS 마스터 및 슬레이브 조직 매핑 설정	✓
spacewalk-manage-channel-lifecycle	채널 라이프사이클 관리	✓

1.3. 지원되는 SLES 및 openSUSE 기능

아래 표는 SUSE 및 openSUSE 클라이언트에서 다양한 기능의 사용 가능 여부를 정리한 것입니다. 이 표에서는 SAP용 SLES, SLED, SUSE Linux Enterprise Server 및 HPC용 SUSE Linux Enterprise Server를 포함한 SUSE Linux Enterprise 운영 체제의 모든 변형에 대해 설명합니다.



클라이언트에서 실행하는 운영 체제는 운영 체제를 제공하는 조직이 지원합니다. SUSE Linux Enterprise는 SUSE가 지원합니다. openSUSE는 SUSE 커뮤니티가 지원합니다.

이 표의 아이콘에는 다음과 같은 의미가 있습니다.

- ✓ 이 기능을 사용할 수 있음
- ✗ 이 기능을 사용할 수 없습니다.

- ?이 기능은 현재 고려 중이며, 이후에 사용할 수도 있고 사용하지 못할 수도 있습니다.

표 3. SUSE 및 openSUSE 운영 체제에서 지원하는 기능

기능	SUSE Linux Enterprise 12	SUSE Linux Enterprise 15	openSUSE 15
클라이언트	✓	✓	✓
시스템 패키지	SUSE	SUSE	openSUSE Community
등록	✓	✓	✓
패키지 설치	✓	✓	✓
패치 적용	✓	✓	✓
원격 명령	✓	✓	✓
시스템 패키지 상태	✓	✓	✓
시스템 사용자 정의 상태	✓	✓	✓
그룹 사용자 정의 상태	✓	✓	✓
조직 사용자 정의 상태	✓	✓	✓
시스템 세트 관리자(SSM)	✓	✓	✓
제품 마이그레이션	✓	✓	✓
기본 가상 게스트 관리 *	✓	✓	✓
고급 가상 게스트 관리 *	✓	✓	✓
호스트 OS로 가상 게스트 설치(AutoYaST)	✗	✗	✗
호스트 OS로 가상 게스트 설치(이미지 템플릿)	✓	✓	✓
가상 게스트 관리	✓	✓	✓
시스템 배포(PXE/AutoYaST)	✓	✓	✓
시스템 재배포(AutoYaST)	✓	✓	✓
연락 방법	✗	✗	✓ ZeroMQ, Salt-SSH
Uyuni 프록시와 함께 작동	✓	✓	✓
작업 체인	✓	✓	✓
스테이징(패키지 사전 다운로드)	✓	✓	✓
중복 패키지 보고	✓	✓	✓
CVE 감사	✓	✓	✓
SCAP 감사	✓	✓	✓
패키지 검증	✗	✗	✗

기능	SUSE Linux Enterprise 12	SUSE Linux Enterprise 15	openSUSE 15
패키지 잠금	✓	✓	✓
시스템 잠금	✗	✗	✗
유지보수 기간	✓	✓	✓
시스템 스냅샷	✗	✗	✗
구성 파일 관리	✓	✓	✓
패키지 프로필	✓ 프로필 지원됨, 동기화 지원되지 않음	✓ 프로필 지원됨, 동기화 지원되지 않음	✓ 프로필 지원됨, 동기화 지원되지 않음
전원 관리	✓	✓	✓
모니터링 서버	✓	✓	✓
모니터링되는 클라이언트	✓	✓	✓
Docker 빌드호스트	✓	✓	?
OS를 사용한 Docker 이미지 빌드	✓	✓	✓
Kiwi 빌드호스트	✓	?	?
OS를 사용한 Kiwi 이미지 빌드	✓	?	✗
반복 작업	✓	✓	✓
AppStreams	N/A	N/A	N/A
Yomi	✗	✓	✓

* 가상 게스트 관리:

이 표에서 가상 게스트 관리는 기본과 고급으로 구분되어 있습니다.

기본 가상 게스트 관리에는 VM, 느린 새로 고침, VM 라이프싸이클 작업(시작, 중지, 다시 시작, 일시 중지) 나열과 VM vCPU 및 메모리 수정이 포함됩니다.

고급 가상 게스트 관리에는 기본 가상 게스트 관리의 모든 기능과 함께 빠른 새로 고침, VM 라이프싸이클 작업(삭제, 재설정, 전원 끄기), VM 디스크, 네트워크, 그래픽 표시 및 그래픽 표시 구성 수정이 포함됩니다.

1.4. Supported SLE Micro Features



The operating system you run on a client is supported by the organization that supplies the operating system. SLE Micro is supported by SUSE.



SLE Micro is only supported as regular minion (default contact method) for the time being. We are working on managing it as Salt SSH client (salt-ssh contact method), too.

이 표의 아이콘에는 다음과 같은 의미가 있습니다.

- ✓ 이 기능을 사용할 수 있음
- ✗ 이 기능을 사용할 수 없습니다.
- ? 이 기능은 현재 고려 중이며, 이후에 사용할 수도 있고 사용하지 못할 수도 있습니다.

표 4. Supported Features on SLE Micro Operating Systems

Feature	SLE Micro
Client	✓
Operating system packages	✓
Registration	✓
Install packages	✓
Apply patches (requires CVE ID)	✓
Remote commands	✓
System package states	✓
System custom states	✓
Group custom states	✓
Organization custom states	✓
System set manager (SSM)	✓
Product migration	✓
Basic Virtual Guest Management *	✓
Advanced Virtual Guest Management *	✓
Virtual Guest Installation (Kickstart), as Host OS	✓
Virtual Guest Installation (image template), as Host OS	✓
System deployment (PXE/Kickstart)	✓
System redeployment (Kickstart)	✓
Contact methods	✓ ZeroMQ
Works with Uyuni Proxy	✓
Action chains	✓
Staging (pre-download of packages)	?
Duplicate package reporting	✓
CVE auditing (requires CVE ID)	✓
SCAP auditing	?
Package verification	?
Package locking	✓

Feature	SLE Micro
System locking	?
Maintenance Windows	?
System snapshot	✗
Configuration file management	✓
Snapshots and profiles	✓ Profiles supported, Sync not supported
Power management	✓
Monitoring server	✗
Monitored clients **	✓
Docker buildhost	✗
Build Docker image with OS	✗
Kiwi buildhost	✗
Build Kiwi image with OS	✓
Recurring Actions	✓
AppStreams	N/A
Yomi	?

* 가상 게스트 관리:

이 표에서 가상 게스트 관리는 기본과 고급으로 구분되어 있습니다.

기본 가상 게스트 관리에는 VM, 느린 새로 고침, VM 라이프싸이클 작업(시작, 중지, 다시 시작, 일시 중지) 나열과 VM vCPU 및 메모리 수정이 포함됩니다.

고급 가상 게스트 관리에는 기본 가상 게스트 관리의 모든 기능과 함께 빠른 새로 고침, VM 라이프싸이클 작업(삭제, 재설정, 전원 끄기), VM 디스크, 네트워크, 그래픽 표시 및 그래픽 표시 구성 수정이 포함됩니다.

****** On SLE Micro, only the Node exporter and the Blackbox exporter are available.

1.5. 지원되는 SL Micro 클라이언트 기능



클라이언트에서 실행하는 운영 체제는 운영 체제를 제공하는 조직이 지원합니다. SUSE는 SL Micro를 지원합니다.



SL Micro는 현재 -- 연락 방법으로만 Salt 클라이언트로 지원됩니다.

이 표의 아이콘에는 다음과 같은 의미가 있습니다.

- ✓ 이 기능을 사용할 수 있음
- ✗ 이 기능을 사용할 수 없습니다.

- ? 이 기능은 현재 고려 중이며, 이후에 사용할 수도 있고 사용하지 못할 수도 있습니다.

표 5. Supported Features on SL Micro Operating Systems

Feature	SL Micro
Client	Salt
Operating system packages	Salt
Registration	Salt
Install packages	Salt
Apply patches (requires CVE ID)	Salt
Remote commands	Salt
System package states	Salt
System custom states	Salt
Group custom states	Salt
Organization custom states	Salt
System set manager (SSM)	Salt
Product migration	Salt
Basic Virtual Guest Management *	?
Advanced Virtual Guest Management *	?
Virtual Guest Installation (Kickstart), as Host OS	✗
Virtual Guest Installation (image template), as Host OS	?
System deployment (PXE/Kickstart)	?
System redeployment (Kickstart)	✗
Contact methods	Salt: ZeroMQ
Works with Uyuni Proxy	Salt
Action chains	Salt
Staging (pre-download of packages)	?
Duplicate package reporting	Salt
CVE auditing (requires CVE ID)	Salt
SCAP auditing	?
Package verification	?
Package locking	Salt
System locking	?
Maintenance Windows	?
System snapshot	✗
Configuration file management	Salt

Feature	SL Micro
Snapshots and profiles	Salt: Profiles supported, Sync not supported
Power management	Salt
Monitoring server	✗
Monitored clients **	Salt
Docker buildhost	✗
Build Docker image with OS	✗
Kiwi buildhost	✗
Build Kiwi image with OS	Salt
Recurring Actions	Salt
AppStreams	N/A
Yomi	?

* 가상 게스트 관리:

이 표에서 가상 게스트 관리는 기본과 고급으로 구분되어 있습니다.

기본 가상 게스트 관리에는 VM, 느린 새로 고침, VM 라이프싸이클 작업(시작, 중지, 다시 시작, 일시 중지) 나열과 VM vCPU 및 메모리 수정이 포함됩니다.

고급 가상 게스트 관리에는 기본 가상 게스트 관리의 모든 기능과 함께 빠른 새로 고침, VM 라이프싸이클 작업(삭제, 재설정, 전원 끄기), VM 디스크, 네트워크, 그래픽 표시 및 그래픽 표시 구성 수정이 포함됩니다.

****** SL Micro의, 노드 엑스포트 및 블랙박스 엑스포트만 사용할 수 있습니다.

1.6. openSUSE Leap Micro 클라이언트 기능



openSUSE Leap Micro는 SUSE 커뮤니티에서 지원합니다.

이 표의 아이콘에는 다음과 같은 의미가 있습니다.

- ✓ 이 기능을 사용할 수 있음
- ✗ 이 기능을 사용할 수 없습니다.
- ? 이 기능은 현재 고려 중이며, 이후에 사용할 수도 있고 사용하지 못할 수도 있습니다.

표 6. openSUSE Leap Micro 운영 체제에서 지원하는 기능

기능	openSUSE Leap Micro
클라이언트	✓
운영 체제 패키지	✓
등록	✓

기능	openSUSE Leap Micro
패키지 설치	✓
패치 적용(CVE ID 필요)	✓
원격 명령	✓
시스템 패키지 상태	✓
시스템 사용자 정의 상태	✓
그룹 사용자 정의 상태	✓
조직 사용자 정의 상태	✓
시스템 세트 관리자(SSM)	✓
제품 마이그레이션	✓
기본 가상 게스트 관리 *	✓
고급 가상 게스트 관리 *	✓
호스트 OS로 가상 게스트 설치(킥스타트)	✓
호스트 OS로 가상 게스트 설치(이미지 템플릿)	✓
시스템 배포(PXE/킥스타트)	✓
시스템 재배포(킥스타트)	✓
연락 방법	✓ ZeroMQ
Uyuni 프록시와 함께 작동	✓
작업 체인	✓
스테이징(패키지 사전 다운로드)	?
중복 패키지 보고	✓
CVE 감사(CVE ID 필요)	✓
SCAP 감사	?
패키지 검증	?
패키지 잠금	✓
시스템 잠금	?
유지보수 기간	?
시스템 스냅샷	✗
구성 파일 관리	✓
스냅샷 및 프로필	✓ 프로필 지원됨, 동기화 지원되지 않음
전원 관리	✓
모니터링 서버	✗
모니터링되는 서버	✓
Docker 빌드호스트	✗

기능	openSUSE Leap Micro
OS를 사용한 Docker 이미지 빌드	✗
Kiwi 빌드호스트	✗
OS를 사용한 Kiwi 이미지 빌드	✓
반복 작업	✓
AppStreams	N/A
Yomi	?

* 가상 게스트 관리:

이 표에서 가상 게스트 관리는 기본과 고급으로 구분되어 있습니다.

기본 가상 게스트 관리에는 VM, 느린 새로 고침, VM 라이프사이클 작업(시작, 중지, 다시 시작, 일시 중지) 나열과 VM vCPU 및 메모리 수정이 포함됩니다.

고급 가상 게스트 관리에는 기본 가상 게스트 관리의 모든 기능과 함께 빠른 새로 고침, VM 라이프사이클 작업(삭제, 재설정, 전원 끄기), VM 디스크, 네트워크, 그래픽 표시 및 그래픽 표시 구성 수정이 포함됩니다.

1.7. 지원하는 Alibaba Cloud Linux 기능

아래 표는 Alibaba Cloud Linux 클라이언트에서 다양한 기능의 사용 가능 여부를 정리한 것입니다.



클라이언트에서 실행하는 운영 체제는 운영 체제를 제공하는 조직이 지원합니다. Alibaba Cloud Linux는 Alibaba Cloud에서 지원됩니다.

이 표의 아이콘에는 다음과 같은 의미가 있습니다.

- ✓ 이 기능을 사용할 수 있음
- ✗ 이 기능을 사용할 수 없습니다.
- ? 이 기능은 현재 고려 중이며, 이후에 사용할 수도 있고 사용하지 못할 수도 있습니다.

표 7. Alibaba Cloud Linux 운영 체제에서 지원하는 기능

기능	Alibaba Cloud Linux 2
클라이언트	✓
운영 체제 패키지	✓
등록	✓
패키지 설치	✓
패치 적용(CVE ID 필요)	✓
원격 명령	✓
시스템 패키지 상태	✓

기능	Alibaba Cloud Linux 2
시스템 사용자 정의 상태	✓
그룹 사용자 정의 상태	✓
조직 사용자 정의 상태	✓
시스템 세트 관리자(SSM)	✓
제품 마이그레이션	N/A
기본 가상 게스트 관리 *	?
고급 가상 게스트 관리 *	?
호스트 OS로 가상 게스트 설치(킥스타트)	✗
호스트 OS로 가상 게스트 설치(이미지 템플릿)	?
시스템 배포(PXE/킥스타트)	?
시스템 재배포(킥스타트)	?
연락 방법	✓ ZeroMQ, Salt-SSH
Uyuni 프록시와 함께 작동	✓
작업 체인	✓
스테이징(패키지 사전 다운로드)	✓
중복 패키지 보고	✓
CVE 감사(CVE ID 필요)	✓
SCAP 감사	✓
패키지 검증	✗
패키지 잠금	✗
시스템 잠금	✗
유지보수 기간	✓
시스템 스냅샷	✗
구성 파일 관리	✓
스냅샷 및 프로필	✓ 프로필 지원됨, 동기화 지원되지 않음
전원 관리	?
모니터링 서버	✗
모니터링되는 클라이언트	✓
Docker 빌드호스트	✓
OS를 사용한 Docker 이미지 빌드	✓
Kiwi 빌드호스트	✓
OS를 사용한 Kiwi 이미지 빌드	✓
반복 작업	✓

기능	Alibaba Cloud Linux 2
AppStreams	N/A
Yomi	N/A

* 가상 게스트 관리:

이 표에서 가상 게스트 관리는 기본과 고급으로 구분되어 있습니다.

기본 가상 게스트 관리에는 VM, 느린 새로 고침, VM 라이프사이클 작업(시작, 중지, 다시 시작, 일시 중지) 나열과 VM vCPU 및 메모리 수정이 포함됩니다.

고급 가상 게스트 관리에는 기본 가상 게스트 관리의 모든 기능과 함께 빠른 새로 고침, VM 라이프사이클 작업(삭제, 재설정, 전원 끄기), VM 디스크, 네트워크, 그래픽 표시 및 그래픽 표시 구성 수정이 포함됩니다.

1.8. 지원하는 AlmaLinux 기능

아래 표는 AlmaLinux 클라이언트에서 다양한 기능의 사용 가능 여부를 정리한 것입니다.



클라이언트에서 실행하는 운영 체제는 운영 체제를 제공하는 조직이 지원합니다.
AlmaLinux는 AlmaLinux 커뮤니티에서 지원됩니다.

이 표의 아이콘에는 다음과 같은 의미가 있습니다.

- ✓ 이 기능을 사용할 수 있음
- ✗ 이 기능을 사용할 수 없습니다.
- ? 이 기능은 현재 고려 중이며, 이후에 사용할 수도 있고 사용하지 못할 수도 있습니다.

표 8. AlmaLinux 운영 체제에서 지원하는 기능

기능	AlmaLinux 9	AlmaLinux 8
클라이언트	✓ (plain AlmaLinux)	✓ (plain AlmaLinux)
시스템 패키지	AlmaLinux Community	AlmaLinux Community
등록	✓	✓
패키지 설치	✓	✓
패치 적용	✓	✓
원격 명령	✓	✓
시스템 패키지 상태	✓	✓
시스템 사용자 정의 상태	✓	✓
그룹 사용자 정의 상태	✓	✓
조직 사용자 정의 상태	✓	✓
시스템 세트 관리자(SSM)	✓	✓
제품 마이그레이션	N/A	N/A

기능	AlmaLinux 9	AlmaLinux 8
기본 가상 게스트 관리 *	✓	✓
고급 가상 게스트 관리 *	✓	✓
호스트 OS로 가상 게스트 설치(킥스타트)	✗	✗
호스트 OS로 가상 게스트 설치(이미지 템플릿)	✓	✓
시스템 배포(PXE/킥스타트)	✓	✓
시스템 재배포(킥스타트)	✓	✓
연락 방법	✓ ZeroMQ, Salt-SSH	✓ ZeroMQ, Salt-SSH
Uyuni 프록시와 함께 작동	✓	✓
작업 체인	✓	✓
스테이징(패키지 사전 다운로드)	✓	✓
중복 패키지 보고	✓	✓
CVE 감사	✓	✓
SCAP 감사	✓	✓
패키지 검증	✗	✗
패키지 잠금	✗	✗
시스템 잠금	✗	✗
유지보수 기간	✓	✓
시스템 스냅샷	✗	✗
구성 파일 관리	✓	✓
스냅샷 및 프로필	✓ 프로필 지원됨, 동기화 지원되지 않음	✓ 프로필 지원됨, 동기화 지원되지 않음
전원 관리	✓	✓
모니터링 서버	✗	✗
모니터링되는 클라이언트	✓	✓
Docker 빌드호스트	✗	✗
OS를 사용한 Docker 이미지 빌드	✗	✗
Kiwi 빌드호스트	✗	✗
OS를 사용한 Kiwi 이미지 빌드	✗	✗
반복 작업	✓	✓
AppStreams	✓	✓
Yomi	N/A	N/A

* 가상 게스트 관리:

이 표에서 가상 게스트 관리는 기본과 고급으로 구분되어 있습니다.

기본 가상 게스트 관리에는 VM, 느린 새로 고침, VM 라이프싸이클 작업(시작, 중지, 다시 시작, 일시 중지) 나열과 VM vCPU 및 메모리 수정이 포함됩니다.

고급 가상 게스트 관리에는 기본 가상 게스트 관리의 모든 기능과 함께 빠른 새로 고침, VM 라이프싸이클 작업(삭제, 재설정, 전원 끄기), VM 디스크, 네트워크, 그래픽 표시 및 그래픽 표시 구성 수정이 포함됩니다.

1.9. 지원되는 Amazon Linux 기능

아래 표는 Amazon Linux 클라이언트에서 다양한 기능의 사용 가능 여부를 정리한 것입니다.



클라이언트에서 실행하는 운영 체제는 운영 체제를 제공하는 조직이 지원합니다. Amazon Linux은 Amazon에서 지원됩니다.

이 표의 아이콘에는 다음과 같은 의미가 있습니다.

- ✓ 이 기능을 사용할 수 있음
- ✗ 이 기능을 사용할 수 없습니다.
- ? 이 기능은 현재 고려 중이며, 이후에 사용할 수도 있고 사용하지 못할 수도 있습니다.

표 9. Amazon Linux 운영 체제에서 지원하는 기능

기능	Amazon Linux 2	Amazon Linux 2023
클라이언트	✓	✓
운영 체제 패키지	✓	✓
등록	✓	✓
패키지 설치	✓	✓
패치 적용(CVE ID 필요)	✓	✓
원격 명령	✓	✓
시스템 패키지 상태	✓	✓
시스템 사용자 정의 상태	✓	✓
그룹 사용자 정의 상태	✓	✓
조직 사용자 정의 상태	✓	✓
시스템 세트 관리자(SSM)	✓	✓
제품 마이그레이션	N/A	N/A
기본 가상 게스트 관리 *	?	?
고급 가상 게스트 관리 *	?	?
호스트 OS로 가상 게스트 설치(킥스타트)	✗	✗

기능	Amazon Linux 2	Amazon Linux 2023
호스트 OS로 가상 게스트 설치(이미지 템플릿)	?	?
시스템 배포(PXE/킥스타트)	?	?
시스템 재배포(킥스타트)	?	?
연락 방법	✓ ZeroMQ, Salt-SSH	✓ ZeroMQ, Salt-SSH
Uyuni 프록시와 함께 작동	✓	✓
작업 체인	✓	✓
스테이징(패키지 사전 다운로드)	✓	✓
중복 패키지 보고	✓	✓
CVE 감사(CVE ID 필요)	✓	✓
SCAP 감사	✓	✓
패키지 검증	✗	✗
패키지 잠금	✗	✗
시스템 잠금	✗	✗
유지보수 기간	✓	✓
시스템 스냅샷	✗	✗
구성 파일 관리	✓	✓
스냅샷 및 프로필	✓ 프로필 지원됨, 동기화 지원되지 않음	✓ 프로필 지원됨, 동기화 지원되지 않음
전원 관리	?	?
모니터링 서버	✗	✗
모니터링되는 클라이언트	✓	✓
Docker 빌드호스트	✓	✓
OS를 사용한 Docker 이미지 빌드	✓	✓
Kiwi 빌드호스트	✓	✓
OS를 사용한 Kiwi 이미지 빌드	✓	✓
반복 작업	✓	✓
AppStreams	N/A	N/A
Yomi	N/A	N/A

* 가상 게스트 관리:

이 표에서 가상 게스트 관리는 기본과 고급으로 구분되어 있습니다.

기본 가상 게스트 관리에는 VM, 느린 새로 고침, VM 라이프사이클 작업(시작, 중지, 다시 시작, 일시 중지) 나열과 VM vCPU 및 메모리 수정이 포함됩니다.

고급 가상 게스트 관리에는 기본 가상 게스트 관리의 모든 기능과 함께 빠른 새로 고침, VM 라이프사이클 작업(삭제, 재설정, 전원 끄기), VM 디스크, 네트워크, 그래픽 표시 및 그래픽 표시 구성 설정이 포함됩니다.

1.10. 지원하는 CentOS 기능

아래 표는 CentOS 클라이언트에서 다양한 기능의 사용 가능 여부를 정리한 것입니다.



클라이언트에서 실행하는 운영 체제는 운영 체제를 제공하는 조직이 지원합니다. CentOS는 CentOS 커뮤니티가 지원합니다.

이 표의 아이콘에는 다음과 같은 의미가 있습니다.

- ✓ 이 기능을 사용할 수 있음
- ✗ 이 기능을 사용할 수 없습니다.
- ? 이 기능은 현재 고려 중이며, 이후에 사용할 수도 있고 사용하지 못할 수도 있습니다.

표 10. CentOS 운영 체제에서 지원하는 기능

기능	CentOS 7
클라이언트	✓ (plain CentOS)
시스템 패키지	CentOS Community
등록	✓
패키지 설치	✓
패치 적용(CVE ID 필요)	✓(오류 수정에 필요한 타사 서비스)
원격 명령	✓
시스템 패키지 상태	✓
시스템 사용자 정의 상태	✓
그룹 사용자 정의 상태	✓
조직 사용자 정의 상태	✓
시스템 세트 관리자(SSM)	✓
제품 마이그레이션	N/A
기본 가상 게스트 관리 *	✓
고급 가상 게스트 관리 *	✓
호스트 OS로 가상 게스트 관리(킥스타트)	✗
호스트 OS로 가상 게스트 관리(이미지 템플릿)	✓
시스템 배포(PXE/킥스타트)	✓
시스템 재배포(킥스타트)	✓
연락 방법	✓ ZeroMQ, Salt-SSH
Uyuni 프록시와 함께 작동	✓

기능	CentOS 7
작업 체인	✓
스테이징(패키지 사전 다운로드)	✓
중복 패키지 보고	✓
CVE 감사(CVE ID 필요)	✓
SCAP 감사	✓
패키지 검증	✗
패키지 잠금	✓
시스템 잠금	✗
유지보수 기간	✓
시스템 스냅샷	✗
구성 파일 관리	✓
스냅샷 및 프로필	✓ 프로필 지원됨, 동기화 지원되지 않음
전원 관리	✓
모니터링 서버	✗
모니터링되는 클라이언트	✓
Docker 빌드호스트	✗
OS를 사용한 Docker 이미지 빌드	✗
Kiwi 빌드호스트	✗
OS를 사용한 Kiwi 이미지 빌드	✗
반복 작업	✓
AppStreams	N/A
Yomi	N/A

* 가상 게스트 관리:

이 표에서 가상 게스트 관리는 기본과 고급으로 구분되어 있습니다.

기본 가상 게스트 관리에는 VM, 느린 새로 고침, VM 라이프사이클 작업(시작, 중지, 다시 시작, 일시 중지) 나열과 VM vCPU 및 메모리 수정이 포함됩니다.

고급 가상 게스트 관리에는 기본 가상 게스트 관리의 모든 기능과 함께 빠른 새로 고침, VM 라이프사이클 작업(삭제, 재설정, 전원 끄기), VM 디스크, 네트워크, 그래픽 표시 및 그래픽 표시 구성 수정이 포함됩니다.

1.11. 지원하는 Debian 기능

아래 표는 Debian 클라이언트에서 다양한 기능의 사용 가능 여부를 정리한 것입니다.



클라이언트에서 실행하는 운영 체제는 운영 체제를 제공하는 조직이 지원합니다. Debian은 Debian 커뮤니티가 지원합니다.

이 표의 아이콘에는 다음과 같은 의미가 있습니다.

- ✓ 이 기능을 사용할 수 있음
- ✗ 이 기능을 사용할 수 없습니다.
- ? 이 기능은 현재 고려 중이며, 이후에 사용할 수도 있고 사용하지 못할 수도 있습니다.

표 11. Debian 운영 체제에서 지원하는 기능

기능	Debian 11	Debian 12
클라이언트	✓	✓
시스템 패키지	Debian Community	Debian Community
등록	✓	✓
패키지 설치	✓	✓
패치 적용	?	?
원격 명령	✓	✓
시스템 패키지 상태	✓	✓
시스템 사용자 정의 상태	✓	✓
그룹 사용자 정의 상태	✓	✓
조직 사용자 정의 상태	✓	✓
시스템 세트 관리자(SSM)	✓	✓
제품 마이그레이션	N/A	N/A
기본 가상 게스트 관리 *	✓	✓
고급 가상 게스트 관리 *	✓	✓
호스트 OS로 가상 게스트 설치(킥스타트)	✓	✓
호스트 OS로 가상 게스트 설치(이미지 템플릿)	✓	✓
시스템 배포(PXE/킥스타트)	✓	✓
시스템 재배포(킥스타트)	✗	✗
연락 방법	✓ ZeroMQ, Salt-SSH	✓ ZeroMQ, Salt-SSH
Uyuni 프록시와 함께 작동	✓	✓
작업 체인	✓	✓
스테이징(패키지 사전 다운로드)	✓	✓
중복 패키지 보고	✓	✓
CVE 감사	?	?

기능	Debian 11	Debian 12
SCAP 감사	?	?
패키지 검증	×	×
패키지 잠금	✓	✓
시스템 잠금	×	×
유지보수 기간	✓	✓
시스템 스냅샷	×	×
구성 파일 관리	✓	✓
패키지 프로필	✓ 프로필 지원됨, 동기화 지원되지 않음	✓ 프로필 지원됨, 동기화 지원되지 않음
전원 관리	✓	✓
모니터링 서버	×	×
모니터링되는 클라이언트	✓	✓
Docker 빌드호스트	?	?
OS를 사용한 Docker 이미지 빌드	Salt	Salt
Kiwi 빌드호스트	×	×
OS를 사용한 Kiwi 이미지 빌드	×	×
반복 작업	✓	✓
AppStreams	N/A	N/A
Yomi	N/A	N/A

* 가상 게스트 관리:

이 표에서 가상 게스트 관리는 기본과 고급으로 구분되어 있습니다.

기본 가상 게스트 관리에는 VM, 느린 새로 고침, VM 라이프싸이클 작업(시작, 중지, 다시 시작, 일시 중지) 나열과 VM vCPU 및 메모리 수정이 포함됩니다.

고급 가상 게스트 관리에는 기본 가상 게스트 관리의 모든 기능과 함께 빠른 새로 고침, VM 라이프싸이클 작업(삭제, 재설정, 전원 끄기), VM 디스크, 네트워크, 그래픽 표시 및 그래픽 표시 구성 수정이 포함됩니다.

1.12. 지원되는 openEuler 기능

이 표에는 openEuler 클라이언트에서 사용할 수 있는 다양한 기능이 나열되어 있습니다.



클라이언트에서 실행하는 운영 체제는 운영 체제를 제공하는 조직에서 지원합니다.
openEuler는 openEuler 커뮤니티에서 지원합니다.

이 표의 아이콘에는 다음과 같은 의미가 있습니다.

- ✓ 이 기능은 두 Salt에서 사용할 수 있습니다.
- ✗ 이 기능을 사용할 수 없습니다.
- ? 이 기능은 현재 고려 중이며, 이후에 사용할 수도 있고 사용하지 못할 수도 있습니다.

표 12. openEuler 운영 체제에서 지원하는 기능

기능	openEuler 22.03
클라이언트	✓ (plain openEuler)
시스템 패키지	openEuler Community
등록	✓
패키지 설치	✓
패치 적용	✓
원격 명령	✓
시스템 패키지 상태	✓
시스템 사용자 정의 상태	✓
그룹 사용자 정의 상태	✓
조직 사용자 정의 상태	✓
시스템 세트 관리자(SSM)	✓
제품 마이그레이션	N/A
기본 가상 게스트 관리 *	✓
고급 가상 게스트 관리 *	✓
호스트 OS로 가상 게스트 설치(킥스타트)	✗
호스트 OS로 가상 게스트 설치(이미지 템플릿)	✓
시스템 배포(PXE/킥스타트)	✓
시스템 재배포(킥스타트)	✓
연락 방법	✓ ZeroMQ, Salt-SSH
Uyuni 프록시와 함께 작동	✓
작업 체인	✓
스테이징(패키지 사전 다운로드)	✓
중복 패키지 보고	✓
CVE 감사	✓
SCAP 감사	✓
패키지 검증	✗
패키지 잠금	✗
시스템 잠금	✗
유지보수 기간	✓

기능	openEuler 22.03
시스템 스냅샷	✗
구성 파일 관리	✓
스냅샷 및 프로필	✓ 프로필 지원됨, 동기화 지원되지 않음
전원 관리	✓
모니터링	✓
Docker 빌드호스트	✗
OS를 사용한 Docker 이미지 빌드	✗
Kiwi 빌드호스트	✗
OS를 사용한 Kiwi 이미지 빌드	✗
반복 작업	✓
Yomi	N/A

* 가상 게스트 관리:

이 표에서 가상 게스트 관리는 기본과 고급으로 구분되어 있습니다.

기본 가상 게스트 관리에는 VM, 느린 새로 고침, VM 라이프싸이클 작업(시작, 중지, 다시 시작, 일시 중지) 나열과 VM vCPU 및 메모리 수정이 포함됩니다.

고급 가상 게스트 관리에는 기본 가상 게스트 관리의 모든 기능과 함께 빠른 새로 고침, VM 라이프싸이클 작업(삭제, 재설정, 전원 끄기), VM 디스크, 네트워크, 그래픽 표시 및 그래픽 표시 구성 수정이 포함됩니다.

1.13. 지원하는 Oracle 기능

아래 표는 Oracle Linux 클라이언트에서 다양한 기능의 사용 가능 여부를 정리한 것입니다.



클라이언트에서 실행하는 운영 체제는 운영 체제를 제공하는 조직이 지원합니다. Oracle Linux는 Oracle이 지원합니다.

이 표의 아이콘에는 다음과 같은 의미가 있습니다.

- ✓ 이 기능을 사용할 수 있음
- ✗ 이 기능을 사용할 수 없습니다.
- ? 이 기능은 현재 고려 중이며, 이후에 사용할 수도 있고 사용하지 못할 수도 있습니다.

표 13. Oracle Linux 운영 체제에서 지원하는 기능

기능	Oracle Linux 7	Oracle Linux 8	Oracle Linux 9
클라이언트	✓	✓	✓
운영 체제 패키지	✓	✓	✓

기능	Oracle Linux 7	Oracle Linux 8	Oracle Linux 9
등록	✓	✓	✓
패키지 설치	✓	✓	✓
패치 적용(CVE ID 필요)	✓	✓	✓
원격 명령	✓	✓	✓
시스템 패키지 상태	✓	✓	✓
시스템 사용자 정의 상태	✓	✓	✓
그룹 사용자 정의 상태	✓	✓	✓
조직 사용자 정의 상태	✓	✓	✓
시스템 세트 관리자(SSM)	✓	✓	✓
제품 마이그레이션	N/A	N/A	N/A
기본 가상 게스트 관리 *	✓	✓	✓
고급 가상 게스트 관리 *	✓	✓	✓
호스트 OS로 가상 게스트 설치(킥스타트)	✗	✓	✓
호스트 OS로 가상 게스트 설치(이미지 템플릿)	✓	✓	✓
시스템 배포(PXE/킥스타트)	✓	✓	✓
시스템 재배포(킥스타트)	✓	✓	✓
연락 방법	✓ ZeroMQ, Salt-SSH	✓ ZeroMQ, Salt-SSH	✓ ZeroMQ, Salt-SSH
Uyuni 프록시와 함께 작동	✓	✓	✓
작업 체인	✓	✓	✓
스테이징(패키지 사전 다운로드)	✓	✓	✓
중복 패키지 보고	✓	✓	✓
CVE 감사(CVE ID 필요)	✓	✓	✓
SCAP 감사	✓	✓	✓
패키지 검증	✗	✗	✗
패키지 잠금	✓	✓	✓
시스템 잠금	✗	✗	✗
유지보수 기간	✓	✓	✓
시스템 스냅샷	✗	✗	✗
구성 파일 관리	✓	✓	✓
스냅샷 및 프로필	✓ 프로필 지원됨, 동기화 지원되지 않음	✓ 프로필 지원됨, 동기화 지원되지 않음	✓ 프로필 지원됨, 동기화 지원되지 않음

기능	Oracle Linux 7	Oracle Linux 8	Oracle Linux 9
전원 관리	✓	✓	✓
모니터링 서버	✗	✗	✗
모니터링되는 클라이언트	✓	✓	✓
Docker 빌드호스트	✗	✗	✗
OS를 사용한 Docker 이미지 빌드	✗	✗	✗
Kiwi 빌드호스트	✗	✗	✗
OS를 사용한 Kiwi 이미지 빌드	✗	✗	✗
반복 작업	✓	✓	✓
AppStreams	N/A	✓	✓
Yomi	N/A	N/A	N/A

* 가상 게스트 관리:

이 표에서 가상 게스트 관리는 기본과 고급으로 구분되어 있습니다.

기본 가상 게스트 관리에는 VM, 느린 새로 고침, VM 라이프싸이클 작업(시작, 중지, 다시 시작, 일시 중지) 나열과 VM vCPU 및 메모리 수정이 포함됩니다.

고급 가상 게스트 관리에는 기본 가상 게스트 관리의 모든 기능과 함께 빠른 새로 고침, VM 라이프싸이클 작업(삭제, 재설정, 전원 끄기), VM 디스크, 네트워크, 그래픽 표시 및 그래픽 표시 구성 수정이 포함됩니다.

1.14. 지원되는 Raspberry Pi OS 기능

이 표에는 Raspberry Pi OS 클라이언트에서 사용할 수 있는 다양한 기능이 나열되어 있습니다.



클라이언트에서 실행하는 운영 체제는 운영 체제를 제공하는 조직에서 지원합니다.
Raspberry Pi OS는 Raspberry Pi OS 커뮤니티에서 지원합니다.

이 표의 아이콘에는 다음과 같은 의미가 있습니다.

- ✓ 이 기능을 사용할 수 있음
- ✗ 이 기능을 사용할 수 없습니다.
- ? 이 기능은 현재 고려 중이며, 이후에 사용할 수도 있고 사용하지 못할 수도 있습니다.

표 14. Raspberry Pi OS 운영 체제에서 지원하는 기능

기능	Raspberry Pi OS 12
클라이언트	✓
시스템 패키지	Raspberry Pi OS Community

기능	Raspberry Pi OS 12
등록	✓
패키지 설치	✓
패치 적용	?
원격 명령	✓
시스템 패키지 상태	✓
시스템 사용자 정의 상태	✓
그룹 사용자 정의 상태	✓
조직 사용자 정의 상태	✓
시스템 세트 관리자(SSM)	✓
제품 마이그레이션	N/A
기본 가상 게스트 관리 *	✓
고급 가상 게스트 관리 *	✓
호스트 OS로 가상 게스트 설치(킥스타트)	✗
호스트 OS로 가상 게스트 설치(이미지 템플릿)	✓
시스템 배포(PXE/킥스타트)	✗
시스템 재배포(킥스타트)	✗
연락 방법	✓ ZeroMQ, Salt-SSH
Uyuni 프록시와 함께 작동	✓
작업 체인	✓
스테이징(패키지 사전 다운로드)	✓
중복 패키지 보고	✓
CVE 감사	?
SCAP 감사	?
패키지 검증	✗
패키지 잠금	✓
시스템 잠금	✗
유지보수 기간	✓
시스템 스냅샷	✗
구성 파일 관리	✓
패키지 프로필	✓ 프로필 지원됨, 동기화 지원되지 않음
전원 관리	✓
모니터링 서버	✗
모니터링되는 클라이언트	✓

기능	Raspberry Pi OS 12
Docker 빌드호스트	?
OS를 사용한 Docker 이미지 빌드	✓
Kiwi 빌드호스트	✗
OS를 사용한 Kiwi 이미지 빌드	✗
반복 작업	✓
AppStreams	N/A
Yomi	N/A

* 가상 게스트 관리:

이 표에서 가상 게스트 관리는 기본과 고급으로 구분되어 있습니다.

기본 가상 게스트 관리에는 VM, 느린 새로 고침, VM 라이프사이클 작업(시작, 중지, 다시 시작, 일시 중지) 나열과 VM vCPU 및 메모리 수정이 포함됩니다.

고급 가상 게스트 관리에는 기본 가상 게스트 관리의 모든 기능과 함께 빠른 새로 고침, VM 라이프사이클 작업(삭제, 재설정, 전원 끄기), VM 디스크, 네트워크, 그래픽 표시 및 그래픽 표시 구성 수정이 포함됩니다.

1.15. 지원하는 Red Hat Enterprise Linux 기능

이 표에는 기본 Red Hat Enterprise Linux 클라이언트에서 사용할 수 있는 다양한 기능이 나열되어 있습니다.



클라이언트에서 실행하는 운영 체제는 운영 체제를 제공하는 조직이 지원합니다. Red Hat Enterprise Linux는 Red Hat이 지원합니다.

이 표의 아이콘에는 다음과 같은 의미가 있습니다.

- ✓ 이 기능을 사용할 수 있음
- ✗ 이 기능을 사용할 수 없습니다.
- ? 이 기능은 현재 고려 중이며, 이후에 사용할 수도 있고 사용하지 못할 수도 있습니다.

표 15. Red Hat Enterprise Linux 운영 체제에서 지원하는 기능

기능	RHEL 7	RHEL 8	RHEL 9
클라이언트	✓	✓	✓
시스템 패키지	Red Hat	Red Hat	Red Hat
등록	✓	✓	✓
패키지 설치	✓	✓	✓
패치 적용	✓	✓	✓
원격 명령	✓	✓	✓
시스템 패키지 상태	✓	✓	✓

기능	RHEL 7	RHEL 8	RHEL 9
시스템 사용자 정의 상태	✓	✓	✓
그룹 사용자 정의 상태	✓	✓	✓
조직 사용자 정의 상태	✓	✓	✓
시스템 세트 관리자(SSM)	✓	✓	✓
제품 마이그레이션	N/A	N/A	N/A
기본 가상 게스트 관리 *	✓	✓	✓
고급 가상 게스트 관리 *	✓	✓	✓
OS를 사용한 가상 게스트 설치(킥스타트)	✗	✗	✗
OS를 사용한 가상 게스트 설치(이미지 템플릿)	✓	✓	✓
시스템 배포(PXE/킥스타트)	✓	✓	✓
시스템 재배포(킥스타트)	✓	✓	✓
연락 방법	✓ ZeroMQ, Salt-SSH	✓ ZeroMQ, Salt-SSH	✓ ZeroMQ, Salt-SSH
Uyuni 프록시와 함께 작동	✓	✓	✓
작업 체인	✓	✓	✓
스테이징(패키지 사전 다운로드)	✓	✓	✓
중복 패키지 보고	✓	✓	✓
CVE 감사	✓	✓	✓
SCAP 감사	✓	✓	✓
패키지 검증	✗	✗	✗
패키지 잠금	✓	✓	✓
시스템 잠금	✗	✗	✗
유지보수 기간	✓	✓	✓
시스템 스냅샷	✗	✗	✗
구성 파일 관리	✓	✓	✓
스냅샷 및 프로필	✓ 프로필 지원됨, 동기화 지원되지 않음	✓ 프로필 지원됨, 동기화 지원되지 않음	✓ 프로필 지원됨, 동기화 지원되지 않음
전원 관리	✓	✓	✓
모니터링 서버	✗	✗	✗
모니터링되는 클라이언트	✓	✓	✓
Docker 빌드호스트	✓	✓	✓
OS를 사용한 Docker 이미지 빌드	?	?	?

기능	RHEL 7	RHEL 8	RHEL 9
Kiwi 빌드호스트	✗	✗	✗
OS를 사용한 Kiwi 이미지 빌드	✗	✗	✗
반복 작업	✓	✓	✓
AppStreams	N/A	✓	✓
Yomi	N/A	N/A	N/A

* 가상 게스트 관리:

이 표에서 가상 게스트 관리는 기본과 고급으로 구분되어 있습니다.

기본 가상 게스트 관리에는 VM, 느린 새로 고침, VM 라이프싸이클 작업(시작, 중지, 다시 시작, 일시 중지) 나열과 VM vCPU 및 메모리 설정이 포함됩니다.

고급 가상 게스트 관리에는 기본 가상 게스트 관리의 모든 기능과 함께 빠른 새로 고침, VM 라이프싸이클 작업(삭제, 재설정, 전원 끄기), VM 디스크, 네트워크, 그래픽 표시 및 그래픽 표시 구성 설정이 포함됩니다.

1.16. 지원하는 Rocky Linux 기능

아래 표는 Rocky Linux 클라이언트에서 다양한 기능의 사용 가능 여부를 정리한 것입니다.



클라이언트에서 실행하는 운영 체제는 운영 체제를 제공하는 조직이 지원합니다. Rocky Linux는 Rocky Linux 커뮤니티가 지원합니다.

이 표의 아이콘에는 다음과 같은 의미가 있습니다.

- ✓ 이 기능을 사용할 수 있음
- ✗ 이 기능을 사용할 수 없습니다.
- ? 이 기능은 현재 고려 중이며, 이후에 사용할 수도 있고 사용하지 못할 수도 있습니다.

표 16. Rocky Linux 운영 체제에서 지원하는 기능

기능	Rocky Linux 8	Rocky Linux 9
클라이언트	✓ (plain Rocky Linux)	✓ (plain Rocky Linux)
시스템 패키지	Rocky Linux Community	Rocky Linux Community
등록	✓	✓
패키지 설치	✓	✓
패치 적용	✓	✓
원격 명령	✓	✓
시스템 패키지 상태	✓	✓
시스템 사용자 정의 상태	✓	✓

기능	Rocky Linux 8	Rocky Linux 9
그룹 사용자 정의 상태	✓	✓
조직 사용자 정의 상태	✓	✓
시스템 세트 관리자(SSM)	✓	✓
제품 마이그레이션	N/A	N/A
기본 가상 게스트 관리 *	✓	✓
고급 가상 게스트 관리 *	✓	✓
OS를 사용한 가상 게스트 설치(킥스타트)	✗	✗
OS를 사용한 가상 게스트 설치(이미지 템플릿)	✓	✓
시스템 배포(PXE/킥스타트)	✓	✓
시스템 재배포(킥스타트)	✓	✓
연락 방법	✓ ZeroMQ, Salt-SSH	✓ ZeroMQ, Salt-SSH
Uyuni 프록시와 함께 작동	✓	✓
작업 체인	✓	✓
스테이징(패지지 사전 다운로드)	✓	✓
중복 패키지 보고	✓	✓
CVE 감사	✓	✓
SCAP 감사	✓	✓
패키지 검증	✗	✗
패키지 잠금	✓	✓
시스템 잠금	✗	✗
유지보수 기간	✓	✓
시스템 스냅샷	✗	✗
구성 파일 관리	✓	✓
스냅샷 및 프로필	✓ 프로필 지원됨, 동기화 지원되지 않음	✓ 프로필 지원됨, 동기화 지원되지 않음
전원 관리	✓	✓
모니터링 서버	✗	✗
모니터링되는 클라이언트	✓	✓
Docker 빌드호스트	✗	✗
OS를 사용한 Docker 이미지 빌드	✗	✗
Kiwi 빌드호스트	✗	✗
OS를 사용한 Kiwi 이미지 빌드	✗	✗

기능	Rocky Linux 8	Rocky Linux 9
반복 작업	✓	✓
AppStreams	✓	✓
Yomi	N/A	N/A

* 가상 게스트 관리:

이 표에서 가상 게스트 관리는 기본과 고급으로 구분되어 있습니다.

기본 가상 게스트 관리에는 VM, 느린 새로 고침, VM 라이프싸이클 작업(시작, 중지, 다시 시작, 일시 중지) 나열과 VM vCPU 및 메모리 수정이 포함됩니다.

고급 가상 게스트 관리에는 기본 가상 게스트 관리의 모든 기능과 함께 빠른 새로 고침, VM 라이프싸이클 작업(삭제, 재설정, 전원 끄기), VM 디스크, 네트워크, 그래픽 표시 및 그래픽 표시 구성 수정이 포함됩니다.

1.17. 지원하는 Ubuntu 기능

아래 표는 Ubuntu 클라이언트에서 다양한 기능의 사용 가능 여부를 정리한 것입니다.



클라이언트에서 실행하는 운영 체제는 운영 체제를 제공하는 조직이 지원합니다. Ubuntu는 Canonical이 지원합니다.

이 표의 아이콘에는 다음과 같은 의미가 있습니다.

- ✓ 이 기능을 사용할 수 있음
- ✗ 이 기능을 사용할 수 없습니다.
- ? 이 기능은 현재 고려 중이며, 이후에 사용할 수도 있고 사용하지 못할 수도 있습니다.

표 17. Ubuntu 운영 체제에서 지원하는 기능

기능	Ubuntu 20.04	Ubuntu 22.04
클라이언트	✓	✓
시스템 패키지	Canonical	Canonical
등록	✓	✓
패키지 설치	✓	✓
패치 적용	✓	✓
원격 명령	✓	✓
시스템 패키지 상태	✓	✓
시스템 사용자 정의 상태	✓	✓
그룹 사용자 정의 상태	✓	✓
조직 사용자 정의 상태	✓	✓
시스템 세트 관리자(SSM)	✓	✓

기능	Ubuntu 20.04	Ubuntu 22.04
제품 마이그레이션	N/A	N/A
기본 가상 게스트 관리 *	✓	✓
고급 가상 게스트 관리 *	✓	✓
호스트 OS로 가상 게스트 설치(킥스타트)	✗	✗
호스트 OS로 가상 게스트 설치(이미지 템플릿)	✓	✓
시스템 배포(PXE/킥스타트)	✗	✗
시스템 재배포(킥스타트)	✗	✗
연락 방법	✓ ZeroMQ, Salt-SSH	✓ ZeroMQ, Salt-SSH
Uyuni 프록시와 함께 작동	✓	✓
작업 체인	✓	✓
스테이징(패키지 사전 다운로드)	✓	✓
중복 패키지 보고	✓	✓
CVE 감사	?	?
SCAP 감사	?	?
패키지 검증	✗	✗
패키지 잠금	✓	✓
시스템 잠금	✗	✗
시스템 스냅샷	✗	✗
구성 파일 관리	✓	✓
패키지 프로필	✓ 프로필 지원됨, 동기화 지원되지 않음	✓ 프로필 지원됨, 동기화 지원되지 않음
전원 관리	✓	✓
모니터링	✓	✓
Docker 빌드호스트	?	?
OS를 사용한 Docker 이미지 빌드	✓	✓
Kiwi 빌드호스트	✗	✗
OS를 사용한 Kiwi 이미지 빌드	✗	✗
반복 작업	✓	✓
AppStreams	N/A	N/A
Yomi	N/A	N/A

* 가상 게스트 관리:

이 표에서 가상 게스트 관리는 기본과 고급으로 구분되어 있습니다.

기본 가상 게스트 관리에는 VM, 느린 새로 고침, VM 라이프사이클 작업(시작, 중지, 다시 시작, 일시 중지) 나열과 VM vCPU 및 메모리 설정이 포함됩니다.

고급 가상 게스트 관리에는 기본 가상 게스트 관리의 모든 기능과 함께 빠른 새로 고침, VM 라이프사이클 작업(삭제, 재설정, 전원 끄기), VM 디스크, 네트워크, 그래픽 표시 및 그래픽 표시 구성 설정이 포함됩니다.

Chapter 2. 구성 기본

Uyuni에서는 클라이언트 등록을 위한 환경을 준비하기 위해 여러 단계를 거친 후 매우 다양한 작업을 활용해야 합니다.

이 섹션에는 성공적인 Uyuni 설치 및 설정 후 환경 작업을 지원하기 위해 필요한 초기 구성 단계에 대한 요약이 포함되어 있습니다.

- Uyuni 설치에 대한 자세한 정보는 [Installation-and-upgrade > Install-uyuni](#)에서 참조하십시오.
- Uyuni 설정에 대한 자세한 정보는 [Installation-and-upgrade > Uyuni-server-setup](#)에서 참조하십시오.

2.1. 소프트웨어 채널

채널은 소프트웨어 패키지를 그룹화하는 방법입니다. 소프트웨어 패키지는 리포지토리가 제공하며, 리포지토리는 채널과 연결되어 있습니다. 클라이언트를 소프트웨어 채널에 가입하면 클라이언트는 자신과 연결된 모든 소프트웨어를 설치하고 업데이트할 수 있습니다.

Uyuni에서 채널은 기본 채널과 하위 채널로 구분됩니다. 이런 식으로 채널을 구성하면 호환되는 패키지만 각 시스템에 설치됩니다. 클라이언트는 등록 중에 클라이언트 운영 체제 및 아키텍처에 따라 할당된 하나의 기본 채널에만 가입되어야 합니다. 벤더가 제공하는 유료 채널에 대해서는 연결된 구독이 있어야 합니다.

기본 채널은 특정 운영 체제 유형, 버전 및 아키텍처를 위해 구축된 패키지로 구성됩니다. 예를 들어 SUSE Linux Enterprise Server 15 x86-64 기본 채널에는 이 운영 체제 및 아키텍처와 호환되는 소프트웨어만 포함되어 있습니다.

하위 채널은 기본 채널과 연결되어 있으며 기본 채널과 호환되는 패키지만 제공합니다. 시스템은 기본 채널의 여러 하위 채널에 가입할 수 있습니다. 시스템이 기본 채널에 할당된 경우 해당 시스템은 관련 하위 채널만 설치할 수 있습니다. 예를 들어 시스템에 SUSE Linux Enterprise Server 15 x86_64 기본 채널이 할당된 경우 호환되는 기본 채널이나 기본 채널과 연결된 하위 채널 중 하나를 통해 사용할 수 있는 패키지만 설치하거나 업데이트할 수 있습니다.

Uyuni Web UI에서 [소프트웨어 > 채널 목록](#), 전체로 이동하여 사용 가능한 채널을 검색할 수 있습니다. [소프트웨어 > 관리 > 채널](#)로 이동하여 새 채널을 수정하거나 생성할 수 있습니다.

사용자 지정 채널을 포함한 채널 사용에 대한 자세한 내용은 [Administration > Channel-management](#)을 참조하십시오.

2.1.1. SUSE Package Hub가 제공하는 패키지

SUSE Package Hub는 SUSE Linux Enterprise 제품의 확장으로, openSUSE 커뮤니티가 제공하는 추가 오픈 소스 소프트웨어를 제공합니다.



- SUSE Package Hub의 패키지는 openSUSE 커뮤니티가 제공합니다. 이 패키지는 SUSE에서 지원되지 않습니다.

클라이언트에서 SUSE Linux Enterprise 운영 체제를 사용 중인 경우 SUSE Package Hub 확장을 활성화하여 이러한 추가 패키지에 액세스할 수 있습니다. 이렇게 하면 클라이언트를 가입할 수 있는 SUSE Package Hub 채널이 제공됩니다.

SUSE Package Hub는 다수의 패키지를 제공하므로 동기화하는 데 시간이 오래 걸리고 대량의 디스크 공간을 소비할 수 있습니다. SUSE Package Hub가 제공하는 패키지가 필요하지 않은 경우 SUSE Package Hub를 활성화하지

마십시오.

지원되지 않는 패키지를 의도치 않게 설치하거나 업데이트하지 않도록 하려면 처음에 모든 SUSE Package Hub 패키지를 거부하는 컨텐트 라이프싸이클 관리 전략을 구현하는 것이 좋습니다. 그러면 필요한 특정 패키지를 명시적으로 활성화할 수 있습니다. 컨텐트 라이프싸이클 관리에 대한 자세한 내용은 **Administration > Content-lifecycle**을 참조하십시오.

2.1.2. AppStream이 제공하는 패키지

Red Hat 기반 클라이언트의 경우 AppStream을 통해 추가 패키지가 제공됩니다. 대부분의 경우 AppStream 패키지가 있어야 필요한 소프트웨어를 모두 보유할 수 있습니다.

2.1.3. EPEL이 제공하는 패키지

Red Hat 기반 클라이언트의 경우 EPEL(엔터프라이즈 Linux용 추가 패키지)을 통해 추가 패키지가 제공됩니다. EPEL은 추가 소프트웨어를 제공하는 옵션 패키지 리포지토리입니다.



- EPEL의 패키지는 Fedora 커뮤니티가 제공합니다. 이 패키지는 SUSE에서 지원하지 않습니다.

클라이언트에서 Red Hat 운영 체제를 사용 중인 경우 EPEL 확장을 활성화하여 이러한 추가 패키지에 액세스할 수 있습니다. 이렇게 하면 클라이언트를 가입할 수 있는 EPEL 채널이 제공됩니다.

EPEL은 다수의 패키지를 제공하므로 동기화하는 데 시간이 오래 걸리고 대량의 디스크 공간을 소비할 수 있습니다. EPEL이 제공하는 패키지가 필요하지 않은 경우 EPEL 리포지토리를 활성화하지 마십시오.

지원되지 않는 패키지를 의도치 않게 설치하거나 업데이트하지 않도록 하려면 처음에 모든 EPEL 패키지를 거부하는 컨텐트 라이프싸이클 관리(CLM) 전략을 구현하는 것이 좋습니다. 그러면 필요한 특정 패키지를 명시적으로 활성화할 수 있습니다. 컨텐트 라이프싸이클 관리에 대한 자세한 내용은 **Administration > Content-lifecycle**을 참조하십시오.

2.1.4. SUSE Linux Enterprise 클라이언트의 Unified Installer 업데이트 채널

이 채널은 운영 체제를 설치하기 전에 최신 상태인지 확인하기 위해 Unified Installer에서 사용합니다. 모든 SUSE Linux Enterprise 제품은 설치 중에 설치 프로그램 업데이트 채널에 액세스할 수 있는 권한이 있어야 합니다.

SUSE Linux Enterprise Server 클라이언트의 경우 설치 프로그램 업데이트 채널은 이 채널을 포함하는 제품을 추가하면 기본적으로 동기화되고 이 제품 채널로 자동 설치 가능한 배포판을 생성하면 활성화됩니다.

SAP용 SUSE Linux Enterprise를 포함한 다른 모든 SUSE Linux Enterprise 변형의 경우 설치 프로그램 업데이트 채널을 수동으로 추가해야 합니다. 이 작업을 수행하려면 SUSE Linux Enterprise 변형의 기본 채널 아래에 적절한 SUSE Linux Enterprise Server 설치 프로그램 업데이트 채널을 복제해야 합니다. 채널을 복제한 후 SUSE Linux Enterprise 변형에 대해 자동 설치 가능한 배포판을 생성할 때 이 채널이 자동으로 사용됩니다.

2.1.5. 소프트웨어 리포지토리

리포지토리는 소프트웨어 패키지를 수집하는 데 사용됩니다. 소프트웨어 리포지토리에 액세스할 수 있으면 리포지토리가 제공하는 모든 소프트웨어를 설치할 수 있습니다. 클라이언트를 채널에 할당하고 클라이언트에서 패키지를 설치하고 업데이트하려면 Uyuni에 소프트웨어 채널과 연결된 리포지토리가 최소 한 개 있어야 합니다.

Uyuni의 기본 채널은 대부분 이미 올바른 리포지토리에 연결되어 있습니다. 사용자 정의 채널을 생성하려면 액세스할 수 있는 또는 사용자가 직접 생성한 리포지토리를 연결해야 합니다.

사용자 정의 리포지토리에 대한 자세한 내용은 **Administration > Custom-channels**에서 참조하십시오.

2.1.5.1. 로컬 리포지토리 위치

클라이언트에서 로컬 리포지토리를 구성하여 Uyuni 채널이 제공하지 않는 패키지를 제공할 수 있습니다.



대부분의 경우 클라이언트 시스템에는 로컬 리포지토리가 필요 없습니다. 로컬 리포지토리로 인해 클라이언트에서 어떤 패키지를 사용할 수 있는지 파악하는 데 문제가 생길 수 있습니다. 그러면 결국 예상치 않은 패키지를 설치하게 될 수 있습니다.

온보딩 중에는 로컬 리포지토리가 비활성화됩니다.

채널 상태가 실행될 때마다 로컬 리포지토리가 비활성화됩니다. 그러한 경우로는 Highstate를 적용하거나 패키지 작업을 수행하는 경우입니다.

온보딩 후에도 로컬 리포지토리를 계속 사용하도록 설정해야 하는 경우 해당 클라이언트에 대해 다음 열을 설정해야 합니다.

/srv/pillar/top.sls 파일 편집:

```
base:
  'minionid':
    - localrepos
```

/srv/pillar/localrepos.sls 파일 편집:

```
mgr_disable_local_repos: False
```

클라이언트가 온보딩을 완료하고 나면 다음 위치에 로컬 리포지토리를 추가할 수 있습니다.

표 18. 로컬 리포지토리 위치

클라이언트 운영 체제	로컬 리포지토리 디렉토리
SUSE Linux Enterprise Server	/etc/zypp/repos.d
openSUSE	/etc/zypp/repos.d
Red Hat Enterprise Linux and similar derivatives	/etc/yum.repos.d/
Ubuntu	/etc/apt/sources.list.d/
Debian	/etc/apt/sources.list.d/

2.1.6. 소프트웨어 제품

Uyuni에서는 소프트웨어를 제품 내에서 제공합니다. SUSE 구독을 통해 다양한 제품에 액세스할 수 있습니다. 이러한 제품은 Uyuni Web UI에서 **관리** > **설치 마법사** > **제품**으로 이동하여 검색하고 선택할 수 있습니다.

제품에는 소프트웨어 채널이 포함되어 있는데, 그 개수는 다양합니다. ⌂ ⌂ ⌂ 아이콘을 클릭하여 제품에 포함된 채널을 확인합니다. 제품을 추가하고 동기화를 완료하고 나면 제품이 제공하는 채널에 액세스할 수 있고 Uyuni 서버 및 클라이언트에서 제품에 있는 패키지를 사용할 수 있습니다.

절차: 소프트웨어 채널 추가

1. Uyuni Web UI에서 **관리** > **설치 마법사** > **제품**으로 이동합니다.
2. 검색 창을 사용하여 클라이언트 운영 체제 및 아키텍처에 적합한 제품을 찾고 해당 제품을 확인하십시오. 모든 필수 채널은 자동으로 확인됩니다. 또한, ⌂ ⌂ 토글이 켜져 있으면 모든 추천 채널이 확인됩니다. 관련 제품의 전체 목록을 보려면 화살표를 클릭하고 필요한 추가 제품이 선택되어 있는지 확인하십시오.
3. **[제품 추가]**를 클릭하고 제품이 동기화를 마칠 때까지 기다립니다.

2.1.7. AppStream 모듈

AppStream 채널에 가입한 클라이언트를 관리하는 경우 Uyuni은 현재 활성화된 AppStream 모듈 세트에 따라 클라이언트에서 사용할 수 있는 패키지를 필터링합니다.

절차: 활성화된 AppStream 모듈 관리

1. Uyuni Web UI에서 **시스템** > **소프트웨어** > **AppStreams**로 이동합니다.
2. 모듈 스트림 테이블에서 클라이언트에서 활성화 할 스트림을 선택합니다.
3. **[변경 사항 적용]**을 클릭합니다.
4. 다음 페이지에서 변경할 목록을 확인하고 작업을 수행할 시간을 선택합니다.
5. **[확인]**을 클릭하여 적용할 변경 사항을 예약합니다.

2.2. 부트스트랩 리포지토리

부트스트랩 리포지토리에는 부트스트랩 중에 클라이언트를 등록하기 위한 필수 패키지가 포함되어 있습니다. 제품이 동기화되면 부트스트랩 리포지토리가 자동으로 생성되고 Uyuni 서버에서 다시 생성됩니다.

2.2.1. 부트스트랩 리포지토리 생성 준비

동기화를 위해 제품을 선택하면 모든 필수 채널이 완전히 미러링되자마자 부트스트랩 리포지토리가 자동으로 생성됩니다.

절차: Web UI에서 동기화 진행 상태 확인

1. Uyuni Web UI에서 **소프트웨어** > **관리** > **채널**로 이동한 다음, 리포지토리에 연결된 채널을 클릭합니다.
2. ⌂ ⌂ 탭으로 이동한 다음, ⌂를 클릭하고 ⌂를 확인합니다.

절차: 명령 프롬프트에서 동기화 진행 상태 확인

1. Uyuni 서버의 명령 프롬프트에서 루트 권한으로 다음과 같이 `tail` 명령을 사용해 동기화 로그 파일을

확인합니다.

```
tail -f /var/log/rhn/reposync/<channel-label>.log
```

- 각 하위 채널은 동기화가 진행되는 동안 자체 로그를 생성합니다. 동기화가 완료되었는지 알아보려면 모든 기본 및 하위 채널 로그 파일을 확인해야 합니다.

2.2.2. 자동 모드를 위한 옵션

자동 부트스트랩 리포지토리가 생성되는 방식을 변경할 수 있습니다. 이 섹션에서는 다양한 설정에 대해 자세히 설명합니다.

플러시 모드::

플러시 모드

기본적으로 기존 리포지토리는 최신 패키지로만 업데이트됩니다. 대신, 빈 리포지토리로 항상 시작하도록 구성할 수 있습니다. 이 동작을 활성화하려면 다음과 같이 `/etc/rhn/rhn.conf`에서 이 값을 추가하거나 편집합니다.

```
server.susemanager.bootstrap_repo_flush = 1
```

자동 모드::

자동 모드

기본적으로 부트스트랩 리포지토리 자동 재생성은 활성화되어 있습니다. 이 작업을 비활성화하려면 다음과 같이 `/etc/rhn/rhn.conf`에서 이 값을 추가하거나 편집합니다.

```
server.susemanager.auto_generate_bootstrap_repo = 0
```

2.2.2.1. 부트스트랩 데이터 파일 구성

도구는 각 배포에 어떤 패키지가 필요한지에 대한 정보가 포함된 데이터 파일을 사용합니다. 이 데이터 파일은 `/usr/share/susemanager/mgr_bootstrap_data.py`에 저장되어 있습니다. SUSE는 이 파일을 정기적으로 업데이트합니다. 이 파일을 변경하고 싶은 경우 직접 편집하지 마십시오. 대신, 다음과 같이 동일한 디렉토리에 사본을 생성하고 이 사본을 편집합니다.

```
cd /usr/share/susemanager/
cp mgr_bootstrap_data.py my_data.py
```

변경을 완료했으면 새 파일을 사용하도록 Uyuni를 구성합니다. 다음과 같이 `/etc/rhn/rhn.conf`에서 이 값을 추가하거나 편집합니다.

```
server.susemanager.bootstrap_repo_datamodule = my_data
```



- 다음 업데이트에서 SUSE의 새 데이터는 새 데이터 파일이 아닌 원본 데이터 파일을 덮어씁니다. SUSE에서 제공하는 변경 사항으로 새 파일을 최신 상태로 유지해야 합니다.

2.2.3. 부트스트랩 리포지토리를 수동으로 생성

기본적으로 부트스트랩 리포지토리는 매일 다시 생성됩니다. 명령 프롬프트에서 부트스트랩 리포지토리를 수동으로 생성할 수 있습니다.

절차: SUSE Linux Enterprise용 부트스트랩 리포지토리 생성

1. Uyuni 서버의 명령 프롬프트에서 루트 권한으로 다음과 같이 부트스트랩 리포지토리를 생성하는 데 사용 가능한 배포를 나열합니다.

```
mgr-create-bootstrap-repo -l
```

2. 다음과 같이 적절한 리포지토리 이름을 제품 레이블로 사용해 부트스트랩 리포지토리를 생성합니다.

```
mgr-create-bootstrap-repo -c SLE-version-x86_64
```

3. 또는 사용 가능한 배포 목록에서 배포 이름 옆에 표시된 번호를 사용합니다.

클라이언트 리포지토리는 `/srv/www/htdocs/pub/repositories`에 있습니다.

제품을 두 개 이상 미러링한 경우(예: SLES, SAP용 SLES) 또는 사용자 지정 채널을 사용하는 경우 부트스트랩 리포지토리를 생성할 때 사용할 상위 채널을 지정해야 합니다. 이는 모든 상황에서 필수가 아닙니다. 예를 들어, 일부 SLES 15 버전에는 공통 코드 기반이 있으므로, 상위 채널을 지정할 필요가 없습니다. 환경에서 필요한 경우에만 이 절차를 사용하십시오.

선택 사항 절차: 부트스트랩 리포지토리에 상위 채널 지정

1. 사용 가능한 상위 채널을 확인합니다.

```
mgr-create-bootstrap-repo -c SLE-15-x86_64
```

여러 개의 상위 채널용 옵션을 찾았습니다.

--with-parent-channel <label>옵션을 사용하고 다음 중 하나를 선택하십시오.
 - sle-product-sles15-pool-x86_64
 - sle-product-sles_sap15-pool-x86_64
 - sle-product-sled15-pool-x86_64

2. 다음과 같이 적절한 상위 채널을 지정합니다.

```
mgr-create-bootstrap-repo -c SLE-15-x86_64 --with-parent-channel  
sle-product-sled15-pool-x86_64
```

2.2.3.1. 아키텍처가 여러 개인 리포지토리

아키텍처가 여러 개 포함된 부트스트랩 리포지토리를 생성하는 경우 모든 아키텍처가 올바르게 업데이트되는지 주의 깊게 살펴봐야 합니다. 예를 들어 SLE용 IBM Z 및 x86-64 아키텍처가 /srv/www/htdocs/pub/repositories/sle/15/2/bootstrap/에서 동일한 부트스트랩 리포지토리 URL을 사용합니다.

flush 옵션이 활성화된 상태에서 여러 아키텍처에 대해 부트스트랩 리포지토리를 생성하려고 하면 하나의 아키텍처만 생성됩니다. 이를 방지하려면 추가 아키텍처를 생성할 때 명령 프롬프트에서 --no-flush 옵션을 사용하십시오. 예:

```
mgr-create-bootstrap-repo -c SLE-15-SP2-x86_64
mgr-create-bootstrap-repo --no-flush -c SLE-15-SP2-s390x
```

2.2.4. 부트스트랩 및 사용자 지정 채널

사용자 지정 채널을 사용 중인 경우 --with-custom-channels 옵션을 mgr-create-bootstrap-repo 명령과 함께 사용할 수 있습니다. 이 경우 사용할 상위 채널도 지정해야 합니다.

사용자 지정 채널을 사용 중인 경우 부트스트랩 리포지토리 자동 생성에 실패할 수 있습니다. 이러한 경우에는 리포지토리를 수동으로 생성해야 합니다.

사용자 지정 채널에 대한 자세한 내용은 **Administration > Custom-channels**에서 참조하십시오.

2.3. 활성화 키

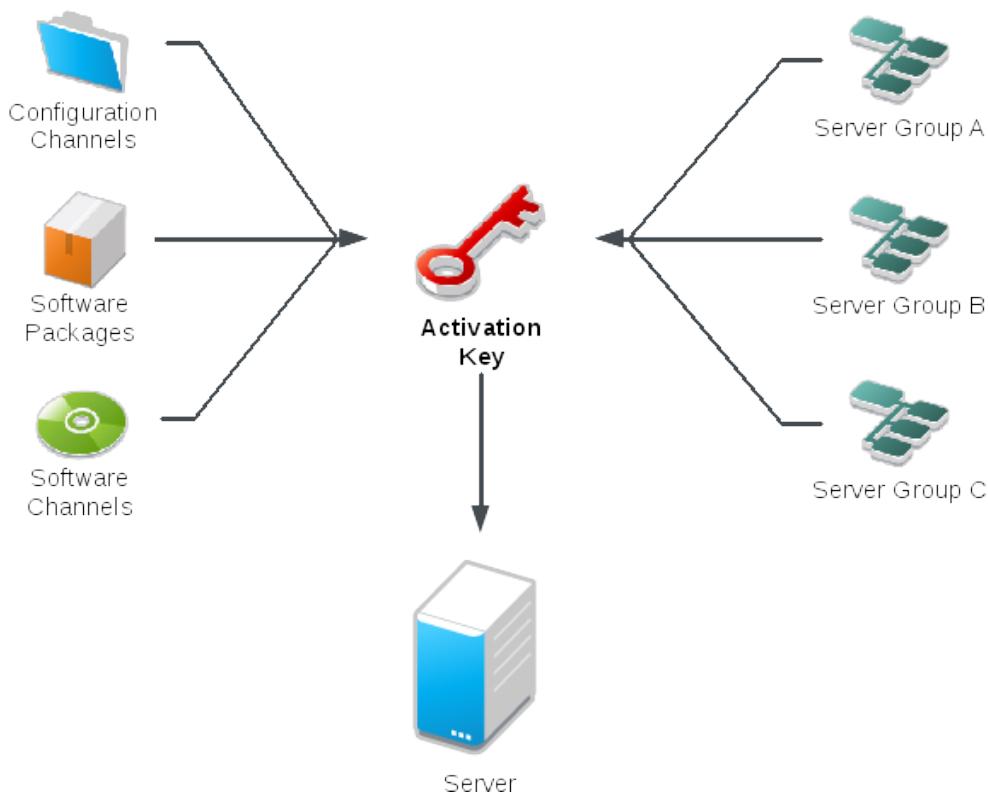
활성화 키는 고객에게 적합한 소프트웨어 자격이 있는지, 적절한 채널에 연결되었는지, 관련 그룹에 가입되어 있는지를 확인하는 데 사용됩니다. 각 활성화 키는 키를 생성할 때 설정할 수 있는 조직에 바인딩됩니다.

Uyuni에서 활성화 키는 레이블이 있는 구성 설정 그룹입니다. 레이블을 부트스트랩 스크립트에 파라미터로 추가하여 활성화 키와 연결된 모든 구성 설정을 적용할 수 있습니다. 활성화 키 레이블을 부트스트랩 스크립트와 함께 사용하는 것이 좋습니다. 부트스트랩 스크립트가 실행될 때 레이블에 연결된 모든 구성 설정은 스크립트가 실행되는 시스템에 적용됩니다.

활성화 키는 다음을 지정할 수 있습니다.

- 채널 할당
- 시스템 유형 또는 추가 자격
- 연락 방법
- 구성 파일
- 설치할 패키지
- AppStreams to be enabled
- 시스템 그룹 할당

활성화 키는 클라이언트 등록 시 사용되며 다시 사용되지 않습니다. 클라이언트는 등록된 후에 지정된 활성화 키와 관계없이 변경할 수 있습니다. 활성화 키와 클라이언트 사이의 관계는 기록 목적으로만 기록됩니다.



절차: 활성화 키 생성

1. Uyuni Web UI에서 관리자 권한으로 **시스템 > 활성화 키**로 이동합니다.
 2. **[키 생성]** 버튼을 클릭합니다.
 3. **활성화 키 이름** 필드에 활성화 키의 설명을 입력합니다.
 4. **활성화 키 이름** 필드에 활성화 키의 이름을 입력합니다. 예를 들어, SUSE Linux Enterprise Server 15 SP5의 경우 **SLES15-SP5**을(를) 입력합니다.
- SUSE 제품인 경우** **활성화 키 이름** 필드에 쉼표를 사용하지 마십시오. 하지만 Red Hat 제품인 경우 **반드시** 쉼표를 사용해야 합니다.

 - 기타 모든 문자는 허용되지만 <> (){}(공백 포함)는 자동으로 제거됩니다.
 - 필드가 비어 있으면 임의의 스트링이 생성됩니다.
5. **설정** 드롭다운 상자에서 적절한 기본 소프트웨어 채널을 선택하고 관련 하위 채널이 채워지도록 허용합니다. 자세한 내용은 [reference:admin/setup-wizard.pdf](#)와 **Administration > Custom-channels**에서 확인할 수 있습니다.
 6. 필요한 하위 채널을 선택합니다(예: 필수 SUSE Manager 도구 및 업데이트 채널).
 7. 옵션을 활성화하려면 **선택** 확인란을 선택합니다.
 8. **선택**은 **선택**으로 설정된 상태 그대로 두는 것이 좋습니다.
 9. **선택** 설정은 확인란이 선택되지 않은 상태 그대로 두는 것이 좋습니다.
 10. **[활성화 키 생성]**을 클릭하여 활성화 키를 생성합니다.

11. →→→ 확인란을 선택하여 이 키에 대한 구성 관리를 활성화하고, **[활성화 키 업데이트]**를 클릭하여 이 변경 사항을 저장합니다.



→→→ 확인란은 활성화 키를 생성한 후에야 표시됩니다. 구성 관리를 활성화해야 하는 경우 되돌아가 확인란을 선택하십시오.

When created, tabs such as

- Packages
- Configuration
- Groups
- Activated Systems
- AppStreams (available whenever a modular channel is associated with the activation key)

allow you to check and set additional features.

2.3.1. 활성화 키

클라이언트를 다시 등록하고 모든 Uyuni 설정을 다시 확보하기 위한 목적으로 재활성화 키를 한 번만 사용할 수 있습니다. 재활성화 키는 클라이언트 전용이며 시스템 ID, 이력, 그룹 및 채널을 포함합니다.

재활성화 키를 생성하려면 →→으로 이동한 후 클라이언트를 클릭하여 재활성화 키를 생성하고, **상세 정보** > **재활성** 탭으로 이동하십시오. **[새 키 생성]**을 클릭하여 재활성화 키를 생성합니다. 나중에 사용할 수 있도록 키의 상세 정보를 기록해 둡니다. 특정 시스템 ID와 연결되지 않은 일반적인 활성화 키와 달리 여기에서 생성한 키는 **시스템** > **활성화 키** 페이지에 표시되지 않습니다.

재활성화 키를 생성한 후에는 /etc/salt/minion.d/susemanager.conf에서 management_key 입자로 사용할 수 있습니다. 예:

```
grains:
  susemanager:
    management_key: "re-1-daf44db90c0853edbb5db03f2b37986e"
```

재활성화 키를 적용하려면 venv-salt-minion 또는 salt-minion 프로세스를 다시 시작합니다.

재활성화 키를 부트스트랩 스크립트와 함께 사용할 수 있습니다. 부트스트랩 스크립트에 대한 자세한 내용은 **Client-configuration** > **Registration-bootstrap**을 참조하십시오.



클라이언트를 기존 Uyuni 프로파일과 함께 자동 설치하는 경우 프로파일은 재활성화 키를 사용해 시스템을 다시 등록하고 설정을 복원합니다. 프로파일 기반 자동 설치가 진행 중일 때는 이 키를 재생성, 삭제 또는 사용하지 마십시오. 그러면 자동 설치에 실패하게 됩니다.

2.3.2. 활성화 키 모범 사례

기본 상위 채널

SUSE Manager --- 상위 채널을 사용하지 마십시오. 이 설정은 Uyuni가 설치된 운영 체제와 가장 부합하는 상위 채널을 선택하도록 강제합니다. 이로 인해 때로 예기치 않은 동작이 발생할 수 있습니다. 대신에 각 배포 및 아키텍처별로 활성화 키를 생성하는 것이 좋습니다.

활성화 키를 이용한 부트스트래핑

부트스트랩 스크립트를 사용 중인 경우 각 스크립트에 대해 활성화 키를 생성하는 것을 고려하십시오. 이렇게 하면 채널 할당, 패키지 설치, 시스템 그룹 구성원, 구성 채널 할당을 조정하는 데 도움이 됩니다. 또한 등록 후 시스템에 대한 수동 개입이 덜 필요합니다.

LTSS 클라이언트 부트스트래핑

LTSS 구독을 사용하여 클라이언트를 부스트랩하는 경우 활성화 키 생성 중에 LTSS 채널을 포함시켜야 합니다.

대역폭 요구사항

활성화 키를 사용하면 등록 시점에 소프트웨어가 자동으로 다운로드될 수 있는데, 대역폭이 제한된 환경에서는 바람직하지 않을 수 있습니다.

이러한 옵션은 대역폭 사용량을 생성합니다.

- SUSE 제품 풀 채널을 할당하면 해당 제품 설명자 패키지가 자동으로 설치됩니다.
- 섹션의 모든 패키지가 Packages 설치됩니다.
- -- 섹션의 모든 Salt 상태가 그 내용에 따라 다운로드를 트리거할 수 있습니다.

키 레이블 명명

활성화 키에 대해 읽을 수 있는 이름을 입력하지 않으면 시스템이 수 스트링을 자동으로 생성하므로 키를 관리하기 어려울 수 있습니다.

키를 추적하는 데 도움이 되는 활성화 키 명명 스키마를 고려하십시오. 조직의 인프라와 관련이 있는 이름을 생성하면 더 복잡한 작업을 더 쉽게 수행할 수 있습니다.

키 레이블을 생성할 때 다음과 같은 팁을 참고하십시오.

- OS 명명(필수): 키는 항상 자신이 설정을 제공하는 대상인 OS를 참조해야 합니다.
- 아키텍처 명명(권장): 귀사가 단 하나의 아키텍처(예: x86_64)만 운영하고 있지 않다면 레이블에 아키텍처 유형을 제공하는 것이 좋습니다.
- 서버 유형 명명: 이 서버를 어떤 목적으로 사용 중입니까?
- 위치 명명: 서버가 어디에 있습니까? 서버룸, 건물 또는 부서 내?
- 날짜 명명: 유지보수 기간, 분기 등
- 사용자 정의 명명: 어떤 명명 스키마가 조직의 필요에 적합합니까?

활성화 키 레이블 이름의 예:

```
sles15-sp4-web_server-room_129-x86_64
```

```
sles15-sp4-test_packages-blg_502-room_21-ppc64le
```

포함된 채널

활성화 키를 생성할 때는 어떤 소프트웨어 채널이 연결되어 있는지도 기억해 두어야 합니다. 키에는 할당된 특정 기본 채널이 있어야 합니다. 기본값으로 설정된 기본 채널은 사용하지 않는 것이 좋습니다. 자세한 내용은 **Client-configuration > Registration-overview**에서 설치하려는 클라이언트 운영 체제를 참조하십시오.

2.4. GPG 키

클라이언트는 소프트웨어 패키지가 설치되기 전에 GPG 키를 사용해 신뢰성을 확인합니다. 신뢰할 수 있는 소프트웨어만 클라이언트에 설치할 수 있습니다.

대부분의 경우 GPG 설정을 조정하지 않아도 클라이언트에 소프트웨어를 설치할 수 있습니다.

RPM 패키지에는 직접 서명할 수 있지만, Debian 기반 시스템은 메타데이터에만 서명하고 체크섬 체인을 사용하여 패키지를 보호합니다. 대부분의 RPM 기반 시스템은 서명된 패키지뿐만 아니라 서명된 메타데이터를 사용합니다.

2.4.1. 클라이언트의 GPG 키 신뢰

운영 체제는 자체 GPG 키를 직접 신뢰하거나 최소 시스템 이상과 함께 설치된 상태로 제공됩니다. 그러나 다른 GPG 키로 서명된 타사 패키지는 수동으로 처리해야 합니다. 클라이언트는 신뢰할 수 있는 GPG 키가 없어도 성공적으로 부트스트래핑할 수 있습니다. 하지만 키를 신뢰할 수 있을 때까지는 새 클라이언트 도구 패키지를 설치하거나 업데이트할 수 없습니다.

이제 클라이언트는 소프트웨어 채널에 대해 입력한 GPG 키 정보를 사용하여 신뢰된 키를 관리합니다. GPG 키 정보가 있는 소프트웨어 채널이 클라이언트에 할당되면 채널을 새로 고치거나 이 채널에서 첫 번째 패키지를 설치할 때 키가 신뢰됩니다.

소프트웨어 채널 페이지의 GPG 키 URL 파라미터에는 "공백"으로 구분된 여러 키 URL이 포함될 수 있습니다. 파일 URL인 경우 클라이언트에 GPG 키 파일을 배포한 후 소프트웨어 채널을 사용해야 합니다.

Red Hat 기반 클라이언트의 클라이언트 도구 채널에 대한 GPG 키는 클라이언트의 `/etc/pki/rpm-gpg/`에 배포되며 파일 URL로 참조할 수 있습니다.

소프트웨어 채널이 클라이언트에 할당된 경우에만 시스템에서 가져와서 신뢰하게 됩니다.



Debian 기반 시스템은 메타데이터에만 서명하므로 단일 채널에 대해 추가 키를 지정할 필요가 없습니다. **Administration > Repo-metadata**에서 "자체 GPG 키 사용"의 설명과 같이 사용자가 자체 GPG 키를 구성하여 메타데이터에 서명하는 경우 해당 키의 배포 및 신뢰가 자동으로 실행됩니다.

2.4.1.1. 사용자 정의 GPG 키

사용자는 클라이언트에 배포할 사용자 정의 GPG 키를 정의할 수 있습니다.

일부 열 데이터를 제공하고 Salt 파일 시스템에 GPG 키 파일을 제공하면 클라이언트에 자동으로 배포됩니다.

이러한 키는 RPM 기반 운영 체제의 `/etc/pki/rpm-gpg/` 및 Debian 시스템의 `/usr/share/keyrings/`에 배포됩니다.

키를 배포할 클라이언트에 대한 열 키 `custom_gpgkeys`를 정의하고 키 파일의 이름을 나열합니다.

```
cat /srv/pillar/mypillar.sls
custom_gpgkeys:
  - my_first_gpg.key
  - my_second_gpgkey.gpg
```

추가적으로 Salt 파일 시스템에서 `gpg` 디렉토리를 생성한 후 해당 디렉토리에 `custom_gpgkeys` 열 데이터에 지정된 이름으로 GPG 키 파일을 저장합니다.

```
ls -la /srv/salt/gpg/
/srv/salt/gpg/my_first_gpg.key
/srv/salt/gpg/my_second_gpgkey.gpg
```

키는 클라이언트에 `/etc/pki/rpm-gpg/my_first_gpg.key` 및 `/etc/pki/rpm-gpg/my_second_gpgkey.gpg`에 배포됩니다.

마지막 단계로 소프트웨어 채널의 GPG 키 URL 필드에 URL을 추가합니다. **소프트웨어 > 관리 > 채널**로 이동하고 수정할 채널을 선택합니다. GPG → URL에 `file:///etc/pki/rpm-gpg/my_first_gpg.key` 값을 추가합니다.

2.4.1.2. 부트스트랩 스크립트의 GPG 키

절차: 부트스트랩 스크립트를 사용하여 클라이언트의 GPG 키 신뢰

- Uyuni 서버의 명령 프롬프트에서 `/srv/www/htdocs/pub/` 디렉토리의 내용을 확인합니다. 이 디렉토리에는 사용 가능한 모든 공용 키가 포함되어 있습니다. 등록하려는 클라이언트에 할당된 채널에 적용되는 키를 적어 두십시오.
- 관련 부트스트랩 스크립트를 열어 `ORG_GPG_KEY=` 파라미터를 찾고 필요한 키를 추가합니다. 예:

```
uyuni-gpg-pubkey-0d20833e.key
```

이전에 저장한 키는 삭제하지 않아도 됩니다.



클라이언트의 보안을 위해 GPG 키의 신뢰성이 중요합니다. 어떤 키가 필요하고 신뢰할 수 있는지 결정하는 것은 관리자가 수행해야 하는 작업입니다. GPG 키를 신뢰할 수 없으면 클라이언트에 소프트웨어 채널을 할당할 수 없습니다.

Chapter 3. 클라이언트 연락 방법

Uyuni 서버는 클라이언트와 다양한 방법으로 통신할 수 있습니다. 통신은 Salt 프로토콜을 기반으로 합니다.

사용되는 Salt 연락 방법은 클라이언트 유형과 네트워크 아키텍처에 따라 다릅니다.

기본값(Salt)

이 기본값이며 특별한 요구 사항이 있는 경우를 제외하고 권장됩니다. 자세한 내용은 **Client-configuration > Contact-methods-default**에서 확인할 수 있습니다.

SSH 푸시

는 네트워크 제한으로 인해 클라이언트가 서버에 연결할 수 없는 경우에만 유용합니다. 이는 Salt 번들에서만 지원됩니다. 이 연락 방법에는 심각한 제한이 있습니다. 자세한 내용은 **Client-configuration > Contact-methods-saltssh**에서 확인할 수 있습니다.

SSH 푸시(터널 사용)

는 SSH 푸시와 동일하지만 보안 통신 터널을 사용한다는 점이 다릅니다. 자세한 내용은 **Client-configuration > Contact-methods-saltssh-tunnel**에서 확인할 수 있습니다.

Salt 통신 소프트웨어 옵션 구현:

Salt Bundle

은 Salt를 포함하는 단일 바이너리 패키지입니다. Minion, Python 3, 필수 Python 모듈 및 라이브러리가 포함된 단일 바이너리 패키지입니다. 따라서 Salt 연락 방법은 클라이언트에 설치된 소프트웨어와 독립적입니다. Salt 번들 사용이 기본값입니다. 이 방법은 SSH 푸시 또는 SSH 푸시(터널 사용) 연락 방법에 대해 유일하게 지원되는 구현입니다. 자세한 내용은 **Client-configuration > Contact-methods-saltbundle**에서 확인할 수 있습니다.

Salt 미니언

은 클라이언트 시스템에 설치된 Salt 소프트웨어입니다.



Uyuni 5.0 이상에서는 기존 연락 프로토콜이 더 이상 지원되지 않습니다. 기존 프록시를 포함한 모든 기존 클라이언트를 Salt로 마이그레이션하거나 Salt 프록시로 바꾼 후 Uyuni 4에서 5로 업그레이드해야 합니다.

기존 Uyuni 4 클라이언트를 Salt 클라이언트로 마이그레이션하는 방법에 대한 자세한 내용은 <https://documentation.suse.com/suma/4.3/en/suse-manager/client-configuration/contact-methods-migrate-traditional.html>에서 확인할 수 있습니다.

3.1. 기본 연락 방법(Salt)

특별한 필요가 있는 경우를 제외하고 Salt 프로토콜을 사용하는 기본 연락 방법을 사용하는 것이 좋습니다. 일반적인 Salt에 대한 자세한 내용은 **Specialized-guides > Salt**에서 확인할 수 있습니다.

소프트웨어 업데이트는 일반적으로 서버에서 클라이언트로 푸시됩니다. 연결은 클라이언트에서 시작됩니다. 즉, 클라이언트가 아닌 서버의 포트를 열어야 합니다. Salt 클라이언트를 Salt 미니언이라고도 합니다. Uyuni 서버는 모든 클라이언트에 데몬을 설치합니다.

연결이 해제된 설정에서 Salt 클라이언트를 사용해야 하는 경우 SSH 푸시를 연락 방법으로 구성할 수 있습니다. 이 연락 방법을 사용하면 클라이언트가 방화벽으로 보호되는 이름이 DMZ인 영역에 위치할 수 있습니다. SSH 푸시에 대한 자세한 내용은 **Client-configuration > Contact-methods-saltssh**에서 확인할 수 있습니다.

3.1.1. 온보딩 정보

Salt에는 미니언의 키를 보관하는 자체 데이터베이스가 있습니다. 이 데이터베이스는 Uyuni 데이터베이스와 동기화 상태를 유지해야 합니다. Salt에서 키가 수락되는 즉시 Uyuni에서 온보딩 프로세스가 시작됩니다.

온보딩 프로세스는 Uyuni 데이터베이스에서 'minion_id' 와 'machine-id' 를 검색하여 기존 시스템을 찾습니다. 결과에 따라, 다음과 같은 시나리오가 가능합니다.

- 아무것도 찾을 수 없으면 새 시스템이 생성됩니다.
- minion_id 또는 machine-id가 있는 항목을 찾으면 해당 시스템은 새 시스템과 일치하도록 마이그레이션됩니다.
- 두 항목이 모두 일치하지만 동일한 시스템이 아닌 경우 오류가 발생하고 온보딩이 종단됩니다.
- 이 경우 관리자는 시스템 중 하나 이상을 제거하여 충돌을 해결해야 합니다.

3.2. SSH Push 연락 방법

SSH Push([literal] ssh-push)는 Salt 클라이언트가 Uyuni 서버에 직접 연결할 수 없는 환경에서 사용됩니다. 이 환경에서 클라이언트는 방화벽으로 보호되는 이름이 DMZ인 영역에 위치합니다. DMZ 내의 어떤 시스템도 Uyuni 서버가 있는 내부 네트워크에 대한 연결을 열 수 있는 권한이 없습니다.

SSH 푸시는 클라이언트에 데몬 에이전트를 설치할 수 없는 경우에도 사용할 수 있습니다.



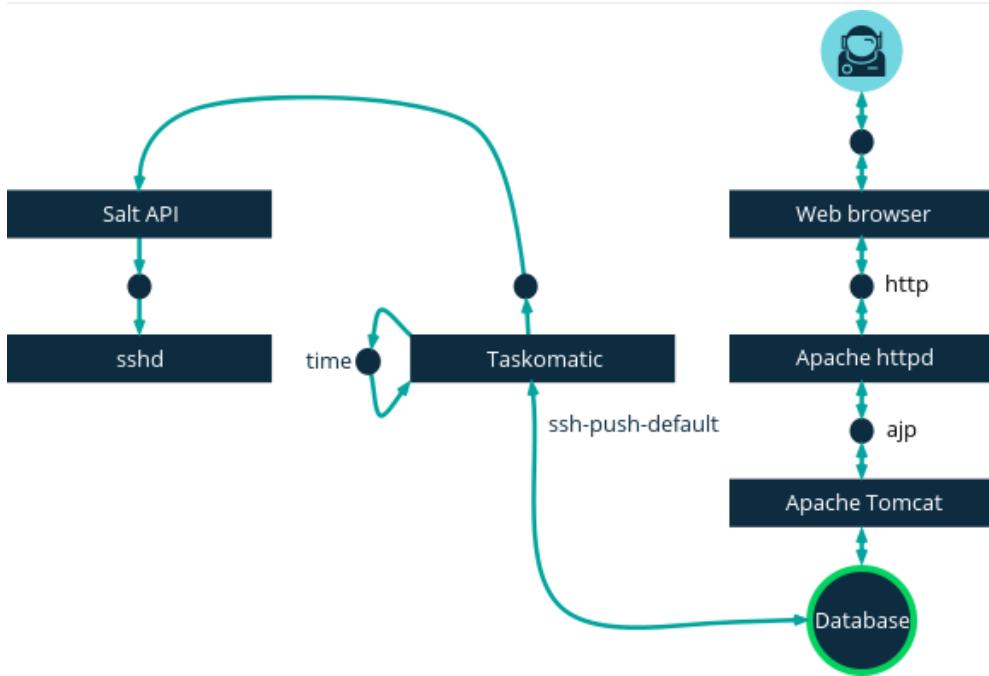
SSH 푸시 방법에는 심각한 제한이 있습니다. 크기 조정이 원활하지 않으며, 일반 Salt 방법([literal] --)보다 더 많은 서버 리소스와 네트워크 대역폭을 소비합니다. 푸시 SSH 방식은 대규모 설정(1000명 이상의 클라이언트)의 경우에는 전혀 지원되지 않습니다.

서버는 SSH Push를 사용하여 정기적으로 클라이언트에 연락해 체크인하고 예약된 작업과 이벤트를 수행합니다.



프로비저닝 모델을 사용하여 시스템을 다시 설치하는 것은 현재 SSH Push로 관리되는 클라이언트에서는 지원되지 않습니다.

이 이미지는 SSH Push 프로세스 경로를 보여줍니다. Taskomatic 블록의 왼쪽에 있는 모든 항목은 Uyuni 클라이언트에서 실행 중인 프로세스를 나타냅니다.



SSH Push를 사용하려면 클라이언트에서 실행 중인 SSH 데몬이 있어야 하며, Uyuni 서버에서 실행 중인 salt-api 데몬에 연결할 수 있어야 합니다. 또한 요구사항에 해당하는 Python 버전이 클라이언트 시스템에 Salt 번들과 함께 배포됩니다.

다음 등록 절차를 시작하기 전에 SSH 푸시 연락 방법을 구성하여 활성화 키를 정의합니다. 이 방법에서는 서버에 HTTPS로 직접 연결해야 합니다.

이러한 클라이언트를 Uyuni 서버에 등록하려면 Web UI 또는 API를 사용해야 합니다. 다음 절차 또는 예제를 참조하십시오.

절차: SSH Push를 사용하여 클라이언트 등록

1. Uyuni Web UI에서 **시스템 > 부트스트랩**으로 이동하여 해당 필드를 완성합니다.
2. SSH Push 연락 방법이 구성된 활성화 키를 선택합니다. 활성화 키에 대한 자세한 내용은 **Client-configuration > Activation-keys**에서 확인할 수 있습니다.
3. SSH 푸시 활성화 키 확인란을 선택합니다.
4. **[부트스트랩]**을 클릭하여 등록을 시작합니다.
5. **시스템 > 개요**로 이동하여 시스템이 올바르게 등록되었는지 확인합니다.

예: SSH를 통한 푸시에 대한 API 액세스

API를 사용해 어떤 연결 방법을 사용할지 관리할 수 있습니다. 다음 Python 코드 예제에서는 연결 방법을 ssh-push로 설정합니다.

유용한 값은 다음과 같습니다.

- default(풀)
- ssh-push
- ssh-push-tunnel

```
client = xmlrpclib.Server(SUMA_HOST + '/rpc/api', verbose=0)
key = client.auth.login(SUMA_LOGIN, SUMA_PASSWORD)
client.system.setDetails(key, 1000012345, {'contact_method' : 'ssh-push'})
```

3.2.1. 사용 가능한 파라미터

SSH Push를 구성할 때 호스트, 활성화 키, 비밀번호 등 시스템의 등록 시에 사용되는 파라미터를 수정할 수 있습니다. 비밀번호는 부트스트랩에만 사용되며 어디에도 저장되지 않습니다. 향후 모든 SSH 세션은 키/인증서 쌍을 통해 인증됩니다. 이러한 파라미터는 **시스템 > 부트스트래핑**에서 구성할 수 있습니다.

또한 시스템 전체에 사용되는 영구 파라미터를 구성할 수 있으며, sudo 사용자가 루트 대신 권한 없는 사용자로 시스템에 액세스하도록 설정할 수도 있습니다.

절차: 권한 없는 SSH 액세스 구성

1. Uyuni 서버에 최신 spacewalk-taskomatic 및 spacewalk-certs-tools 패키지가 설치되어 있는지 확인합니다.
2. 각 클라이언트 시스템에서 적절한 권한 없는 사용자를 생성합니다.
3. 각 클라이언트에서 다음과 같이 sudoers 파일을 편집합니다.

```
sudo visudo
```

4. 이 줄을 sudoers 파일 끝에 추가하여 사용자에게 sudo 액세스 권한을 부여합니다. 다음과 같이 <user>를 Web UI에서 클라이언트를 부트스트래핑하고 있는 사용자의 이름으로 교체합니다.

```
<user> ALL=NOPASSWD: /usr/bin/python3, /var/tmp/venv-salt-minion/bin/python
```



이 절차는 클라이언트 등록에 필요한 비밀번호가 없어도 루트에 액세스 권한을 부여합니다. 클라이언트는 성공적으로 설치된 후 루트 권한으로 실행되므로 액세스 권한이 더 이상 필요 없습니다. 클라이언트가 성공적으로 설치된 후 sudoers 파일에서 이 줄을 제거하는 것이 좋습니다.

5. Uyuni 서버의 /etc/rhn/rhn.conf 구성 파일에서 다음과 같은 줄을 추가하거나 수정하여 권한이 없는 사용자 이름을 포함합니다.

```
ssh_push_sudo_user = <user>
```

이 구성 파라미터를 변경한 후에는 salt-secrets-config.service, tomcat.service, taskomatic.service와 같은 서비스를 다시 시작해야 합니다. 필요한 모든 서비스를 적용하려면 Uyuni 서버를 루트로 다시 시작하는 것이 가장 좋은 방법입니다.

```
mgradm restart
```

3.2.2. 작업 실행

SSH Push 기능은 taskomatic을 사용하여 salt-ssh를 수행해 예약된 작업을 실행합니다. taskomatic 작업은 주기적으로 예약된 작업을 확인하고 실행합니다. SSH Push 기능은 예약된 작업을 기반으로 완전한 salt-ssh 호출을 실행합니다.

기본적으로 한 번에 20건의 Salt SSH 작업을 실행할 수 있습니다. 구성 파일에 다음과 같은 줄을 추가하고 parallel_threads의 값을 상향 조정하여 병렬로 실행할 수 있는 작업의 수를 늘릴 수 있습니다. 문제를 방지하기 위해 병렬 작업의 수를 다음과 같이 낮게 유지하는 것이 좋습니다.

```
taskomatic.sshminion_action_executor.parallel_threads = <number>
org.quartz.threadPool.threadCount = <value of parallel_threads + 20>
```

이렇게 하여 어느 한 곳의 클라이언트에서 병렬로 실행할 수 있는 작업의 수와 Taskomatic이 사용하는 작업자 스레드의 총 개수를 조정할 수 있습니다. 작업을 여러 클라이언트에서 실행해야 하는 경우 작업은 항상 각 클라이언트에서 순차적으로 실행됩니다.

클라이언트가 프록시를 통해 연결되어 있는 경우 프록시에서 MaxSessions 설정을 조정해야 합니다. 이 경우 병렬 연결의 수를 클라이언트 총 수의 세 배로 설정합니다.

3.2.3. 향후 출시될 기능

SSH Push를 통한 퓨시에서는 아직 지원하지 않는 몇 가지 기능이 있습니다. 다음 기능은 Salt SSH 클라이언트에서 작동하지 않습니다.

- OpenSCAP 감사
- Beacons, 이로 인해
 - zypper를 사용해 시스템에 패키지를 설치해도 패키지 새로 고침이 호출되지 않습니다.
 - 가상 호스트 기능(예: 게스트에 대한 호스트)은 가상 호스트 시스템이 Salt SSH 기반인 경우 작동하지 않습니다.

3.2.3.1. 자세한 내용은

- 일반적인 Salt SSH와 관련하여 **Specialized-guides** > **Salt** 및 <https://docs.saltstack.com/en/latest/topics/ssh>에서 확인할 수 있습니다.
- SSH 키 순환에 대한 내용은 [specialized-guides:salt/salt-ssh.pdf](#)에서 확인할 수 있습니다.

3.3. SSH Push(터널 사용) 연락 방법

SSH Push(터널 사용)(ssh-push-tunnel)는 클라이언트가 Uyuni 서버에 직접 연결할 수 없는 환경에서 사용됩니다. 이 환경에서는 클라이언트가 방화벽으로 보호되는 영역인 DMZ에 위치합니다. DMZ 내의 어떤 시스템도 Uyuni 서버를 포함하여 내부 네트워크에 대한 연결을 열 수 있는 권한이 없습니다.

이 SSH 방법은 내부 네트워크의 Uyuni 서버에서 DMZ에 위치한 클라이언트로 암호화된 터널을 생성합니다. 모든 작업과 이벤트가 실행된 후 터널이 종료됩니다.

서버는 SSH를 사용해 정기적으로 클라이언트에 연결하여 체크인하고 예약된 작업과 이벤트를 수행합니다.



- 프로비저닝 모델을 사용하여 시스템을 다시 설치하는 것은 현재 SSH Push로 관리되는 클라이언트에서는 지원되지 않습니다.



- 터널은 암호화된 터널을 통해 서버로의 액세스를 제공하기 위해 사용됩니다. SSH 푸시 클라이언트(터널 사용)에 할당된 리포지토리는 이 터널을 통해서만 제공되므로 터널이 작동하는 동안에만 리포지토리를 사용할 수 있으므로 클라이언트 시스템에서 직접 패키지 관리자 도구를 사용할 수 없습니다. 즉, 서버에서 세션을 시작한 경우에만 액세스가 가능합니다. 클라이언트의 모든 패키지 관리 작업은 서버 쪽에서만 수행할 수 있습니다.

SSH를 통한 터널링 연결의 경우, HTTPS를 통한 터널링에는 포트 번호가 필요합니다. 기본적으로 사용되는 포트 번호는 `1233`입니다. 이를 덮어쓰려면 /etc/rhn/rhn.conf에 1024보다 큰 사용자 정의 포트 번호를 추가하면 됩니다.

```
ssh_push_port_https = high_port
```

이 구성 파라미터를 변경한 후에는 salt-secrets-config.service, tomcat.service, taskomatic.service와 같은 서비스를 다시 시작해야 합니다. 필요한 모든 서비스를 포함하려면 spacewalk-service를 루트로 다시 시작하는 것이 가장 좋은 방법입니다.

```
spacewalk-service restart
```

보안상의 이유로 sudo를 SSH와 함께 사용해 루트 권한 대신에 권한이 없는 사용자로 시스템에 액세스하고 싶을 때가 있을 수 있습니다.

절차: 권한이 없는 SSH 액세스 구성

1. 각 클라이언트 시스템에서 적절한 권한 없는 사용자를 생성합니다.
2. 각 클라이언트에서 다음과 같이 sudoers 파일을 편집합니다.

```
sudo visudo
```

3. 이 줄을 sudoers 파일 끝에 추가하여 사용자에게 sudo 액세스 권한을 부여합니다. 다음과 같이 <user>를 Web UI에서 클라이언트를 부트스트래핑하고 있는 사용자의 이름으로 교체합니다.

```
<user> ALL=NOPASSWD: /usr/bin/python3, /var/tmp/venv-salt-minion/bin/python
```



이 절차는 클라이언트 등록에 필요한 비밀번호가 없어도 루트에 액세스 권한을 부여합니다. 클라이언트는 성공적으로 설치된 후 루트 권한으로 실행되므로 액세스 권한이 더 이상 필요 없습니다. 클라이언트가 성공적으로 설치된 후 sudoers 파일에서 이 줄을 제거하는 것이 좋습니다.

4. Uyuni 서버의 /etc/rhn/rhn.conf 구성 파일에서 다음과 같은 줄을 추가하거나 수정하여 권한이 없는 사용자 이름을 포함합니다.

```
ssh_push_sudo_user = <user>
```

이 구성 парамет를 변경한 후에는 salt-secrets-config.service, tomcat.service, taskomatic.service와 같은 서비스를 다시 시작해야 합니다. 필요한 모든 서비스를 포함하려면 spacewalk-service를 루트로 다시 시작하는 것이 가장 좋은 방법입니다.

```
spacewalk-service restart
```

이러한 클라이언트를 Uyuni 서버에 등록하려면 Web UI 또는 API를 사용해야 합니다.

시작하기 전에 SSH 터널링에 사용할 포트를 지정했는지 확인해야 합니다. 포트 번호를 변경하기 전에 클라이언트를 등록한 경우, 다시 활성화 키를 사용하여 다시 등록해야 합니다.

- 부트스트래핑에 대한 자세한 내용은 **Client-configuration** > **Registration-bootstrap**에서 확인할 수 있습니다.
- 부트스트래핑에 대한 자세한 내용은 [client-configuration:activation-keys.pdf](#)에서 확인할 수 있습니다.

예: SSH Push에 대한 API 액세스(터널 사용)

API를 사용하여 어떤 연락 방법을 사용할지 관리할 수 있습니다. 이 예제 Python 코드는 연락 방법을 ssh-push-tunnel로 설정합니다.

유효한 값은 다음과 같습니다.

- default(풀)
- ssh-push
- ssh-push-tunnel

```
client = xmlrpclib.Server(SUMA_HOST + "/rpc/api", verbose=0)
key = client.auth.login(SUMA_LOGIN, SUMA_PASSWORD)
client.system.setDetails(key, 1000012345, {'contact_method' : 'ssh-push-tunnel'})
```

3.4. Salt Bundle

3.4.1. Salt Bundle이란 무엇입니까?

Salt 번들은 Salt Minion, Python 3, 필수 Python 모듈 및 라이브러리가 포함된 단일 바이너리 패키지입니다.

Salt 번들은 Python 3과 Salt가 실행되기 위한 모든 요구사항과 함께 제공됩니다. 따라서 Salt 번들은 클라이언트에 설치된 Python 버전을 시스템 소프트웨어로 사용하지 않습니다. 해당 Salt 버전에 대한 요구사항을 충족하지 않는 클라이언트에도 Salt 번들을 설치할 수 있습니다.

Uyuni Salt Master가 아닌 Salt Master에 연결된 Salt Minion을 실행하는 시스템에서도 Salt 번들을 사용할 수 있습니다.

3.4.2. Salt Bundle을 Minion으로 클라이언트 등록

Salt 번들을 통한 등록 방법이 권장되는 등록 방법입니다. 이 섹션에서는 현재 구현의 장점과 한계에 대해 설명합니다. Salt 번들은 Salt, Python 3, Salt가 사용하는 Python 모듈로 구성된 `venv-salt-minion`으로 제공됩니다. Web UI를 통한 부트스트랩도 Salt 번들을 사용하므로, Web UI를 사용한 부트스트랩은 Python 종속적이지 않습니다. Salt 번들을 사용하면 클라이언트가 더 이상 Python 인터프리터나 모듈을 제공할 필요가 없습니다.

새 클라이언트를 부트스트랩하는 경우 Salt Bundle에 등록하는 것이 기본 방법입니다. 기존 클라이언트를 Salt Bundle 방식으로 전환할 수 있습니다. 전환하면 `salt-minion` 패키지와 종속 항목이 설치된 상태로 유지됩니다.

3.4.2.1. Salt Minion과 함께 Salt 번들 사용

Salt 번들은 Salt와 함께 사용할 수 있습니다. 미니언이 관리하는 Salt Master 이외의 Uyuni 서버에서 동시에 사용할 수 있습니다. Salt 번들을 클라이언트에 설치한 경우 Uyuni 서버가 Salt 번들의 구성 파일을 관리하게 되며, 이 경우 `salt-minion`의 구성 파일은 관리되지 않습니다. 자세한 내용은 [Salt 번들 구성](#)에서 확인할 수 있습니다.



- Uyuni 서버가 아닌 Salt Master가 관리하는 Salt 미니언을 사용하여 클라이언트를 부트스트랩하려면 클라이언트를 Salt로 부트스트랩하려면 부트스트랩 스크립트 생성 시 `mgr-bootstrap --force-bundle`을 사용하거나 부트스트랩 스크립트에서 `FORCE_VENV_SALT_MINION`을 `1`로 설정하는 것이 좋습니다.
- Web UI `mgr_force_venv_salt_minion`이 `true`로 설정된 부트스트래핑의 경우 열을 전역적으로 지정할 수 있습니다. 자세한 내용은 [Specialized-guides > Salt](#)에서 확인할 수 있습니다.

3.4.2.2. Salt 미니언에서 Salt 번들로 전환

Salt 상태 `util.mgr_switch_to_venv_minion`을 사용하여 `salt-minion`에서 `venv-salt-minion`으로 전환할 수 있습니다. 프로세스 이동 문제를 방지하려면 두 단계에 걸쳐 `venv-salt-minion`으로 전환하는 것이 좋습니다.

절차: `util.mgr_switch_to_venv_minion` 상태를 `venv-salt-minion`으로 전환

- 우선 열을 지정하지 않고 `util.mgr_switch_to_venv_minion`을 적용하십시오. 그러면 구성 파일 등을 복사하는 `venv-salt-minion`으로 전환됩니다. 원래 `salt-minion` 구성 및 해당 패키지는 정리되지 않습니다.

```
salt <minion_id> state.apply util.mgr_switch_to_venv_minion
```

`mgr_purge_non_venv_salt`가 `True`로 설정된 `util.mgr_switch_to_venv_minion`을 적용하여 `salt-minion`을 제거하고 `mgr_purge_non_venv_salt_files`을 `True`로 설정하여 `salt-minion`과 관련된 모든 파일을 제거하십시오. 이 두 번째 단계는 첫 번째 단계가 처리되었는지 확인한 후 이전 구성 파일과 더 이상 사용되지 않는 `salt-minion` 패키지를 제거합니다.

```
salt <minion_id> state.apply util.mgr_switch_to_venv_minion
pillar='{"mgr_purge_non_venv_salt_files": True,
"mgr_purge_non_venv_salt": True}'
```



첫 번째 단계를 건너뛰고 두 번째 전환 단계를 실행하는 경우 클라이언트 측에서 명령을 실행하기 위해 사용되는 `salt-minion`을 중지해야 하므로 상태 적용 프로세스가 실패할 수 있습니다.

반면에 Salt Bundle을 설치하지 않고 대신 `salt-minion`을 계속 사용하는 것도 가능합니다. 이 경우 다음 옵션 중 하나를 지정:

- `--no-bundle` 옵션으로 `mgr-bootstrap`을 실행하십시오.
- 생성된 부트스트랩 스크립트에서 `AVOID_VENV_SALT_MINION`을 1로 설정합니다.
- 부트스트랩 상태에 대하여 `mgr_avoid_venv_salt_minion` 열을 `True`로 설정합니다.

3.4.3. Salt 번들을 사용한 SSH Push

Salt 번들은 클라이언트로 SSH Push 작업을 수행할 때도 사용됩니다.

셀 스크립트는 Salt 명령이 실행되기 전에 `venv-salt-minion`을 설치하지 않고 대상 시스템에 Salt 번들을 배포합니다. Salt 번들에는 전체 Salt 코드 베이스가 포함되어 있으므로 `salt-thin`은 배포되지 않습니다. SSH Push(Web UI를 사용한 부트스트랩 포함)는 번들 내의 Python 3 인터프리터를 사용합니다. 대상 시스템에는 다른 Python 인터프리터가 설치되어 있지 않아도 됩니다.

번들과 함께 배포된 Python 3은 클라이언트에서 SSH Push 세션을 처리하는 데 사용되므로 SSH Push(Web UI를 사용한 부트스트랩 포함)는 시스템에 설치된 Python에 종속되지 않습니다.

다른 방법으로 `salt-thin`을 사용할 수 있지만, 클라이언트에 Python 3이 설치되어 있어야 합니다. 이 방법은 권장되거나 지원되지 않으며 개발 목적으로만 존재합니다. `web.ssh_use_salt_thin`을 `/etc/rhn/rhn.conf` 구성 파일에서 `true`로 설정합니다.

- 부트스트랩 리포지토리는 Web UI를 사용하여 클라이언트를 부트스트랩하기 전에 생성해야 합니다. SSH Push는 감지된 대상 운영 체제를 기반으로 부트스트랩 리포지토리에서 가져온 Salt 번들을 사용합니다. 자세한 내용은 [client-configuration:bootstrap-repository.pdf](#)에서 확인할 수 있습니다.
- SSH Push는 /var/tmp를 사용하여 번들된 Python을 사용하여 클라이언트에서 Salt 명령을 실행하기 위해 Salt 번들을 배포합니다. 따라서 noexec 옵션으로 /var/tmp를 마운트하지 않아야 합니다. noexec 옵션을 사용하여 /var/tmp가 마운트된 클라이언트는 부트스트랩 프로세스가 SSH Push를 사용하여 클라이언트에 도달하기 때문에 Web UI를 사용하여 부트스트랩할 수 없습니다.

3.4.4. pip를 사용하여 Python 패키지와 함께 Salt Bundle 확장

Salt 번들에는 [literal] pip이 포함되어 있어 번들된 Salt 미니언의 기능을 추가 Python 패키지로 확장할 수 있습니다.

기본적으로 salt <minion_id> pip.install <package-name> 은 <package_name>에 의해 지정된 Python 패키지를 /var/lib/venv-salt-minion/local에 설치합니다.

필요한 경우 /var/lib/venv-salt-minion/local 경로는 venv-salt-minion.service에 대한 VENV_PIP_TARGET 환경 변수를 설정하여 재정의할 수 있습니다. 서비스에 대한 systemd drop-in 구성 파일을 사용하는 것이 좋습니다. 이는 구성 파일 /etc/systemd/system/venv-salt-minion.service.d/10-pip-destination.conf를 사용하여 수행할 수 있습니다.

```
[Service]
Environment=VENV_PIP_TARGET=/new/path/local/venv-salt-
minion/pip
```

[literal] pip을 사용하여 설치된 Python 패키지는 Salt 번들을 업데이트할 때 변경되지 않습니다. 업데이트 후에도 해당 패키지를 사용할 수 있고 정상적으로 작동하도록 하려면 Salt 번들 업데이트 후 Salt 상태를 적용하여 설치하는 것이 좋습니다.

Chapter 4. 클라이언트 등록

클라이언트를 Uyuni 서버에 등록하는 방법이 몇 가지 있습니다. 이 섹션에서는 사용할 수 있는 다양한 방법에 대해 설명합니다. 또한 클라이언트에서 실행하려는 운영 체제에 대한 정보도 포함되어 있습니다.

시작하기 전에 다음 사항을 확인하십시오.

- 클라이언트는 등록 전에 날짜 및 시간을 Uyuni 서버와 정확하게 동기화한 상태이어야 합니다.
- 활성화 키 생성을 완료한 상태이어야 합니다. 활성화 키 생성에 대한 자세한 내용은 [Client-configuration > Activation-keys](#)를 참조하십시오.



Uyuni 서버 기본 OS를 Uyuni 자체에 등록하지 마십시오. Uyuni 서버 기본 OS는 개별적으로 또는 다른 Uyuni 서버를 사용해 관리해야 합니다.
Uyuni 서버 컨테이너를 관리하려면 [literal] `mgradm` 도구를 사용합니다.

4.1. 등록 방법

클라이언트를 Uyuni 서버에 등록하는 방법이 몇 가지 있습니다.

- 적은 수의 클라이언트만 등록하는 경우에는 Uyuni Web UI를 사용하여 클라이언트를 등록하는 것이 좋습니다. 자세한 내용은 [Client-configuration > Registration-webui](#)를 참조하십시오.
- 프로세스에 대한 더 많은 제어권을 가지고 싶거나 많은 수의 클라이언트를 등록해야 하는 경우 부트스트랩 스크립트를 생성하는 것이 좋습니다. 자세한 내용은 [Client-configuration > Registration-bootstrap](#)에서 참조하십시오.
- 클라이언트와 프로세스에 대한 더 많은 제어 권한을 위해서는 명령줄에서 단일 명령을 실행하면 도움이 될 수 있습니다. 자세한 내용은 [Client-configuration > Registration-cli](#)를 참조하십시오.

클라이언트는 등록 전에 날짜 및 시간을 Uyuni 서버와 정확하게 동기화한 상태이어야 합니다.

부트스트랩 스크립트 또는 명령줄 방법을 사용하려면 먼저 활성화 키를 생성해야 합니다. 활성화 키 생성에 대한 자세한 내용은 [Client-configuration > Activation-keys](#)를 참조하십시오.



Uyuni 서버에 서버 자신을 등록하지 마십시오. Uyuni 서버는 개별 관리하거나 별도의 다른 Uyuni 서버를 사용하여 관리해야 합니다. 여러 서버 사용에 대한 자세한 내용은 [Specialized-guides > Large-deployments](#)에서 확인할 수 있습니다.

4.1.1. Web UI로 클라이언트 등록

Web UI를 사용하여 부트스트랩하는 클라이언트는 [Specialized-guides > Salt](#)를 사용하여 클라이언트에서 부트스트랩 프로세스를 실행합니다. Salt SSH는 Salt 번들과 포함된 Python 인터프리터를 사용합니다. 따라서 클라이언트에 다른 Python 인터프리터를 설치할 필요가 없습니다.



Salt Bundle은 부트스트랩 리포지토리와 함께 제공되므로 리포지토리를 생성한 후 클라이언트의 부트스트랩 프로세스를 시작해야 합니다. 셀 스크립트가 클라이언트에서 운영 체제를 감지하고 부트스트랩 스크립트와 동일한 논리를 사용하여 적절한 부트스트랩 리포지토리로부터 Salt Bundle을 배포합니다. 자세한 내용은 [부트스트랩 리포지토리 생성 준비](#)를 참조하십시오.



Uyuni 서버에 서버 자신을 등록하지 마십시오. Uyuni 서버는 개별 관리하거나 별도의 다른 Uyuni 서버를 사용하여 관리해야 합니다. 여러 서버 사용에 대한 자세한 내용은 [Specialized-guides > Large-deployments](#)에서 확인할 수 있습니다.

절차: Web UI로 클라이언트 등록

1. Uyuni Web UI에서 **시스템 > 부트스트랩**으로 이동합니다.
2. **URL** 필드에 부트스트랩 할 클라이언트의 정규화된 도메인 이름(FQDN)을 입력합니다.
3. **SSH URL** 필드에서 클라이언트를 연결하고 부트스트랩하는 데 사용할 SSH 포트 번호를 입력합니다. 기본적으로 SSH 포트는 22입니다.
4. **Username** 필드에서 클라이언트에 로그인할 사용자 이름을 입력합니다. 기본적으로 사용자 이름은 `root`입니다.
5. SSH로 클라이언트를 부트스트래핑하려면 **Auth Type** 필드에서 `SSH`을 확인하고 클라이언트에 로그인할 때 사용할 SSH 개인 키를 업로드하십시오. SSH 개인 키에 암호문이 필요한 경우 **SSH Key Password** 필드에 입력하십시오. 또는 어떤 암호문에 대해서도 이 필드를 공백 상태로 두지 마십시오.
6. 비밀번호로 클라이언트를 부트스트래핑하려면 **Auth Type** 필드에서 `SSH`을 확인하고 비밀번호를 입력하여 클라이언트에 로그인하십시오.
7. **Activation Keys** 필드에서 클라이언트를 부트스트랩하는 데 사용할 소프트웨어 채널과 연결된 활성화 키를 선택합니다. 자세한 내용은 [Client-configuration > Activation-keys](#)에서 확인할 수 있습니다.
8. 옵션: **Lock Client** 필드에서 클라이언트를 등록할 프록시를 선택합니다.
9. 기본적으로 `Strict Host Key Checking` 확인란이 선택되어 있습니다. 따라서 사용자가 수동으로 인증할 필요 없이 부트스트랩 프로세스가 자동으로 SSH 호스트 키를 수락할 수 있습니다.
10. 옵션: `Forward X11` 확인란을 선택합니다. 이 옵션을 선택하면 서버에 연결하기 위해 SSH를 사용하도록 클라이언트가 구성되고 다른 연결 방법은 구성되지 않습니다.
11. **[부트스트랩]**을 클릭하여 등록을 시작합니다.

부트스트랩 프로세스가 완료되면 클라이언트가 **시스템 > 시스템 목록**에 나열됩니다.



SSH 개인 키는 부트스트래핑 프로세스가 진행되는 중에만 저장되며, 부트스트래핑이 완료되자마자 Uyuni 서버에서 삭제됩니다.



Uyuni를 사용해 클라이언트에 새 패키지 또는 업데이트를 설치하면 최종 사용자 라이선스 계약(EULA)이 자동으로 수락됩니다. 패키지 EULA를 검토하려면 Web UI에서 패키지 상세 정보 페이지를 여십시오.

4.1.1.1. 로컬 할당 리포지토리 처리

Uyuni에서 제공하지 않는 클라이언트에 리포지토리를 직접 할당하는 것은 일반적인 사용 사례가 아닙니다. 이로 인해

문제가 발생할 수 있습니다. 그러므로 Salt를 통한 부트스트래핑은 부트스트랩 프로세스를 시작할 때 모든 로컬 리포지토리를 비활성화합니다.

나중에 Highstate 또는 패키지 설치를 실행할 때처럼 채널 상태를 사용할 때마다 로컬에 할당된 모든 리포지토리가 다시 비활성화됩니다.

클라이언트에서 사용되는 모든 소프트웨어 패키지는 Uyuni에서 제공하는 채널에서 가져와야 합니다. 사용자 정의 채널을 생성하는 방법에 대한 자세한 내용은 **Administration > Custom-channels**에서 을 참조하십시오.

4.1.2. 부트스트랩 스크립트로 클라이언트 등록

부트스트랩 스크립트로 클라이언트를 등록하면 파라미터를 제어할 수 있고, 다수의 클라이언트를 한 번에 등록해야 하는 경우 도움이 됩니다.

부트스트랩 스크립트를 사용하여 클라이언트를 등록하려면 먼저 템플릿 부트스트랩 스크립트를 생성한 다음 복사하고 수정하는 것이 좋습니다. 생성한 부트스트랩 스크립트는 클라이언트가 등록될 때 클라이언트에서 실행되며, 필요한 모든 패키지가 클라이언트에 배포되도록 합니다. 부트스트랩 스크립트의 일부 파라미터는 활성화 키와 GPG 키를 사용하여 클라이언트 시스템을 기본 채널에 할당할 수 있도록 합니다.

리포지토리 정보를 주의 깊게 확인하여 기본 채널 리포지토리와 일치하게 하는 것이 중요합니다. 리포지토리 정보가 정확히 일치하지 않으면 부트스트랩 스크립트가 패키지를 정확히 다운로드할 수 없습니다.



모든 클라이언트에는 부트스트랩 리포지토리가 필요합니다. 부트스트랩 리포지토리는 제품이 동기화될 때 Uyuni 서버에 자동으로 생성 및 재생성됩니다. 부트스트랩 리포지토리에는 클라이언트에 Salt를 설치하고 클라이언트를 등록하기 위한 패키지가 포함됩니다.

부트스트랩 리포지토리 생성에 대한 자세한 내용은 **Client-configuration > Bootstrap-repository**에서 확인할 수 있습니다.



GPG 키 및 Uyuni 클라이언트 도구

Uyuni 클라이언트 도구가 사용하는 GPG 키는 기본적으로 신뢰할 수 없습니다. 부트스트랩 스크립트를 생성할 때 ORG_GPG_KEY 파라미터와 함께 공용 키 지문이 포함된 파일의 경로를 추가하십시오.



openSUSE Leap 15 및 SLES 15는 기본적으로 Python 3를 사용합니다. Python 2에 기반을 둔 부트스트랩 스크립트는 openSUSE Leap 15 및 SLE 15 시스템에 대해 다시 생성해야 합니다. Python 2를 사용해 openSUSE Leap 15 또는 SLE 15 시스템을 등록하는 경우 부트스트랩 스크립트는 실패합니다.

4.1.2.1. mgr-bootstrap 명령을 사용한 부트스트랩 스크립트 생성

`mgr-bootstrap` 명령은 사용자 정의 부트스트랩 스크립트를 생성합니다. 초기 등록 및 구성을 단순화하기 위해 Uyuni 클라이언트 시스템은 부트스트랩 스크립트를 사용합니다.

`--activation-keys` 및 `--script` 인수가 유일한 필수 인수입니다. Uyuni 서버에서 명령줄의 루트로 필수 인수를 사용하여 실행하십시오. 다음과 같이 `<ACTIVATION_KEYS>` 및 `<EDITED_NAME>`을 사용자의 값으로 바꿉니다.

```
mgr-bootstrap --activation-keys=<ACTIVATION_KEY> --script=bootstrap
-<EDITED_NAME>.sh
```

`mgr-bootstrap` 명령은 특정 호스트 이름을 설정하고 특정 GPG 키를 설정하며 등록 방법(salt-minion 또는 salt-bundle)을 설정하는 기능 등 몇 가지 다른 옵션을 제공합니다.

자세한 내용은 `mgr-bootstrap` 사용자 지정 페이지를 참조하거나 `mgr-bootstrap --help`를 실행하십시오.

4.1.2.2. Web UI에서 부트스트랩 스크립트 생성

Uyuni Web UI를 사용하면 편집 가능한 부트스트랩 스크립트를 생성할 수 있습니다.

절차: 부트스트랩 스크립트 생성

1. Uyuni Web UI에서 관리 > 관리자 구성 > **부트스트랩 스크립트**로 이동합니다.
2. 필수 필드는 이전의 설치 단계에서 얻은 값으로 미리 채워져 있습니다. 각 설정에 대한 자세한 내용은 **Reference > Admin**에서 확인할 수 있습니다.
3. **[업데이트]**를 클릭하여 스크립트를 생성합니다.
4. 부트스트랩 스크립트는 생성되어 서버의 `/srv/www/htdocs/pub/bootstrap` 디렉토리에 저장됩니다. 또는, HTTPS를 통해 부트스트랩 스크립트에 액세스할 수 있습니다. `<example.com>`을 다음과 같은 Uyuni 서버의 호스트 이름으로 바꿉니다.

```
https://<example.com>/pub/bootstrap/bootstrap.sh
```



- 부트스트랩 스크립트에서 SSL을 비활성화하지 마십시오. Web UI에서 SSL --- 확인란이 선택되어 있는지, 또는 `USING_SSL=1`이라는 설정이 부트스트랩 스크립트에 있는지 확인합니다. SSL을 비활성화하면 등록 프로세스에 사용자 정의 SSL 인증서가 필요합니다.
- 사용자 정의 채널에 대한 자세한 내용은 **Administration > Ssl-certs**에서 확인할 수 있습니다.

4.1.2.3. 부트스트랩 스크립트 편집

생성한 템플릿 부트스트랩 스크립트를 복사 및 수정하여 사용자 정의할 수 있습니다. Uyuni에서 사용할 부트스트랩 스크립트를 수정할 때 충족해야 할 최소 요구사항은 활성화 키를 포함해야 한다는 것입니다. 대부분의 패키지는 GPG로 서명되므로 패키지를 설치할 시스템에 신뢰할 수 있는 GPG 키도 있어야 합니다.

이 절차에서 활성화 키의 정확한 이름을 알아야 합니다. 홈 > **개요**로 이동하여 -- 상자에서 --- ↗ --를 클릭합니다. 채널에 대해 생성된 모든 키는 이 페이지에 나열되어 있습니다. 부트스트랩 스크립트에서 사용하려는 키의 전체 이름을 키 필드에 표시된 대로 정확히 입력해야 합니다. 활성화 키에 대한 자세한 내용은 **Client-configuration > Activation-keys**를 참조하십시오.

절차: 부트스트랩 스크립트 수정

1. Uyuni 서버에서 명령줄의 루트 권한으로 다음과 같이 부트스트랩 디렉토리로 변경합니다.

```
cd /srv/www/htdocs/pub/bootstrap/
```

- 각 클라이언트에서 사용할 템플릿 부트스트랩 스크립트 사본 2개를 생성하여 이름을 변경합니다.

```
cp bootstrap.sh bootstrap-sles12.sh
cp bootstrap.sh bootstrap-sles15.sh
```

- 수정을 위해 `bootstrap-sles15.sh`를 엽니다. 아래와 같은 텍스트가 나올 때까지 아래로 스크롤합니다. `exit 1`이 파일에 있는 경우 해시 또는 파운드 기호(#)를 줄 첫머리에 입력하여 코멘트 아웃합니다. 이렇게 하면 스크립트가 활성화됩니다. 이 스크립트의 키 이름을 `ACTIVATION_KEYS=` 필드에 다음과 같이 입력합니다.

```
echo "Enable this script: comment (with #'s) this block (or, at
least just"
echo "the exit below)"
echo
#exit 1

# 수정할 수 있으나 아마 정확할 것입니다(초기 설치 중에 생성하지 않은 경우):
# 참고: ACTIVATION_KEYS는 클라이언트 시스템을 부트스트래핑하는 데 *반드시*
사용해야 합니다.
ACTIVATION_KEYS=1-sles15
ORG_GPG_KEY=
```

- 완료한 후 파일을 저장하고 두 번째 부트스트랩 스크립트에 대해 이 절차를 반복하십시오.



기본적으로 부트스트랩 스크립트는 부트스트랩 리포지토리에 Salt 번들이 있으면 `venv-salt-minion`을 설치하고 부트스트랩 리포지토리에 `salt-minion`이 없으면 `salt-minion`을 설치하려고 시도합니다. 부득이하게 필요한 경우 Salt 번들을 설치하지 않고 `salt-minion`을 계속 사용하는 것이 가능합니다.

자세한 내용은 **Client-configuration > Contact-methods-saltbundle**에서 확인할 수 있습니다.

4.1.2.4. 부트스트랩 스크립트를 실행하는 클라이언트 등록

스크립트 생성을 완료했으면 이 스크립트를 사용해 클라이언트를 등록할 수 있습니다.

절차: 부트스트랩 스크립트 실행

- Uyuni 서버에서 루트로 로그인합니다. 명령 프롬프트에서 부트스트랩 디렉토리로 변경합니다.

```
cd /srv/www/htdocs/pub/bootstrap/
```

2. 다음 명령을 실행하여 클라이언트에서 부트스트랩 스크립트를 실행하고, EXAMPLE.COM을 클라이언트의 호스트 이름으로 교체합니다.

```
cat bootstrap-sles15.sh | ssh root@EXAMPLE.COM /bin/bash
```

3. 또는 클라이언트에서 다음 명령을 실행합니다.

```
curl -Sks https://server_hostname/pub/bootstrap/bootstrap-sles15.sh  
| /bin/bash
```



문제를 해결하려면 bash 명령을 사용하여 부트스트랩 스크립트를 실행해 확인하십시오.

이 스크립트는 앞서 생성한 리포지토리 디렉토리에 있는 필수 종속성을 다운로드합니다.

4. 스크립트 실행이 완료되면 클라이언트가 올바르게 등록되었는지 확인할 수 있습니다. productname} Web UI를 열고 시스템 > 개요로 이동하여 새 클라이언트가 나열되었는지 확인합니다. 클라이언트가 나열되지 않으면 Uyuni Web UI에서 Salt > 키로 이동하여 클라이언트 키가 허용되는지 확인합니다.



Uyuni를 사용해 클라이언트에 새 패키지 또는 업데이트를 설치하면 최종 사용자 라이선스 계약(EULA)이 자동으로 수락됩니다. 패키지 EULA를 검토하려면 Web UI에서 패키지 상세 정보 페이지를 여십시오.

4.1.3. 명령줄에서 클라이언트 등록

4.1.3.1. 수동 클라이언트 등록

대부분의 경우 Salt 클라이언트는 기본 부트스트랩 방법으로 정확히 등록됩니다. 하지만 Salt를 사용하면 클라이언트의 Salt Minion 파일을 편집하고 서버의 전체 도메인 이름(FQDN)을 제공하여 수동으로 Uyuni 서버에 클라이언트를 등록할 수 있습니다. 이 방법에서는 서버에 대한 4505 및 4506 인바운드 포트가 사용됩니다. 이 방법은 해당 포트가 열려 있는지 확인하는 것 외에 Uyuni 서버에서 구성할 필요가 없습니다.

이 절차를 수행하려면 Salt 클라이언트에 venv-salt-minion (Salt 번들) 또는 salt-minion 패키지를 설치한 후 등록해야 합니다. 둘 다 다른 위치에서 구성 파일을 사용하며 파일 이름은 동일하게 유지됩니다. systemd 서비스 파일 이름이 다릅니다.



이 방법으로 부트스트랩하면 클라이언트 도구 채널 또는 공식 SUSE 배포판의 일부인 salt-minion을 사용하는 경우에만 작동합니다.

4.1.3.2. Salt 번들 구성

Salt 번들(venv-salt-minion)

- /etc/venv-salt-minion/

- /etc/venv-salt-minion/minion
- /etc/venv-salt-minion/minion.d/NAME.conf
- systemd 서비스 파일: venv-salt-minion.service

Salt 번들에 대한 자세한 내용은 **Client-configuration > Contact-methods-saltbundle**에서 참조하십시오.

절차: Salt Bundle 구성 파일로 클라이언트 등록

1. Salt 클라이언트에서 minion 구성 파일을 엽니다. 구성 파일의 위치는 다음 중 하나입니다.

```
/etc/venv-salt-minion/minion
```

또는:

```
/etc/venv-salt-minion/minion.d/NAME.conf
```

2. 파일에서 Uyuni 서버 또는 프록시의 FQDN과 활성화 키(있는 경우)를 추가하거나 편집합니다. 또한 아래 나열된 다른 구성 파라미터를 추가합니다.

```
master: SERVER.EXAMPLE.COM
```

세분화:

```
susemanager:
    activation_key: "<Activation_Key_Name>"

server_id_use_crc: adler32
enable_legacy_startup_events: False
enable_fqdns_grains: False
```

3. 다음과 같이 venv-salt-minion 서비스를 재시작합니다.

```
systemctl restart venv-salt-minion
```

4. Uyuni 서버에서 새로운 클라이언트 키를 수락하고, 다음과 같이 <client>를 클라이언트의 이름으로 교체합니다.

```
salt-key -a '<client>'
```

4.1.3.3. 클라이언트 구성

클라이언트(salt-minion)

- /etc/salt/
- /etc/salt/minion
- /etc/salt/minion.d/NAME.conf
- systemd 서비스 파일: salt-minion.service

절차: Salt Minion 구성 파일로 클라이언트 등록

1. Salt 클라이언트에서 minion 구성 파일을 엽니다. 구성 파일의 위치는 다음 중 하나입니다.

```
/etc/salt/minion
```

또는:

```
/etc/salt/minion.d/NAME.conf
```

2. 파일에서 Uyuni 서버 또는 프록시의 FQDN과 활성화 키(있는 경우)를 추가하거나 편집합니다. 또한 아래에 나열된 다른 구성 파라미터도 추가합니다.

```
master: SERVER.EXAMPLE.COM

세분화:
susemanager:
    activation_key: "<Activation_Key_Name>"

server_id_use_crc: adler32
enable_legacy_startup_events: False
enable_fqdns_grains: False
```

3. 다음과 같이 salt-minion 서비스를 다시 시작합니다.

```
systemctl restart salt-minion
```

4. Uyuni 서버에서 새로운 클라이언트 키를 수락하고, 다음과 같이 <client>를 클라이언트의 이름으로 교체합니다.

```
salt-key -a '<client>'
```

Salt 미니언 구성 파일에 대한 자세한 내용은 <https://docs.saltstack.com/en/latest/ref/configuration/>

[minion.html](#)을 참조하십시오.

4.2. SUSE 클라이언트 등록

SUSE Linux Enterprise 클라이언트를 Uyuni 서버에 등록할 수 있습니다.

방법과 자세한 내용은 클라이언트의 운영 체제에 따라 다릅니다.

시작하기 전에 클라이언트의 날짜 및 시간이 Uyuni 서버와 정확히 동기화되었는지 확인하십시오.

또한 활성화 키 생성을 완료한 상태이어야 합니다. 활성화 키 생성에 대한 자세한 내용은 **Client-configuration > Activation-keys**를 참조하십시오.



Uyuni 서버를 자체에 등록하지 마십시오. Uyuni 서버는 개별적으로 또는 별도의 Uyuni 서버를 사용해 관리해야 합니다. 여러 서버를 사용하는 방법에 대한 자세한 내용은 **Specialized-guides > Large-deployments**에서 확인할 수 있습니다.

4.2.1. SUSE Linux Enterprise 클라이언트 등록

이 섹션에는 SUSE Linux Enterprise 운영 체제를 실행하는 클라이언트를 등록하는 방법에 대한 정보가 포함되어 있습니다.

다음을 포함하여 모든 SUSE Linux Enterprise 제품을 준비하려면 이 장의 지침을 사용하십시오.

- SUSE Linux Enterprise Server for SAP
- SUSE Linux Enterprise 데스크톱
- SUSE Linux Enterprise
- SUSE Linux Enterprise Real Time

이전 SUSE Linux Enterprise 버전 및 서비스 팩에서도 이러한 지침을 사용할 수 있습니다.

4.2.1.1. 소프트웨어 채널 추가



다음 섹션에서 설명은 종종 x86_64 아키텍처로 기본 설정됩니다. 해당될 경우 다른 아키텍처로 교체하십시오.

SUSE Linux Enterprise 클라이언트를 Uyuni 서버에 등록하기 전에 필요한 소프트웨어 채널을 추가하고 이 채널을 동기화해야 합니다.

이 절차에 필요한 제품은 다음과 같습니다.

표 19. SLE 제품 - WebUI

OS 버전	제품 이름
SUSE Linux Enterprise Server 15 SP6	SUSE Linux Enterprise Server 15 SP6 x86_64
SUSE Linux Enterprise Server 15 SP5	SUSE Linux Enterprise Server 15 SP5 x86_64

OS 버전	제품 이름
SUSE Linux Enterprise Server 15 SP4	SUSE Linux Enterprise Server 15 SP4 x86_64
SUSE Linux Enterprise Server 15 SP3	SUSE Linux Enterprise Server 15 SP3 x86_64
SUSE Linux Enterprise Server 15 SP2	SUSE Linux Enterprise Server 15 SP2 x86_64
SUSE Linux Enterprise Server 15 SP1	SUSE Linux Enterprise Server 15 SP1 x86_64
SUSE Linux Enterprise Server 12 SP5	SUSE Linux Enterprise Server 12 SP5 x86_64

절차: 소프트웨어 채널 추가

1. Uyuni Web UI에서 **관리 > 설치 마법사 > 제품**으로 이동합니다.
2. 검색 창을 사용하여 클라이언트 운영 체제 및 아키텍처에 적합한 제품을 찾고 해당 제품을 확인하십시오. 모든 필수 채널은 자동으로 확인됩니다. 또한, ⇧ 키를 누르면 모든 추천 채널이 확인됩니다. 관련 제품의 전체 목록을 보려면 화살표를 클릭하고 필요한 추가 제품이 선택되어 있는지 확인하십시오.
3. **[제품 추가]**를 클릭하고 제품이 동기화를 마칠 때까지 기다립니다.

또는 명령 프롬프트에서 채널을 추가할 수 있습니다. 이 절차에 필요한 채널은 다음과 같습니다.

표 20. SLE 제품 - CLI

OS 버전	기본 채널
SUSE Linux Enterprise Server 15 SP6	sle-product-sles15-sp6-pool-x86_64
SUSE Linux Enterprise Server 15 SP5	sle-product-sles15-sp5-pool-x86_64
SUSE Linux Enterprise Server 15 SP4	sle-product-sles15-sp4-pool-x86_64
SUSE Linux Enterprise Server 15 SP3	sle-product-sles15-sp3-pool-x86_64
SUSE Linux Enterprise Server 15 SP2	sle-product-sles15-sp2-pool-x86_64
SUSE Linux Enterprise Server 15 SP1	sle-product-sles15-sp1-pool-x86_64
SUSE Linux Enterprise Server 12 SP5	sle-product-sles15-sp5-pool-x86_64

이전 제품의 채널 이름을 찾으려면 Uyuni 서버의 명령 프롬프트에서 루트 권한으로 `mgr-sync` 명령을 사용합니다.

```
mgr-sync list --help
```

그런 다음 원하는 인수를 지정합니다. 예를 들어, `channels`:

```
mgr-sync list channels [-c]
```

명령 프롬프트에서 소프트웨어 채널 추가

1. Uyuni 서버의 명령 프롬프트에서 root 권한으로 다음과 같이 `mgr-sync` 명령을 사용해 적절한 채널을 추가합니다.

```
mgr-sync add channel <channel_label_1>
mgr-sync add channel <channel_label_2>
mgr-sync add channel <channel_label_n>
```

2. 동기화가 자동으로 시작됩니다. 채널을 수동으로 동기화하려면 다음 명령을 사용하십시오.

```
mgr-sync sync --with-children <channel_name>
```

3. 계속하기 전에 동기화가 완료되었는지 확인합니다.

클라이언트 도구를 추가하려면 명령 프롬프트에서 다음 채널을 추가합니다.

표 21. SUSE Linux Enterprise 채널 - CLI

OS 버전	클라이언트 채널
SUSE Linux Enterprise Server 15 SP6	sles15-sp6-uyuni-client
SUSE Linux Enterprise Server 15 SP5	sles15-sp5-uyuni-client
SUSE Linux Enterprise Server 15 SP4	sles15-sp4-uyuni-client
SUSE Linux Enterprise Server 15 SP3	sles15-sp3-uyuni-client
SUSE Linux Enterprise Server 15 SP2	sles15-sp2-uyuni-client
SUSE Linux Enterprise Server 15 SP1	sles15-sp1-uyuni-client
SUSE Linux Enterprise Server 12 SP5	sles12-sp5-uyuni-client

명령 프롬프트에서 소프트웨어 채널 추가

1. Uyuni 서버의 명령 프롬프트에서 루트로 spacewalk-common-channels 명령을 사용하여 적절한 채널을 추가합니다.

```
spacewalk-common-channels \
<base_channel_label> \
<child_channel_label_1> \
<child_channel_label_2> \
... <child_channel_label_n>
```

2. **automatic synchronization**이 꺼진 경우 다음 채널을 동기화하십시오.

```
spacewalk-repo-sync -p <base_channel_label>
```

3. 계속하기 전에 동기화가 완료되었는지 확인합니다.

4.2.1.2. 동기화 상태 확인

절차: Web UI에서 동기화 진행 상태 확인

1. Uyuni Web UI에서 **소프트웨어** > **관리** > **채널**로 이동한 다음, 리포지토리에 연결된 채널을 클릭합니다.
 2. **-----** 템으로 이동한 다음, **---**를 클릭하고 **---** **--**를 확인합니다.

절차: 명령 프롬프트에서 동기화 진행 상태 확인

1. Uyuni 서버의 명령 프롬프트에서 루트 권한으로 다음과 같이 `tail` 명령을 사용해 동기화 로그 파일을 확인합니다.

```
tail -f /var/log/rhn/reposync/<channel-label>.log
```

2. 각 하위 채널은 동기화가 진행되는 동안 자체 로그를 생성합니다. 동기화가 완료되었는지 알아보려면 모든 기본 및 하위 채널 로그 파일을 확인해야 합니다.



SUSE Linux Enterprise 채널은 규모가 매우 클 수 있습니다. 때로 동기화에 몇 시간이 걸릴 수 있습니다.

4.2.1.3. GPG 키 관리

클라이언트는 소프트웨어 패키지가 설치되기 전에 GPG 키를 사용해 신뢰성을 확인합니다. 신뢰할 수 있는 소프트웨어만 클라이언트에 설치할 수 있습니다.



클라이언트의 보안을 위해 GPG 키를 신뢰하는 것이 중요합니다. 어떤 키가 필요하고 신뢰할 수 있는지 결정하는 것은 관리자가 수행해야 하는 작업입니다. GPG 키를 신뢰할 수 없는 경우 소프트웨어 채널을 사용할 수 없기 때문에 채널을 클라이언트에 할당하는 결정은 키를 신뢰하는 결정에 따라 달라집니다.

GPG 키에 대한 자세한 내용은 [Client-configuration > Gpg-keys](#)에서 확인할 수 있습니다.



SUSE Linux Enterprise Server 15 및 SUSE Linux Enterprise Server 12 클라이언트에
동일한 GPG 키를 사용합니다. 올바른 키는 sle12-gpg-pubkey-
39db7c82.key입니다.

4.2.1.4. 클라이언트 등록

클라이언트를 등록하려면 부트스트랩 리포지토리가 필요합니다. 기본적으로 부트스트랩 리포지토리는 자동으로 생성되며 동기화된 모든 제품에 대해 매일 재생성됩니다. 명령 프롬프트에서 다음 명령을 사용해 부트스트랩 리포지토리를 수동으로 생성할 수 있습니다.

```
mr-create-bootstrap-repo
```

클라이언트 등록에 대한 자세한 내용은 [Client-configuration > Registration-overview](#)에서 참조하십시오.

4.2.2. Registering SLE Micro Clients

This section contains information about registering clients running SLE Micro operating systems 5.1, 5.2, 5.3, 5.4, and 5.5 on x86-64, arm64, and IBM Z (s390x) architectures.

The SLE Micro is an ultra-reliable, lightweight operating system purpose built for edge computing. It leverages the enterprise hardened security and compliance components of SUSE Linux Enterprise and merges them with a modern, immutable, developer-friendly OS platform.

The SLE Micro uses transactional updates.

Transactional updates are atomic (all updates are applied only if all updates succeed) and support rollbacks. They do not affect a running system as no changes are activated until after the system is rebooted. This information is displayed in the **Systems > Details > Overview** subtab.

트랜잭션 업데이트 및 재부팅에 대한 자세한 내용은 <https://documentation.suse.com/sles/html/SLES-all/cha-transactional-updates.html>에서 확인할 수 있습니다.

4.2.2.1. 소프트웨어 채널 추가

Before you register SLE Micro clients to your Uyuni Server, you need to add the required software channels, and synchronize them.



다음 섹션에서 설명은 종종 x86_64 아키텍처로 기본 설정됩니다. 해당될 경우 다른 아키텍처로 교체하십시오.

이 절차에 필요한 제품은 다음과 같습니다.

표 22. SLE Micro 제품 - WebUI

OS Version	Product Name
SLE Micro 5.5 x86-64	SUSE Linux Enterprise Micro 5.5 x86_64
SLE Micro 5.5 arm64	SUSE Linux Enterprise Micro 5.5 aarch64
SLE Micro 5.5 s390x	SUSE Linux Enterprise Micro 5.5 s390x
SLE Micro 5.4 x86-64	SUSE Linux Enterprise Micro 5.4 x86_64
SLE Micro 5.4 arm64	SUSE Linux Enterprise Micro 5.4 aarch64
SLE Micro 5.4 s390x	SUSE Linux Enterprise Micro 5.4 s390x
SLE Micro 5.3 x86-64	SUSE Linux Enterprise Micro 5.3 x86_64
SLE Micro 5.3 arm64	SUSE Linux Enterprise Micro 5.3 aarch64
SLE Micro 5.3 s390x	SUSE Linux Enterprise Micro 5.3 s390x
SLE Micro 5.2 x86-64	SUSE Linux Enterprise Micro 5.2 x86_64
SLE Micro 5.2 arm64	SUSE Linux Enterprise Micro 5.2 aarch64
SLE Micro 5.2 s390x	SUSE Linux Enterprise Micro 5.2 s390x
SLE Micro 5.1 x86-64	SUSE Linux Enterprise Micro 5.1 x86_64

OS Version	Product Name
SLE Micro 5.1 arm64	SUSE Linux Enterprise Micro 5.1 aarch64
SLE Micro 5.1 s390x	SUSE Linux Enterprise Micro 5.1 s390x

절차: 소프트웨어 채널 추가

1. Uyuni Web UI에서 관리 > 설치 마법사 > 제품으로 이동합니다.
2. 검색 창을 사용하여 클라이언트 운영 체제 및 아키텍처에 적합한 제품을 찾고 해당 제품을 확인하십시오. 모든 필수 채널은 자동으로 확인됩니다. 또한, ⇧ 키 토큰이 켜져 있으면 모든 추천 채널이 확인됩니다. 관련 제품의 전체 목록을 보려면 화살표를 클릭하고 필요한 추가 제품이 선택되어 있는지 확인하십시오.
3. [제품 추가]를 클릭하고 제품이 동기화를 마칠 때까지 기다립니다.

또는 명령 프롬프트에서 채널을 추가할 수 있습니다. 이 절차에 필요한 채널은 다음과 같습니다.

표 23. SLE Micro 제품 - CLI

OS Version	Base Channel	Updates Channel
SLE Micro 5.5 x86-64	sle-micro-5.5-pool-x86_64	sle-micro-5.5-updates-x86_64
SLE Micro 5.5 arm64	sle-micro-5.5-pool-arm64	sle-micro-5.5-updates-arm64
SLE Micro 5.5 IBM Z (s390x)	sle-micro-5.5-pool-s390x	sle-micro-5.5-updates-s390x
SLE Micro 5.4 x86-64	sle-micro-5.4-pool-x86_64	sle-micro-5.4-updates-x86_64
SLE Micro 5.4 arm64	sle-micro-5.4-pool-arm64	sle-micro-5.4-updates-arm64
SLE Micro 5.4 IBM Z (s390x)	sle-micro-5.4-pool-s390x	sle-micro-5.4-updates-s390x
SLE Micro 5.3 x86-64	sle-micro-5.3-pool-x86_64	sle-micro-5.3-updates-x86_64
SLE Micro 5.3 arm64	sle-micro-5.3-pool-arm64	sle-micro-5.3-updates-arm64
SLE Micro 5.3 IBM Z (s390x)	sle-micro-5.3-pool-s390x	sle-micro-5.3-updates-s390x
SLE Micro 5.2 x86-64	suse-microos-5.2-pool-x86_64	suse-microos-5.2-updates-x86_64
SLE Micro 5.2 arm64	suse-microos-5.2-pool-aarch64	suse-microos-5.2-updates-aarch64
SLE Micro 5.2 IBM Z (s390x)	suse-microos-5.2-pool-s390x	suse-microos-5.2-updates-s390x
SLE Micro 5.1 x86-64	suse-microos-5.1-pool-x86_64	suse-microos-5.1-updates-x86_64
SLE Micro 5.1 arm64	suse-microos-5.1-pool-aarch64	suse-microos-5.1-updates-aarch64
SLE Micro 5.1 IBM Z (s390x)	suse-microos-5.1-pool-s390x	suse-microos-5.1-updates-s390x

명령 프롬프트에서 소프트웨어 채널 추가

1. Uyuni 서버의 명령 프롬프트에서 root 권한으로 다음과 같이 `mgr-sync` 명령을 사용해 적절한 채널을 추가합니다.

```
mgr-sync add channel <channel_label_1>
mgr-sync add channel <channel_label_2>
mgr-sync add channel <channel_label_n>
```

2. 동기화가 자동으로 시작됩니다. 채널을 수동으로 동기화하려면 다음 명령을 사용하십시오.

```
mgr-sync sync --with-children <channel_name>
```

3. 계속하기 전에 동기화가 완료되었는지 확인합니다.

클라이언트 도구를 추가하려면 명령 프롬프트에서 다음 채널을 추가합니다.

표 24. SLE Micro 채널 - CLI

OS Version	Client Channel
SLE Micro 5.4	sle-micro-5.4-uyuni-client
SLE Micro 5.3	suse-micro-5.3-uyuni-client
SLE Micro 5.2	suse-microos-5.2-uyuni-client
SLE Micro 5.1	sle-microos-5.1-uyuni-client

명령 프롬프트에서 소프트웨어 채널 추가

1. Uyuni 서버의 명령 프롬프트에서 루트로 spacewalk-common-channels 명령을 사용하여 적절한 채널을 추가합니다.

```
spacewalk-common-channels \
<base_channel_label> \
<child_channel_label_1> \
<child_channel_label_2> \
... <child_channel_label_n>
```

2. **automatic synchronization**이 꺼진 경우 다음 채널을 동기화하십시오.

```
spacewalk-repo-sync -p <base_channel_label>
```

3. 계속하기 전에 동기화가 완료되었는지 확인합니다.

4.2.2.2. 동기화 상태 확인

절차: Web UI에서 동기화 진행 상태 확인

1. Uyuni Web UI에서 **소프트웨어** > **관리** > **채널**로 이동한 다음, 리포지토리에 연결된 채널을 클릭합니다.

2. ⌘ 템으로 이동한 다음, ⌘를 클릭하고 ⌘ ⌘를 확인합니다.

절차: 명령 프롬프트에서 동기화 진행 상태 확인

1. Uyuni 서버의 명령 프롬프트에서 루트 권한으로 다음과 같이 tail 명령을 사용해 동기화 로그 파일을 확인합니다.

```
tail -f /var/log/rhn/reposync/<channel-label>.log
```

2. 각 하위 채널은 동기화가 진행되는 동안 자체 로그를 생성합니다. 동기화가 완료되었는지 알아보려면 모든 기본 및 하위 채널 로그 파일을 확인해야 합니다.

4.2.2.3. 클라이언트 등록



SLE Micro clients require a reboot after registering.

Following the bootstrapping process, automatic booting is disabled on SLE Micro clients. This change was implemented due to intermittent automatic reboots, which could occur before Salt could relay the results of applying the bootstrap Salt state.

등록이 완료되면 재부팅이 자동으로 예약되지만, 기본 재부팅 관리자 유지보수 기간이 적용됩니다. 이 기간은 클라이언트가 등록되고 몇 시간이 지난 후가 될 수 있습니다. 등록 스크립트가 완료된 후 등록 속도를 향상하고 시스템이 시스템 목록에 표시되도록 하려면 클라이언트를 수동으로 재부팅하는 것이 좋습니다.

클라이언트를 등록하려면 부트스트랩 리포지토리가 필요합니다. 기본적으로 부트스트랩 리포지토리는 자동으로 생성되며 동기화된 모든 제품에 대해 매일 재생성됩니다. 명령 프롬프트에서 다음 명령을 사용해 부트스트랩 리포지토리를 수동으로 생성할 수 있습니다.

```
mgr-create-bootstrap-repo
```

클라이언트 등록에 대한 자세한 내용은 [Client-configuration > Registration-overview](#)에서 참조하십시오.

When using a bootstrap script with SLE Micro systems, ensure that the certificate section of the script has this content:

```
ORG_CA_CERT=RHN-ORG-TRUSTED-SSL-CERT
ORG_CA_CERT_IS_RPM_YN=0
```

부트스트랩 스크립트를 직접 편집하고 설정을 추가하거나 다음 매개변수를 사용하여 부트스트랩 스크립트를 생성하십시오.

```
mgr-bootstrap --script=bootstrap-sle-micro.sh \
--ssl-cert=/srv/www/htdocs/pub/RHN-ORG-TRUSTED-SSL-CERT
```

4.2.2.4. Reboot SLE Micro

SLE Micro is a transactional system. Transactional updates in general support several reboot methods. It is recommended to use `systemd` for rebooting in systems managed by Uyuni. Using other methods can lead to undesired behavior.

Uyuni에서 트랜잭션 시스템을 부트스트랩할 때 시스템이 기본 구성인 경우 `systemd`는 재부팅 방법(REBOOT_METHOD)대로 구성됩니다. 이러한 구성을 사용하면 Uyuni가 재부팅 작업을 제어할 수 있으며 재부팅을 즉시 수행하거나 Uyuni에서 원하는 일정을 예약할 수 있습니다.

4.2.2.4.1. 배경 정보

기본적으로 클라이언트를 설치하는 동안 재부팅 방법은 `auto`로 설정됩니다. `auto` 부팅 방법을 사용하면 서비스가 실행 중인 경우 구성된 정책에 따라 시스템을 재부팅하기 위해 `rebootmgrd`가 사용됩니다. 정책은 즉시 재부팅 또는 유지보수 기간 중 재부팅입니다. 자세한 내용은 `rebootmgrd(8)` 사용자 지정 페이지를 참조하십시오. `rebootmgrd`가 실행 중이 아닌 경우 Uyuni는 `systemctl reboot`를 호출합니다.



- `systemd` 이외의 방법을 사용하면 원하지 않은 동작이 발생할 수 있습니다.

4.2.3. SL Micro 클라이언트 등록

이 섹션에는 SL Micro 운영 체제 6.0 x86-64, arm64, and IBM Z (s390x)를 실행하는 클라이언트를 등록하는 방법에 대한 정보가 포함되어 있습니다.

SL Micro는 에지 컴퓨팅을 위해 특별히 제작된 매우 안정적인 경량 운영 체제입니다. 그리고 SUSE Linux Enterprise의 엔터프라이즈 강화 보안 및 규정 준수 구성 요소를 활용하며, 최신의 변경 불가능하고 개발자 친화적인 OS 플랫폼과 병합합니다.

SL Micro는 트랜잭션 업데이트를 사용합니다. 트랜잭션 업데이트는 원자적(모든 업데이트가 성공한 경우에만 모든 업데이트가 적용됨)이며 룰백을 지원합니다. 시스템이 재부팅될 때까지 변경 사항이 적용되지 않으므로 실행 중인 시스템에 영향을 주지 않습니다. 이와 관련된 정보는 **Systems** > **세부 정보** > **개요** 하위 탭에 표시됩니다.

트랜잭션 업데이트 및 재부팅에 대한 자세한 내용은 <https://documentation.suse.com/sles/html/SLES-all/cha-transactional-updates.html>에서 확인할 수 있습니다.

4.2.3.1. 소프트웨어 채널 추가

SL Micro 클라이언트를 Uyuni 서버에 등록하기 전에 필요한 소프트웨어 채널을 추가하고 이 채널을 동기화해야 합니다.



- 다음 섹션에서 설명은 종종 x86_64 아키텍처로 기본 설정됩니다. 해당될 경우 다른 아키텍처로 교체하십시오.

이 절차에 필요한 제품은 다음과 같습니다.

표 25. SL Micro 제품 - WebUI

OS 버전	제품 이름
SL Micro 6.0 x86-64	SUSE Linux Micro 6.0 x86_64
SL Micro 6.0 arm64	SUSE Linux Micro 6.0 arch64
SL Micro 6.0 s390x	SUSE Linux Micro 6.0 s390x

절차: 소프트웨어 채널 추가

- Uyuni Web UI에서 **관리** > **설치 마법사** > **제품**으로 이동합니다.
- 검색 창을 사용하여 클라이언트 운영 체제 및 아키텍처에 적합한 제품을 찾고 해당 제품을 확인하십시오. 모든 필수 채널은 자동으로 확인됩니다. 또한, ↗↗ 토글이 켜져 있으면 모든 추천 채널이 확인됩니다. 관련 제품의 전체 목록을 보려면 화살표를 클릭하고 필요한 추가 제품이 선택되어 있는지 확인하십시오.
- [**제품 추가**]를 클릭하고 제품이 동기화를 마칠 때까지 기다립니다.

또는 명령 프롬프트에서 채널을 추가할 수 있습니다. 이 절차에 필요한 채널은 다음과 같습니다.

표 26. SL Micro 제품 - CLI

OS 버전	기본 채널
SL Micro 6.0 x86-64	sl-micro-6.0-pool-x86_64

명령 프롬프트에서 소프트웨어 채널 추가

- Uyuni 서버의 명령 프롬프트에서 root 권한으로 다음과 같이 `mgr-sync` 명령을 사용해 적절한 채널을 추가합니다.

```
mgr-sync add channel <channel_label_1>
mgr-sync add channel <channel_label_2>
mgr-sync add channel <channel_label_n>
```

- 동기화가 자동으로 시작됩니다. 채널을 수동으로 동기화하려면 다음 명령을 사용하십시오.

```
mgr-sync sync --with-children <channel_name>
```

- 계속하기 전에 동기화가 완료되었는지 확인합니다.

4.2.3.2. 동기화 상태 확인

절차: Web UI에서 동기화 진행 상태 확인

- Uyuni Web UI에서 **소프트웨어** > **관리** > **채널**로 이동한 다음, 리포지토리에 연결된 채널을 클릭합니다.
- ☞☞☞ 탭으로 이동한 다음, ☞☞☞를 클릭하고 ☞☞☞를 확인합니다.

절차: 명령 프롬프트에서 동기화 진행 상태 확인

- Uyuni 서버의 명령 프롬프트에서 루트 권한으로 다음과 같이 `tail` 명령을 사용해 동기화 로그 파일을 확인합니다.

```
tail -f /var/log/rhn/reposync/<channel-label>.log
```

- 각 하위 채널은 동기화가 진행되는 동안 자체 로그를 생성합니다. 동기화가 완료되었는지 알아보려면 모든 기본 및 하위 채널 로그 파일을 확인해야 합니다.

4.2.3.3. 클라이언트 등록



SL Micro 클라이언트를 등록한 후 재부팅해야 합니다. 등록이 완료된 후 재부팅이 자동으로 예약되지만, 기본 재부팅 관리자 유지보수 기간을 따릅니다. 이 기간은 클라이언트가 등록된 후 몇 시간이 걸릴 수 있습니다. 등록 속도를 향상하고 시스템이 시스템 목록에 표시되는 것을 확인하려면 등록 스크립트가 완료된 후 클라이언트를 수동으로 재부팅하는 것이 좋습니다.

클라이언트를 등록하려면 부트스트랩 리포지토리가 필요합니다. 기본적으로 부트스트랩 리포지토리는 자동으로 생성되며 동기화된 모든 제품에 대해 매일 재생성됩니다. 명령 프롬프트에서 다음 명령을 사용해 부트스트랩 리포지토리를 수동으로 생성할 수 있습니다.

```
mgr-create-bootstrap-repo
```

클라이언트 등록에 대한 자세한 내용은 [Client-configuration > Registration-overview](#)에서 참조하십시오.

SL Micro 시스템에서 부트스트랩 스크립트를 사용할 때 스크립트의 인증서 섹션에 다음 내용이 있는지 확인하십시오.

```
ORG_CA_CERT=RHN-ORG-TRUSTED-SSL-CERT
ORG_CA_CERT_IS_RPM_YN=0
```

부트스트랩 스크립트를 직접 편집하고 설정을 추가하거나 다음 매개변수를 사용하여 부트스트랩 스크립트를 생성하십시오.

```
mgr-bootstrap --script=bootstrap-sl-micro.sh \
--ssl-cert=/srv/www/htdocs/pub/RHN-ORG-TRUSTED-SSL-CERT
```

4.2.3.4. SL Micro 재부팅

SL Micro는 트랜잭션 시스템입니다. 일반적으로 트랜잭션 업데이트는 다양한 재부팅 방법을 지원합니다. Uyuni에서 관리하는 시스템에서 재부팅을 수행하려면 `systemd`을 사용하는 것이 좋습니다. 다른 방법을 사용하면 원하지 않은 동작이 발생할 수 있습니다.

Uyuni에서 트랜잭션 시스템을 부트스트랩할 때 시스템이 기본 구성인 경우 `systemd`은 재부팅 방법(`REBOOT_METHOD`)대로 구성됩니다. 이러한 구성을 사용하면 Uyuni가 재부팅 작업을 제어할 수 있으며 재부팅을

즉시 수행하거나 Uyuni에서 원하는 일정을 예약할 수 있습니다.

4.2.3.4.1. 배경 정보

기본적으로 클라이언트를 설치하는 동안 재부팅 방법은 `auto`로 설정됩니다. `auto` 부팅 방법을 사용하면 서비스가 실행 중인 경우 구성된 정책에 따라 시스템을 재부팅하기 위해 `rebootmgrd`가 사용됩니다. 정책은 즉시 재부팅 또는 유지보수 기간 중 재부팅입니다. 자세한 내용은 `rebootmgrd(8)` 사용자 지정 페이지를 참조하십시오. `rebootmgrd`가 실행 중이 아닌 경우 Uyuni는 `systemctl reboot`를 호출합니다.



`systemd` 이외의 방법을 사용하면 원하지 않은 동작이 발생할 수 있습니다.

4.3. openSUSE 클라이언트 등록

openSUSE 및 openSUSE Leap Micro 클라이언트를 Uyuni 서버에 등록할 수 있습니다. 방법과 자세한 내용은 클라이언트의 운영 체제에 따라 달라집니다.

시작하기 전에 클라이언트의 날짜 및 시간이 Uyuni 서버와 정확히 동기화되었는지 확인하십시오.

또한 활성화 키 생성을 완료한 상태이어야 합니다. 활성화 키 생성에 대한 자세한 내용은 [Client-configuration > Activation-keys](#)를 참조하십시오.



Uyuni 서버를 자체에 등록하지 마십시오. Uyuni 서버는 개별적으로 또는 별도의 Uyuni 서버를 사용해 관리해야 합니다. 여러 서버를 사용하는 방법에 대한 자세한 내용은 [Specialized-guides > Large-deployments](#)에서 확인할 수 있습니다.

4.3.1. openSUSE Leap 클라이언트 등록

이 섹션에는 openSUSE 운영 체제를 실행하는 클라이언트를 등록하는 방법에 대한 정보가 포함되어 있습니다. Uyuni은(는) Salt를 사용하는 openSUSE Leap 15 클라이언트를 지원합니다.

부트스트래핑은 openSUSE 클라이언트 시작과 초기 상태 실행 수행(예: 리포지토리 설정, 프로파일 업데이트 수행)에 대해 지원됩니다.

4.3.1.1. 소프트웨어 채널 추가

openSUSE 클라이언트를 Uyuni 서버에 등록하기 전에 필요한 소프트웨어 채널을 추가하고 이 채널을 동기화해야 합니다.

현재 지원되는 아키텍처는 `x86_64` 및 `aarch64`입니다. 지원되는 제품 및 아키텍처의 전체 목록은 [Client-configuration > Supported-features](#)에서 참조하십시오.



다음 섹션에서 설명은 종종 `x86_64` 아키텍처로 기본 설정됩니다. 해당될 경우 다른 아키텍처로 교체하십시오.

예를 들어, `x86_64` 아키텍처에서 작업 시 필요한 제품은 다음과 같습니다.

표 27. openSUSE 채널 - CLI

OS 버전	openSUSE Leap 15.6	openSUSE Leap 15.5	openSUSE Leap 15.4
기본 채널	opensuse_leap15_6	opensuse_leap15_5	opensuse_leap15_4
클라이언트 채널	opensuse_leap15_6-uyuni-client	opensuse_leap15_5-uyuni-client	opensuse_leap15_4-uyuni-client
업데이트 채널	opensuse_leap15_6-updates	opensuse_leap15_5-updates	opensuse_leap15_4-updates
비OSS 채널	opensuse_leap15_6-non-oss	opensuse_leap15_5-non-oss	opensuse_leap15_4-non-oss
비OSS 업데이트 채널	opensuse_leap15_6-non-oss-updates	opensuse_leap15_5-non-oss-updates	opensuse_leap15_4-non-oss-updates
백포트 업데이트 채널	opensuse_leap15_6-backports-updates	opensuse_leap15_5-backports-updates	opensuse_leap15_4-backports-updates
SLE 업데이트 채널	opensuse_leap15_6-sle-updates	opensuse_leap15_5-sle-updates	opensuse_leap15_4-sle-updates

명령 프롬프트에서 소프트웨어 채널 추가

- Uyuni 서버의 명령 프롬프트에서 루트로 spacewalk-common-channels 명령을 사용하여 적절한 채널을 추가합니다.

```
spacewalk-common-channels \
<base_channel_label> \
<child_channel_label_1> \
<child_channel_label_2> \
... <child_channel_label_n>
```

- automatic synchronization이 꺼진 경우 다음 채널을 동기화하십시오.

```
spacewalk-repo-sync -p <base_channel_label>
```

- 계속하기 전에 동기화가 완료되었는지 확인합니다.

4.3.1.2. 동기화 상태 확인

절차: Web UI에서 동기화 진행 상태 확인

- Uyuni Web UI에서 **소프트웨어** > **관리** > **채널**로 이동한 다음, 리포지토리에 연결된 채널을 클릭합니다.
- bbbbbb 탭으로 이동한 다음, bbbb를 클릭하고 bbbb bbb를 확인합니다.

절차: 명령 프롬프트에서 동기화 진행 상태 확인

- Uyuni 서버의 명령 프롬프트에서 루트 권한으로 다음과 같이 tail 명령을 사용해 동기화 로그 파일을 확인합니다.

```
tail -f /var/log/rhn/reposync/<channel-label>.log
```

2. 각 하위 채널은 동기화가 진행되는 동안 자체 로그를 생성합니다. 동기화가 완료되었는지 알아보려면 모든 기본 및 하위 채널 로그 파일을 확인해야 합니다.



openSUSE 채널은 규모가 매우 클 수 있습니다. 때로 동기화에 몇 시간이 걸릴 수 있습니다.

4.3.1.3. GPG 키 관리

클라이언트는 소프트웨어 패키지가 설치되기 전에 GPG 키를 사용해 신뢰성을 확인합니다. 신뢰할 수 있는 소프트웨어만 클라이언트에 설치할 수 있습니다.



클라이언트의 보안을 위해 GPG 키를 신뢰하는 것이 중요합니다. 어떤 키가 필요하고 신뢰할 수 있는지 결정하는 것은 관리자가 수행해야 하는 작업입니다. GPG 키를 신뢰할 수 없는 경우 소프트웨어 채널을 사용할 수 없기 때문에 채널을 클라이언트에 할당하는 결정은 키를 신뢰하는 결정에 따라 달라집니다.

GPG 키에 대한 자세한 내용은 [Client-configuration > Gpg-keys](#)에서 확인할 수 있습니다.

4.3.1.4. 클라이언트 등록

클라이언트를 등록하려면 부트스트랩 리포지토리가 필요합니다. 기본적으로 부트스트랩 리포지토리는 자동으로 생성되며 동기화된 모든 제품에 대해 매일 재생성됩니다. 명령 프롬프트에서 다음 명령을 사용해 부트스트랩 리포지토리를 수동으로 생성할 수 있습니다.

```
mgr-create-bootstrap-repo
```

클라이언트 등록에 대한 자세한 내용은 [Client-configuration > Registration-overview](#)에서 참조하십시오.

4.3.2. openSUSE Leap Micro 클라이언트 등록

이 섹션에는 x86-64 및 aarch64 아키텍처에서 openSUSE Leap Micro 운영 체제를 실행하는 클라이언트를 등록하는 방법에 대한 정보가 포함되어 있습니다.

openSUSE Leap Micro는 에지 컴퓨팅을 위해 특별히 제작된 매우 안정적인 경량 운영 체제입니다. 그리고 SUSE Linux Enterprise의 엔터프라이즈 강화 보안 및 규정 준수 구성 요소를 활용하며, 최신의 변경 불가능하고 개발자 친화적인 OS 플랫폼이 통합되어 있습니다.

openSUSE Leap Micro는 트랜잭션 업데이트를 사용합니다. 트랜잭션 업데이트는 원자적(모든 업데이트가 성공한 경우에만 모든 업데이트가 적용됨)이며 롤백을 지원합니다. 시스템이 재부팅될 때까지 변경 사항이 적용되지 않으므로 실행 중인 시스템에 영향을 주지 않습니다. 이 정보는 [시스템 > 세부사항 > 개요](#) 하위 탭에 표시됩니다.

트랜잭션 업데이트 및 재부팅에 대한 자세한 내용은 <https://documentation.suse.com/sles/html/SLES-all/cha-transactional-updates.html>에서 확인할 수 있습니다.

표 28. openSUSE 채널 - CLI

OS 버전	openSUSE Leap Micro 5.5	openSUSE Leap Micro 5.4	openSUSE Leap Micro 5.3
기본 채널	opensuse_micro5_5	opensuse_micro5_4	opensuse_micro5_3
클라이언트 채널	opensuse_micro5_5-uyuni-client	opensuse_micro5_4-uyuni-client	opensuse_micro5_3-uyuni-client
SLE 업데이트 채널	opensuse_micro5_5-sle-updates	opensuse_micro5_4-sle-updates	opensuse_micro5_3-sle-updates

명령 프롬프트에서 소프트웨어 채널 추가

- Uyuni 서버의 명령 프롬프트에서 루트로 spacewalk-common-channels 명령을 사용하여 적절한 채널을 추가합니다.

```
spacewalk-common-channels \
<base_channel_label> \
<child_channel_label_1> \
<child_channel_label_2> \
... <child_channel_label_n>
```

- automatic synchronization이 꺼진 경우 다음 채널을 동기화하십시오.

```
spacewalk-repo-sync -p <base_channel_label>
```

- 계속하기 전에 동기화가 완료되었는지 확인합니다.

4.3.2.1. 동기화 상태 확인

절차: Web UI에서 동기화 진행 상태 확인

- Uyuni Web UI에서 **소프트웨어** > **관리** > **채널**로 이동한 다음, 리포지토리에 연결된 채널을 클릭합니다.
- 탭으로 이동한 다음, **동기화**를 클릭하고 **확인**을 확인합니다.

절차: 명령 프롬프트에서 동기화 진행 상태 확인

- Uyuni 서버의 명령 프롬프트에서 루트 권한으로 다음과 같이 tail 명령을 사용해 동기화 로그 파일을 확인합니다.

```
tail -f /var/log/rhn/reposync/<channel-label>.log
```

- 각 하위 채널은 동기화가 진행되는 동안 자체 로그를 생성합니다. 동기화가 완료되었는지 알아보려면 모든 기본 및 하위 채널 로그 파일을 확인해야 합니다.



openSUSE Leap Micro 채널은 규모가 매우 클 수 있습니다. 일부 경우 동기화에 몇 시간이 걸릴 수 있습니다.

4.3.2.2. 클라이언트 등록



openSUSE Leap Micro 클라이언트는 등록한 후 재부팅해야 합니다.

부트스트랩 프로세스 이후 openSUSE Leap Micro에서 자동 부팅이 비활성화됩니다. 이 변경 사항은 간헐적인 자동 재부팅의 발생으로 인해 구현되었으며, 이 현상은 Salt가 부트스트랩 salt 상태 적용 결과를 전달하기 전에 발생할 수 있습니다.

재부팅은 등록이 완료된 후 자동으로 예약되지만, 기본 재부팅 관리자 유지보수 기간이 적용됩니다. 이 기간은 클라이언트가 등록되고 몇 시간 후가 될 수 있습니다. 등록 속도를 향상하려면 등록 스크립트가 완료된 후 클라이언트를 수동으로 재부팅합니다.

클라이언트를 등록하려면 부트스트랩 리포지토리가 필요합니다. 기본적으로 부트스트랩 리포지토리는 자동으로 생성되며 동기화된 모든 제품에 대해 매일 재생성됩니다. 명령 프롬프트에서 다음 명령을 사용해 부트스트랩 리포지토리를 수동으로 생성할 수 있습니다.

```
mgr-create-bootstrap-repo
```

클라이언트 등록에 대한 자세한 내용은 [Client-configuration > Registration-overview](#)에서 참조하십시오.

4.4. Alibaba Cloud Linux 클라이언트 등록

Alibaba Cloud Linux 클라이언트를 Uyuni 서버에 등록할 수 있습니다. 방법과 상세 정보는 클라이언트의 운영 체제에 따라 다릅니다.

시작하기 전에 클라이언트의 날짜 및 시간이 Uyuni 서버와 정확히 동기화되었는지 확인하십시오.

또한 활성화 키 생성을 완료한 상태이어야 합니다. 활성화 키 생성에 대한 자세한 내용은 [Client-configuration > Activation-keys](#)를 참조하십시오.

4.4.1. Alibaba Cloud Linux 클라이언트 등록

이 섹션에는 Alibaba Cloud Linux 운영 체제를 실행하는 클라이언트를 등록하는 방법에 대한 정보가 포함되어 있습니다.



일부 Alibaba Cloud Linux 2 인스턴스를 등록하려면 2회 시도해야 합니다.

4.4.1.1. 소프트웨어 채널 추가

Alibaba Cloud Linux 클라이언트를 Uyuni 서버에 등록하기 전에 필요한 소프트웨어 채널을 추가하고 이 채널을 동기화해야 합니다.



다음 섹션에서 설명은 종종 x86_64 아키텍처로 기본 설정됩니다. 해당될 경우 다른 아키텍처로 교체하십시오.

이 절차에 필요한 채널은 다음과 같습니다.

표 29. Alibaba Cloud Linux 채널 - CLI

OS 버전	Core Channel	Updates Channel	Client Channel
Alibaba Cloud Linux 2	alibaba-2	alibaba-2-updates	alibaba-2-uyuni-client

명령 프롬프트에서 소프트웨어 채널 추가

- Uyuni 서버의 명령 프롬프트에서 루트로 spacewalk-common-channels 명령을 사용하여 적절한 채널을 추가합니다.

```
spacewalk-common-channels \
<base_channel_label> \
<child_channel_label_1> \
<child_channel_label_2> \
... <child_channel_label_n>
```

- automatic synchronization이 꺼진 경우 다음 채널을 동기화하십시오.

```
spacewalk-repo-sync -p <base_channel_label>
```

- 계속하기 전에 동기화가 완료되었는지 확인합니다.



spacewalk-common-channels가 제공하는 클라이언트 도구 채널은 SUSE가 아닌 Uyuni가 공급합니다.

4.4.1.2. 동기화 상태 확인

절차: Web UI에서 동기화 진행 상태 확인

- Uyuni Web UI에서 **소프트웨어** > **관리** > **채널**로 이동한 다음, 리포지토리에 연결된 채널을 클릭합니다.
- 탭으로 이동한 다음, **동기화**를 클릭하고 **확인**을 확인합니다.

절차: 명령 프롬프트에서 동기화 진행 상태 확인

- Uyuni 서버의 명령 프롬프트에서 루트 권한으로 다음과 같이 tail 명령을 사용해 동기화 로그 파일을 확인합니다.

```
tail -f /var/log/rhn/reposync/<channel-label>.log
```

- 각 하위 채널은 동기화가 진행되는 동안 자체 로그를 생성합니다. 동기화가 완료되었는지 알아보려면 모든 기본 및 하위 채널 로그 파일을 확인해야 합니다.

4.4.1.3. 활성화 키 생성

Alibaba Cloud Linux 채널에 연결된 활성화 키를 생성해야 합니다.

활성화 키에 대한 자세한 내용은 [Client-configuration > Activation-keys](#)를 참조하십시오.

4.4.1.4. 클라이언트 등록

클라이언트를 등록하려면 부트스트랩 리포지토리가 필요합니다. 기본적으로 부트스트랩 리포지토리는 자동으로 생성되며 동기화된 모든 제품에 대해 매일 재생성됩니다. 명령 프롬프트에서 다음 명령을 사용해 부트스트랩 리포지토리를 수동으로 생성할 수 있습니다.

```
mgr-create-bootstrap-repo
```

클라이언트 등록에 대한 자세한 내용은 [Client-configuration > Registration-overview](#)에서 참조하십시오.

일부 Alibaba Cloud Linux 2 인스턴스는 첫 번째 시도에서 등록에 실패합니다. 이 문제는 Alibaba Cloud Linux 2 이미지의 알려진 버그로 인한 것입니다.

`python-urlgrabber3` 패키지는 Python pip 패키지 및 RPM 패키지로 제공되며, 이로 인해 첫 번째 등록 시도에서 충돌이 발생할 수 있습니다.

인스턴스가 영향을 받는 이미지 버전 중 하나에 해당하는 경우 두 번째 등록 시도에서 클라이언트가 올바르게 등록되어야 합니다.

4.5. AlmaLinux 클라이언트 등록

AlmaLinux 클라이언트를 Uyuni 서버에 등록할 수 있습니다. 방법과 상세 정보는 클라이언트의 운영 체제에 따라 다릅니다.

시작하기 전에 클라이언트의 날짜 및 시간이 Uyuni 서버와 정확히 동기화되었는지 확인하십시오.

활성화 키도 생성해야 합니다.

- 활성화 키 만들기에 대한 자세한 내용은 [Client-configuration > Activation-keys](#)에서 확인할 수 있습니다.
- AlmaLinux에서 SUSE Liberty Linux로 마이그레이션에 대한 자세한 내용은 [client-configuration:clients-sleses.pdf](#)에서 확인할 수 있습니다.

4.5.1. AlmaLinux 클라이언트 등록

이 섹션에는 AlmaLinux 운영 체제를 실행하는 클라이언트를 등록하는 방법에 대한 정보가 포함되어 있습니다.



AWS에서 생성할 때 AlmaLinux 인스턴스에서는 항상 `/etc/machine-id` 에서 `machine-id` 가 동일합니다. 인스턴스가 생성된 후에 `machine-id`를 다시 생성해야 합니다. 자세한 설명은 [Administration > Troubleshooting](#)에서 확인할 수 있습니다.

4.5.1.1. 소프트웨어 채널 추가



AlmaLinux 클라이언트를 Uyuni에 등록하는 기능은 그 정책으로 그하는 기본 SELinux 구성으로 테스트됩니다. SELinux를 비활성화해야 AlmaLinux 클라이언트를 Uyuni에 등록할 수 있는 것은 아닙니다.

AlmaLinux 클라이언트를 Uyuni 서버에 등록하려면 먼저 필요한 소프트웨어 채널을 추가하고 이 채널을 동기화해야 합니다.

현재 지원되는 아키텍처는 `x86_64` 및 `aarch64`이며, 버전 9의 경우 추가적으로 `ppc64le` 및 `s390x`가 지원됩니다. 지원되는 제품 및 아키텍처의 전체 목록은 [Client-configuration > Supported-features](#)에서 확인할 수 있습니다.



다음 섹션에서 설명은 종종 `x86_64` 아키텍처로 기본 설정됩니다. 해당될 경우 다른 아키텍처로 교체하십시오.

이 절차에 필요한 채널은 다음과 같습니다.

표 30. AlmaLinux 채널 - CLI

OS 버전	기본 채널	클라이언트 채널	AppStream 채널
AlmaLinux 9	<code>almalinux9</code>	<code>almalinux9-uyuni-client</code>	<code>almalinux9-appstream</code>
AlmaLinux 8	<code>almalinux8</code>	<code>almalinux8-uyuni-client</code>	<code>almalinux8-appstream</code>

명령 프롬프트에서 소프트웨어 채널 추가

- Uyuni 서버의 명령 프롬프트에서 root 권한으로 `spacewalk-common-channels` 명령을 사용해 적절한 채널을 추가합니다. 다음과 같이 올바른 아키텍처를 지정했는지 확인합니다.

```
spacewalk-common-channels \
-a <architecture> \
<base_channel_name> \
<child_channel_name_1> \
<child_channel_name_2> \
... <child_channel_name_n>
```

- `automatic synchronization`이 꺼진 경우 다음 채널을 동기화하십시오.

```
spacewalk-repo-sync -p <base_channel_label>-<architecture>
```

- 계속하기 전에 동기화가 완료되었는지 확인합니다.



spacewalk-common-channels가 제공하는 클라이언트 도구 채널은 SUSE가 아닌 Uyuni가 공급합니다.



AlmaLinux 9 및 AlmaLinux 8 클라이언트의 경우 기본 채널과 AppStream 채널을 모두 추가하십시오. 사용자에게는 두 채널의 패키지가 모두 필요합니다. 두 채널을 모두 추가하지 않으면 패키지가 누락되어 부트스트랩 리포지토리를 생성할 수 없습니다.

모듈형 채널을 사용하는 경우 AlmaLinux 8 클라이언트에서 Python 3.6 모듈 스트림을 활성화해야 합니다. Python 3.6이 제공되지 않으면 spacecmd 패키지의 설치가 실패합니다.



AppStream 채널에서 사용할 수 있는 패키지의 수가 업스트림과 Uyuni 채널 간에 서로 약간 불일치함을 알 수 있습니다. 또한 다른 시점에 추가한 동일 채널을 비교하면 그 수가 다른 것을 알 수 있습니다. 이는 AlmaLinux가 리포지토리를 관리하는 방식으로 인한 것입니다. AlmaLinux는 새 버전이 릴리스되면 이전 버전의 패키지를 제거하지만 Uyuni는 사용 기간과 관계없이 모두 그대로 유지합니다.

4.5.1.2. 동기화 상태 확인

절차: Web UI에서 동기화 진행 상태 확인

1. Uyuni Web UI에서 **소프트웨어 > 관리 > 채널**로 이동한 다음, 리포지토리에 연결된 채널을 클릭합니다.
2. ⌘ 키 템으로 이동한 다음, ⌘ 키를 클릭하고 ⌘ 키를 확인합니다.

절차: 명령 프롬프트에서 동기화 진행 상태 확인

1. Uyuni 서버의 명령 프롬프트에서 루트 권한으로 다음과 같이 `tail` 명령을 사용해 동기화 로그 파일을 확인합니다.

```
tail -f /var/log/rhn/reposync/<channel-label>.log
```

2. 각 하위 채널은 동기화가 진행되는 동안 자체 로그를 생성합니다. 동기화가 완료되었는지 알아보려면 모든 기본 및 하위 채널 로그 파일을 확인해야 합니다.

4.5.1.3. 활성화 키 생성

AlmaLinux 채널에 연결된 활성화 키를 생성해야 합니다.

활성화 키에 대한 자세한 내용은 **Client-configuration > Activation-keys**를 참조하십시오.

4.5.1.4. GPG 키 관리

클라이언트는 소프트웨어 패키지가 설치되기 전에 GPG 키를 사용해 신뢰성을 확인합니다. 신뢰할 수 있는 소프트웨어만 클라이언트에 설치할 수 있습니다.



클라이언트의 보안을 위해 GPG 키의 신뢰성이 중요합니다. 어떤 키가 필요하고 신뢰할 수 있는지 결정하는 것은 관리자가 수행해야 하는 작업입니다. GPG 키를 신뢰하지 않으면 소프트웨어 채널을 사용할 수 없으므로 클라이언트에 채널을 할당할지 여부는 키를 신뢰하는 지에 따라 달라집니다.

GPG 키에 대한 자세한 내용은 [Client-configuration > Gpg-keys](#)에서 확인할 수 있습니다.

4.5.1.5. 클라이언트 등록

클라이언트를 등록하려면 부트스트랩 리포지토리가 필요합니다. 기본적으로 부트스트랩 리포지토리는 자동으로 생성되며 동기화된 모든 제품에 대해 매일 재생성됩니다. 명령 프롬프트에서 다음 명령을 사용해 부트스트랩 리포지토리를 수동으로 생성할 수 있습니다.

```
mgr-create-bootstrap-repo
```

클라이언트 등록에 대한 자세한 내용은 [Client-configuration > Registration-overview](#)에서 참조하십시오.

4.5.1.6. 정오표 관리

AlmaLinux 클라이언트를 업데이트할 때 패키지에는 업데이트에 대한 메타데이터가 포함됩니다.

4.6. Amazon Linux 클라이언트 등록

Amazon Linux 클라이언트를 Uyuni 서버에 등록할 수 있습니다. 방법과 상세 정보는 클라이언트의 운영 체제에 따라 다릅니다.

시작하기 전에 클라이언트의 날짜 및 시간이 Uyuni 서버와 정확히 동기화되었는지 확인하십시오.

또한 활성화 키 생성을 완료한 상태이어야 합니다. 활성화 키 생성에 대한 자세한 내용은 [Client-configuration > Activation-keys](#)를 참조하십시오.

4.6.1. Amazon Linux 클라이언트 등록

이 섹션에는 Amazon Linux 운영 체제를 실행하는 클라이언트를 등록하는 방법에 대한 정보가 포함되어 있습니다.



AWS에서 생성된 경우, Amazon Linux 2 인스턴스는 항상 `/etc/machine-id`에 `machine-id` ID를 갖습니다. Amazon Linux 2 인스턴스를 생성하는 경우 인스턴스가 생성된 후 `/etc/machine-id`를 다시 생성해야 합니다.

Amazon Linux 2023은 이로 인해 영향을 받지 않습니다.

4.6.1.1. 소프트웨어 채널 추가

Amazon Linux 클라이언트를 Uyuni 서버에 등록하기 전에 필요한 소프트웨어 채널을 추가하고 이 채널을 동기화해야 합니다.

현재 지원되는 아키텍처는 `x86_64` 및 `aarch64`입니다. 지원되는 제품 및 아키텍처의 전체 목록은 **Client-configuration > Supported-features**에서 참조하십시오.



다음 섹션에서 설명은 종종 `x86_64` 아키텍처로 기본 설정됩니다. 해당될 경우 다른 아키텍처로 교체하십시오.

이 절차에 필요한 채널은 다음과 같습니다.

표 31. Amazon Linux 채널 - CLI

OS 버전	기본 채널	클라이언트 채널
Amazon Linux 2023	<code>amazonlinux2023</code>	<code>amazonlinux2023-uyuni-client</code>
Amazon Linux 2	<code>amazonlinux2-core</code>	<code>amazonlinux2-uyuni-client</code>



Amazon Linux 2의 경우, Amazon Linux 인스턴스에서 Docker를 사용하려면 `amazonlinux2-extra-docker` 채널도 추가하고 동기화해야 합니다.



Amazon Linux 2023의 경우, Amazon Linux 인스턴스에서 커널 라이브 패치를 사용하려면 `amazonlinux2023-kernel-livepatch` 채널도 추가하고 동기화해야 합니다.

명령 프롬프트에서 소프트웨어 채널 추가

1. Uyuni 서버의 명령 프롬프트에서 루트로 `spacewalk-common-channels` 명령을 사용하여 적절한 채널을 추가합니다.

```
spacewalk-common-channels \
<base_channel_label> \
<child_channel_label_1> \
<child_channel_label_2> \
... <child_channel_label_n>
```

2. `automatic synchronization`이 꺼진 경우 다음 채널을 동기화하십시오.

```
spacewalk-repo-sync -p <base_channel_label>
```

3. 계속하기 전에 동기화가 완료되었는지 확인합니다.



`spacewalk-common-channels`가 제공하는 클라이언트 도구 채널은 SUSE가 아닌 Uyuni가 공급합니다.

4.6.1.2. 동기화 상태 확인

절차: Web UI에서 동기화 진행 상태 확인

- Uyuni Web UI에서 **소프트웨어 > 관리 > 채널**로 이동한 다음, 리포지토리에 연결된 채널을 클릭합니다.
- 탭으로 이동한 다음, 를 클릭하고 를 확인합니다.

절차: 명령 프롬프트에서 동기화 진행 상태 확인

- Uyuni 서버의 명령 프롬프트에서 루트 권한으로 다음과 같이 tail 명령을 사용해 동기화 로그 파일을 확인합니다.

```
tail -f /var/log/rhn/reposync/<channel-label>.log
```

- 각 하위 채널은 동기화가 진행되는 동안 자체 로그를 생성합니다. 동기화가 완료되었는지 알아보려면 모든 기본 및 하위 채널 로그 파일을 확인해야 합니다.

4.6.1.3. 활성화 키 생성

Amazon Linux 채널에 연결된 활성화 키를 생성해야 합니다.

활성화 키에 대한 자세한 내용은 **Client-configuration > Activation-keys**를 참조하십시오.

4.6.1.4. GPG 키 관리

클라이언트는 소프트웨어 패키지가 설치되기 전에 GPG 키를 사용해 신뢰성을 확인합니다. 신뢰할 수 있는 소프트웨어만 클라이언트에 설치할 수 있습니다.



클라이언트의 보안을 위해 GPG 키를 신뢰하는 것이 중요합니다. 어떤 키가 필요하고 신뢰할 수 있는지 결정하는 것은 관리자가 수행해야 하는 작업입니다. GPG 키를 신뢰할 수 없는 경우 소프트웨어 채널을 사용할 수 없기 때문에 채널을 클라이언트에 할당하는 결정은 키를 신뢰하는 결정에 따라 달라집니다.

GPG 키에 대한 자세한 내용은 **Client-configuration > Gpg-keys**에서 확인할 수 있습니다.

4.6.1.5. 클라이언트 등록

클라이언트를 등록하려면 부트스트랩 리포지토리가 필요합니다. 기본적으로 부트스트랩 리포지토리는 자동으로 생성되며 동기화된 모든 제품에 대해 매일 재생성됩니다. 명령 프롬프트에서 다음 명령을 사용해 부트스트랩 리포지토리를 수동으로 생성할 수 있습니다.

```
mgr-create-bootstrap-repo
```

클라이언트 등록에 대한 자세한 내용은 **Client-configuration > Registration-overview**에서 참조하십시오.

4.7. CentOS 클라이언트 등록

CentOS 클라이언트를 Uyuni 서버에 등록할 수 있습니다. 방법과 상세 정보는 클라이언트의 운영 체제에 따라 달라집니다.

시작하기 전에 클라이언트의 날짜 및 시간이 Uyuni 서버와 정확히 동기화되었는지 확인하십시오.

활성화 키도 생성해야 합니다.

- 활성화 키 만들기에 대한 자세한 내용은 [Client-configuration > Activation-keys](#)에서 확인할 수 있습니다.
- CentOS에서 SUSE Liberty Linux로 마이그레이션에 대한 자세한 내용은 [client-configuration:clients-sleses.pdf](#)에서 확인할 수 있습니다.

4.7.1. CentOS 클라이언트 등록

이 섹션에는 CentOS 운영 체제를 실행하는 클라이언트를 등록하는 방법에 대한 정보가 포함되어 있습니다.



CentOS 기본 미디어 리포지토리 및 CentOS 설치 미디어에 대한 액세스를 준비하고 Uyuni 서버를 CentOS 콘텐츠 전송 네트워크에 연결할 책임은 사용자에게 있습니다.



CentOS 클라이언트를 Uyuni에 등록하는 기능은 --정책으로 --하는 기본 SELinux 구성으로 테스트됩니다. SELinux를 비활성화해야 CentOS 클라이언트를 Uyuni에 등록할 수 있는 것은 아닙니다.

4.7.1.1. 소프트웨어 채널 추가

CentOS 클라이언트를 Uyuni 서버에 등록하려면 먼저 필요한 소프트웨어 채널을 추가하고 이 채널을 동기화해야 합니다.

현재 지원되는 아키텍처는 x86_64 및 aarch64입니다. 지원되는 제품 및 아키텍처의 전체 목록은 [Client-configuration > Supported-features](#)에서 참조하십시오.



다음 섹션에서 설명은 종종 x86_64 아키텍처로 기본 설정됩니다. 해당될 경우 다른 아키텍처로 교체하십시오.

이 절차에 필요한 채널은 다음과 같습니다.

표 32. CentOS 채널 - CLI

OS 버전	기본 채널	클라이언트 채널	업데이트/Appstream 채널
CentOS 7	centos7	centos7-uyuni-client	centos7-updates

명령 프롬프트에서 소프트웨어 채널 추가

- Uyuni 서버의 명령 프롬프트에서 root 권한으로 spacewalk-common-channels 명령을 사용해 적절한 채널을 추가합니다. 다음과 같이 올바른 아키텍처를 지정했는지 확인합니다.

```
spacewalk-common-channels \
-a <architecture> \
<base_channel_name> \
<child_channel_name_1> \
<child_channel_name_2> \
... <child_channel_name_n>
```

2. **automatic synchronization**이 꺼진 경우 다음 채널을 동기화하십시오.

```
spacewalk-repo-sync -p <base_channel_label>-<architecture>
```

3. 계속하기 전에 동기화가 완료되었는지 확인합니다.



spacewalk-common-channels가 제공하는 클라이언트 도구 채널은 SUSE가 아닌 Uyuni가 공급합니다.

모듈형 채널을 사용하는 경우 클라이언트에서 Python 3.6 모듈 스트림을 활성화해야 합니다. Python 3.6이 제공되지 않으면 spacecmd 패키지의 설치가 실패합니다.



AppStream 채널에서 사용할 수 있는 패키지의 수가 업스트림과 Uyuni 채널 간에 서로 약간 불일치함을 알 수 있습니다. 또한 다른 시점에 추가한 동일 채널을 비교하면 그 수가 다른 것을 알 수 있습니다. 이는 CentOS가 리포지토리를 관리하는 방식에 원인이 있습니다. CentOS는 새 버전이 릴리스되면 이전 버전의 패키지를 제거하지만 Uyuni는 사용 기간과 관계없이 모두 그대로 유지합니다.

4.7.1.2. 동기화 상태 확인

절차: Web UI에서 동기화 진행 상태 확인

1. Uyuni Web UI에서 **소프트웨어**, **관리**, **채널**로 이동한 다음, 리포지토리에 연결된 채널을 클릭합니다.
2. ⏵탭으로 이동한 다음, ⏵를 클릭하고 ⏵를 확인합니다.

절차: 명령 프롬프트에서 동기화 진행 상태 확인

1. Uyuni 서버의 명령 프롬프트에서 루트 권한으로 다음과 같이 `tail` 명령을 사용해 동기화 로그 파일을 확인합니다.

```
tail -f /var/log/rhn/reposync/<channel-label>.log
```

2. 각 하위 채널은 동기화가 진행되는 동안 자체 로그를 생성합니다. 동기화가 완료되었는지 알아보려면 모든 기본 및 하위 채널 로그 파일을 확인해야 합니다.

4.7.1.3. 활성화 키 생성

CentOS 채널에 연결된 활성화 키를 생성해야 합니다.

활성화 키에 대한 자세한 내용은 [Client-configuration > Activation-keys](#)를 참조하십시오.

4.7.1.4. GPG 키 관리

클라이언트는 소프트웨어 패키지가 설치되기 전에 GPG 키를 사용해 신뢰성을 확인합니다. 신뢰할 수 있는 소프트웨어만 클라이언트에 설치할 수 있습니다.



- 클라이언트의 보안을 위해 GPG 키를 신뢰하는 것이 중요합니다. 어떤 키가 필요하고 신뢰할 수 있는지 결정하는 것은 관리자가 수행해야 하는 작업입니다. GPG 키를 신뢰할 수 없는 경우 소프트웨어 채널을 사용할 수 없기 때문에 채널을 클라이언트에 할당하는 결정은 키를 신뢰하는 결정에 따라 달라집니다.

GPG 키에 대한 자세한 내용은 [Client-configuration > Gpg-keys](#)에서 확인할 수 있습니다.

4.7.1.5. 클라이언트 등록

클라이언트를 등록하려면 부트스트랩 리포지토리가 필요합니다. 기본적으로 부트스트랩 리포지토리는 자동으로 생성되며 동기화된 모든 제품에 대해 매일 재생성됩니다. 명령 프롬프트에서 다음 명령을 사용해 부트스트랩 리포지토리를 수동으로 생성할 수 있습니다.

```
mgr-create-bootstrap-repo
```

클라이언트 등록에 대한 자세한 내용은 [Client-configuration > Registration-overview](#)에서 참조하십시오.

4.7.1.6. 정오표 관리



- 이 섹션은 컨테이너화되지 않은 Uyuni에만 적용되며, CentOS 7의 수명 종료 이후에 제거됩니다.

CentOS 클라이언트를 업데이트할 때 패키지에는 업데이트에 대한 메타데이터가 포함되지 않습니다. 타사 정오표 서비스를 사용해 이 정보를 얻을 수 있습니다.



CEFS의 원저자는 도움이 됐으면 하는 마음으로 최선을 다해 패치나 정오표를 제공하지만 정확성이나 최신 상태를 보장하지는 않습니다. 따라서 패치 날짜가 부정확할 수 있습니다. 한번은 게시 날짜가 한 달 이상 늦은 날짜로 표시된 적도 있습니다. 이러한 경우에 대한 자세한 내용은 <https://github.com/stevemeier/cefs/issues/28#issuecomment-656579382> 및 <https://github.com/stevemeier/cefs/issues/28#issuecomment-656573607>을 참조하십시오.

패치 데이터 관련 문제나 지역으로 인해 Uyuni 서버로 신뢰할 수 없는 패치 정보가 임포트될 수 있습니다. 이로 인해 보고서, 감사, CVE 업데이트 또는 기타 패치 관련 정보가 부정확해질 수도 있습니다. 패치 데이터를 독립적으로 확인하거나 보안 관련 요구사항 및 인증 기준에 따라 다른 운영 체제를 선택하는 등 이 서비스 사용을 대체할 수 있는 수단을 고려해 보시기 바랍니다.

절차: 정오표 서비스 설치

- Uyuni 서버의 명령 프롬프트에서 루트 권한으로 `sle-module-development-tools` 모듈을 추가합니다.

```
SUSEConnect --product sle-module-development-tools/15.2/x86_64
```

- 다음과 같이 오류 서비스 종속성을 설치합니다.

```
zypper in perl-Text-Unidecode
```

- `/etc/rhn/rhn.conf`에서 다음 줄을 추가하거나 편집합니다.

```
java.allow_adding_patches_via_api = centos7-x86_64-updates,centos7-x86_64,centos7-x86_64-extras
```

- Tomcat을 다시 시작합니다.

```
systemctl restart tomcat
```

- 다음과 같이 오류 스크립트에 대해 파일을 생성합니다.

```
touch /usr/local/bin/cent-errata.sh
```

- 이 스크립트를 포함할 새 파일을 편집하고, 필요에 따라 리포지토리 상세 정보를 편집합니다. 이 스크립트는 외부 오류 서비스에서 오류 상세 정보를 가져와 압축을 풀고 세부 정보를 게시합니다.

```
#!/bin/bash
mkdir -p /usr/local/centos
cd /usr/local/centos
rm *.xml
wget -c http://cefs.steve-meier.de/errata.latest.xml
wget -c https://www.redhat.com/security/data/oval/v2/RHEL7/rhel-
7.oval.xml.bz2
bzip2 -d rhel-7.oval.xml.bz2
wget -c http://cefs.steve-meier.de/errata-import.tar
tar xvf errata-import.tar
chmod +x /usr/local/centos/errata-import.pl
export SPACEWALK_USER='<adminname>' ; export
SPACEWALK_PASS='<password>'
/usr/local/centos/errata-import.pl --server '<servername>' \
--errata /usr/local/centos/errata.latest.xml \
--include-channels=centos7-x86_64-updates,centos7-x86_64,centos7
-x86_64-extras \
--publish --rhsa-oval /usr/local/centos/rhel-7.oval.xml
```

7. 다음과 같이 cron 작업을 설정하여 매일 스크립트를 실행합니다.

```
ln -s /usr/local/bin/cent-errata.sh /etc/cron.daily
```

이 도구에 대한 자세한 내용은 <https://cefs.steve-meier.de/>를 참조하십시오.

4.8. Debian 클라이언트 등록

Debian 클라이언트를 Uyuni 서버에 등록할 수 있습니다. 방법과 상세 정보는 클라이언트의 운영 체제에 따라 달라집니다.

시작하기 전에 클라이언트의 날짜 및 시간이 Uyuni 서버와 정확히 동기화되었는지 확인하십시오.

또한 활성화 키 생성을 완료한 상태이어야 합니다. 활성화 키 생성에 대한 자세한 내용은 **Client-configuration > Activation-keys**를 참조하십시오.

4.8.1. Debian 클라이언트 등록

이 섹션에는 Debian 운영 체제를 실행하는 클라이언트를 등록하는 방법에 대한 정보가 포함되어 있습니다.

Debian 클라이언트에서는 부트스트래핑을 사용해 초기 상태 실행을 수행하고 프로파일을 업데이트할 수 있습니다.

4.8.1.1. 등록 준비

Debian 클라이언트를 Uyuni 서버에 등록하려면 다음과 같은 약간의 준비가 필요합니다.

- DNS가 올바르게 구성되어 있는지 확인하고 클라이언트에 항목을 제공합니다. 또는 Uyuni 서버와 클라이언트에서 적절한 항목으로 /etc/hosts 파일을 구성할 수 있습니다.
- 클라이언트의 날짜 및 시간을 Uyuni 서버와 동기화한 후 등록해야 합니다.

4.8.1.2. 소프트웨어 채널 추가

Debian 클라이언트를 Uyuni 서버에 등록하려면 먼저 필요한 소프트웨어 채널을 추가하고 이 채널을 동기화해야 합니다.



다음 섹션에서 설명은 종종 x86_64 아키텍처로 기본 설정됩니다. 해당될 경우 다른 아키텍처로 교체하십시오.

이 절차에 필요한 채널은 다음과 같습니다.

표 33. Debian 채널 - CLI

OS 버전	기본 채널	클라이언트 채널	업데이트 채널	보안 채널
Debian 12	debian-12-pool-amd64-uyuni	debian-12-amd64-uyuni-client	debian-12-amd64-main-updates-uyuni	debian-12-amd64-main-security-uyuni
Debian 11	debian-11-pool-amd64-uyuni	debian-11-amd64-uyuni-client	debian-11-amd64-main-updates-uyuni	debian-11-amd64-main-security-uyuni

명령 프롬프트에서 소프트웨어 채널 추가

1. Uyuni 서버의 명령 프롬프트에서 루트로 spacewalk-common-channels 명령을 사용하여 적절한 채널을 추가합니다.

```
spacewalk-common-channels \
<base_channel_label> \
<child_channel_label_1> \
<child_channel_label_2> \
... <child_channel_label_n>
```

2. **automatic synchronization**이 꺼진 경우 다음 채널을 동기화하십시오.

```
spacewalk-repo-sync -p <base_channel_label>
```

3. 계속하기 전에 동기화가 완료되었는지 확인합니다.

4.8.1.3. 동기화 상태 확인

절차: Web UI에서 동기화 진행 상태 확인

1. Uyuni Web UI에서 **소프트웨어** > **관리** > **채널**로 이동한 다음, 리포지토리에 연결된 채널을 클릭합니다.
 2. **-----** 템으로 이동한 다음, **-----**를 클릭하고 **-----** **-----**를 확인합니다.

절차: 명령 프롬프트에서 동기화 진행 상태 확인

1. Uyuni 서버의 명령 프롬프트에서 루트 권한으로 다음과 같이 `tail` 명령을 사용해 동기화 로그 파일을 확인합니다.

```
tail -f /var/log/rhn/reposync/<channel-label>.log
```

2. 각 하위 채널은 동기화가 진행되는 동안 자체 로그를 생성합니다. 동기화가 완료되었는지 알아보려면 모든 기본 및 하위 채널 로그 파일을 확인해야 합니다.



Debian 채널은 규모가 매우 클 수 있습니다. 때로 동기화에 몇 시간이 걸릴 수 있습니다.

4.8.1.4. GPG 키 관리

클라이언트는 소프트웨어 패키지가 설치되기 전에 GPG 키를 사용해 신뢰성을 확인합니다. 신뢰할 수 있는 소프트웨어만 클라이언트에 설치할 수 있습니다.



클라이언트의 보안을 위해 GPG 키를 신뢰하는 것이 중요합니다. 어떤 키가 필요하고 신뢰할 수 있는지 결정하는 것은 관리자가 수행해야 하는 작업입니다. GPG 키를 신뢰할 수 없는 경우 소프트웨어 채널을 사용할 수 없기 때문에 채널을 클라이언트에 할당하는 결정은 키를 신뢰하는 결정에 따라 달라집니다.

GPG 키에 대한 자세한 내용은 [Client-configuration > Gpg-keys](#)에서 확인할 수 있습니다.



Debian 클라이언트를 설치하려면 여러 개의 GPG 키가 필요할 수 있습니다.

타사 Debian 리포지토리를 동기화하는 경우 서버에서 적절한 GPG 키를 가져와야 합니다. GPG 키가 누락되면 동기화에 실패합니다.

Debian 리포지토리의 경우, 메타데이터만 서명됩니다. 그러므로 소프트웨어 채널용 GPG 키를 가져올 필요가 없습니다.
Uyuni는 패키지에 다시 서명하지 않습니다.

Uyuni 서버로 이미 임포트한 GPG 키를 살펴보려면 다음 명령을 실행합니다.

```
mgrctl exec -- gpg --homedir /var/lib/spacewalk/gpgdir --list-keys
```

새 GPG 키를 임포트하려면 다음 명령을 실행합니다.

```
mgradm gpg add <filename>.gpg
```

4.8.1.5. 루트 액세스

Debian의 루트 사용자는 SSH 액세스에 대해 기본적으로 비활성화되어 있습니다.

일반 사용자를 사용하여 온보딩하려면 sudoers 파일을 편집해야 합니다.

절차: 루트 사용자에게 액세스 권한 부여

1. 클라이언트에서 다음과 같이 sudoers 파일을 편집합니다.

```
sudo visudo
```

이 줄을 sudoers 파일 끝에 추가하여 사용자에게 sudo 액세스 권한을 부여합니다. 다음과 같이 <user>를 Web UI에서 클라이언트를 부트스트래핑하고 있는 사용자의 이름으로 교체합니다.

```
<user>  ALL=NOPASSWD: /usr/bin/python, /usr/bin/python2,  
/usr/bin/python3, /var/tmp/venv-salt-minion/bin/python
```



이 절차는 클라이언트 등록에 필요한 비밀번호가 없어도 루트에 액세스 권한을 부여합니다. 클라이언트는 성공적으로 설치된 후 루트 권한으로 실행되므로 액세스 권한이 더 이상 필요 없습니다. 클라이언트가 성공적으로 설치된 후 sudoers 파일에서 이 줄을 제거하는 것이 좋습니다.

4.8.1.6. 클라이언트 등록

클라이언트를 등록하려면 부트스트랩 리포지토리가 필요합니다. 기본적으로 부트스트랩 리포지토리는 자동으로 생성되며 동기화된 모든 제품에 대해 매일 재생성됩니다. 명령 프롬프트에서 다음 명령을 사용해 부트스트랩 리포지토리를 수동으로 생성할 수 있습니다.

```
mgr-create-bootstrap-repo
```

클라이언트 등록에 대한 자세한 내용은 [Client-configuration > Registration-overview](#)에서 참조하십시오.

4.9. OpenEuler 클라이언트 등록

openEuler 클라이언트를 Uyuni 서버에 등록할 수 있습니다. 방법과 자세한 내용은 클라이언트의 운영 체제에 따라 달라집니다.

시작하기 전에 클라이언트의 날짜 및 시간이 Uyuni 서버와 정확히 동기화되었는지 확인하십시오.

또한 활성화 키 생성을 완료한 상태이어야 합니다. 활성화 키 생성에 대한 자세한 내용은 [Client-configuration >](#)

Activation-keys를 참조하십시오.

4.9.1. openEuler 클라이언트 등록

이 섹션에는 openEuler 운영 체제를 실행하는 클라이언트를 등록하는 방법에 대한 정보가 포함되어 있습니다.



AWS에서 생성되었을 때 openEuler 인스턴스는 항상 `/etc/machine-id` 파일에 `machine-id` ID를 갖습니다. 인스턴스가 생성된 후 `machine-id`를 다시 생성해야 합니다. 자세한 내용은 **Administration > Troubleshooting**에서 확인할 수 있습니다.

4.9.1.1. 소프트웨어 채널 추가

Uyuni 서버에 openEuler 클라이언트를 등록하기 전에 요구사항이 있는 소프트웨어 채널을 추가하고 동기화해야 합니다.

현재 지원되는 아키텍처는 x86_64 및 aarch64입니다. 지원되는 제품 및 아키텍처의 전체 목록은 **Client-configuration > Supported-features**에서 참조하십시오.



다음 섹션에서 설명은 종종 x86_64 아키텍처로 기본 설정됩니다. 해당될 경우 다른 아키텍처로 교체하십시오.

이 절차에 필요한 채널은 다음과 같습니다.

표 34. openEuler Channels - CLI

OS 버전	코어 채널	클라이언트 채널	업데이트 채널	EPOL 채널	Everything 채널
openEuler 22.03	openeuler2203	openeuler2203-uyuni-client	openeuler2203-update	openeuler2203-epol	openeuler2203-everything

명령 프롬프트에서 소프트웨어 채널 추가

1. Uyuni 서버의 명령 프롬프트에서 루트로 `spacewalk-common-channels` 명령을 사용하여 적절한 채널을 추가합니다.

```
spacewalk-common-channels \
<base_channel_label> \
<child_channel_label_1> \
<child_channel_label_2> \
... <child_channel_label_n>
```

2. `automatic synchronization`이 꺼진 경우 다음 채널을 동기화하십시오.

```
spacewalk-repo-sync -p <base_channel_label>
```

3. 계속하기 전에 동기화가 완료되었는지 확인합니다.



spacewalk-common-channels가 제공하는 클라이언트 도구 채널은 SUSE가 아닌 Uyuni가 공급합니다.

4.9.1.2. 동기화 상태 확인

절차: Web UI에서 동기화 진행 상태 확인

1. Uyuni Web UI에서 **소프트웨어 > 관리 > 채널**로 이동한 다음, 리포지토리에 연결된 채널을 클릭합니다.
2. ←→ 템으로 이동한 다음, ←→를 클릭하고 ←→ ←→를 확인합니다.

절차: 명령 프롬프트에서 동기화 진행 상태 확인

1. Uyuni 서버의 명령 프롬프트에서 루트 권한으로 다음과 같이 `tail` 명령을 사용해 동기화 로그 파일을 확인합니다.

```
tail -f /var/log/rhn/reposync/<channel-label>.log
```

2. 각 하위 채널은 동기화가 진행되는 동안 자체 로그를 생성합니다. 동기화가 완료되었는지 알아보려면 모든 기본 및 하위 채널 로그 파일을 확인해야 합니다.

4.9.1.3. 활성화 키 생성

openEuler 채널에 연결된 활성화 키를 생성해야 합니다.

활성화 키에 대한 자세한 내용은 **Client-configuration > Activation-keys**를 참조하십시오.

4.9.1.4. 클라이언트의 GPG 키 신뢰

4.9.1.5. 클라이언트의 GPG 키 신뢰

운영 체제는 자체 GPG 키를 직접 신뢰하거나 최소 시스템 이상과 함께 설치된 상태로 제공됩니다. 그러나 다른 GPG 키로 서명된 타사 패키지는 수동으로 처리해야 합니다. 클라이언트는 신뢰할 수 있는 GPG 키가 없어도 성공적으로 부트스트래핑할 수 있습니다. 하지만 키를 신뢰할 수 있을 때까지는 새 클라이언트 도구 패키지를 설치하거나 업데이트할 수 없습니다.

이제 클라이언트는 소프트웨어 채널에 대해 입력한 GPG 키 정보를 사용하여 신뢰된 키를 관리합니다. GPG 키 정보가 있는 소프트웨어 채널이 클라이언트에 할당되면 채널을 새로 고치거나 이 채널에서 첫 번째 패키지를 설치할 때 키가 신뢰됩니다.

소프트웨어 채널 페이지의 GPG 키 URL 파라미터에는 "공백"으로 구분된 여러 키 URL이 포함될 수 있습니다. 파일 URL인 경우 클라이언트에 GPG 키 파일을 배포한 후 소프트웨어 채널을 사용해야 합니다.

Red Hat 기반 클라이언트의 클라이언트 도구 채널에 대한 GPG 키는 클라이언트의 `/etc/pki/rpm-gpg/`에 배포되며 파일 URL로 참조할 수 있습니다.

소프트웨어 채널이 클라이언트에 할당된 경우에만 시스템에서 가져와서 신뢰하게 됩니다.



Debian 기반 시스템은 메타데이터에만 서명하므로 단일 채널에 대해 추가 키를 지정할 필요가 없습니다. **Administration > Repo-metadata**에서 "자체 GPG 키 사용"의 설명과 같이 사용자가 자체 GPG 키를 구성하여 메타데이터에 서명하는 경우 해당 키의 배포 및 신뢰가 자동으로 실행됩니다.

4.9.1.5.1. 사용자 정의 GPG 키

사용자는 클라이언트에 배포할 사용자 정의 GPG 키를 정의할 수 있습니다.

일부 열 데이터를 제공하고 Salt 파일 시스템에 GPG 키 파일을 제공하면 클라이언트에 자동으로 배포됩니다.

이러한 키는 RPM 기반 운영 체제의 `/etc/pki/rpm-gpg/` 및 Debian 시스템의 `/usr/share/keyrings/`에 배포됩니다.

키를 배포할 클라이언트에 대한 열 키 `custom_gpgkeys`를 정의하고 키 파일의 이름을 나열합니다.

```
cat /srv/pillar/mypillar.sls
custom_gpgkeys:
  - my_first_gpg.key
  - my_second_gpgkey.gpg
```

추가적으로 Salt 파일 시스템에서 `gpg` 디렉토리를 생성한 후 해당 디렉토리에 `custom_gpgkeys` 열 데이터에 지정된 이름으로 GPG 키 파일을 저장합니다.

```
ls -la /srv/salt/gpg/
/srv/salt/gpg/my_first_gpg.key
/srv/salt/gpg/my_second_gpgkey.gpg
```

키는 클라이언트에 `/etc/pki/rpm-gpg/my_first_gpg.key` 및 `/etc/pki/rpm-gpg/my_second_gpgkey.gpg`에 배포됩니다.

마지막 단계로 소프트웨어 채널의 GPG 키 URL 필드에 URL을 추가합니다. **소프트웨어 > 관리 > 채널**로 이동하고 수정할 채널을 선택합니다. GPG → URL에 `file:///etc/pki/rpm-gpg/my_first_gpg.key` 값을 추가합니다.

4.9.1.5.2. 부트스트랩 스크립트의 GPG 키

절차: 부트스트랩 스크립트를 사용하여 클라이언트의 GPG 키 신뢰

- Uyuni 서버의 명령 프롬프트에서 `/srv/www/htdocs/pub/` 디렉토리의 내용을 확인합니다. 이 디렉토리에는 사용 가능한 모든 공용 키가 포함되어 있습니다. 등록하려는 클라이언트에 할당된 채널에 적용되는 키를 적어 두십시오.
- 관련 부트스트랩 스크립트를 열어 `ORG_GPG_KEY=` 파라미터를 찾고 필요한 키를 추가합니다. 예:

```
uyuni-gpg-pubkey-0d20833e.key
```

이전에 저장한 키는 삭제하지 않아도 됩니다.



클라이언트의 보안을 위해 GPG 키의 신뢰성이 중요합니다. 어떤 키가 필요하고 신뢰할 수 있는지 결정하는 것은 관리자가 수행해야 하는 작업입니다. GPG 키를 신뢰할 수 없으면 클라이언트에 소프트웨어 채널을 할당할 수 없습니다.

4.9.1.6. 클라이언트 등록

openEuler 클라이언트는 기타 모든 클라이언트와 동일한 방식으로 등록됩니다. 자세한 내용은 **Client-configuration > Registration-overview**에서 확인할 수 있습니다.

4.10. Oracle 클라이언트 등록

Oracle Linux 클라이언트를 Uyuni 서버에 등록할 수 있습니다. 방법과 상세 정보는 클라이언트의 운영 체제에 따라 달라집니다.

시작하기 전에 클라이언트의 날짜 및 시간이 Uyuni 서버와 정확히 동기화되었는지 확인하십시오.

활성화 키도 생성해야 합니다.

- 활성화 키 만들기에 대한 자세한 내용은 **Client-configuration > Activation-keys**에서 확인할 수 있습니다.
- Oracle Linux에서 SUSE Liberty Linux로 마이그레이션에 대한 자세한 내용은 [client-configuration:clients-sleses.pdf](#)에서 확인할 수 있습니다.

4.10.1. Oracle Linux 클라이언트 등록

이 섹션에는 Oracle Linux 운영 체제를 실행하는 클라이언트를 등록하는 방법에 대한 정보가 포함되어 있습니다.



ULN(Unbreakable Linux Network) 리포지토리와 Uyuni의 직접 동기화 기능은 현재 지원되지 않습니다. ULN용 Oracle Local Distribution을 사용해야 합니다. 로컬 ULN 미러 설정에 대한 자세한 내용은 <https://docs.oracle.com/en/operating-systems/oracle-linux/software-management/sfw-mgmt-UseSoftwareDistributionMirrors.html#local-uln-mirror>에서 제공되는 Oracle 설명서를 참조하십시오.

4.10.1.1. 소프트웨어 채널 추가

Oracle Linux 클라이언트를 Uyuni 서버에 등록하기 전에 필요한 소프트웨어 채널을 추가하고 이 채널을 동기화해야 합니다.

현재 지원되는 아키텍처는 x86_64 및 aarch64입니다. 지원되는 제품 및 아키텍처의 전체 목록은 **Client-configuration > Supported-features**에서 참조하십시오.



다음 섹션에서 설명은 종종 x86_64 아키텍처로 기본 설정됩니다. 해당될 경우 다른 아키텍처로 교체하십시오.

이 절차에 필요한 채널은 다음과 같습니다.

표 35. Oracle 채널 - CLI

OS 버전	기본 채널	클라이언트 채널	업데이트 채널
Oracle Linux 9	oraclelinux9	oraclelinux9-uyuni-client	oraclelinux9-appstream
Oracle Linux 8	oraclelinux8	oraclelinux8-uyuni-client	oraclelinux8-appstream
Oracle Linux 7	oraclelinux7	oraclelinux7-uyuni-client	-

명령 프롬프트에서 소프트웨어 채널 추가

1. Uyuni 서버의 명령 프롬프트에서 루트로 `spacewalk-common-channels` 명령을 사용하여 적절한 채널을 추가합니다.

```
spacewalk-common-channels \
<base_channel_label> \
<child_channel_label_1> \
<child_channel_label_2> \
... <child_channel_label_n>
```

2. `automatic synchronization`이 꺼진 경우 다음 채널을 동기화하십시오.

```
spacewalk-repo-sync -p <base_channel_label>
```

3. 계속하기 전에 동기화가 완료되었는지 확인합니다.



spacewalk-common-channels가 제공하는 클라이언트 도구 채널은 SUSE가 아닌 Uyuni가 공급합니다.



Oracle Linux 9 및 Oracle Linux 8 클라이언트의 경우 기본 채널과 AppStream 채널을 모두 추가하십시오. 사용자에게는 두 채널의 패키지가 모두 필요합니다. 두 채널을 모두 추가하지 않으면 패키지가 누락되어 부트스트랩 리포지토리를 생성할 수 없습니다.

모듈형 채널을 사용하는 경우 클라이언트에서 Python 3.6 모듈 스트림을 활성화해야 합니다. Python 3.6이 제공되지 않으면 `spacecmd` 패키지의 설치가 실패합니다.

4.10.1.2. 동기화 상태 확인

절차: Web UI에서 동기화 진행 상태 확인

1. Uyuni Web UI에서 **소프트웨어** > **관리** > **채널**로 이동한 다음, 리포지토리에 연결된 채널을 클릭합니다.
2. **탭으로 이동** 템으로 이동한 다음, **탭**을 클릭하고 **선택** > **확인**을 확인합니다.

절차: 명령 프롬프트에서 동기화 진행 상태 확인

- Uyuni 서버의 명령 프롬프트에서 루트 권한으로 다음과 같이 `tail` 명령을 사용해 동기화 로그 파일을 확인합니다.

```
tail -f /var/log/rhn/reposync/<channel-label>.log
```

- 각 하위 채널은 동기화가 진행되는 동안 자체 로그를 생성합니다. 동기화가 완료되었는지 알아보려면 모든 기본 및 하위 채널 로그 파일을 확인해야 합니다.

4.10.1.3. 활성화 키 생성

Oracle Linux 채널에 연결된 활성화 키를 생성해야 합니다.

활성화 키에 대한 자세한 내용은 [Client-configuration > Activation-keys](#)를 참조하십시오.

4.10.1.4. GPG 키 관리

클라이언트는 소프트웨어 패키지가 설치되기 전에 GPG 키를 사용해 신뢰성을 확인합니다. 신뢰할 수 있는 소프트웨어만 클라이언트에 설치할 수 있습니다.



클라이언트의 보안을 위해 GPG 키를 신뢰하는 것이 중요합니다. 어떤 키가 필요하고 신뢰할 수 있는지 결정하는 것은 관리자가 수행해야 하는 작업입니다. GPG 키를 신뢰할 수 없는 경우 소프트웨어 채널을 사용할 수 없기 때문에 채널을 클라이언트에 할당하는 결정은 키를 신뢰하는 결정에 따라 달라집니다.

GPG 키에 대한 자세한 내용은 [Client-configuration > Gpg-keys](#)에서 확인할 수 있습니다.



Oracle Linux 9 및 Oracle Linux 8 클라이언트의 경우

```
ol8-gpg-pubkey-82562EA9AD986DA3.key
```

Oracle Linux 7 클라이언트의 경우

```
ol67-gpg-pubkey-72F97B74EC551F0A3.key
```

4.10.1.5. 클라이언트 등록

클라이언트를 등록하려면 부트스트랩 리포지토리가 필요합니다. 기본적으로 부트스트랩 리포지토리는 자동으로 생성되며 동기화된 모든 제품에 대해 매일 재생성됩니다. 명령 프롬프트에서 다음 명령을 사용해 부트스트랩 리포지토리를 수동으로 생성할 수 있습니다.

```
mgr-create-bootstrap-repo
```

클라이언트 등록에 대한 자세한 내용은 [Client-configuration > Registration-overview](#)에서 참조하십시오.

4.11. Raspberry Pi OS 클라이언트 등록

Raspberry Pi OS 클라이언트를 Uyuni 서버에 등록할 수 있습니다. 방법과 자세한 내용은 클라이언트의 운영 체제에 따라 다릅니다.

시작하기 전에 클라이언트의 날짜 및 시간이 Uyuni 서버와 정확히 동기화되었는지 확인하십시오.

또한 활성화 키 생성을 완료한 상태이어야 합니다. 활성화 키 생성에 대한 자세한 내용은 [Client-configuration > Activation-keys](#)를 참조하십시오.

4.11.1. Raspberry Pi OS 클라이언트 등록

이 섹션에는 Raspberry Pi OS 운영 체제를 실행하는 클라이언트를 등록하는 방법에 대한 정보가 포함되어 있습니다.

Raspberry Pi OS 클라이언트에서는 부트스트래핑을 사용해 초기 상태 실행을 수행하고 프로파일을 업데이트할 수 있습니다.

4.11.1.1. 등록 준비

Raspberry Pi OS 클라이언트를 Uyuni 서버에 등록하려면 다음과 같은 약간의 준비가 필요합니다.

- DNS가 올바르게 구성되어 있는지 확인하고 클라이언트에 항목을 제공합니다. 또는 Uyuni 서버와 클라이언트에서 적절한 항목으로 /etc/hosts 파일을 구성할 수 있습니다.
- 클라이언트의 날짜 및 시간을 Uyuni 서버와 동기화한 후 등록해야 합니다.

4.11.1.2. 소프트웨어 채널 추가

Raspberry Pi OS 클라이언트를 Uyuni 서버에 등록하기 전에 필요한 소프트웨어 채널을 추가하고 이 채널을 동기화해야 합니다.

현재 지원되는 아키텍처는 arm64 및 armhf입니다. 지원되는 제품과 아키텍처의 전체 목록은 [Client-configuration > Supported-features](#)에서 확인할 수 있습니다.

이 절차에 필요한 채널은 다음과 같습니다.

표 36. Raspberry Pi OS 채널 - CLI

채널 설명	arm64	armhf
기본 채널	raspberrypios-12-pool-arm64-uyuni	raspberrypios-12-pool-armhf-uyuni
클라이언트 채널	raspberrypios-12-arm64-uyuni-client	raspberrypios-12-armhf-uyuni-client
업데이트 채널	raspberrypios-12-arm64-main-updates-uyuni	-

채널 설명	arm64	armhf
기여 채널	raspberrypios-12-arm64-contrib-uyuni	raspberrypios-12-armhf-contrib-uyuni
비자유 채널	raspberrypios-12-arm64-non-free-uyuni	raspberrypios-12-armhf-non-free-uyuni
비자유 펌웨어 채널	raspberrypios-12-arm64-non-free-firmware-uyuni	-
Raspberry 채널	raspberrypios-12-arm64-raspberry-uyuni	raspberrypios-12-armhf-raspberry-uyuni
기여 업데이트	raspberrypios-12-arm64-contrib-updates-uyuni	-
비자유 업데이트	raspberrypios-12-arm64-non-free-updates-uyuni	-
비자유 펌웨어 업데이트	raspberrypios-12-arm64-non-free-firmware-updates-uyuni	-
보안 기본 채널	raspberrypios-12-arm64-main-security-uyuni	-
보안 기여 채널	raspberrypios-12-arm64-contrib-security-uyuni	-
보안 비자유 채널	raspberrypios-12-arm64-non-free-security-uyuni	-
보안 비자유 펌웨어 채널	raspberrypios-12-arm64-non-free-firmware-security-uyuni	-
RPI 채널	-	raspberrypios-12-armhf-rpi-uyuni

명령 프롬프트에서 소프트웨어 채널 추가

1. Uyuni 서버의 명령 프롬프트에서 루트로 `spacewalk-common-channels` 명령을 사용하여 적절한 채널을 추가합니다.

```
spacewalk-common-channels \
<base_channel_label> \
<child_channel_label_1> \
<child_channel_label_2> \
... <child_channel_label_n>
```

2. `automatic synchronization`이 꺼진 경우 다음 채널을 동기화하십시오.

```
spacewalk-repo-sync -p <base_channel_label>
```

3. 계속하기 전에 동기화가 완료되었는지 확인합니다.



spacewalk-common-channels가 제공하는 클라이언트 도구 채널은 SUSE가 아닌 Uyuni가 공급합니다.



새 채널을 완전히 동기화한 후에 모든 Raspberry Pi OS 클라이언트를 부트스트랩해야 합니다.

4.11.1.3. 동기화 상태 확인

절차: Web UI에서 동기화 진행 상태 확인

1. Uyuni Web UI에서 **소프트웨어** > **관리** > **채널**로 이동한 다음, 리포지토리에 연결된 채널을 클릭합니다.
2. ←→ 템으로 이동한 다음, ←→를 클릭하고 ←→ ←→를 확인합니다.

절차: 명령 프롬프트에서 동기화 진행 상태 확인

1. Uyuni 서버의 명령 프롬프트에서 루트 권한으로 다음과 같이 `tail` 명령을 사용해 동기화 로그 파일을 확인합니다.

```
tail -f /var/log/rhn/reposync/<channel-label>.log
```

2. 각 하위 채널은 동기화가 진행되는 동안 자체 로그를 생성합니다. 동기화가 완료되었는지 알아보려면 모든 기본 및 하위 채널 로그 파일을 확인해야 합니다.



Raspberry Pi OS 채널은 규모가 매우 클 수 있습니다. 일부 경우 동기화에 몇 시간이 걸릴 수 있습니다.

4.11.1.4. 활성화 키 생성

Raspberry Pi OS 채널에 연결된 활성화 키를 생성해야 합니다.

활성화 키에 대한 자세한 내용은 **Client-configuration** > **Activation-keys**를 참조하십시오.

4.11.1.5. GPG 키 관리

클라이언트는 소프트웨어 패키지가 설치되기 전에 GPG 키를 사용해 신뢰성을 확인합니다. 신뢰할 수 있는 소프트웨어만 클라이언트에 설치할 수 있습니다.



클라이언트의 보안을 위해 GPG 키를 신뢰하는 것이 중요합니다. 어떤 키가 필요하고 신뢰할 수 있는지 결정하는 것은 관리자가 수행해야 하는 작업입니다. GPG 키를 신뢰할 수 없는 경우 소프트웨어 채널을 사용할 수 없기 때문에 채널을 클라이언트에 할당하는 결정은 키를 신뢰하는 결정에 따라 달라집니다.

GPG 키에 대한 자세한 내용은 **Client-configuration** > **Gpg-keys**에서 확인할 수 있습니다.



Raspberry Pi OS 클라이언트를 설치하려면 여러 개의 GPG 키가 필요할 수 있습니다.

타사 Raspberry Pi OS 리포지토리를 동기화하는 경우 서버에서 적절한 GPG 키를 가져와야 합니다. GPG 키가 누락되면 동기화에 실패합니다.

Raspberry Pi OS 리포지토리의 경우, 메타데이터만 서명됩니다. 그러므로 소프트웨어 채널용 GPG 키를 가져올 필요가 없습니다. 패키지는 Uyuni에 의해 재서명되지 않습니다.

Uyuni 서버로 이미 임포트한 GPG 키를 살펴보려면 다음 명령을 실행합니다.

```
mgrctl exec -- gpg --homedir /var/lib/spacewalk/gpgdir --list-keys
```

새 GPG 키를 임포트하려면 다음 명령을 실행합니다.

```
mgradm gpg add <filename>.gpg
```

4.11.1.6. 루트 액세스

Raspberry Pi OS의 루트 사용자는 SSH 액세스에 대해 기본적으로 비활성화되어 있습니다.

일반 사용자를 사용하여 온보딩하려면 sudoers 파일을 편집해야 합니다.

절차: 루트 사용자에게 액세스 권한 부여

1. 클라이언트에서 다음과 같이 sudoers 파일을 편집합니다.

```
sudo visudo
```

이 줄을 sudoers 파일 끝에 추가하여 사용자에게 sudo 액세스 권한을 부여합니다. 다음과 같이 <user>를 Web UI에서 클라이언트를 부트스트래핑하고 있는 사용자의 이름으로 교체합니다.

```
<user>  ALL=NOPASSWD: /usr/bin/python, /usr/bin/python2,  
/usr/bin/python3, /var/tmp/venv-salt-minion/bin/python
```



- 이 절차는 클라이언트 등록에 필요한 비밀번호가 없어도 루트에 액세스 권한을 부여합니다.
- 클라이언트는 성공적으로 설치된 후 루트 권한으로 실행되므로 액세스 권한이 더 이상 필요 없습니다. 클라이언트가 성공적으로 설치된 후 sudoers 파일에서 이 줄을 제거하는 것이 좋습니다.

4.11.1.7. 클라이언트 등록

클라이언트를 등록하려면 부트스트랩 리포지토리가 필요합니다. 기본적으로 부트스트랩 리포지토리는 자동으로 생성되며 동기화된 모든 제품에 대해 매일 재생성됩니다. 명령 프롬프트에서 다음 명령을 사용해 부트스트랩 리포지토리를 수동으로 생성할 수 있습니다.

```
mgr-create-bootstrap-repo
```

클라이언트 등록에 대한 자세한 내용은 [Client-configuration > Registration-overview](#)에서 참조하십시오.

4.12. Red Hat 클라이언트 등록

Red Hat 컨텐트 제공 네트워크(CDN) 또는 Red Hat 업데이트 인프라(RHUI)를 사용해 Uyuni 서버에 Red Hat Enterprise Linux 클라이언트를 등록할 수 있습니다. 방법과 상세 정보는 클라이언트의 운영 체제에 따라 달라집니다.

시작하기 전에 클라이언트의 날짜 및 시간이 Uyuni 서버와 정확히 동기화되었는지 확인하십시오.

활성화 키도 생성해야 합니다.

- 활성화 키 만들기에 대한 자세한 내용은 [Client-configuration > Activation-keys](#)에서 확인할 수 있습니다.
- Red Hat Enterprise Linux에서 SUSE Liberty Linux로 마이그레이션에 대한 자세한 내용은 [client-configuration:clients-sleses.pdf](#)에서 확인할 수 있습니다.

4.12.1. Red Hat Enterprise Linux 클라이언트를 CDN으로 등록

이 섹션에는 Red Hat Enterprise Linux 운영 체제를 실행하는 클라이언트를 등록하기 위해 Red Hat CDN(콘텐츠 전송 네트워크)을 사용하는 방법에 대한 정보가 포함되어 있습니다.

대신에 Red Hat 업데이트 인프라(RHUI) 사용법에 대한 정보는 [Client-configuration > Clients-rh-rhui](#)를 참조하십시오.



Red Hat 기본 미디어 리포지토리 및 RHEL 설치 미디어에 대한 액세스를 준비하고 Uyuni 서버를 Red Hat 콘텐츠 전송 네트워크에 연결할 책임은 사용자에게 있습니다. 사용자는 모든 RHEL 시스템과 관련하여 Red Hat의 지원을 받아야 합니다. 그렇지 않으면 Red Hat과의 약관을 위반하게 될 수 있습니다.

4.12.1.1. 자격 및 인증서 임포트

Red Hat 클라이언트는 Red Hat 인증 주체(CA)와 자격 인증서 및 자격 키가 있어야 합니다.

자격 인증서에는 지원 구독 기간과 일치하는 만료일이 포함되어 있습니다. 중단을 방지하려면 지원 구독 기간이 종료되는 시점이 될 때마다 이 프로세스를 반복해야 합니다.

Red Hat은 구독 할당을 관리할 수 있도록 구독 관리자 도구를 제공합니다. 이 도구는 로컬로 실행되어 설치된 제품 및 구독을 추적합니다. 클라이언트는 구독 관리자로 등록되어 있어야 인증서를 취득할 수 있습니다.

Red Hat 클라이언트는 URL을 사용해 리포지토리를 복제합니다. URL은 Red Hat 클라이언트가 어디에 등록되었는지에 따라 달라집니다.

Red Hat 클라이언트는 다음 세 가지 방식으로 등록할 수 있습니다.

- redhat.com의 Red Hat 컨텐트 제공 네트워크(CDN)

- Red Hat Satellite 서버
- 클라우드의 Red Hat 업데이트 인프라(RHUI)

이 안내서에서는 Red Hat CDN에 등록된 클라이언트에 대해 설명합니다. CDN에 등록된 시스템이 최소 한 개 있어야 하고, 이와 함께 리포지토리 컨텐트에 대한 인증 구독이 있어야 합니다.

대신에 Red Hat 업데이트 인프라(RHUI) 사용법에 대한 정보는 [Client-configuration > Clients-rh-rhui](#)를 참조하십시오.



클라이언트 시스템에 대한 Satellite 서버는 Satellite 서버 및 구독이 있어야 합니다. Satellite 인증서를 사용하는 클라이언트는 Uyuni 서버에서 지원하지 않습니다.



자격 인증서에는 지원 구독 기간과 일치하는 만료일이 포함되어 있습니다. 중단을 방지하려면 지원 구독 기간이 종료되는 시점이 될 때마다 이 프로세스를 반복해야 합니다.

Red Hat은 구독 할당을 관리할 수 있도록 구독 관리자 도구를 제공합니다. 이 도구는 클라이언트 시스템에서 로컬로 실행되어 설치된 제품 및 구독을 추적합니다. 구독 관리자로 redhat.com에 등록한 후, 이 절차에 따라 인증서를 취득하십시오.

절차: 클라이언트를 구독 관리자에 등록

1. 클라이언트 시스템의 명령 프롬프트에서 다음과 같이 구독 관리자 도구로 등록합니다.

```
subscription-manager register
```

프롬프트가 표시되면 Red Hat 포털 사용자 이름 및 비밀번호를 입력합니다.

2. 명령 실행:

```
subscription-manager activate
```

3. 다음과 같이 클라이언트 시스템에서 Uyuni 서버가 액세스할 수 있는 위치로 자격 인증서 및 키를 복사합니다.

```
cp /etc/pki/entitlement/ /<example>/entitlement/
```



자격 인증서 및 키에는 .pem이라는 파일 확장자가 있습니다. 또한 키는 파일 이름에 key가 포함되어 있습니다.

4. 다음과 같이 클라이언트 시스템에서 자격 인증서 및 키와 동일한 웹 위치로 Red Hat CA 인증서 파일을 복사합니다.

```
cp /etc/rhsm/ca/redhat-uep.pem /<example>/entitlement
```

Red Hat 클라이언트에서 리포지토리를 관리하려면 CA 및 자격 인증서를 Uyuni 서버로 임포트해야 합니다. 이렇게 하려면 임포트 절차를 세 차례 수행하여 자격 인증서, 자격 키, Red Hat 인증서 각각에 대해 하나씩 세 항목을 생성해야 합니다.

절차: 인증서를 서버에 임포트

1. Uyuni 서버 Web UI에서 **시스템 > 자동 설치 > GPG 및 SSL 키**로 이동합니다.
2. **[저장된 키/인증서 생성]**을 클릭하고 자격 인증서에 대해 다음 파라미터를 설정합니다.
 - -- 필드에 Entitlement-Cert-date를 입력합니다.
 - -- 필드에서 SSL을 선택합니다.
 - -- 필드에서 자격 인증서를 저장한 위치로 이동하여 .pem 인증서 파일을 선택합니다.
3. **[키 생성]**을 클릭합니다.
4. **[저장된 키/인증서 생성]**을 클릭하고 자격 키에 대해 다음 파라미터를 설정합니다.
 - -- 필드에 Entitlement-key-date를 입력합니다.
 - -- 필드에서 SSL을 선택합니다.
 - -- 필드에서 자격 키를 저장한 위치로 이동하여 .pem 키 파일을 선택합니다.
5. **[키 생성]**을 클릭합니다.
6. **[저장된 키/인증서 생성]**을 클릭하고 Red Hat 인증서에 대해 다음 파라미터를 설정합니다.
 - -- 필드에 redhat-uep를 입력합니다.
 - -- 필드에서 SSL을 선택합니다.
 - -- 필드에서 Red Hat 인증서를 저장한 위치로 이동하여 인증서 파일을 선택합니다.
7. **[키 생성]**을 클릭합니다.

4.12.1.2. 소프트웨어 채널 추가

Red Hat 클라이언트를 Uyuni 서버에 등록하기 전에 필요한 소프트웨어 채널을 추가하고 이 채널을 동기화해야 합니다.



다음 섹션에서 설명은 종종 x86_64 아키텍처로 기본 설정됩니다. 해당될 경우 다른 아키텍처로 교체하십시오.

이 절차에 필요한 채널은 다음과 같습니다.

표 37. Red Hat 채널- CLI

OS 버전	기본 채널	클라이언트 채널	도구 채널
Red Hat 9	rhel9-pool-uyuni	-	rhel9-uyuni-client
Red Hat 8	rhel8-pool-uyuni	-	rhel8-uyuni-client
Red Hat 7	rhel7-pool-uyuni	-	rhel7-uyuni-client

명령 프롬프트에서 소프트웨어 채널 추가

- Uyuni 서버의 명령 프롬프트에서 root 권한으로 `spacewalk-common-channels` 명령을 사용해 적절한 채널을 추가합니다. 다음과 같이 올바른 아키텍처를 지정했는지 확인합니다.

```
spacewalk-common-channels \
-a <architecture> \
<base_channel_name> \
<child_channel_name_1> \
<child_channel_name_2> \
... <child_channel_name_n>
```

- `automatic synchronization`이 꺼진 경우 다음 채널을 동기화하십시오.

```
spacewalk-repo-sync -p <base_channel_label>-<architecture>
```

- 계속하기 전에 동기화가 완료되었는지 확인합니다.



spacewalk-common-channels가 제공하는 클라이언트 도구 채널은 SUSE가 아닌 Uyuni가 공급합니다.

4.12.1.3. 사용자 정의 리포지토리 및 채널 준비

Red Hat CDN에서 소프트웨어를 미러링하려면 URL로 CDN에 링크된 사용자 정의 채널 및 리포지토리를 Uyuni에 생성해야 합니다. 이 작업을 올바르게 수행하려면 Red Hat 포털에 이러한 제품에 대한 자격이 있어야 합니다. 미러링하려는 리포지토리의 URL은 다음과 같이 구독 관리자 도구를 사용해 가져올 수 있습니다.

```
subscription-manager repos
```

이 리포지토리 URL을 사용해 사용자 정의 리포지토리를 생성할 수 있습니다. 이렇게 하면 클라이언트를 관리하는 데 필요한 컨텐트만 미러링할 수 있습니다.



Red Hat 포털에 올바른 자격이 있는 경우 Red Hat 리포지토리의 사용자 정의 버전만 생성할 수 있습니다.

이 절차에 필요한 상세 정보는 다음과 같습니다.

표 38. Red Hat 사용자 정의 리포지토리 설정

옵션	설정
리포지토리 URL	Red Hat CDN이 제공하는 컨텐트 URL
서명 메타데이터가 있습니까?	모든 Red Hat Enterprise 리포지토리를 선택 취소
SSL CA 인증서	redhat-uep
SSL 클라이언트 인증서	Entitlement-Cert-date

옵션	설정
SSL 클라이언트 키	Entitlement-Key-date

절차: 사용자 정의 리포지토리 생성

1. Uyuni 서버 Web UI에서 **소프트웨어 > 관리 > 리포지토리**로 이동합니다.
2. **[리포지토리 생성]**을 클릭하여 리포지토리에 적절한 파라미터를 설정합니다.
3. **[리포지토리 생성]**을 클릭합니다.
4. 생성해야 하는 모든 리포지토리에 대해 이를 반복합니다.

이 절차에 필요한 채널은 다음과 같습니다.

표 39. Red Hat 사용자 정의 채널

OS 버전	기본 채널
Red Hat 9	rhel9-pool-uyuni
Red Hat 8	rhel8-pool-uyuni
Red Hat 7	rhel7-pool-uyuni

절차: 사용자 정의 채널 생성

1. Uyuni 서버 Web UI에서 **소프트웨어 > 관리 > 채널**로 이동합니다.
2. **[채널 생성]**을 클릭하여 채널에 적절한 파라미터를 설정합니다.
3. **-- 필드에서 적절한 기본 채널을 선택합니다.**
4. **[채널 생성]**을 클릭합니다.
5. 생성해야 하는 모든 채널에 대해 이를 반복합니다. 각 사용자 정의 리포지토리에는 하나의 사용자 정의 채널이 있어야 합니다.

소프트웨어 > 채널 목록 > 전체로 이동하여 적절한 채널 및 리포지토리를 모두 생성했는지 확인할 수 있습니다.



For Red Hat 9 and Red Hat 8 clients, add both the Base and AppStream channels. You require packages from both channels. If you do not add both channels, you cannot create the bootstrap repository, due to missing packages.

모듈형 채널을 사용하는 경우 클라이언트에서 Python 3.6 모듈 스트림을 활성화해야 합니다. Python 3.6이 제공되지 않으면 spacecmd 패키지의 설치가 실패합니다.

모든 채널을 생성했으면 생성한 리포지토리를 다음과 같이 채널과 연결할 수 있습니다.

절차: 채널을 리포지토리와 연결

1. Uyuni 서버 Web UI에서 **소프트웨어 > 관리 > 채널**로 이동한 다음, 연결할 채널을 클릭합니다.
2. **----- 탭으로 이동하여 이 채널에 연결할 리포지토리의 확인란을 선택합니다.**
3. **[리포지토리 업데이트]**를 클릭하여 채널과 리포지토리를 연결합니다.

4. 연결하려는 모든 채널과 리포지토리에 대해 이를 반복합니다.
5. 옵션: →→ 템으로 이동하여 이 리포지토리의 동기화에 대해 정기적 일정을 설정합니다.
6. [지금 동기화]를 클릭하여 즉시 동기화를 시작합니다.

4.12.1.4. 동기화 상태 확인

절차: Web UI에서 동기화 진행 상태 확인

1. Uyuni Web UI에서 **소프트웨어** > **관리** > **채널**로 이동한 다음, 리포지토리에 연결된 채널을 클릭합니다.
2. →→→→ 템으로 이동한 다음, →→를 클릭하고 →→ →→를 확인합니다.

절차: 명령 프롬프트에서 동기화 진행 상태 확인

1. Uyuni 서버의 명령 프롬프트에서 루트 권한으로 다음과 같이 tail 명령을 사용해 동기화 로그 파일을 확인합니다.

```
tail -f /var/log/rhn/reposync/<channel-label>.log
```

2. 각 하위 채널은 동기화가 진행되는 동안 자체 로그를 생성합니다. 동기화가 완료되었는지 알아보려면 모든 기본 및 하위 채널 로그 파일을 확인해야 합니다.



Red Hat Enterprise Linux 채널은 규모가 매우 클 수 있습니다. 일부 경우 동기화에 몇 시간이 걸릴 수 있습니다.

절차: 옵션: Salt 상태를 생성하여 구성 파일 배포

1. Uyuni 서버 Web UI에서 **구성** > **채널**로 이동합니다.
2. [상태 채널 생성]을 클릭합니다.
 - → 필드에 subscription-manager: disable yum plugins를 입력합니다.
 - →→ 필드에 subscription-manager-disable-yum-plugins를 입력합니다.
 - → 필드에 subscription-manager: disable yum plugins를 입력합니다.
 - SLS →→ 필드를 빈 상태로 둡니다.
3. [구성 채널 생성]을 클릭합니다.
4. [구성 파일 생성]을 클릭합니다.
 - →→ → 필드에 /etc/yum/pluginconf.d/subscription-manager.conf를 입력합니다.
 - →→ 필드에 다음과 같이 입력합니다.

```
[main]
enabled=0
```

5. [구성 파일 생성]을 클릭합니다.

6. Salt :: :: 필드의 값을 적어 둡니다.
7. 구성 채널의 이름을 클릭합니다.
8. 'init.sls' :: :: 을 클릭합니다.
 - :: 필드에 다음과 같이 입력합니다.

```
configure_subscription-manager-disable-yum-plugins:
cmd.run:
  - name: subscription-manager config
--rhsm.auto_enable_yum_plugins=0
  - watch:
    - file: /etc/yum/pluginconf.d/subscription-manager.conf
file.managed:
  - name: /etc/yum/pluginconf.d/subscription-manager.conf
  - source: salt:///etc/yum/pluginconf.d/subscription-
manager.conf
```

9. [구성 파일 업데이트]를 클릭합니다.



Salt :: 절차는 선택 사항입니다.

절차: Red Hat Enterprise Linux 클라이언트에 대해 시스템 그룹 생성

1. Uyuni 서버 Web UI에서 **시스템** > **시스템 그룹**으로 이동합니다.
2. [그룹 생성]을 클릭합니다.
 - :: 필드에 rhel-systems라고 입력합니다.
 - :: 필드에 :: RHEL ::이라고 입력합니다.
3. [그룹 생성]을 클릭합니다.
4. :: 탭을 클릭합니다.
5. :: :: 탭을 클릭합니다.
6. 검색 상자에 subscription-manager: disable yum plugins라고 입력합니다.
7. [검색]을 클릭하여 상태를 확인합니다.
8. :: 열에서 상태의 확인란을 클릭합니다.
9. [변경 사항 저장]을 클릭합니다.
10. [확인]을 클릭합니다.

RHEL 시스템을 Uyuni에 이미 추가했다면 새 시스템 그룹에 할당하고 Highstat를 적용합니다.

절차: 시스템 그룹을 활성화 키에 추가

위에서 생성한 시스템 그룹을 포함하려면 RHEL 시스템에 사용한 활성화 키를 수정해야 합니다.

1. Uyuni 서버 Web UI에서 시스템 > 활성화 키로 이동합니다.
2. RHEL 시스템에 사용한 각 활성화 키를 클릭하고 다음과 같이 합니다.
3. ⇩ 탭의 ⇩ 하위 탭으로 이동합니다.
4. rhel-systems ⇩의 확인란을 선택합니다.
5. [선택한 그룹 조인]을 클릭합니다.

4.12.1.5. GPG 키 관리

클라이언트는 소프트웨어 패키지가 설치되기 전에 GPG 키를 사용해 신뢰성을 확인합니다. 신뢰할 수 있는 소프트웨어만 클라이언트에 설치할 수 있습니다.



클라이언트의 보안을 위해 GPG 키를 신뢰하는 것이 중요합니다. 어떤 키가 필요하고 신뢰할 수 있는지 결정하는 것은 관리자가 수행해야 하는 작업입니다. GPG 키를 신뢰할 수 없는 경우 소프트웨어 채널을 사용할 수 없기 때문에 채널을 클라이언트에 할당하는 결정은 키를 신뢰하는 결정에 따라 달라집니다.

GPG 키에 대한 자세한 내용은 [Client-configuration > Gpg-keys](#)에서 확인할 수 있습니다.



Red Hat 사용자 정의 채널의 경우, GPG ⇩ ⇩ 필드를 확인하려면 GPG ⇩ URL 필드를 채워야 합니다. Red Hat 미니언의 /etc/pki/rpm-gpg 디렉토리에 저장된 GPG 키의 파일 URL을 사용할 수 있습니다.

예: `file:///etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release`

Red Hat 제품 서명 키의 전체 목록은 <https://access.redhat.com/security/team/key>를 참조하십시오.

4.12.1.6. 클라이언트 등록

클라이언트를 등록하려면 부트스트랩 리포지토리가 필요합니다. 기본적으로 부트스트랩 리포지토리는 자동으로 생성되며 동기화된 모든 제품에 대해 매일 재생성됩니다. 명령 프롬프트에서 다음 명령을 사용해 부트스트랩 리포지토리를 수동으로 생성할 수 있습니다.

```
mgr-create-bootstrap-repo
```

클라이언트 등록에 대한 자세한 내용은 [Client-configuration > Registration-overview](#)에서 참조하십시오.

4.12.2. RHUI를 사용하여 Red Hat Enterprise Linux 클라이언트 등록

이 섹션에는 Red Hat Enterprise Linux 운영 체제를 실행하는 클라이언트를 등록하기 위해 Red Hat RHUI(업데이트 인프라)를 사용하는 방법에 대한 정보가 포함되어 있습니다.

Amazon EC2와 같은 공용 클라우드에서 클라이언트를 실행하는 경우 이 방법을 사용합니다.

RHUI를 Red Hat 컨텐트 제공 네트워크(CDN)와 함께 사용하여 Red Hat Enterprise Linux 구독을 관리할 수

있습니다. Red Hat CDN 사용에 대한 정보는 [Client-configuration > Clients-rh-cdn](#)을 참조하십시오.



사용자는 Uyuni 서버를 Red Hat 업데이트 인프라에 연결할 책임이 있습니다. 이 RHUI 인증서를 사용해 업데이트를 가져오는 모든 클라이언트는 올바른 라이선스를 보유해야 합니다. 자세한 내용은 해당 클라우드 공급자에게 문의하거나 Red Hat 서비스 약관을 확인하시기 바랍니다.



RHUI로 등록한 Red Hat Enterprise Linux 클라이언트가 켜져 있지 않은 경우 Red Hat은 인증서가 잘못되었다고 선언할 수 있습니다. 이 경우 클라이언트를 다시 켜거나 새 RHUI 인증서를 가져와야 합니다.

4.12.2.1. 자격 및 인증서 임포트

이전에는 인증서 및 자격 데이터 설명서를 Uyuni 서버로 임포트해야 했습니다. 이 작업은 SUSE PAYG 인스턴스와 동일한 메커니즘을 사용하여 자동화되었습니다. 자세한 내용은 [Installation-and-upgrade > Connect-payg](#)에서 확인할 수 있습니다.

이 안내서에서는 Red Hat 업데이트 인프라(RHUI)에 등록된 클라이언트에 대해 설명합니다. RHUI에 등록된 시스템이 최소 한 개 있어야 하고, 이와 함께 리포지토리 컨텐트에 대한 인증 구독이 있어야 합니다.

Red Hat 컨텐트 제공 네트워크(CDN)를 사용하는 방법에 대한 정보 대신에 [Client-configuration > Clients-rh-cdn](#)을 참조하십시오.



클라이언트 시스템에 대한 Satellite 서버는 Satellite 서버 및 구독이 있어야 합니다. Satellite 인증서를 사용하는 클라이언트는 Uyuni 서버에서 지원하지 않습니다.



PAYG 연결은 클라이언트를 정기적으로 확인하여 최신 인증 데이터를 가져옵니다. 클라이언트가 계속 실행 중이고 정기적으로 업데이트하는 것이 중요합니다. 그러지 않으면 특정 시점에 인증 오류가 발생하며 리포지토리 동기화가 실패합니다.



Red Hat 7 인스턴스를 업데이트한 후 연결하십시오.



Red Hat 9 인스턴스를 연결하려면 LEGACY 암호화 정책을 사용하여 구성해야 합니다. `sudo update-crypto-policies --set LEGACY`를 실행하여 적절하게 구성하십시오.

4.12.2.2. Red Hat 업데이트 인프라에 연결

절차: 새 Red Hat 인스턴스 연결

1. Uyuni Web UI에서 관리 > 설치 마법사 > PAYG으로 이동하고 **[PAYG 추가]**를 클릭합니다.
2. 페이지 섹션 PAYG → → → 으로 시작하십시오.
3. → → 필드에 설명을 추가합니다.
4. → → → SSH → → → 페이지 섹션으로 이동합니다.

5. 필드에 Uyuni에서 연결할 인스턴스 DNS 또는 IP 주소를 입력합니다.
6. SSH 필드에 포트 번호를 입력하거나 기본값 22를 사용합니다.
7. 필드에 클라우드에 지정된 대로 사용자 이름을 입력합니다.
8. 필드에 암호를 입력합니다.
9. SSH 필드에 인스턴스 키를 입력합니다.
10. SSH 필드에 키 암호 문구를 입력합니다.



인증 키는 항상 PEM 형식이어야 합니다.

인스턴스에 직접 연결하지 않고 SSH 배스천을 통해 연결하는 경우 [절차: SSH 배스천 연결 데이터 추가](#)를 진행합니다.

그러지 않으면, [절차: Red Hat 연결 종료](#)를 진행합니다.

절차: SSH 배스천 연결 데이터 추가

1. SSH 페이지 섹션으로 이동합니다.
2. 필드에 배스천 호스트 이름을 입력합니다.
3. SSH 필드에 배스천 포트 번호를 입력합니다.
4. 필드에 배스천 사용자 이름을 입력합니다.
5. 필드에 배스천 암호를 입력합니다.
6. SSH 필드에 배스천 키를 입력합니다.
7. SSH 필드에 배스천 키 암호 문구를 입력합니다.

[절차: Red Hat 연결 종료](#)로 설정을 완료합니다.

절차: Red Hat 연결 종료

1. 새 Red Hat 연결 데이터 추가를 완료하려면 **[생성]**을 클릭합니다.
2. PAYG 연결 데이터 페이지로 돌아갑니다. 업데이트된 연결 상태는 상단 섹션에 표시됩니다.
3. 연결 상태는 > > 화면에도 표시됩니다.
4. 인스턴스에 대한 인증 데이터가 올바른 경우 열에 표시됩니다.



언제라도 잘못된 데이터가 입력되면 새로 생성된 인스턴스가 > > PAYG에 표시되고 열에는 오류 메시지가 표시됩니다.

서버에서 인증 데이터를 사용할 수 있게 되면, 연결된 인스턴스에서 사용 가능한 모든 리포지토리에 대한 리포지토리가 추가됩니다. 리포지토리는 > > 에서 볼 수 있습니다.



Red Hat 연결은 기본적으로 조직 1이 소유하는 사용자 정의 리포지토리를 생성합니다. 다른 조직이 자동 생성된 리포지토리를 소유해야 하는 경우, /etc/rhn/rhn.conf에서 java.rhui_default_org_id를 구성합니다.

이는 리포지토리만 정의하고 업데이트합니다. 관리 클라이언트용 리포지토리를 사용하려면 소프트웨어 채널을 지정하고 해당 채널에 리포지토리를 연결해야 합니다.

4.12.2.3. 소프트웨어 채널 추가

Red Hat 클라이언트를 Uyuni 서버에 등록하기 전에 필요한 소프트웨어 채널을 추가하고 이 채널을 동기화해야 합니다.



다음 섹션에서 설명은 종종 `x86_64` 아키텍처로 기본 설정됩니다. 해당될 경우 다른 아키텍처로 교체하십시오.

이 절차에 필요한 채널은 다음과 같습니다.

표 40. Red Hat 채널-CLI

OS 버전	기본 채널	클라이언트 채널	도구 채널
Red Hat 7	<code>rhel7-pool-uyuni</code>	-	<code>rhel7-uyuni-client</code>
Red Hat 8	<code>rhel8-pool-uyuni</code>	-	<code>rhel8-uyuni-client</code>
Red Hat 9	<code>rhel9-pool-uyuni</code>	-	<code>rhel9-uyuni-client</code>

명령 프롬프트에서 소프트웨어 채널 추가

- Uyuni 서버의 명령 프롬프트에서 root 권한으로 `spacewalk-common-channels` 명령을 사용해 적절한 채널을 추가합니다. 다음과 같이 올바른 아키텍처를 지정했는지 확인합니다.

```
spacewalk-common-channels \
-a <architecture> \
<base_channel_name> \
<child_channel_name_1> \
<child_channel_name_2> \
... <child_channel_name_n>
```

- `automatic synchronization`이 꺼진 경우 다음 채널을 동기화하십시오.

```
spacewalk-repo-sync -p <base_channel_label>-<architecture>
```

- 계속하기 전에 동기화가 완료되었는지 확인합니다.



`spacewalk-common-channels`가 제공하는 클라이언트 도구 채널은 SUSE가 아닌 Uyuni가 공급합니다.

4.12.2.4. 사용자 정의 채널 준비

RHUI에서 소프트웨어를 미러링하려면 자동 생성 리포지토리에 연결된 Uyuni에 사용자 정의 채널을 생성해야 합니다.

이 절차에 필요한 채널은 다음과 같습니다.

표 41. Red Hat 사용자 정의 채널

OS 버전	기본 채널
Red Hat 7	rhel7-pool-uyuni
Red Hat 8	rhel8-pool-uyuni
Red Hat 9	rhel9-pool-uyuni

절차: 사용자 정의 채널 생성

1. Uyuni 서버 Web UI에서 **소프트웨어 > 관리 > 채널**로 이동합니다.
2. **[채널 생성]**을 클릭하여 채널에 적절한 파라미터를 설정합니다.
3. → 필드에서 적절한 기본 채널을 선택합니다.
4. **[채널 생성]**을 클릭합니다.
5. 생성해야 하는 모든 채널에 대해 이를 반복합니다. 각 사용자 정의 리포지토리에는 하나의 사용자 정의 채널이 있어야 합니다.

소프트웨어 > 채널 목록 > 전체로 이동하여 적절한 채널 및 리포지토리를 모두 생성했는지 확인할 수 있습니다.



For Red Hat 9 and Red Hat 8 clients, add both the Base and AppStream channels. You require packages from both channels. If you do not add both channels, you cannot create the bootstrap repository, due to missing packages.

모든 채널을 생성했으면 생성한 리포지토리를 다음과 같이 채널과 연결할 수 있습니다.

절차: 채널을 리포지토리와 연결

1. Uyuni 서버 Web UI에서 **소프트웨어 > 관리 > 채널**로 이동한 다음, 연결할 채널을 클릭합니다.
2. → 템으로 이동하여 이 채널에 연결할 리포지토리의 확인란을 선택합니다.
3. **[리포지토리 업데이트]**를 클릭하여 채널과 리포지토리를 연결합니다.
4. 연결하려는 모든 채널과 리포지토리에 대해 이를 반복합니다.
5. 옵션: → 템으로 이동하여 이 리포지토리의 동기화에 대해 정기적 일정을 설정합니다.
6. **[지금 동기화]**를 클릭하여 즉시 동기화를 시작합니다.

4.12.2.5. 동기화 상태 확인

절차: Web UI에서 동기화 진행 상태 확인

1. Uyuni Web UI에서 **소프트웨어 > 관리 > 채널**로 이동한 다음, 리포지토리에 연결된 채널을 클릭합니다.
2. → 템으로 이동한 다음, →를 클릭하고 →를 확인합니다.

절차: 명령 프롬프트에서 동기화 진행 상태 확인

1. Uyuni 서버의 명령 프롬프트에서 루트 권한으로 다음과 같이 `tail` 명령을 사용해 동기화 로그 파일을 확인합니다.

```
tail -f /var/log/rhn/reposync/<channel-label>.log
```

2. 각 하위 채널은 동기화가 진행되는 동안 자체 로그를 생성합니다. 동기화가 완료되었는지 알아보려면 모든 기본 및 하위 채널 로그 파일을 확인해야 합니다.



Red Hat Enterprise Linux 채널은 규모가 매우 클 수 있습니다. 일부 경우 동기화에 몇 시간이 걸릴 수 있습니다.

4.12.2.6. GPG 키 관리

클라이언트는 소프트웨어 패키지가 설치되기 전에 GPG 키를 사용해 신뢰성을 확인합니다. 신뢰할 수 있는 소프트웨어만 클라이언트에 설치할 수 있습니다.



클라이언트의 보안을 위해 GPG 키를 신뢰하는 것이 중요합니다. 어떤 키가 필요하고 신뢰할 수 있는지 결정하는 것은 관리자가 수행해야 하는 작업입니다. GPG 키를 신뢰할 수 없는 경우 소프트웨어 채널을 사용할 수 없기 때문에 채널을 클라이언트에 할당하는 결정은 키를 신뢰하는 결정에 따라 달라집니다.

GPG 키에 대한 자세한 내용은 [Client-configuration > Gpg-keys](#)에서 확인할 수 있습니다.

4.12.2.7. 클라이언트 등록

클라이언트를 등록하려면 부트스트랩 리포지토리가 필요합니다. 기본적으로 부트스트랩 리포지토리는 자동으로 생성되며 동기화된 모든 제품에 대해 매일 재생성됩니다. 명령 프롬프트에서 다음 명령을 사용해 부트스트랩 리포지토리를 수동으로 생성할 수 있습니다.

```
mgr-create-bootstrap-repo
```

클라이언트 등록에 대한 자세한 내용은 [Client-configuration > Registration-overview](#)에서 참조하십시오.

4.13. Rocky Linux 클라이언트 등록

Rocky Linux 클라이언트를 Uyuni 서버에 등록할 수 있습니다. 방법과 상세 정보는 클라이언트의 운영 체제에 따라 다릅니다.

시작하기 전에 클라이언트의 날짜 및 시간이 Uyuni 서버와 정확히 동기화되었는지 확인하십시오.

활성화 키도 생성해야 합니다.

- 활성화 키 만들기에 대한 자세한 내용은 [Client-configuration > Activation-keys](#)에서 확인할 수 있습니다.
- Rocky Linux에서 SUSE Liberty Linux로 마이그레이션에 대한 자세한 내용은 [client-configuration:clients-sleses.pdf](#)에서 확인할 수 있습니다.

4.13.1. Rocky Linux 클라이언트 등록

이 섹션에는 Rocky Linux 운영 체제를 실행하는 클라이언트를 등록하는 방법에 대한 정보가 포함되어 있습니다.



Rocky Linux 클라이언트를 Uyuni에 등록하는 기능은 -- 정책으로 --하는 기본 SELinux 구성으로 테스트됩니다. SELinux를 비활성화해야 Rocky Linux 클라이언트를 Uyuni에 등록할 수 있는 것은 아닙니다.

4.13.1.1. 소프트웨어 채널 추가

Rocky Linux 클라이언트를 Uyuni 서버에 등록하려면 먼저 필요한 소프트웨어 채널을 추가하고 이 채널을 동기화해야 합니다.

현재 지원되는 아키텍처는 x86_64 및 aarch64이며, 버전 9의 경우 추가적으로 ppc64le 및 s390x가 지원됩니다. 지원되는 제품 및 아키텍처의 전체 목록은 [Client-configuration > Supported-features](#)에서 확인할 수 있습니다.



다음 섹션에서 설명은 종종 x86_64 아키텍처로 기본 설정됩니다. 해당될 경우 다른 아키텍처로 교체하십시오.

이 절차에 필요한 채널은 다음과 같습니다.

표 42. Rocky Linux 채널 - CLI

OS 버전	기본 채널	클라이언트 채널	AppStream 채널
Rocky Linux 9	rockylinux9	rockylinux9-uyuni-client	rockylinux9-appstream
Rocky Linux 8	rockylinux8	rockylinux8-uyuni-client	rockylinux8-appstream

명령 프롬프트에서 소프트웨어 채널 추가

- Uyuni 서버의 명령 프롬프트에서 root 권한으로 spacewalk-common-channels 명령을 사용해 적절한 채널을 추가합니다. 다음과 같이 올바른 아키텍처를 지정했는지 확인합니다.

```
spacewalk-common-channels \
-a <architecture> \
<base_channel_name> \
<child_channel_name_1> \
<child_channel_name_2> \
... <child_channel_name_n>
```

- automatic synchronization이 꺼진 경우 다음 채널을 동기화하십시오.

```
spacewalk-repo-sync -p <base_channel_label>-<architecture>
```

- 계속하기 전에 동기화가 완료되었는지 확인합니다.



spacewalk-common-channels가 제공하는 클라이언트 도구 채널은 SUSE가 아닌 Uyuni가 공급합니다.



For Rocky Linux 9 and Rocky Linux 8 clients, add both the Base and AppStream channels. You require packages from both channels. If you do not add both channels, you cannot create the bootstrap repository, due to missing packages.



AppStream 채널에서 사용할 수 있는 패키지의 수가 업스트림과 Uyuni 채널 간에 서로 약간 불일치함을 알 수 있습니다. 또한 다른 시점에 추가한 동일 채널을 비교하면 그 수가 다른 것을 알 수 있습니다. 이는 Rocky Linux가 리포지토리를 관리하는 방식에 원인이 있습니다. Rocky Linux는 새 버전이 릴리스되면 이전 버전의 패키지를 제거하지만 Uyuni는 사용 기간과 관계없이 모두 그대로 유지합니다.

Rocky Linux 8을 통해 모듈형 채널을 사용하는 경우 클라이언트에서 Python 3.6 모듈 스트림을 활성화해야 합니다. Python 3.6을 제공하지 않으면 spacecmd 패키지의 설치가 실패합니다.

4.13.1.2. 동기화 상태 확인

절차: Web UI에서 동기화 진행 상태 확인

1. Uyuni Web UI에서 **소프트웨어** > **관리** > **채널**로 이동한 다음, 리포지토리에 연결된 채널을 클릭합니다.
2. ←→ 템으로 이동한 다음, ←→를 클릭하고 ←→ ←→를 확인합니다.

절차: 명령 프롬프트에서 동기화 진행 상태 확인

1. Uyuni 서버의 명령 프롬프트에서 루트 권한으로 다음과 같이 `tail` 명령을 사용해 동기화 로그 파일을 확인합니다.

```
tail -f /var/log/rhn/reposync/<channel-label>.log
```

2. 각 하위 채널은 동기화가 진행되는 동안 자체 로그를 생성합니다. 동기화가 완료되었는지 알아보려면 모든 기본 및 하위 채널 로그 파일을 확인해야 합니다.

4.13.1.3. 활성화 키 생성

Rocky Linux 채널에 연결된 활성화 키를 생성해야 합니다.

활성화 키에 대한 자세한 내용은 **Client-configuration** > **Activation-keys**를 참조하십시오.

4.13.1.4. GPG 키 관리

클라이언트는 소프트웨어 패키지가 설치되기 전에 GPG 키를 사용해 신뢰성을 확인합니다. 신뢰할 수 있는 소프트웨어만 클라이언트에 설치할 수 있습니다.



클라이언트의 보안을 위해 GPG 키의 신뢰성이 중요합니다. 어떤 키가 필요하고 신뢰할 수 있는지 결정하는 것은 관리자가 수행해야 하는 작업입니다. GPG 키를 신뢰할 수 없으면 클라이언트에 소프트웨어 채널을 할당할 수 없습니다.

GPG 키에 대한 자세한 내용은 [Client-configuration > Gpg-keys](#)에서 확인할 수 있습니다.

4.13.1.5. 클라이언트 등록

클라이언트를 등록하려면 부트스트랩 리포지토리가 필요합니다. 기본적으로 부트스트랩 리포지토리는 자동으로 생성되며 동기화된 모든 제품에 대해 매일 재생성됩니다. 명령 프롬프트에서 다음 명령을 사용해 부트스트랩 리포지토리를 수동으로 생성할 수 있습니다.

```
mgr-create-bootstrap-repo
```

클라이언트 등록에 대한 자세한 내용은 [Client-configuration > Registration-overview](#)에서 참조하십시오.

4.13.1.6. 정오표 관리

Rocky Linux 클라이언트를 업데이트할 때 패키지에는 업데이트에 대한 메타데이터가 포함됩니다.

4.14. Ubuntu 클라이언트 등록

Ubuntu 클라이언트를 Uyuni 서버에 등록할 수 있습니다. 방법과 상세 정보는 클라이언트의 운영 체제에 따라 달라집니다.

시작하기 전에 클라이언트의 날짜 및 시간이 Uyuni 서버와 정확히 동기화되었는지 확인하십시오.

또한 활성화 키 생성을 완료한 상태이어야 합니다. 활성화 키 생성에 대한 자세한 내용은 [Client-configuration > Activation-keys](#)를 참조하십시오.

4.14.1. Ubuntu 20.04 및 22.04 클라이언트 등록

이 섹션에는 Ubuntu 20.04 LTS 및 22.04 LTS 운영 체제를 실행하는 클라이언트를 등록하는 방법에 대한 정보가 포함되어 있습니다.

부트스트래핑은 Ubuntu 클라이언트 시작과 초기 상태 실행 수행(예: 리포지토리 설정, 프로파일 업데이트 수행)에 대해 지원됩니다. 하지만 Ubuntu의 루트 사용자는 기본적으로 비활성화되므로 부트스트래핑을 사용하려면 Python에 대한 sudo 권한이 있는 기존 사용자가 필요합니다.



Canonical은 Uyuni를 지지하거나 지원하지 않습니다.

4.14.1.1. 소프트웨어 채널 추가

Ubuntu 클라이언트를 Uyuni 서버에 등록하기 전에 필요한 소프트웨어 채널을 추가하고 이 채널을 동기화해야 합니다.



다음 섹션에서 설명은 종종 x86_64 아키텍처로 기본 설정됩니다. 해당될 경우 다른 아키텍처로 교체하십시오.

이 절차에 필요한 채널은 다음과 같습니다.

표 43. Ubuntu 채널 - CLI

OS 버전	기본 채널	주 채널	업데이트 채널	보안 채널	클라이언트 채널
Ubuntu 22.04	ubuntu-2204-pool-amd64-uyuni	ubuntu-2204-amd64-main-uyuni	ubuntu-2204-amd64-main-updates-uyuni	ubuntu-2204-amd64-main-security-uyuni	ubuntu-2204-amd64-uyuni-client
Ubuntu 20.04	ubuntu-2004-pool-amd64-uyuni	ubuntu-2004-amd64-main-uyuni	ubuntu-2004-amd64-main-updates-uyuni	ubuntu-2004-amd64-main-security-uyuni	ubuntu-2004-amd64-uyuni-client

20.04 버전에도 Universe 채널이 필요합니다.

표 44. Ubuntu 20.04 Universe 채널 - CLI

Ubuntu 20.04	
Universe 채널	ubuntu-2004-amd64-universe-uyuni
Universe 업데이트 채널	ubuntu-2004-amd64-universe-updates-uyuni
Universe 보안 업데이트 채널	ubuntu-2004-amd64-universe-security-uyuni
Universe 백포트 채널	ubuntu-2004-amd64-universe-backports-uyuni

명령 프롬프트에서 소프트웨어 채널 추가

1. Uyuni 서버의 명령 프롬프트에서 루트로 spacewalk-common-channels 명령을 사용하여 적절한 채널을 추가합니다.

```
spacewalk-common-channels \
<base_channel_label> \
<child_channel_label_1> \
<child_channel_label_2> \
... <child_channel_label_n>
```

2. **automatic synchronization**이 꺼진 경우 다음 채널을 동기화하십시오.

```
spacewalk-repo-sync -p <base_channel_label>
```

3. 계속하기 전에 동기화가 완료되었는지 확인합니다.



새 채널을 완전히 동기화한 후에 Ubuntu 클라이언트를 부트스트랩해야 합니다.

4.14.1.2. Ubuntu ESM 패키지 미러링

Canonical은 Ubuntu Pro 사용자 및 고객을 위한 **Expanded Security Maintenance (ESM)** 패키지를 제공합니다. 이 패키지는 여러 운영 체제 구성 요소와 일부 애플리케이션에 대해 더 긴 유지보수 기간(10~12년)을 제공합니다.

이러한 리포지토리는 Ubuntu Pro에 등록된 시스템에서 필요한 **GPG 키** 및 개인 **Bearer 토큰**을 추출하면 Uyuni 내에서 동기화할 수도 있습니다.

4.14.1.2.1. GPG 키 및 Bearer 토큰 추출

Ubuntu Pro에 Ubuntu 호스트를 등록합니다. 개인 등록 토큰은 [Ubuntu Pro Dashboard](#)에서 찾을 수 있습니다. 이를 위해서는 [Ubuntu One account](#)이 필요합니다.

```
sudo apt-get install ubuntu-advantage-tools
sudo pro attach <personal_token>
```

등록 후, /etc/apt/auth.conf.d/90ubuntu-advantage 파일에서 Bearer 토큰을 찾을 수 있습니다

```
machine esm.ubuntu.com/apps/ubuntu/ login bearer password <token> #
ubuntu-pro-client
machine esm.ubuntu.com/infra/ubuntu/ login bearer password <token> #
ubuntu-pro-client
```



리포지토리당 1개의 전용 bearer 토큰이 사용됩니다.

Uyuni 내에 다음 리포지토리를 구성합니다.

4.14.1.2.2. Ubuntu ESM 리포지토리 구성

리포지토리를 만들려면 다음 URL을 사용합니다.

표 45. Ubuntu ESM 리포지토리

URL	설명
<a href="https://bearer:<token>@esm.ubuntu.com/infra/ubuntu/dists/<release>-infra-updates/main/binary-<arch>/">https://bearer:<token>@esm.ubuntu.com/infra/ubuntu/dists/<release>-infra-updates/main/binary-<arch>/	운영 체제 기능 업데이트
<a href="https://bearer:<token>@esm.ubuntu.com/infra/ubuntu/dists/<release>-infra-security/main/binary-<arch>/">https://bearer:<token>@esm.ubuntu.com/infra/ubuntu/dists/<release>-infra-security/main/binary-<arch>/	운영 체제 보안 업데이트
<a href="https://bearer:<token>@esm.ubuntu.com/apps/ubuntu/dists/<release>-apps-updates/main/binary-<arch>/">https://bearer:<token>@esm.ubuntu.com/apps/ubuntu/dists/<release>-apps-updates/main/binary-<arch>/	애플리케이션 기능 업데이트

URL	설명
<a href="https://bearer:<token>@esm.ubuntu.com/apps/ubuntu/dists/<release>-apps-security/main/binary-<arch>/">https://bearer:<token>@esm.ubuntu.com/apps/ubuntu/dists/<release>-apps-security/main/binary-<arch>/	애플리케이션 보안 업데이트

`<token>` 을 개인 Bearer 토큰으로 바꿉니다. 또한 `arch` 와 `release` 는 다음 값 중 하나로 바꿔야 합니다.

표 46. Ubuntu ESM 아키텍처 및 릴리스

아키텍처	릴리스
amd64, arm64, armel, armhf, i386, powerpc, ppc64el, s390x	bionic, focal, jammy, noble, trusty, xenial

Uyuni(가) 리포지토리를 동기화하려면 해당 GPG 키(ubuntu-advantage-esm-infra-trusty.gpg, ubuntu-advantage-esm-apps.gpg)를 가져와야 합니다. 이러한 키는 Ubuntu Pro에 등록된 시스템의 `/etc/apt/trusted.gpg.d`에 있습니다. 이러한 파일을 Uyuni 시스템에 복사하고 다음과 같이 가져옵니다.

```
mgradm gpg add /path/to/gpg.key
```

이미 동기화된 Ubuntu 상위 채널 아래에 적절한 하위 채널을 만듭니다. 그런 다음 리포지토리를 동기화할 수 있습니다.



여기에 표시된 절차를 사용하여 구독 제한을 우회할 수 있지만, 이는 서비스 이용 약관을 위반하는 것이며 법적 결과를 초래할 수 있습니다. 항상 사용하는 시스템 수에 해당하는 충분한 구독을 보유해야 합니다.

4.14.1.3. 동기화 상태 확인

절차: Web UI에서 동기화 진행 상태 확인

1. Uyuni Web UI에서 **소프트웨어 > 관리 > 채널**로 이동한 다음, 리포지토리에 연결된 채널을 클릭합니다.
2. ➤➤➤➤ 탭으로 이동한 다음, ➤➤➤를 클릭하고 ➤➤ ➤➤를 확인합니다.

절차: 명령 프롬프트에서 동기화 진행 상태 확인

1. Uyuni 서버의 명령 프롬프트에서 루트 권한으로 다음과 같이 tail 명령을 사용해 동기화 로그 파일을 확인합니다.

```
tail -f /var/log/rhn/reposync/<channel-label>.log
```

2. 각 하위 채널은 동기화가 진행되는 동안 자체 로그를 생성합니다. 동기화가 완료되었는지 알아보려면 모든 기본 및 하위 채널 로그 파일을 확인해야 합니다.



Ubuntu 채널은 규모가 매우 클 수 있습니다. 때로 동기화에 몇 시간이 걸릴 수 있습니다.

4.14.1.4. GPG 키 관리

클라이언트는 소프트웨어 패키지가 설치되기 전에 GPG 키를 사용해 신뢰성을 확인합니다. 신뢰할 수 있는 소프트웨어만 클라이언트에 설치할 수 있습니다.



클라이언트의 보안을 위해 GPG 키를 신뢰하는 것이 중요합니다. 어떤 키가 필요하고 신뢰할 수 있는지 결정하는 것은 관리자가 수행해야 하는 작업입니다. GPG 키를 신뢰할 수 없는 경우 소프트웨어 채널을 사용할 수 없기 때문에 채널을 클라이언트에 할당하는 결정은 키를 신뢰하는 결정에 따라 달라집니다.

GPG 키에 대한 자세한 내용은 [Client-configuration > Gpg-keys](#)에서 확인할 수 있습니다.

4.14.1.5. 루트 액세스

Ubuntu의 루트 사용자는 SSH 액세스에 대해 기본적으로 비활성화되어 있습니다.

일반 사용자를 사용하여 온보딩하려면 sudoers 파일을 편집해야 합니다.



이 문제는 Ubuntu의 자체 설치 버전에서 발생합니다. 설치 중에 기본 사용자에게 관리자 권한이 부여된 경우 sudo를 사용하여 권한 에스컬레이션을 수행하려면 비밀번호가 필요합니다. 클라우드 인스턴스에서는 이런 일이 발생하지 않습니다. 왜냐하면 cloud-init는 /etc/sudoers.d 아래에 자동으로 파일을 생성하고 비밀번호를 사용할 필요 없이 sudo를 사용하여 권한 에스컬레이션을 부여하기 때문입니다.

절차: 루트 사용자에게 액세스 권한 부여

1. 클라이언트에서 다음과 같이 sudoers 파일을 편집합니다.

```
sudo visudo
```

이 줄을 sudoers 파일 끝에 추가하여 사용자에게 sudo 액세스 권한을 부여합니다. 다음과 같이 <user>를 Web UI에서 클라이언트를 부트스트래핑하고 있는 사용자의 이름으로 교체합니다.

```
<user>  ALL=NOPASSWD: /usr/bin/python, /usr/bin/python2,
/usr/bin/python3, /var/tmp/venv-salt-minion/bin/python
```



이 절차는 클라이언트 등록에 필요한 비밀번호가 없어도 루트에 액세스 권한을 부여합니다. 클라이언트는 성공적으로 설치된 후 루트 권한으로 실행되므로 액세스 권한이 더 이상 필요 없습니다. 클라이언트가 성공적으로 설치된 후 sudoers 파일에서 이 줄을 제거하는 것이 좋습니다.

4.14.1.6. 클라이언트 등록

클라이언트를 등록하려면 부트스트랩 리포지토리가 필요합니다. 기본적으로 부트스트랩 리포지토리는 자동으로 생성되며 동기화된 모든 제품에 대해 매일 재생성됩니다. 명령 프롬프트에서 다음 명령을 사용해 부트스트랩

리포지토리를 수동으로 생성할 수 있습니다.

```
mgr-create-bootstrap-repo
```

클라이언트 등록에 대한 자세한 내용은 [Client-configuration > Registration-overview](#)에서 참조하십시오.

4.15. 프록시에 클라이언트 등록

프록시 서버는 Salt 및 기존 클라이언트를 위한 브로커 및 패키지 캐시의 역할을 할 수 있습니다. 프록시에 클라이언트를 등록하는 작업은 몇 가지 주요 차이점 외에는 Uyuni 서버에 직접 등록하는 것과 유사합니다.

이 섹션에는 Web UI, 명령줄의 명령 또는 부트스트랩 스크립트를 사용하여 프록시에 클라이언트를 등록하는 내용이 포함되어 있습니다. Uyuni 프록시에서 다른 프록시 또는 Uyuni 서버로 클라이언트를 이동하는 절차에 대한 설명도 제공됩니다.

Web UI 내에서 프록시 페이지에는 클라이언트에 대한 정보가 표시됩니다. 프록시에 연결된 클라이언트 목록은 **시스템**, **시스템 목록**, **프록시**에서 프록시 이름을 클릭한 후 **...** 하위 탭의 **...** 탭을 선택하여 확인할 수 있습니다.

클라이언트의 체인으로 연결된 프록시 목록은 **시스템**, **전체**에서 클라이언트 이름을 클릭한 후 **...** 탭에서 **...** 하위 탭을 선택하여 확인할 수 있습니다.

4.15.1. 프록시 간 클라이언트 이동

등록 프로세스를 반복할 필요 없이 프록시 간에 클라이언트(Salt)와 Salt SSH 푸시 클라이언트를 이동할 수 있습니다.



체인으로 연결된 프록시는 이동할 수 없습니다. 체인으로 연결된 프록시를 이동하는 대신, 새 프록시를 만들고 클라이언트를 이동한 다음 이전 프록시를 삭제하십시오.

절차: 프록시 간에 클라이언트 또는 Salt SSH 푸시 클라이언트 이동

1. Uyuni Web UI에서 프록시 간에 이동하려는 클라이언트의 **...** **...** 페이지로 이동합니다.
2. **...** 탭으로 이동합니다. 그런 다음 **...** **...** 링크를 따라 드롭다운 메뉴를 확인합니다.
3. **...** 드롭다운 메뉴에서 클라이언트가 이동할 프록시를 선택하고 **[프록시 변경]**을 클릭합니다.

절차: SSM을 사용하여 프록시 간에 여러 클라이언트 또는 Salt SSH 푸시 클라이언트 이동

1. Uyuni Web UI에서 **시스템** > **시스템 목록**으로 이동하여 이동할 각 클라이언트의 확인란을 선택하면 클라이언트가 시스템 세트 관리자에 추가됩니다.
2. **시스템** > **시스템 세트 관리자**로 이동한 후 **기타** > **프록시** 탭으로 이동합니다.
3. **...** 드롭다운 메뉴에서 클라이언트를 이동할 프록시를 선택한 후 **[프록시 변경]**을 클릭합니다.

`system.changeProxy` API 호출을 통해서도 동일한 기능을 이용할 수 있습니다.

4.15.1.1. 배경 정보

이 함수의 효과는 일반 Salt 클라이언트와 Salt SSH 푸시 클라이언트 간에 다릅니다.

4.15.1.1.1. 기본 클라이언트

이 함수는 Salt 상태 작업을 예약하여 `susemanager.conf` Salt 클라이언트 구성 파일에서 `master:` 설정을 수정해 새 프록시를 가리키도록 합니다. 그런 다음 함수가 Salt 클라이언트를 다시 시작합니다.



`susemanager.conf` 파일을 수동으로 편집하여 `master:`를 변경해도 효과가 같으며 이러한 기능도 지원됩니다.

클라이언트가 다시 시작하고 새 프록시를 통해 다시 연결하면 서버는 데이터베이스에서 프록시 경로를 업데이트하고 채널 URL을 새로 고치는 다른 작업을 예약합니다.

4.15.1.1.2. Salt SSH 푸시 클라이언트

이 함수는 데이터베이스에서 즉시 프록시 경로를 업데이트하며 채널 URL을 새로 고치는 새로운 작업이 예약됩니다.

4.15.2. 프록시에서 서버로 클라이언트 이동

클라이언트를 프록시에서 서버로 이동하려면 프록시 목록에서 `None`을 선택합니다.

4.15.3. Web UI로 클라이언트를 프록시에 등록

Web UI를 사용하여 클라이언트를 Uyuni 프록시에 등록할 수 있습니다.

부트스트랩 리포지토리는 비 SLE 클라이언트의 경우 일반적으로 필요하고, SLE 클라이언트의 경우 버전 15 이전 버전에 대해 필요합니다.

부트스트랩 리포지토리 생성에 대한 자세한 내용은 **Client-configuration > Bootstrap-repository**에서 확인할 수 있습니다.

절차: Web UI로 클라이언트를 프록시에 등록

1. Uyuni Web UI에서 **시스템 > 부트스트랩**으로 이동합니다.
2. **---** 필드에 부트스트랩 할 클라이언트의 정규화된 도메인 이름(FQDN)을 입력합니다.
3. **SSH** **--** 필드에서 클라이언트를 연결하고 부트스트랩하는 데 사용할 SSH 포트 번호를 입력합니다. 기본적으로 SSH 포트는 22입니다.
4. **--** 필드에서 클라이언트에 로그인할 사용자 이름을 입력합니다. 기본적으로 사용자 이름은 `root`입니다.
5. **--** **--** 필드에서 클라이언트를 부트스트랩하는 데 사용할 인증 방법을 선택합니다.
 - 비밀번호 인증의 경우 **----** 필드에서 클라이언트에 로그인할 때 사용할 비밀번호를 입력합니다.
 - SSH 개인 키 인증의 경우 개인 키와 개인 키에 연결된 비밀번호를 입력합니다. 이 키는 부트스트랩 프로세스를 완료하는 데 필요한 기간 동안만 보관됩니다.
6. **--** **-** 필드에서 클라이언트를 부트스트랩하는 데 사용할 소프트웨어 채널과 연결된 활성화 키를 선택합니다.
7. **----** 필드에서 등록하려는 프록시 서버를 선택합니다.
8. 기본적으로 **SSH Strict** **--** **--** **----** 확인란이 선택되어 있습니다. 따라서 사용자가 수동으로 인증할 필요 없이 부트스트랩 프로세스가 자동으로 SSH 호스트 키를 수락할 수 있습니다.
9. 옵션: **SSH** **--** **--** **----** **----** 확인란을 선택합니다. 이 옵션을 선택하면 서버에 연결하기 위해 SSH를

사용하도록 클라이언트가 구성되고 다른 연결 방법은 구성되지 않습니다.

10. [부트스트랩]을 클릭하여 등록을 시작합니다.

부트스트랩 프로세스가 완료되면 클라이언트가 시스템 > 시스템 목록에 나열됩니다.

4.15.4. 명령줄에서 클라이언트 등록

Web UI 대신 명령줄을 사용하여 클라이언트를 프록시에 등록할 수 있습니다. 이 절차를 수행하려면 등록하기 전에 클라이언트에 Salt 패키지를 설치해야 합니다. SLE 12 기반 클라이언트의 경우, glibc 모듈도 활성화해야 합니다.

절차: 명령줄을 사용해 클라이언트를 프록시에 등록

1. 다음 위치에 있는 클라이언트 구성 파일을 선택합니다.

```
/etc/salt/minion
```

또는:

```
/etc/salt/minion.d/NAME.conf
```

이 파일을 종종 minion 파일이라고도 합니다.

2. 다음과 같이 프록시 FQDN을 클라이언트 구성 파일에 master로 추가합니다.

```
master: PROXY123.EXAMPLE.COM
```

3. 다음과 같이 salt-minion 서비스를 다시 시작합니다.

```
systemctl restart salt-minion
```

4. 서버에서 새로운 클라이언트 키를 수락하고, <client>를 클라이언트 이름으로 대체합니다.

```
salt-key -a '<client>'
```

4.16. 공용 클라우드에서 클라이언트 등록

Uyuni 서버를 설정한 후에는 클라이언트 등록을 시작할 수 있습니다.

4.16.1. 제품 추가 및 리포지토리 동기화

클라이언트에 해당하는 제품을 이미 추가했고 리포지토리를 Uyuni에 동기화했는지 확인하십시오. 클라이언트 등록에 사용되는 부트스트랩 리포지토리를 생성하려면 이 작업은 필수입니다.

자세한 내용은 **Specialized-guides** > **Public-cloud-guide**에서 확인할 수 있습니다.

4.16.2. 온디맨드 이미지 준비

SUSE가 제공한 온디맨드 이미지에서 시작된 인스턴스는 자동으로 등록되고 업데이트 인프라 및 SUSE Linux Enterprise 모듈이 활성화됩니다. 온디맨드 이미지를 Uyuni 클라이언트로 사용하려면 이 자동화를 비활성화한 후에 시작해야 합니다.

절차: 온디맨드 이미지 준비

1. 온디맨드 인스턴스에 로그인합니다.
2. 명령 프롬프트에서 루트 권한으로 등록 데이터 및 리포지토리를 제거합니다.

```
registercloudguest --clean
```

3. 자동 등록을 위한 트리거 서비스 제거:

```
systemctl disable guestregister.service
```

4. Microsoft Azure에서는 비활성화해야 하는 추가 서비스가 있습니다.

```
systemctl disable regionsrv-enabler-azure.timer
```

4.16.3. 클라이언트 등록

Uyuni Web UI에서 **시스템** > **부트스트래핑**으로 이동하여 **IP**, **SSH 키**, **FQDN** 및 **호스트 이름** 필드를 입력합니다. **FQDN** 필드에서 안정적인 FQDN을 사용할 수 있는지 확인합니다. 그렇지 않으면 Uyuni가 공용 클라우드에서 다른 일시적인 FQDNs를 제공하는 경우 호스트를 찾을 수 없습니다.

일반적으로 공용 클라우드 이미지는 사용자 이름 및 암호를 사용한 SSH 로그인을 허용하지 않고 인증서가 있는 SSH만 허용합니다. Web UI에서 부트스트랩하려면 사용자 이름 및 SSH 키를 사용하여 SSH 로그인을 활성화해야 합니다. 이 작업은 **시스템** > **부트스트랩**으로 이동한 후 권한 부여 방법을 변경하여 수행할 수 있습니다.

클라우드 공급자가 Microsoft Azure인 경우 사용자 이름 및 비밀번호를 사용하여 로그인할 수 있습니다. 이렇게 하려면 AzureUser가 비밀번호를 사용하지 않고 루트 권한으로 명령을 실행할 수 있도록 허용해야 합니다. 이 작업을 수행하려면 `/etc/sudoers.d/waagent` 파일을 열고 다음 줄을 추가하거나 편집합니다.

```
AzureUser ALL=(ALL) NOPASSWD: ALL
```



AzureUser가 암호를 사용하지 않고 루트 권한으로 명령을 실행할 수 있도록 허용하면 보안 위험이 발생합니다. 테스트 목적으로만 이 방법을 사용하십시오. 프로덕션 시스템에서는 이 기능을 사용하지 마십시오.

부트스트랩 프로세스가 완료된 후에는 클라이언트가 **시스템 > 시스템 목록**에 나열됩니다.

- 프로세스에 대한 더 많은 제어 권한을 원하거나 여러 클라이언트를 등록해야 하거나 기존 클라이언트를 등록하려면 부트스트랩 스크립트를 생성합니다. 자세한 내용은 **Client-configuration > Registration-bootstrap**에서 확인할 수 있습니다.
- Salt 클라이언트와 프로세스에 대한 더 많은 제어 권한을 위해서는 명령줄에서 단일 명령을 실행하면 도움이 될 수 있습니다. 자세한 설명은 **Client-configuration > Registration-cli**를 참조하십시오.
- 공용 클라우드 이미지(예: AWS AMI)에서 실행된 클라이언트를 등록하는 경우에는 몇 가지 추가 구성은 수행하여 서로 덮어쓰지 않도록 해야 합니다. 복제 클라이언트 등록에 대한 자세한 내용은 **Administration > Troubleshooting**에서 확인할 수 있습니다.

4.16.4. 활성화 키

활성화 키는 고객에게 적합한 소프트웨어 자격이 있는지, 적절한 채널에 연결되었는지, 관련 그룹에 가입되어 있는지를 확인하는 데 사용됩니다. 각 활성화 키를 생성할 때 설정할 수 있는 조직에 바인딩됩니다.

활성화 키에 대한 자세한 설명은 **Client-configuration > Activation-keys**에서 참조하십시오.

4.16.5. Terraform에서 생성한 클라이언트의 자동 등록

Terraform에서 생성한 새 클라이언트는 Uyuni에 자동으로 등록될 수 있습니다. 등록은 다음의 두 가지 방법으로 수행할 수 있습니다.

- `cloud-init` 기반 등록
- 원격 실행 제공자 기반 등록

4.16.5.1. `cloud-init` 기반 클라이언트 등록

`cloud-init`를 활용하여 등록하는 방법은 새로 생성된 가상 머신을 자동으로 등록하는 기본 방법입니다. 이 솔루션은 호스트에 대한 SSH 연결 구성을 방지합니다. 클라이언트 생성에 사용된 도구와 상관없이 사용할 수도 있습니다.

사용자는 Uyuni에 머신을 자동으로 등록하기 위해 Terraform으로 이미지를 배포할 때 사용자 데이터 세트를 전달할 수 있습니다. `user_data`는 부트스트랩에서 한 번만 실행되며 시스템이 처음 시작될 때만 실행됩니다.

`cloud-init`를 사용하여 클라이언트를 등록하기 전 사용자는 다음을 구성해야 합니다.

- 부트스트랩 스크립트입니다. 자세한 내용은 **Client-configuration > Registration-bootstrap**에서 확인할 수 있습니다.
- 활성화 키입니다. 자세한 내용은 **Client-configuration > Activation-keys**에서 확인할 수 있습니다.

다음 명령은 부트스트랩 스크립트를 다운로드하고 새 머신이 생성될 때 등록하며, `cloud-init` 구성에 추가해야 합니다.

```
curl -s http://hub-server.tf.local/pub/bootstrap/bootstrap-default.sh |  
bash -s
```



- 프로비저닝을 변경하기 위해 user_data가 업데이트될 때마다 Terraform은 시스템을 제거한 후 새 IP 등으로 다시 생성합니다.

AWS 기반 cloud-init에 대한 자세한 내용은 https://registry.terraform.io/providers/hashicorp/template/latest/docs/data-sources/cloudinit_config에서 확인할 수 있습니다.

cloud-init 예제는 https://registry.terraform.io/providers/hashicorp/cloudinit/latest/docs/data-sources/cloudinit_config#example-usage에서 확인할 수 있습니다.

4.16.5.2. remote-exec 프로비저너 기반 등록

Terraform의 remote-exec 프로비저너를 사용하여 새로 생성된 가상 머신을 자동으로 등록하기 위한 두 번째 솔루션입니다.

remote-exec 프로비저너는 새로 생성된 시스템과 상호 작용하며, SSH 연결을 열고 해당 시스템에서 명령을 실행할 수 있습니다.



- remote-exec 구축 프로그램을 사용하여 클라이언트를 등록할 때 사용자는 Terraform을 실행하는 시스템이 생성 후 새 가상 시스템에 액세스할 수 있는지 확인해야 합니다.

나머지 요구 사항은 [cloud-init 기반 클라이언트 등록](#)을 사용할 때와 동일합니다.

- 부트스트랩 스크립트입니다. 자세한 내용은 [Client-configuration > Registration-bootstrap](#)에서 확인할 수 있습니다.
- 활성화 키입니다. 자세한 내용은 [Client-configuration > Activation-keys](#)에서 확인할 수 있습니다.

다음 명령은 부트스트랩 스크립트를 다운로드하고 새 머신이 생성될 때 등록하며, 실행할 원격 명령으로 정의해야 합니다.

```
curl -s http://hub-server.tf.local/pub/bootstrap/bootstrap-default.sh |  
bash -s
```

remote-exec 프로비저너에 대한 자세한 내용은 <https://www.terraform.io/docs/provisioners/remote-exec.html>에서 확인할 수 있습니다.

Chapter 5. 클라이언트 업그레이드

클라이언트는 자체 기본 운영 체제의 버전 관리 시스템을 사용하며 정기적인 업그레이드가 필요합니다.

SUSE 운영 체제를 사용하는 SCC 등록 클라이언트의 경우 Uyuni Web UI 내에서 업그레이드를 수행할 수 있습니다. 지원되는 SUSE Linux Enterprise 15 업그레이드 경로는 <https://documentation.suse.com/sles/15-SP4/html/SLES-all/cha-upgrade-paths.html>에서 확인할 수 있습니다.

SLE 12를 실행하는 클라이언트를 SLE 15로 업그레이드하는 경우 업그레이드가 자동화되지만 시작하기 전에 몇 가지 준비 단계를 수행해야 합니다. 자세한 내용은 **Client-configuration > Client-upgrades-major**를 참조하십시오.

컨텐트 라이프싸이클 관리자를 사용하여 클라이언트 업그레이드를 자동화할 수도 있습니다. 자세한 내용은 **Client-configuration > Client-upgrades-lifecycle**을 참조하십시오.

서비스 팩 업그레이드, openSUSE Leap 부 버전 업그레이드 또는 SUSE Linux Enterprise로의 openSUSE Leap 마이그레이션과 같은 제품 마이그레이션에 대한 자세한 내용은 **Client-configuration > Client-upgrades-product-migration**에서 확인할 수 있습니다.

등록되지 않은 openSUSE Leap 클라이언트를 업그레이드하는 방법에 대한 자세한 내용은 **Client-configuration > Client-upgrades-uyuni**를 참조하십시오.

5.1. 주 버전 업그레이드

클라이언트에는 설치된 운영 체제에 대해 사용 가능한 최신 서비스 팩(SP)이 설치되고 모든 최신 업데이트가 적용되어 있어야 합니다. 시작하기 전에 시스템이 최신 상태이고 모든 업데이트가 성공적으로 설치되었는지 확인합니다.

업그레이드는 YaST 및 AutoYaST에서 제어하며 Zypper는 사용하지 않습니다.

5.1.1. 마이그레이션 준비

클라이언트를 SLE 12에서 SLE 15로 마이그레이션하기 전에 다음 작업을 수행해야 합니다.

1. 설치 미디어 준비
2. 자동 설치 배포 선언
3. 활성화 키 생성
4. 자동 설치 프로파일 생성

5.1.1.1. 설치 미디어 준비

프로시저: 설치 미디어 준비

1. 컨테이너 호스트에서 설치 소스가 포함된 ISO 이미지를 다운로드합니다.
2. mgradm을 사용하여 ISO 이미지에서 설치 데이터를 가져옵니다.

```
mgradm distribution copy <image_name>.iso <image_name>
```

3. `mgradm`이 보고한 배포 경로를 기록해 둡니다. 이 정보는 배포를 Uyuni(으)로 선언할 때 필요합니다.



이 이미지는 여러 자동 설치 배포에 사용할 수 있습니다.

자세한 설명은 **Client-configuration > Autoinst-distributions**를 참조하십시오.

5.1.1.2. 자동 설치 배포 선언

절차: 자동 설치 배포 선언

1. Uyuni Web UI에서 **시스템 > 자동 설치 > 배포**로 이동합니다.
2. **...**을 클릭하고 다음 필드를 입력합니다.

... 필드에 자동 설치 배포를 식별할 수 있는 이름을 입력합니다. 예를 들어, `sles15sp6-x86_64`를 입력합니다. * **...** 필드에 이미 저장한 설치 미디어의 경로를 입력합니다. * 일치하는 **...**을 선택합니다. 이는 설치 미디어와 일치해야 합니다. * **...**을 선택합니다. 이는 설치 미디어와 일치해야 합니다. * 선택 사항: 이 배포를 부팅할 때 사용할 커널 옵션을 지정합니다. 커널 옵션은 다양한 방법으로 제공할 수 있습니다. 배포에 일반적인 옵션만 여기에 추가해야 합니다.

3. **[자동 설치 가능 배포 생성]**을 클릭합니다.

5.1.1.3. 활성화 키 생성

예를 들어, 기존 SLE 12 기본 채널에서 새 SLE 15 채널로 전환하려면 활성화 키가 필요합니다.

절차: 활성화 키 생성

1. Uyuni 서버 Web UI에서 **시스템 > 활성화 키**로 이동하여 **...**을 클릭합니다.
2. 키에 대한 설명을 입력합니다.
3. 키를 입력하거나 공백으로 두어 자동 키를 생성합니다.
4. 옵션: 사용을 제한하려면 **...** 텍스트 필드에 값을 입력합니다.
5. **SLE-Product-SLES15-SP6-Pool for x86_64** 기본 채널을 선택합니다.
6. 옵션: **...**을 선택합니다. 자세한 내용은 <https://documentation.suse.com/sles/15-SP4/html/SLES-all/article-modules.html>에서 확인할 수 있습니다.
7. **[활성화 키 생성]**을 클릭합니다.
8. **...** 탭을 클릭하고 필요한 채널을 선택합니다.
9. **[키 업데이트]**를 클릭합니다.

5.1.1.4. 자동 설치 프로파일 생성

자동 설치 프로파일에는 시스템을 설치하는 데 필요한 모든 설치 및 구성 데이터가 포함되어 있습니다. 설치 완료 후 실행될 스크립트도 포함할 수 있습니다. 시작 지점으로 사용할 수 있는 예제 스크립트는 <https://github.com/SUSE/manager-build-profiles/tree/master/AutoYaST>를 참조하십시오.

유효한 AutoYaST 업그레이드 설정에 대한 내용은 <https://doc.opensuse.org/projects/autoyast/#CreateProfile->

[upgrade](#)에서 확인할 수 있습니다.

절차: 자동 설치 프로파일 생성

1. Uyuni Web UI에서 **시스템** > **자동 설치** > **프로파일**로 이동하여 자동 설치 프로파일 스크립트를 업로드합니다.

시작 지점으로 사용할 수 있는 스크립트의 예는 다음을 참조하십시오.

<https://github.com/SUSE/manager-build-profiles/tree/master/AutoYaST>.

2. **→→ 필드**에 `autoupgrade=1`을 입력합니다.

선택 사항으로 `Y2DEBUG=1` 옵션을 포함해도 됩니다. 디버그 설정은 필수까지는 아니지만, 향후 발생 가능한 문제를 조사하는 데 유용하게 사용될 수 있습니다.



Azure 클라우드에서 실행 중인 클라이언트는 **→→**에 `textmode=1` `console=ttyS0`을 추가해야 합니다.

3. 자동 설치 프로파일을 붙여 넣거나 파일 업로드 필드를 사용합니다.

4. **[생성]**을 클릭하여 저장합니다.

5. 업로드한 프로파일에 변수를 설정해야 하는 경우 **시스템** > **자동 설치** > **프로파일**로 이동하여 편집할 프로파일을 선택하고 **→→** 탭으로 이동합니다.

다음 형식을 사용하여 필수 변수를 지정합니다.

```
<key>=<value>
```

5.1.2. 마이그레이션

시작하기 전, 자동 설치 프로파일에서 참조된 모든 채널을 사용할 수 있고 완전히 동기화되었는지 확인합니다.

`/var/log/rhn/reposync/<channel-label>.log`에서 미러링 프로세스를 모니터링할 수 있습니다.

절차: 마이그레이션

1. Uyuni 서버 Web UI에서 **→→**으로 이동하여 업그레이드할 클라이언트를 선택합니다.
2. **→→** 탭으로 이동하여 업로드한 자동 설치 프로파일을 선택합니다.
3. **[자동 설치 일정 잡기 후 완료]**를 클릭합니다. 시스템이 필요한 파일을 다운로드하고, 부트로더 항목을 변경하고, 재부팅한 후 업그레이드를 시작합니다.

클라이언트는 다음번에 Uyuni 서버와 동기화될 때 재설치 작업을 수행합니다. 재설치 작업은 새로운 커널 및 `initrd` 패키지를 가져옵니다. 또한 새 커널 및 `initrd` 패키지에 대한 포인터가 포함된 새로운 `/boot/grub/menu.lst` (GRUB Legacy) or `/boot/grub2/grub.cfg` (GRUB 2)를 작성합니다.

클라이언트는 다음번 부팅 시 `grub`을 사용해 `initrd`로 새 커널을 부팅합니다. 이 프로세스 중에는 PXE 부팅이 사용되지 않습니다.

작업을 가져온 후 약 3분이 지나면 클라이언트는 재부팅을 시작합니다.



- 클라이언트의 경우 마이그레이션이 완료된 이후에 `spacewalk/minion_script` 조각을 사용하여 클라이언트를 다시 등록합니다.

5.2. 컨텐트 라이프싸이클 관리자를 사용하여 업그레이드

관리할 SUSE Linux Enterprise Server 클라이언트가 많은 경우 컨텐트 라이프싸이클 관리자를 사용하여 현재 위치에 업그레이드를 자동화할 수 있습니다.

5.2.1. 업그레이드 준비

클라이언트를 업그레이드하려면 먼저 다음과 같은 준비 작업을 마쳐야 합니다.

- 새 컨텐트 라이프싸이클 프로젝트 만들기
- 활성화 키 생성
- 자동 설치 가능한 배포판 생성
- 자동 설치 프로파일 생성

절차: 컨텐트 라이프싸이클 프로젝트 생성

1. 배포를 위한 컨텐트 라이프싸이클 프로젝트를 생성합니다.

자세한 내용은 **Administration > Content-lifecycle**을 참조하십시오.

2. 프로젝트에 대해 설명하는 짧은 이름을 선택해야 합니다.
3. 배포에 필요한 모든 소스 채널 모듈을 포함합니다.
4. 필요에 따라 필터를 추가하고 하나 이상의 환경을 설정합니다.

절차: 활성화 키 생성

1. 배포를 위한 활성화 키를 생성합니다.

자세한 내용은 **Client-configuration > Activation-keys**에서 확인할 수 있습니다.

2. 활성화 키에 필터링된 프로젝트 채널이 모두 포함되어 있어야 합니다.

절차: 자동 설치 가능한 배포 생성

1. 마이그레이션하려는 모든 기본 채널에 대해 자동 설치 가능한 배포판을 생성합니다.

자세한 설명은 **Client-configuration > Autoinst-distributions**를 참조하십시오.

2. 컨텐트 라이프싸이클 프로젝트의 이름을 참조하는 레이블을 배포에 지정합니다.
3. `Autoinst` 필드에서 사용 중인 SLES 버전을 선택합니다.

절차: 자동 설치 프로파일 생성

- 업그레이드하여 도달하려는 모든 대상 배포 및 서비스 팩에 대해 자동 설치 프로파일을 생성합니다.

자세한 설명은 **Client-configuration > Autoinst-profiles**를 참조하십시오.

- 프로파일의 변수를 사용해 다양한 라이프사이클 환경을 구별할 수 있습니다.

자동 설치 프로파일의 예는 <https://github.com/SUSE/manager-build-profiles/tree/master/AutoYaST>를 참조하십시오.

현재 위치에 업그레이드를 자동화하기 위해 자동 설치 프로파일에서 다음 변수를 사용합니다.

목록 1. 예: 자동 설치 프로파일에서 사용할 변수

```
registration_key=1-15sp1-demo-test
org=1
channel_prefix=15sp1-demo-test
distro_label=15sp1-demo-test
```

목록 2. 예: 자동 설치 프로파일에서 사용할 항목

```
<listentry>
  <ask_on_error config:type="boolean">true</ask_on_error>

  <media_url>https://$redhat_management_server/ks/dist/child/$channel_prefix-sle-module-web-scripting15-sp1-pool-x86_64/$distro_label</media_url>
    <name>$channel_prefix SLE-Module-Web-Scripting15-SP1 Pool for x86_64
  </name>
    <product>Web Scripting Module 15 SP1 x86_64 Pool</product>
</listentry>
```

5.2.2. 업그레이드

업그레이드를 위해 서버 준비를 완료했으면 이제 클라이언트를 조달할 수 있습니다.

절차: 클라이언트 조달

- Uyuni Web UI에서 **시스템 > 시스템 목록**으로 이동하여 프로비저닝할 클라이언트를 선택하고 선택한 클라이언트에 시스템 세트 관리자를 추가합니다.
- 시스템 > 시스템 세트 관리자 > 개요**로 이동하여 **----- 탭**을 클릭합니다.
- 사용할 자동 설치 프로파일을 선택합니다.

PXE를 사용할 수 있는 클라이언트의 경우 마이그레이션은 클라이언트를 조달하자마자 자동화됩니다. 기타 모든 클라이언트의 경우에는 Cobbler를 사용해 업그레이드를 수행할 수 있습니다.

절차: Cobbler를 사용한 클라이언트 업그레이드

- 명령 프롬프트에서 루트 권한으로 컨테이너 내부의 셸로 이동하려면 다음을 실행합니다.

```
mgrctl term
```

- 사용할 수 있는 Cobbler 프로파일을 확인합니다.

```
cobbler 프로파일 목록
```

- 다음과 같이 선택한 프로파일 및 배포로 ISO 파일을 빌드합니다.

```
cobbler buildiso --iso=/tmp/SLE_15-sp1.iso --profiles=SLE_15-sp1:1:Example --distro=SLE_15-sp1
```

CD-ROM을 사용하여 클라이언트를 프로비저닝하는 방법은 **Client-configuration > Autoinst-cdrom**을 참조하십시오.

5.3. 제품 마이그레이션

제품 마이그레이션을 통해 SLE 기반 클라이언트 시스템을 서비스 팩(SP) 수준에서 이후 수준으로 업그레이드할 수 있습니다. 예를 들어, SUSE Linux Enterprise Server 15 SP1을 SUSE Linux Enterprise Server 15 SP2로 마이그레이트할 수 있습니다.

제품 마이그레이션은 동일한 주 버전 내에서 업그레이드하기 위한 것입니다. 제품 마이그레이션을 사용하여 SUSE Linux Enterprise Server 12에서 SUSE Linux Enterprise Server 15으로 마이그레이트할 수는 없습니다. 주 업그레이드에 대한 자세한 설명은 **Client-configuration > Client-upgrades-major**를 참조하십시오.

또한 openSUSE Leap을 이후 부 버전으로 또는 해당 SUSE Linux Enterprise Server SP 수준으로 마이그레이트할 수 있습니다. 예는 다음과 같습니다.

- openSUSE Leap 15.1에서 15.2로
- openSUSE Leap 15.1에서 SUSE Linux Enterprise Server 15 SP1으로
- openSUSE Leap 15.5에서 SUSE Linux Enterprise Server 15 SP5로

SUSE Customer Center가 제공하는 경우 SUSE는 SUSE Linux Enterprise Server 12 이상 버전에서 서비스 팩 건너뛰기를 지원합니다. 예를 들어, SP1을 설치하지 않아도 SUSE Linux Enterprise Server 15에서 SP2로 업그레이드할 수 있습니다.

지원되는 SUSE Linux Enterprise Server 업그레이드 경로에 대한 내용은 <https://documentation.suse.com/en-us/sles/15-SP4/html/SLES-all/cha-upgrade-paths.html#sec-upgrade-paths-supported>에서 확인할 수 있습니다.



마이그레이션 과정에서 Uyuni는 설치 전에 필요한 모든 라이선스(EULA)를 자동으로 수락합니다.

5.3.1. 단일 시스템 마이그레이션

제품 마이그레이션을 시작하기 전에:

- 보류 중인 업데이트 또는 패치가 없는지 확인합니다. 클라이언트 시스템 세부 정보 > 개요 페이지에서 해당 항목을 확인하고 제공된 모든 업데이트 또는 패치를 설치합니다. 클라이언트 시스템이 최신이 아닌 경우 제품 마이그레이션이 실패할 수 있습니다.
- 대상 제품의 모든 채널이 완전히 동기화되었는지 확인합니다. Web UI에서 동기화 상태를 확인하려면, 관리자 > 설정 마법사 > 제품 페이지로 이동합니다.
- 비상 상황에서 사용할 수 있는 시스템 백업이 있는지 확인하십시오. 제품 마이그레이션에는 룰백 기능이 없습니다. 마이그레이션 절차가 시작되면 룰백이 불가능합니다.

절차: 단일 시스템 마이그레이션 수행

- 시스템 > 개요 페이지에서 클라이언트를 선택합니다.
- 클라이언트의 시스템 정보 페이지에서 소프트웨어 > 제품 마이그레이션 탭으로 이동합니다.
- 대상 마이그레이션 경로를 선택하고 [채널 선택]을 클릭합니다.
- 제작자 목록에서 원하는 채널을 선택합니다. 기본 채널은 기본 채널입니다.
- 선택 사항: 원하는 채널을 선택하여 벤더가 변경된 패키지를 설치할 수 있도록 허용합니다. 이 경우 마이그레이션이 시작되기 전에 세부 정보와 함께 알림이 표시됩니다.



openSUSE Leap을 SUSE Linux Enterprise Server로 마이그레이트하려면 해당 옵션을 선택해야 합니다.

- 채널이 적절히 구성되었으면 [마이그레이션 예약]을 클릭합니다.

5.3.2. 제품 대량 마이그레이션

다수의 클라이언트를 다음 SP 버전으로 마이그레이션하고 싶은 경우 Uyuni API 호출을 사용하면 됩니다.

spacecmd 명령줄 도구는 system_scheduleproductmigration 하위 명령을 제공하며, 이 하위 명령을 사용하여 많은 수의 클라이언트에 대해 다음 부 버전으로의 마이그레이션을 예약할 수 있습니다.

5.3.2.1. 제품 대량 마이그레이션 수행



제품 대량 마이그레이션 작업은 위험하므로 프로세스를 철저히 테스트해야 합니다. 최소한으로 시험 실행을 우선 수행하십시오.

실수로 시스템을 업그레이드하지 않도록 주의하십시오.

절차: 제품 대량 마이그레이션 수행

- 다음과 같이 사용 가능한 마이그레이션 대상을 나열하고, 마이그레이션하려는 시스템 ID를 적어 둡니다.

```
spacecmd api -- system.listMigrationTargets -A 1000010001
```

2. 각 시스템 ID에 대해 `listMigrationTarget`을 호출하고 원하는 대상 제품이 사용 가능한지 확인합니다.

- 시스템 ID에 사용 가능한 대상이 있는 경우 `system.scheduleProductMigration`을 호출합니다.
- 원하는 대상을 사용할 수 없는 경우 시스템을 건너뛰십시오.

3. 다음 템플릿을 환경에 맞게 조정합니다.

```
target = '....'
basechannel = 'channel-label'
system_ids = [1, 2, 3]

session = auth.login(user, pass)
for system in system_ids
    if system.listMigrationTargets(session, system).ident == target
        system.scheduleProductMigration(session, system, target,
basechannel, [], False, <now>)
    else
        print "요청된 대상으로 마이그레이션할 수 없음 -- 시스템 건너뛰기"
    endif
endfor
```

5.3.2.2. 예: SLES 15 SP2에서 SLES 15 SP3로

이 예에서는 대량 마이그레이션을 편리하게 수행할 수 있도록 그룹이 임시로 생성됩니다.

절차: 대량 제품 마이그레이션 그룹 생성

1. Uyuni Web UI에서 **시스템** > **시스템 그룹**으로 이동하여 **[그룹 생성]**을 클릭합니다.
2. `mpm-target-sles15sp3` 그룹의 이름을 지정하십시오.
 - 동일한 기본 채널에 등록된 시스템만 생성된 그룹에 추가해야 합니다. 이 예에서는 `SLE-Product-SLES15-SP2-Pool for x86_64`에 등록된 시스템만 그룹에 추가되었습니다.

그룹에 클라이언트를 추가하는 방법에 대한 자세한 내용은 **Client-configuration** > **System-groups**에서 확인할 수 있습니다.

절차: 시스템을 그룹에 추가

1. 그룹 내의 모든 시스템에 대한 대상을 가져오려면 다음을 실행합니다.

```
spacecmd -- system_listmigrationtargets group:mpm-target-sles15sp3
```

2. 이 명령을 수행하면 "ID" 스트링이 출력됩니다.

- **모든** 시스템에 대해 보고되는 대상만 선택하십시오.

- 스트링은 다른 명령의 MIGRATIONTARGET에 대한 식별자입니다.



spacecmd 하위 명령 system_scheduleproductmigration 및 system_listmigrationtargets는 그룹 일부에 해당하는 모든 시스템을 반복합니다.

그룹에 시스템 100개가 있는 경우 작업 100개가 예약으로 표시됩니다.

그룹의 모든 시스템은 동일한 마이그레이션 대상을 지원해야 합니다.

절차: 대량 마이그레이션 명령 실행

- system_scheduleproductmigration 명령의 구문은 다음과 같습니다.

```
spacecmd -- system_scheduleproductmigration <SYSTEM>
<BASE_CHANNEL_LABEL> \
    <MIGRATION_TARGET> [옵션]
```

- 이 예에서 mpm-target-sles15sp3 그룹의 모든 시스템을 SLES 12 SP2에서 SLES 15 SP로 업그레이드하려면 명령줄에 다음을 입력:

```
spacecmd -- system_scheduleproductmigration group:mpm-target-
sles15sp3 \
    sle-product-sles15-sp3-pool-x86_64 "[190,203,195,1242]" -d
```

5.3.2.2.1. 필수 구문 설명

system_scheduleproductmigration에 대한 구문 사용 및 옵션을 보려면 다음을 실행하십시오.

```
spacecmd system_scheduleproductmigration help
```

<시스템>

이 예에서는 해당 그룹에서 모든 시스템을 선택하기 위해 생성한 그룹을 사용합니다.

```
group:mpm-target-sles15sp3
```

<BASE_CHANNEL_LABEL>

대상 기본 채널의 레이블입니다. 이 경우 시스템은 SLES 15 SP3로 업그레이드되고 레이블은 sle-product-sles15-sp3-pool-x86_64입니다.

현재 미러링되는 모든 기본 채널 목록을 보려면 다음을 실행:

```
spacecmd softwarechannel_listbasechannels.
```

현재 기본 채널에 대해 사용할 수 있는 대상이 아닌 경우 채널로 업그레이드할 수 없음에 유의하십시오.

<MIGRATION_TARGET>

group:mpm-target-sles15sp3 그룹의 시스템에 대해 이 값을 식별하려면 다음을 실행:

```
spacecmd -- system_listmigrationtargets group:mpm-target-sles15sp3
```

MIGRATION_TARGET 매개 변수가 전달되어야 하는 형식이며, 괄호 사용으로 인한 부작용을 방지하기 위해 셸 인용구 필요:

```
"[190,203,195,1242]"
```

옵션

- -s START_TIME
- 시험 실행을 원할 경우 -d가 이 플래그를 전달합니다(실제 마이그레이션을 수행하기 전 시험 실행을 실행하는 것이 권장됨).
- -c CHILD_CHANNELS(공백 없이 쉼표로 구분된 하위 채널 레이블)

이 경우 성공적인 시험 실행 후 제거할 수 있는 -d 옵션이 포함되었습니다.

성공하면 예약된 각 시스템의 명령 출력은 다음과 같습니다.

```
mpm-sles152-1 시스템에 대한 제품 마이그레이션 예약
예약된 작업 ID: 66
```

그룹의 지정된 시스템에 대해 Web UI에서 작업(이 경우 시험 실행)을 추적할 수도 있습니다. 클라이언트의 시스템 세부 정보 페이지에서 **이벤트** > **이력**으로 이동합니다. 시험 실행 중 오류가 발생하면 시스템을 조사해야 합니다.

모두 정상이면 명령에서 -d 옵션을 제거하여 실제 마이그레이션을 실행할 수 있습니다. 마이그레이션이 완료되면 Uyuni Web UI에서 시스템을 재부팅할 수 있습니다.

5.4. Uyuni 클라이언트 업그레이드

이 섹션에서는 openSUSE Leap을 예로 사용합니다.

5.4.1. 업그레이드 준비

절차: 클라이언트 업그레이드 준비

- Uyuni 서버의 명령 프롬프트에서 루트 권한으로 spacewalk-common-channels 명령을 사용해 적절한 채널을 추가합니다.

```
spacewalk-common-channels \
opensuse_leap15_4 \
opensuse_leap15_4-non-oss \
opensuse_leap15_4-non-oss-updates \
opensuse_leap15_4-updates \
opensuse_leap15_4-uyuni-client
```

- spacewalk-repo-sync로 모든 채널을 완전히 동기화합니다. 이미 정의된 리포지토리 URL인 경우 [installation-and-upgrade:proxy-uyuni.pdf](#)로 계속합니다.
- Uyuni 서버 Web UI에서 **소프트웨어 > 관리 > 채널**로 이동하여 이름이 Uyuni Client Tools for openSUSE Leap 15.4(x86_64)인 채널을 클릭합니다.
- 오른쪽 상단에서 **[채널 관리]**를 클릭합니다.
- 선택** 탭을 클릭하고 **- openSUSE Leap 15.3(x86_64)** - Uyuni **선택**을 선택합니다.
- [리포지토리 업데이트]**를 클릭합니다.
- 리포지토리 > 동기화** 하위 탭으로 이동하여 **[지금 동기화]**를 클릭합니다.
- openSUSE Leap 15.4(x86_64) 및 **- openSUSE Leap 15.3(x86_64)**에 대해서도 동일한 작업을 합니다.

openSUSE Leap 15.4(x86_64)를 펼쳐 패키지로 채워져 있는 모든 하위 채널을 표시합니다.

5.4.2. 업그레이드

클라이언트를 업그레이드하려면 소프트웨어 리포지토리를 교체하고 소프트웨어를 업데이트한 후 끝으로 클라이언트를 재부팅하십시오.

절차: 클라이언트 업그레이드

- Uyuni 서버 Web UI에서 **시스템**으로 이동하여 클라이언트의 이름을 클릭합니다.
- 소프트웨어 > 소프트웨어 채널**을 클릭하고 Custom Channels 목록에 나열된 openSUSE Leap 15.5 채널을 기본 채널로 선택합니다.
- 창**에서 15.5 하위 채널을 선택합니다.
- [다음]**을 클릭하고 **선택**에서 **[확인]**을 클릭합니다.
- 소프트웨어 > 패키지 > 업그레이드**를 클릭하고 클라이언트에서 업데이트할 패키지를 모두 선택한 다음, 선택을 적용합니다. **[패키지 업그레이드]**를 클릭하고, 상세 정보를 확인한 후 **[확인]**을 클릭하여 업데이트를 완료합니다.
- 클라이언트를 재부팅합니다.

여러 클라이언트를 업데이트해야 하는 경우 Uyuni 서버에서 이 명령 시퀀스의 작업 체인을 생성하면 됩니다. 이 작업 체인을 사용해 여러 클라이언트에서 업데이트를 동시에 수행할 수 있습니다.

Chapter 6. 클라이언트 삭제

Uyuni 서버에서 클라이언트를 제거해야 하는 경우 다음을 수행할 수 있습니다.

- Web UI를(를) 사용하여 삭제
- 명령 줄에서 클라이언트를 제거합니다.

6.1. Web UI를 사용한 클라이언트 삭제

절차: 클라이언트 삭제

1. Uyuni Web UI에서 **시스템 > 시스템 목록**으로 이동하여 삭제하려는 클라이언트를 선택합니다.
2. **[시스템 삭제]**를 클릭합니다.
3. 상세 정보를 확인하고 **[프로파일 삭제]**를 클릭하여 확인합니다.
4. Salt 클라이언트의 경우 Uyuni는 추가 구성은 정리하려 합니다. 클라이언트에 연결할 수 없는 경우 삭제를 취소하거나 구성 파일을 정리하지 않고 클라이언트를 삭제할 수 있는 옵션이 제공됩니다.

시스템 세트 관리자를 사용하여 여러 클라이언트를 삭제할 수도 있습니다. 시스템 세트 관리자에 대한 자세한 내용은 **Client-configuration > System-set-manager**에서 확인할 수 있습니다.



클라이언트를 정리하면 Salt만 비활성화하고 가능한 경우 서비스를 중지합니다. 패키지는 설치 해제되지 않습니다.

6.2. 명령 줄에서 클라이언트 삭제(API 호출 사용)

절차: 서버에서 클라이언트 삭제

1. FQDN(정규화된 도메인 이름)이 있는 클라이언트를 삭제합니다.

```
spacecmd system_delete FQDN
```

spacecmd system_delete는 Salt 키도 삭제합니다.

system_delete에서 제공되는 옵션은 다음과 같습니다.

```
usage: system_delete [options] <SYSTEMS>
```

옵션:

- c TYPE - 사용할 수 있는 값:
 - * 'FAIL_ON_CLEANUP_ERR' - 정리 오류 발생 시 실패,
 - * 'NO_CLEANUP' - 삭제만 수행하고 정리하지 않음,
 - * 'FORCE_DELETE' - 우선 정리를 시도하지만, 오류 발생 시 서버 삭제

6.3. 명령 줄에서 클라이언트 삭제



이 프로세스는 Uyuni 클라이언트에만 적용됩니다. Uyuni 서버 자체에서는 실행하지 마십시오.



Red Hat Enterprise Linux, Debian 또는 클론을 실행하는 클라이언트에서 다음 절차를 실행해서는 안 됩니다. zypper 대신, yum, dnf 또는 apt와 같은 패키지 작성 도구 명령을 사용하십시오.

절차: SLES 12 및 15 클라이언트 삭제

1. 다음과 같이 salt-minion 서비스를 중지합니다.

```
systemctl stop salt-minion
```

2. 리포지토리 및 구성 파일 제거:

```
rm /etc/zypp/repos.d/susemanager\channels.repo
rm -r /etc/sysconfig/rhn/
rm -r /etc/salt/
```

3. 클라이언트 패키지 제거:

```
zypper rm salt salt-minion python*-salt sle-manager-tools-release
```

절차: Salt 번들 클라이언트 - 수동 등록 정리

1. 등록을 취소하려면 다음을 실행합니다.

```
systemctl stop venv-salt-minion
zypper rm -y venv-salt-minion
rm /etc/zypp/repos.d/susemanager\channels.repo /etc/venv-salt-
minion/*
rm -r /etc/venv-salt-minion/*
```

Salt 번들에 대한 내용은 [Client-configuration > Contact-methods-saltbundle](#)에서 확인할 수 있습니다.

Chapter 7. 클라이언트 작업

클라이언트의 등록, 업그레이드, 소프트웨어 설치 또는 삭제 작업 외에 다른 작업도 수행할 수 있습니다. Uyuni 클라이언트는 개별적으로 또는 클라이언트 세트로 관리할 수 있습니다. 클라이언트 세트는 시스템 세트 관리자를 사용하여 일회성 작업을 위해 그룹화하거나 시스템 그룹을 사용하여 영구적으로 그룹화할 수 있습니다.

고급 클라이언트 구성 옵션에서는 일반 구성 관리를 사용할 수 있습니다. 사용자 지정 시스템 정보를 가져오고, 전원을 켜거나 끄고, Uyuni Web UI(를) 사용하여 클라이언트를 재부팅할 수 있습니다.

다음 섹션에서는 이러한 각 작업에 대한 자세한 설명을 제공합니다.

7.1. 패키지 관리

클라이언트는 패키지를 사용해 소프트웨어를 설치, 제거 및 업그레이드합니다.



패키지가 설치되거나 업그레이드되면 라이선스 또는 EULA가 자동으로 수락됩니다.

클라이언트에서 패키지를 관리하려면 →→으로 이동하여 관리할 클라이언트를 클릭한 후 **시스템** > **소프트웨어** > **패키지** 하위 탭으로 이동하십시오. 이 섹션에서 사용 가능한 옵션은 선택한 클라이언트의 유형과 현재 채널 구독에 따라 달라집니다.

대부분의 패키지 관리 작업을 작업 체인에 추가할 수 있습니다. 작업 체인에 대한 자세한 내용은 **Reference** > **Schedule**을 참조하십시오.

7.1.1. Compare Packages Using Profiles

클라이언트에 설치한 패키지를 저장한 프로파일 또는 다른 클라이언트에 설치한 패키지와 비교할 수 있습니다. 비교할 때 선택한 클라이언트가 일치하도록 수정하는 쪽을 선택할 수 있습니다.

패키지를 프로파일과 비교하려면 프로파일을 이미 저장했어야 합니다. 프로파일은 현재 설치되어 있는 클라이언트의 패키지에서 생성합니다. 프로파일을 생성했으면 이 프로파일을 사용해 동일한 패키지가 설치된 클라이언트를 추가 설치할 수 있습니다.

절차: 저장 프로파일 생성

1. Uyuni Web UI에서 →→으로 이동하여 프로파일이 기반을 두고 있는 클라이언트를 클릭하고 **시스템** > **소프트웨어** > **패키지** > **프로파일** 하위 탭으로 이동합니다.
2. **[시스템 프로파일 생성]**을 클릭합니다.
3. 프로파일 이름 및 설명을 입력하고 **[프로파일 생성]**을 클릭합니다.

절차: 클라이언트 패키지 비교

1. Uyuni Web UI에서 →→으로 이동하여 비교할 클라이언트를 클릭하고 **시스템** > **소프트웨어** > **패키지** > **프로파일** 하위 탭으로 이동합니다. 저장된 프로파일과 비교하려면 프로파일을 선택하고 **[비교]**를 클릭하십시오.
2. 다른 클라이언트와 비교하려면 클라이언트 이름을 선택하고 **[비교]**를 클릭하여 두 클라이언트 간의 차이점 목록을 살펴봅니다.

7.2. 패치 관리

조직 내 사용자 정의 패치를 사용해 클라이언트를 관리할 수 있습니다. 이를 통해 사용자 정의 채널에 있는 패키지에 대해 패치 경고를 발급하고, 패치 설치 일정을 잡고, 여러 조직에 걸쳐 패치를 관리할 수 있습니다.

7.2.1. 패치 생성

사용자 정의 패치를 사용하려면 패치를 생성하여 패키지를 추가하고 패치를 하나 이상의 채널에 추가하십시오.

절차: 사용자 정의 패치 생성

1. Uyuni Web UI에서 **패치 > 패치 관리**로 이동하여 **[패치 생성]**을 클릭합니다.
2. **---** 섹션에서 다음과 같은 상세 정보를 사용합니다.
 - **--** 필드에 패치에 대한 짧은 설명을 입력합니다.
 - **--** 필드에 패치의 레이블을 입력합니다. 조직이 패치 관리를 더 쉽게 할 수 있도록 명명 규칙을 고안하는 것이 좋습니다.
 - **--** 필드에 패치의 릴리스 번호를 입력합니다. 예를 들어 해당 패치의 첫 버전인 경우 1을 사용하십시오.
 - **--** 필드에서 사용할 패치의 유형을 선택합니다. 예를 들어 오류를 수정하는 패치의 경우 **--**입니다.
 - **--**라는 권고 유형을 선택한 경우 **--** 필드에서, 사용할 심각도 수준을 선택하십시오.
 - **--** 필드에서 이 패치가 참조하는 제품의 이름을 입력합니다.
 - 옵션: **--** 필드에서 패치 원저자의 이름을 입력합니다.
 - **--**, **--**, **--** 필드를 패치에 대한 추가 정보로 완성합니다.
3. 옵션: **---** 섹션에서 다음 상세 정보를 사용해 관련이 있는 모든 버그에 관한 정보를 지정합니다.
 - **ID** 필드에 버그 번호를 입력합니다.
 - **--** 필드에 버그에 대한 짧은 설명을 입력합니다.
 - **--** URL 필드에 버그에 대한 짧은 설명을 입력합니다.
 - **--** 필드에 버그와 관련이 있는 모든 키워드를 입력합니다. 각 키워드 사이에는 쉼표를 사용하십시오.
 - **--** 및 **--** 필드를 버그에 대한 추가 정보로 완성합니다.
 - 새 패치를 추가할 채널을 한 개 이상 선택합니다.
4. **[패치 생성]**을 클릭합니다.

기존 패치를 복제하여 패치를 생성할 수도 있습니다. 복제를 통해 패키지 연결이 유지되고 패치 발급이 간소화됩니다.

절차: 패치 복제

1. Uyuni Web UI에서 **패치 > 패치 복제**로 이동합니다.
2. **---** 필드에서 복제하려는 패치에 대한 소프트웨어 채널을 선택합니다.
3. 복제하려는 패치를 한 개 또는 여러 개 선택하고 **[패치 복제]**를 클릭합니다.
4. 복제한 패치를 추가할 채널을 한 개 이상 선택합니다.

5. 상세 정보를 확인하여 복제를 시작합니다.

패치를 생성하였으면 이 패치에 패키지를 할당할 수 있습니다.

절차: 패키지를 패치에 할당

1. Uyuni Web UI에서 **패치** > **패치 관리**로 이동하여 패치 상세 정보를 확인할 패치의 권고 이름을 클릭합니다.
2. **패키지** > **추가** 탭으로 이동합니다.
3. -- 필드에서 패치에 할당하려는 패키지가 포함된 소프트웨어 채널을 선택하고 **[패키지 보기]**를 클릭합니다.
--> --> --> 를 선택하여 모든 채널에서 사용 가능한 패키지를 확인할 수 있습니다.
4. 포함하려는 패키지의 확인란을 선택하고 **[패키지 추가]**를 클릭합니다.
5. 패키지의 상세 정보를 확인하고, **[확인]**을 클릭하여 패키지를 패치에 할당합니다.
6. **패키지** > **나열/제거** 탭으로 이동하여 패키지가 올바르게 할당되었는지 확인합니다.

패키지가 패치에 할당되면 패치 캐시가 업데이트되어 변경 사항을 반영합니다. 캐시 업데이트에 몇 분 정도 걸릴 수 있습니다.

기존 패치의 상세 정보를 변경해야 하는 경우 -- -- 페이지에서 변경할 수 있습니다.

절차: 기존 패치 경고 편집 및 삭제

1. Uyuni Web UI에서 **패치** > **패치 관리**로 이동합니다.
2. 패치의 권고 이름을 클릭하여 패치 상세 정보를 확인합니다.
3. 필요에 따라 변경하고 **[패치 업데이트]**를 클릭합니다.
4. 패치를 삭제하려면 -- -- 페이지에서 패치를 선택하고 **[패치 삭제]**를 클릭합니다. 패치를 삭제하는 데 몇 분이 걸릴 수 있습니다.

7.2.2. 패치를 클라이언트에 적용

패치가 준비되면 이 패치만 클라이언트에 적용하거나 다른 패치와 함께 적용할 수 있습니다.

패치 내 각 패키지는 한 개 이상의 채널에 속해 있습니다. 클라이언트가 해당 채널에 가입되어 있지 않으면 업데이트가 설치되지 않습니다.

이미 설치된 패키지보다 더 최근에 나온 버전이 클라이언트에 있는 경우 업데이트가 설치되지 않습니다. 설치된 패키지 이전에 나온 버전이 클라이언트에 있는 경우 패키지가 업그레이드됩니다.

절차: 해당하는 모든 패치 적용

1. Uyuni Web UI에서 **시스템** > **개요**로 이동하여 업데이트하려는 클라이언트를 선택합니다.
2. **소프트웨어** > **패치** 탭으로 이동합니다.
3. **[모두 선택]**을 클릭하여 해당하는 모든 패치를 선택합니다.
4. **[패치 적용]**을 클릭하여 클라이언트를 업데이트합니다.

관리자 권한으로 로그인한 경우 클라이언트에 대해 더 큰 규모의 배치 업그레이드를 수행할 수도 있습니다.

절차: 하나의 패치를 여러 클라이언트에 적용

1. Uyuni Web UI에서 **패치 > 패치 목록**으로 이동합니다.
2. 적용하려는 패치를 찾은 다음, 이 패치에 대한 **...** 열 아래의 숫자를 클릭합니다.
3. 패치를 적용하려는 클라이언트를 선택하고 **[패치 적용]**을 클릭합니다.
4. 업데이트를 수행할 클라이언트의 목록을 확인합니다.

절차: 여러 패치를 여러 클라이언트에 적용

1. Uyuni Web UI에서 **시스템 > 개요**로 이동하여 업데이트하려는 클라이언트를 확인하고 이 클라이언트를 시스템 세트 관리자에 추가합니다.
2. **시스템 > 시스템 세트 관리자**로 이동한 후 **...** 탭으로 이동합니다.
3. 클라이언트에 적용하려는 패치를 선택하고 **[패치 적용]**을 클릭합니다.
4. 업데이트 실행 날짜 및 시간을 예약하고 **[확인]**을 클릭합니다.
5. 업데이트의 진도를 확인하려면 **일정 > 보류 중인 작업**으로 이동합니다.



예약된 패키지 업데이트는 각 클라이언트에 구성한 연결 방법을 사용해 설치됩니다. 자세한 내용은 **Client-configuration > Contact-methods-intro**를 참조하십시오.

7.3. 시스템 잠금

시스템 잠금은 클라이언트에서 작업이 이루어지지 못하게 하는 데 사용됩니다. 예를 들어 시스템 잠금은 클라이언트가 업데이트되거나 다시 시작되지 못하게 합니다. 이 기능은 프로덕션 소프트웨어를 실행하는 클라이언트나 실수로 인한 변경 방지에 유용합니다. 작업을 수행할 준비가 되었으면 시스템 잠금을 비활성화할 수 있습니다.

7.3.1. 클라이언트의 시스템 잠금

클라이언트가 잠겨 있거나 차단 모드 상태인 경우 작업을 예약할 수 없고, 실행 명령이 비활성화되며, **...** **...** 페이지에 노란색 배너가 표시됩니다. 이 모드에서는 Web UI 또는 API를 사용해 잠긴 클라이언트에 대해 작업을 예약할 수 있지만 작업이 실패합니다.



Salt SSH 클라이언트에 대해서는 잠금 메커니즘을 사용할 수 없습니다.

절차: 클라이언트의 시스템 잠금

1. Uyuni Web UI에서 잠그려는 클라이언트의 **...** **...** 페이지로 이동합니다.
2. **...** 탭으로 이동하여 시스템 잠금 수식의 확인란을 선택한 후 **[저장]**을 클릭합니다.
3. **수식 > 시스템 잠금** 탭으로 이동하여 **...** **...**의 확인란을 선택한 후 **[저장]**을 클릭합니다. 이 페이지에서는 클라이언트가 잠겨 있는 동안 특정 Salt 모듈을 활성화할 수도 있습니다.
4. 변경한 후 **highstate**를 적용해야 할 수 있습니다. 이 경우 Web UI의 배너가 이를 알려줍니다. 시스템 잠금 수식을 제거할 때까지 클라이언트는 잠긴 상태를 유지합니다.

Salt의 차단 모드에 대한 자세한 내용은 <https://docs.saltstack.com/en/latest/topics/blackout/index.html>에서 확인할 수 있습니다.

7.3.2. 패키지 잠금

패키지 잠금은 여러 클라이언트에서 사용할 수 있지만, 서로 다른 기능 집합을 사용할 수 있습니다. SUSE Linux Enterprise 및 openSUSE(zypp 기반) 클라이언트와 Red Hat Enterprise Linux 또는 Debian 클라이언트를 구분해야 합니다.

7.3.2.1. Zypp 기반 시스템의 패키지 잠금

패키지 잠금은 무단 설치나 소프트웨어 패키지로 업그레이드하는 것을 방지하는 데 사용됩니다. 패키지가 잠겨 있으면 자물쇠 아이콘이 표시되어 설치할 수 없음을 나타냅니다. 잠긴 패키지를 설치하려는 모든 시도는 이벤트 로그에 오류로 보고됩니다.

잠긴 패키지는 Uyuni Web UI를 통해 또는 패키지 관리자를 사용하는 클라이언트 시스템에서 직접 설치, 업그레이드 또는 제거할 수 없습니다. 잠긴 패키지는 모든 종속 패키지도 간접적으로 잠깁니다.

절차: 패키지 잠금 사용하기

- 관리되는 시스템에서 **소프트웨어** > **패키지** > **잠금** 탭으로 이동하여 사용 가능한 모든 패키지의 목록을 확인합니다.
- 잠글 패키지를 선택하고 **[잠금 요청]**을 클릭합니다. 잠금을 활성화할 날짜와 시간을 선택합니다. 기본적으로 잠금은 가능한 한 빨리 활성화됩니다. 잠금이 즉시 활성화되지 않을 수 있습니다. 잠금이 즉시 활성화되지 않을 수 있다는 점에 유의하십시오.
- 패키지 잠금을 제거하려면 잠금을 해제할 패키지를 선택하고 **[잠금 해제 요청]**을 클릭하십시오. 잠금을 활성화할 때와 같이 날짜와 시간을 선택합니다.

7.3.2.2. Red Hat Enterprise Linux 및 Debian 유사 시스템의 패키지 잠금



일부 Red Hat Enterprise Linux 및 Debian 유사 시스템에는 클라이언트에서 사용할 수 있는 패키지 잠금이 있습니다.

Red Hat Enterprise Linux 및 Debian과 같은 시스템에서 패키지 잠금은 소프트웨어 패키지에 대한 무단 업그레이드 또는 제거하는 것을 방지하는 데만 사용됩니다. 패키지가 잠겨 있으면 자물쇠 아이콘이 표시되어 변경할 수 없음을 나타냅니다. 잠긴 패키지를 설치하려는 모든 시도는 이벤트 로그에 오류로 보고됩니다.

잠긴 패키지는 Uyuni Web UI를 통해 또는 패키지 관리자를 사용하는 클라이언트 시스템에서 직접 업그레이드 또는 제거할 수 없습니다. 잠긴 패키지는 모든 종속 패키지도 간접적으로 잠깁니다.

절차: 패키지 잠금 사용하기

- Red Hat Enterprise Linux 7 시스템에서는 yum-plugin-versionlock을 설치하고, Red Hat Enterprise Linux 8 또는 9 시스템에서는 python3-dnf-plugin-versionlock을 --로 설치합니다. Debian 시스템에서는 apt 도구에 잠금 기능이 포함되어 있습니다.
- 관리되는 시스템에서 **소프트웨어** > **패키지** > **잠금** 탭으로 이동하여 사용 가능한 모든 패키지의 목록을 확인합니다.
- 잠글 패키지를 선택하고 **[잠금 요청]**을 클릭합니다. 잠금을 활성화할 날짜와 시간을 선택합니다. 기본적으로 잠금은 가능한 한 빨리 활성화됩니다. 잠금이 즉시 활성화되지 않을 수 있습니다. 잠금이 즉시 활성화되지 않을 수 있다는 점에 유의하십시오.
- 패키지 잠금을 제거하려면 잠금을 해제할 패키지를 선택하고 **[잠금 해제 요청]**을 클릭하십시오. 잠금을 활성화할 때와 같이 날짜와 시간을 선택합니다.

7.4. 구성 관리

각 클라이언트를 수동으로 구성하는 대신에 구성 파일 및 채널을 사용해 클라이언트에 대한 구성을 관리할 수 있습니다.

구성 파라미터는 스크립트되고 구성 파일에 저장됩니다. Uyuni Web UI를 사용해 직접 구성 파일을 작성할 수 있습니다. 또는 다른 위치에 있는 파일을 업로드하거나 그러한 파일에 링크할 수 있습니다.

구성 파일은 중앙에서 관리하거나 로컬에서 관리할 수 있습니다. 중앙에서 관리되는 구성 파일은 전역 구성 채널에서 제공되며, Uyuni 서버를 구독하는 모든 클라이언트에 적용할 수 있습니다.

구성 채널은 구성 파일을 체계적으로 정리하는 데 사용됩니다. 클라이언트를 구성 채널에 가입하고 필요에 따라 구성 파일을 배포할 수 있습니다.

구성 파일은 버전이 관리되므로 구성 설정을 추가하고, 이를 클라이언트에서 테스트하고, 필요에 따라 이전 리비전으로 롤백할 수 있습니다. 구성 채널을 생성했다면 다양한 구성 파일 간에, 그리고 동일한 구성 파일의 리비전 간에 비교를 수행할 수도 있습니다.

중앙에서 관리되는 구성 파일은 전역 구성 채널에서 제공합니다.

이 표에는 지원되는 기능이 나와 있으며, 여기서:

- ✓ 기능은 SUSE에서 지원합니다.
- ✗ 기능은 SUSE에서 지원하지 않습니다.
- ? 기능은 현재 지원 고려 중이며, 이후에 지원될 수도 있고 지원되지 않을 수도 있습니다.

표 47. 지원되는 구성 관리 기능

기능	상태
전역 구성 채널	✓
파일 배포	✓
파일 비교	?
로컬로 관리되는 파일	✓(Salt 기능 포함)
샌드박스 파일	✗
Highstat 적용	✓
클라이언트에서 파일 임포트	✗
Jinja 템플릿	✓
구성 매크로	✓(Salt 기능 사용: 입자, 열 데이터 등)

7.4.1. 구성 채널 생성

7.4.1.1. 중앙 구성 채널

새로운 중앙 구성 채널을 생성하려면:

절차: 중앙 구성 채널 생성

1. Uyuni Web UI에서 구성 > 채널로 이동하여 **[구성 채널 생성]**을 클릭합니다.
2. 채널의 이름을 입력합니다.
3. 채널의 레이블을 입력합니다. 이 필드는 글자, 숫자, 하이픈(-), 밑줄(_)만 포함해야 합니다.
4. 다른 채널과 구별할 수 있게 해주는 채널 설명을 입력합니다.
5. **[구성 채널 생성]**을 클릭하여 새 채널을 생성합니다.

7.4.1.2. Salt 상태 채널

구성 채널을 사용하여 클라이언트의 Salt 상태를 관리할 수도 있습니다.

절차: Salt 상태 채널 생성

1. Uyuni Web UI에서 구성 > 채널로 이동하여 **[상태 채널 생성]**을 클릭합니다.
2. 채널의 이름을 입력합니다.
3. 채널의 레이블을 입력합니다. 이 필드는 글자, 숫자, 하이픈(-), 밑줄(_)만 포함해야 합니다.
4. 다른 채널과 구별할 수 있게 해주는 채널 설명을 입력합니다.
5. init.sls 파일에 대해 SLS 그룹을 입력합니다.
6. **[구성 채널 생성]**을 클릭하여 새 채널을 생성합니다.

7.4.2. 구성 파일, 디렉토리 또는 심볼 링크 추가

구성 채널을 생성했으면 이제 구성 파일, 디렉토리 또는 심볼 링크를 추가할 수 있습니다.

절차: 구성 파일, 디렉토리 또는 심볼 링크 추가

1. Uyuni Web UI에서 구성 > 채널로 이동하여 구성 파일을 추가하려는 구성 채널의 이름을 클릭한 후, 파일 추가 > 파일 생성 하위 탭으로 이동합니다.
2. 그룹 필드에서 텍스트 파일, 디렉토리, 심볼 링크 중 생성하려는 것을 선택합니다.
3. 그룹/그룹 필드에 파일을 배포해야 하는 위치의 절대 경로를 입력합니다.
4. 심볼 링크를 생성하려면 그룹 그룹 그룹 그룹 필드에 대상 파일 및 경로를 입력하십시오.
5. 그룹 필드에서 파일의 그룹 그룹 및 그룹 그룹을 입력하고 그룹 그룹 그룹을 입력합니다.
6. 클라이언트가 SELinux를 활성화한 경우 SELinux 그룹을 구성하여 필수 파일 속성(예: 사용자, 역할, 파일 형식)을 활성화할 수 있습니다.
7. 구성 파일에 매크로가 포함된 경우 매크로의 시작과 끝을 표시하는 기호를 입력하십시오.
8. 그룹 그룹 텍스트 상자에 구성 파일 내용을 입력합니다. 이때 스크립트 드롭다운 상자를 사용해 적절한 스크립팅 언어를 선택합니다.
9. **[구성 파일 생성]**을 클릭합니다.

7.4.3. 클라이언트를 구성 채널에 가입

시스템 > 시스템 목록으로 이동하여 가입하려는 클라이언트를 선택하고 그룹 탭으로 이동하여 개별 클라이언트를 구성 채널에 가입할 수 있습니다. 여러 클라이언트를 구성 채널에 가입하려면 시스템 세트 관리자(SSM)를 사용하면 됩니다.

절차: 여러 클라이언트를 구성 채널에 가입

1. Uyuni Web UI에서 **시스템** > **시스템 목록**으로 이동하여 작업하려는 클라이언트를 선택합니다.
2. **시스템** > **시스템 세트 관리자**로 이동하여 **구성** > **채널에 가입** 하위 탭에서 사용 가능한 구성 채널의 목록을 확인합니다.
3. 옵션: ⇧ ⇨ ⇨ 열의 숫자를 클릭하여 현재 어떤 클라이언트가 구성 채널에 가입되어 있는지 확인합니다.
4. 가입하려는 구성 채널을 확인하고 **[계속]**을 클릭합니다.
5. 위 및 아래 방향 화살표를 사용해 구성 채널의 순위를 지정합니다. 설정이 구성 채널 간에 서로 충돌하는 경우 목록 상단에 더 가까이 있는 채널이 우선합니다.
6. 선택한 클라이언트에 채널이 적용되는 방식을 결정합니다. **[최하위 우선순위로 가입]**을 클릭하여 새 채널을 현재 가입된 채널보다 낮은 우선순위로 추가합니다. **[최상위 우선순위로 가입]**을 클릭하여 새 채널을 현재 가입된 채널보다 높은 우선순위로 추가합니다. **[기존 구독 대체]**을 클릭하여 기존 채널을 제거하고 새 채널로 교체합니다.
7. **[구독 적용]**을 클릭합니다.



새 구성 채널 우선순위가 기존 채널과 충돌하는 경우 중복 채널이 제거되고 새로운 우선순위에 따라 교체됩니다. 작업에 의해 클라이언트의 구성 우선순위가 변경되는 경우 Web UI는 작업을 진행하기 전에 이러한 변경을 확인하도록 사용자에게 요구합니다.

7.4.4. 구성 파일 비교

또한 시스템 세트 관리자(SSM)를 사용해 클라이언트에 배포된 구성 파일을 Uyuni 서버에 저장된 구성 파일과 비교할 수 있습니다.

절차: 구성 파일 비교

1. Uyuni Web UI에서 **시스템** > **시스템 목록**으로 이동하여 비교하려는 구성 파일에 가입된 클라이언트를 선택합니다.
2. **시스템** > **시스템 세트 관리자**로 이동하여 **구성** > **파일 비교** 하위 탭에서 사용 가능한 구성 채널의 목록으로 이동합니다.
3. 옵션: ⇧ ⇨ ⇨ 열의 숫자를 클릭하여 현재 어떤 클라이언트가 구성 파일에 가입되어 있는지 확인합니다.
4. 비교할 구성 파일을 확인하고 **[파일 비교 작업 일정 잡기]**를 클릭합니다.

7.4.5. 클라이언트의 Jinja 템플릿

Salt 클라이언트에서 Jinja 템플릿을 사용할 수 있습니다. Jinja는 열 또는 입자에서 변수를 제공합니다. 이러한 변수는 구성 파일 또는 Salt 상태에서 사용할 수 있습니다.

자세한 내용은 다음 예제와 함께 <https://docs.saltproject.io/salt/user-guide/en/latest/topics/jinja.html>에서 확인할 수 있습니다.

```

{% if grains.os_family == 'RedHat' %}
  {% set dns_cfg = '/etc/named.conf' %}
{% elif grains.os_family == 'Debian' %}
  {% set dns_cfg = '/etc/bind/named.conf' %}
{% else %}
  {% set dns_cfg = '/etc/named.conf' %}
{% endif %}

dns_conf:
  file.managed:
    - name: {{ dns_cfg }}
    - source: salt://dns/files/named.conf

```

7.5. 전원 관리

Uyuni Web UI를 사용해 클라이언트를 켜고, 끄고, 다시 부팅할 수 있습니다.

이 기능은 IPMI 또는 Redfish 프로토콜을 사용하며 Cobbler 프로파일을 사용해 관리됩니다. 선택한 클라이언트에는 이러한 프로토콜 중 하나를 지원하는 전원 관리 컨트롤러가 있어야 합니다.

Redfish의 경우 클라이언트와 Uyuni 서버 간의 타당한 SSL 연결을 설정할 수 있는지 확인하십시오. 사용자는 Redfish 관리 컨트롤러의 SSL 서버 인증서에 서명하는 데 사용되는 인증 주체를 신뢰해야 합니다. CA 인증서는 .pem 형식이어야 하며 Uyuni 서버의 /etc/pki/trust/anchors/에 저장됩니다. 인증서 저장을 완료했으면 update-ca-certificate를 실행하십시오.

절차: 전원 관리 활성화

1. Uyuni Web UI에서 **시스템** > **시스템 목록**으로 이동하여 관리하려는 클라이언트를 선택하고 **조달** > **전원 관리** 탭으로 이동합니다.
2. → 필드에서 사용할 전원 관리 프로토콜을 선택합니다.
3. 전원 관리 서버의 상세 정보를 모두 입력하고, 취할 조치에 해당하는 적절한 버튼을 클릭하거나 **[저장만]**을 클릭하여 아무 조치도 취하지 않고 상세 정보를 저장합니다.

여러 클라이언트를 시스템 세트 관리자에 추가함으로써 전원 관리 조치를 여러 클라이언트에 동시에 적용할 수 있습니다. 시스템 세트 관리자에 대한 자세한 내용은 **Client-configuration** > **System-set-manager**를 참조하십시오.

7.5.1. 전원 관리 및 Cobbler

전원 관리 기능을 처음 사용할 때 Cobbler 시스템 레코드가 자동으로 생성됩니다(클라이언트에 아직 없는 경우). 이렇게 자동으로 생성된 시스템 레코드는 네트워크에서 부팅할 수 없고 더미 시스템 이미지에 대한 참조를 포함합니다. 이 참조가 필요한 이유는 현재 Cobbler가 프로파일 또는 이미지 없는 시스템 레코드를 지원하지 않기 때문입니다.

Cobbler 전원 관리는 펜스 에이전트 도구를 사용해 IPMI가 아닌 프로토콜을 지원합니다. Uyuni는 IPMI 및 Redfish 프로토콜만 지원합니다. rhn.conf 구성 파일의 java.power_management.types 구성 파라미터에 펜스 에이전트 이름을 쉼표로 구분된 목록으로 추가하여 클라이언트가 다른 프로토콜을 사용하도록 구성할 수 있습니다.

7.6. 사용자 정의 시스템 정보

클라이언트에 대한 사용자 정의 시스템 정보를 포함할 수 있습니다. 시스템 정보는 클라이언트에 할당할 수 있는 key:value 쌍으로 정의됩니다. 예를 들어, 특정 프로세서에 대한 key:value pair를 정의한 다음 해당 프로세서가 설치된 모든 클라이언트에 해당 키를 할당할 수 있습니다. 사용자 정의 시스템 정보는 분류되어 Uyuni Web UI를 사용하여 검색할 수 있습니다.

시작하기 전에 사용자 정의를 저장할 수 있게 허용하는 키를 생성해야 합니다.

절차: 사용자 정의 시스템 정보 키 생성

1. Uyuni Web UI에서 **시스템 > 사용자 정의 시스템 정보**로 이동하여 **[키 생성]**을 클릭합니다.
2. ↵ 필드에 키의 이름을 추가합니다. 공백을 사용하면 안 됩니다. 예를 들어, intel-x86_64-quadcore를 추가합니다.
3. ↵ 필드에 필요한 추가 정보를 입력합니다.
4. 필요한 각 키에 이 작업을 반복합니다.

이 정보는 Salt 열을 통해 확인할 수 있습니다. 이 정보는 다음 명령어를 사용하여 검색할 수 있습니다.

```
salt $minionid pillar.get custom_info:key1
```

이 명령을 실행한 후의 출력은 다음과 같습니다.

```
$minionid:
  val1
```

사용자 정의 시스템 정보 키를 몇 개 생성했으면 이 키를 클라이언트에 적용할 수 있습니다.

절차: 사용자 정의 정보 키를 클라이언트에 적용

1. Uyuni Web UI에서 ↵으로 이동하여 사용자 정의 정보를 적용할 클라이언트를 클릭하고, **상세 정보 > 사용자 정의 정보** 탭으로 이동합니다.
2. **[값 생성]**을 클릭합니다.
3. 적용하려는 값을 찾아 키 레이블을 클릭합니다.
4. ↵ 필드에 추가 정보를 입력합니다.
5. **[키 업데이트]**를 클릭하여 사용자 정의 정보를 클라이언트에 적용합니다.

구성 관리에 대한 자세한 설명은 **Client-configuration > Configuration-management**에서 참조하십시오.

7.7. 시스템 세트 관리자

시스템 세트 관리자(SSM)는 한 번에 두 개 이상의 클라이언트에서 작업을 수행하는 데 사용됩니다. SSM은 수명이 짧은 클라이언트 세트를 생성하므로 다수의 클라이언트에 적용해야 하는 커기-끄기 작업에 유용합니다. 더 오래 지속되는 세트를 원하는 대신에 시스템 그룹을 사용하는 것을 고려하십시오. 시스템 그룹에 대한 자세한 내용은 **Client-**

configuration > System-groups를 참조하십시오.

SSM에 사용할 수 있는 작업이 아래 표에 나열되어 있습니다. 이 표의 아이콘에는 다음과 같은 의미가 있습니다.

- ✓ 이 작업은 이 클라이언트 유형의 SSM에서 사용할 수 있습니다.
- ✗ 이 작업은 이 클라이언트 유형의 SSM에서 사용할 수 없습니다.
- ? 이 작업은 이 클라이언트 유형에 대해 고려 중이며, 이후에 지원될 수도 있고 지원되지 않을 수도 있습니다.

표 48. 사용할 수 있는 SSM 작업

시스템 목록	
패치 설치	✓
패치 업데이트 예약	✓
패키지 업그레이드	✓
패키지 설치	✓
패키지 제거	✓
패키지 확인	✗
그룹 생성	✓
그룹 관리하기	✓
채널 멤버십	✓
채널 구독	✗
채널 배포/차단	✗
클라이언트 자동 설치	✗
스냅샷용 태그	✗
원격 명령	✗
전원 관리	✗
시스템 환경 설정 업데이트	✓
하드웨어 프로파일 업데이트	✓
패키지 프로파일 업데이트	✓
사용자 정의 값 설정/제거	✓
클라이언트 재부팅	✓
클라이언트를 다른 조직으로 마이그레이션	✓
클라이언트 삭제	✓

SSM에 대한 클라이언트를 다음 몇 가지 방식으로 선택할 수 있습니다.

- 시스템 > 시스템 목록으로 이동하여 작업하려는 클라이언트의 확인란을 선택합니다.
- 시스템 > 시스템 그룹으로 이동하여 작업하려는 시스템 그룹에 대해 [SSM에서 사용]을 클릭합니다.

- 시스템 > 시스템 그룹으로 이동하여 작업하려는 그룹의 확인란을 선택하고, [그룹 작업]을 클릭합니다.

작업하려는 클라이언트를 선택했으면 시스템 > 시스템 세트 관리자로 이동하거나 상단 메뉴 모음에서 ➔➔➔ 아이콘을 클릭합니다.



SSM의 상세 정보는 Uyuni Web UI의 다른 부분에 있는 상세 정보와 약간 다를 수 있습니다. SSM에 사용할 수 있는 모든 업데이트가 표시됩니다. 따라서 최신 버전이 아닐 수 있는 패키지로 업그레이드할 수 있습니다.

7.7.1. SSM에서 기본 채널 변경하기

SSM을 사용해 동시에 클라이언트 두 개 이상의 기본 채널을 변경할 수 있습니다.



기본 채널을 상당 부분 변경하면 영향을 받는 클라이언트에서 사용할 수 있는 패키지 및 패치가 달라집니다. 주의해서 사용하십시오.

절차: SSM을 사용해 여러 클라이언트의 기본 채널 변경

- Uyuni Web UI에서 시스템 > 시스템 목록으로 이동하여 작업하려는 클라이언트의 확인란을 선택하고 시스템 > 시스템 세트 관리자로 이동합니다.
- ➔ 하위 탭으로 이동합니다.
- 목록에서 현재 기본 채널을 찾고, ➔ 열에 표시된 숫자가 정확한지 확인합니다. 이 열의 숫자를 클릭하면 변경하려는 클라이언트에 관한 더 자세한 정보를 볼 수 있습니다.
- ➔ ➔ ➔ 필드에서 새 기본 채널을 선택하고, [다음]을 클릭합니다.
- 각 하위 채널에 대해 ➔ ➔ ➔ 또는 ➔ ➔ ➔ 를 선택하고, [다음]을 클릭합니다.
- 변경하려는 사항을 확인하고, 작업이 수행되는 시점을 선택합니다.
- [확인]을 클릭하여 변경 사항을 예약합니다.

7.8. 시스템 그룹

시스템 그룹을 사용해 다수의 클라이언트를 더 쉽게 관리할 수 있습니다. 그룹을 사용해 업데이트, 구성 채널, salt 상태 또는 수식의 적용과 같은 대량 작업을 클라이언트에서 수행할 수 있습니다.

환경에 적합한 방식으로 클라이언트를 그룹으로 편성할 수 있습니다. 예를 들어 클라이언트를 설치되는 운영 체제, 있어야 할 물리적 위치 또는 처리하는 워크로드 유형에 따라 체계적으로 편성할 수 있습니다. 클라이언트가 속할 수 있는 그룹의 수에는 제한이 없으므로 그룹을 다양한 방식으로 정의할 수 있습니다.

클라이언트를 그룹으로 구성한 경우 하나 이상의 그룹에 있는 모든 클라이언트 또는 그룹 간의 교차점에서 업데이트를 수행할 수 있습니다. 예를 들어, 웹 서버 소프트웨어가 있는 모든 클라이언트를 한 그룹을 정의하고 모든 SLES 클라이언트를 다른 그룹을 정의할 수 있습니다. 그런 다음, 웹 서버 소프트웨어가 있는 모든 클라이언트에 대해 업데이트를 수행하거나 그룹 간의 교집합을 사용하여 웹 서버 소프트웨어가 있는 모든 SLES 클라이언트를 업데이트할 수 있습니다.

7.8.1. 그룹 생성

몇 개의 그룹을 먼저 생성해야 이 그룹을 사용해 클라이언트를 편성할 수 있습니다.

절차: 새 시스템 그룹 생성

1. Uyuni Web UI에서 **시스템** > **시스템 그룹**으로 이동합니다.
2. **[그룹 생성]**을 클릭합니다.
3. 새 그룹에 이름 및 설명을 부여합니다.
4. **[그룹 생성]**을 클릭하여 그룹을 저장합니다.
5. 필요한 각 그룹에 이 작업을 반복합니다.

7.8.2. 클라이언트를 그룹에 추가

개별 클라이언트를 그룹에 추가하거나 여러 클라이언트를 동시에 추가할 수 있습니다.

절차: 단일 클라이언트를 그룹에 추가

1. Uyuni Web UI에서 **시스템** > **시스템 목록**으로 이동하여 추가할 클라이언트의 이름을 클릭합니다.
2. **그룹** > **조인** 탭으로 이동합니다.
3. 조인할 그룹의 확인란을 선택하고 **[선택한 그룹 조인]**을 클릭합니다.

절차: 여러 클라이언트를 그룹에 추가

1. Uyuni Web UI에서 **시스템** > **시스템 그룹**으로 이동하여 클라이언트를 추가할 그룹의 이름을 클릭합니다.
2. **---** 탭으로 이동합니다.
3. 추가할 클라이언트의 확인란을 선택하고 **[시스템 추가]**를 클릭합니다.

절차: SSM으로 여러 클라이언트를 그룹에 추가

1. Uyuni Web UI에서 **시스템** > **시스템 목록**으로 이동하여 추가할 각 클라이언트의 확인란을 선택하면 클라이언트가 시스템 세트 관리자에 추가됩니다.
2. **시스템** > **시스템 세트 관리자**로 이동한 후 **--** 탭으로 이동합니다.
3. 조인할 그룹을 찾아 **--**의 확인란을 선택합니다.
4. **[멤버쉽 변경]**을 클릭합니다.
5. **[확인]**을 클릭하여 선택한 그룹에 클라이언트를 조인합니다.

시스템 세트 관리자에 대한 자세한 내용은 **Client-configuration** > **System-set-manager**에서 확인할 수 있습니다.

시스템 > **시스템 그룹**으로 이동하여 그룹 이름을 클릭한 다음 **--** 탭으로 이동하면 그룹에 속한 클라이언트를 확인할 수 있습니다.

7.8.3. 그룹 작업

클라이언트를 그룹에 배치하고 나면 그룹을 사용해 업데이트를 관리할 수 있습니다.

Uyuni Web UI에서 **시스템** > **시스템 그룹**으로 이동합니다. 그룹 내 클라이언트 중 어느 하나에 대해서라도 사용할 수 있는 업데이트가 있는 경우 목록에 아이콘이 표시됩니다. 사용 가능한 업데이트에 대한 자세한 정보를 확인하고 업데이트를 클라이언트에 적용하려면 아이콘을 클릭합니다.

한 번에 두 개 이상의 그룹으로 작업할 수도 있습니다. 작업할 그룹을 선택하고 [합집합 작업]을 클릭하여 선택한 각 그룹에서 각 클라이언트를 선택합니다.

또는 그룹의 교집합에서 작업할 수 있습니다. 두 개 이상의 그룹을 선택하고, **[교집합 작업]**을 클릭하여 선택한 모든 그룹에 있는 클라이언트만 선택하십시오. 예를 들어, 웹 서버 소프트웨어가 있는 모든 클라이언트의 그룹이 1개와 모든 SLES 클라이언트의 그룹 1개가 있을 수 있습니다. 이 그룹의 교집합은 웹 서버 소프트웨어가 있는 모든 SLES 클라이언트가 됩니다.

7.9. 시스템 유형

클라이언트는 시스템 유형을 기준으로 분류됩니다. 각 클라이언트에는 기본 시스템 유형과 할당된 추가 시스템 유형이 있을 수 있습니다.

모든 클라이언트의 기본 시스템 유형은 Salt입니다.

추가 시스템 유형에는 가상 호스트로 작동하는 클라이언트의 경우 ..., ..., 빌드 호스트로 작동하는 클라이언트의 경우 ...를 포함합니다.

Chapter 8. 운영 체제 설치

일반적으로 이미 실행 중인 클라이언트를 등록합니다. 이러한 시스템을 수동으로 설치한 직후 Uyuni를 등록하거나 이미 설치된 기존 시스템의 환경에 Uyuni를 추가할 수 있습니다.

또는 Uyuni를 사용하여 운영 체제를 설치한 후 한 번에 Uyuni에 등록할 수 있습니다. 이 방법은 부분적으로 또는 전체적으로 자동화되어 있으므로, 설치 프로그램 질문에 답하는 시간을 절약할 수 있으며 설치 및 등록하려는 클라이언트가 많은 경우 특히 유용합니다.

Uyuni에서는 다음과 같이 여러 방법으로 운영 체제를 설치할 수 있습니다.

- 클라이언트가 이미 등록된 현재 위치에서 설치
- PXE 부팅을 사용하여 네트워크에서 설치
- 설치 CD-ROM 또는 USB 키를 준비한 후 시스템으로 이동하여 해당 미디어를 부팅하여 설치
- Uyuni for Retail 솔루션의 일부로 설치.

현재 위치 재설치 방법에서는 이전 운영 체제가 클라이언트에 이미 설치되어 있고 클라이언트가 이미 Uyuni에 등록된 경우를 가정합니다.

현재 위치 설치 방법은 **Client-configuration > Autoinst-reinstall**을 참조하십시오.

네트워크 부팅 설치 방법은 포맷되지 않은 시스템에서 작동합니다. 그러나 다음과 같은 특정 네트워크 구성에서만 수행될 수 있습니다.

- Uyuni 서버 또는 프록시 중 하나가 설치하는 시스템과 동일한 로컬 네트워크에 있거나 그 사이의 모든 라우터를 통과하는 DHCP 릴레이가 있는 경우
- 새 DHCP 서버를 설정할 수 있거나 기존 DHCP 서버를 구성할 수 있는 경우
- 설치할 클라이언트가 PXE를 사용하여 부팅할 수 있거나 그렇게 구성할 수 있는 경우.

이동식 미디어 방법을 사용하면 이러한 네트워크 제약을 우회할 수 있습니다. 그러나 이 경우에는 시스템이 CD-ROM 또는 USB 키를 읽어 부팅할 수 있는 것으로 가정합니다. 또한 클라이언트 시스템에 물리적으로도 액세스할 수 있어야 합니다.

이동식 미디어 방법은 **Client-configuration > Autoinst-cdrom**을 참조하십시오.

Uyuni for Retail 방법은 **Retail > Retail-overview**을 참조하십시오.



Ubuntu 및 Debian 클라이언트의 자동 설치는 지원되지 않습니다. 이러한 운영 체제는 수동으로 설치해야 합니다.

Uyuni의 자동 설치 기능은 Cobbler 소프트웨어를 기반으로 합니다. Cobbler에 대한 자세한 설명은 <https://cobbler.readthedocs.io>에서 확인할 수 있습니다.



SUSE는 Uyuni Web UI에서 또는 Uyuni API를 통해 사용할 수 있는 Cobbler 함수만 지원합니다. Cobbler에서 지원하는 유일한 명령줄 명령은 `buildiso`입니다. 여기에서는 지원되는 기능만 설명합니다.

8.1. 등록된 시스템 재설치

현재 위치 다시 설치는 로컬 클라이언트 시스템에서 시작됩니다. 따라서 PXE를 사용하여 네트워크에서 클라이언트를 부팅할 필요가 없습니다.

등록된 클라이언트를 현재 위치에 다시 설치하려면 자동 설치 가능한 배포판 및 자동 설치 프로파일을 정의해야 합니다. 관련 정보는 **Client-configuration > Autoinst-distributions** 및 **Client-configuration > Autoinst-profiles**을 참조하십시오.

자동 설치 프로파일 및 배포판을 정의한 후 다시 설치를 실행할 수 있습니다.

절차: 이미 등록된 클라이언트 다시 설치

- Uyuni Web UI에서 **시스템 > 시스템 목록**으로 이동하여 다시 설치할 클라이언트를 선택한 후 **프로비저닝 > 자동 설치 > 일정** 하위 탭으로 이동합니다.
- 준비한 자동 설치 프로파일을 선택하고 필요한 경우 프록시를 선택한 후 **[자동 설치 스케줄링 후 완료]** 클릭합니다.
- 프로비저닝 > 자동 설치 > 세션 상태**로 이동하거나 클라이언트에서 직접 설치 진행 상황을 모니터링할 수 있습니다. 클라이언트가 다시 부팅되고 부팅 메뉴에서 이름이 `reinstall-system`인 새 항목을 선택합니다.

그러면 HTTP 프로토콜을 통해 설치가 진행됩니다.

8.2. CD-ROM 또는 USB 키를 통한 설치

Uyuni에 등록되지 않고 PXE를 통한 네트워크 부팅을 사용할 수 없는 클라이언트의 경우 부팅 가능한 CD-ROM 또는 USB 키를 사용하여 시스템을 설치할 수 있습니다.

해당 이동식 매체를 준비하기 위한 한 가지 옵션은 Cobbler를 사용하는 것입니다. Cobbler를 사용한 ISO 이미지 준비에 대한 설명은 [Cobbler를 사용한 ISO 이미지 빌드](#)를 참조하십시오.

SUSE 시스템의 경우 KIWI를 사용하여 ISO 이미지를 준비하는 것이 권장되는 경우가 많습니다. 자세한 내용은 [KIWI를 사용한 SUSE ISO 이미지 빌드](#)를 참조하십시오.

모든 경우에 결과 이미지를 CD-ROM으로 굽거나 USB 키를 준비해야 합니다.

8.2.1. Cobbler를 사용한 ISO 이미지 빌드

Cobbler는 PXE 설치와 유사한 방식으로 작동하는 배포 세트, 커널 및 메뉴가 포함된 ISO 부팅 이미지를 생성할 수 있습니다.



IBM Z에서는 Cobbler를 사용한 ISO 빌드를 지원하지 않습니다.

Cobbler를 사용하여 ISO 이미지를 준비하려면 PXE를 통한 네트워크 부팅을 사용하는 것과 유사하게 배포 및 프로파일을 준비해야 합니다. 배포판 생성에 대한 설명은 **Client-configuration > Autoinst-distributions**을 참조하십시오. 프로파일 생성에 대한 설명은 **Client-configuration > Autoinst-profiles**을 참조하십시오.

Cobbler `buildiso` 명령은 부팅 ISO의 이름과 출력 위치를 정의하는 파라미터를 사용합니다. `buildiso` 명령을 실행하는 경우에는 반드시 `--distro`로 배포를 지정해야 합니다.

```
cobbler buildiso --iso=/path/to/boot.iso --distro=<your-distro-label>
```



단순히 UI에 표시된 대로가 아니라 Cobbler에서 나열한 배포판 및 프로파일 레이블을 사용해야 합니다.

Cobbler가 저장한 배포 및 프로파일의 이름을 나열하려면 다음 명령을 실행합니다.

```
# cobbler distro 목록
# cobbler 프로파일 목록
```

부팅 ISO에는 모든 프로파일 및 시스템이 기본적으로 포함되어 있습니다. `--profiles` 및 `--systems` 옵션을 사용하면 사용되는 프로파일 및 시스템을 제한할 수 있습니다. 예:

```
cobbler buildiso --systems="system1 system2 system3" \
--profiles=<your-profile1-label> <your-profile2-label> <your-profile3-label> --distro=<your-distro-label>
```



ISO 이미지를 공용 tmp 디렉토리에 쓸 수 없는 경우 /usr/lib/systemd/system/cobblerd.service에서 systemd 설정을 확인하십시오.

8.2.2. KIWI를 사용한 SUSE ISO 이미지 빌드

KIWI는 이미지 생성 시스템입니다. KIWI를 사용하여 SUSE 시스템을 설치하기 위한 대상 시스템에서 사용되는 부팅 가능한 ISO 이미지를 생성할 수 있습니다. 시스템은 재부팅되거나 커질 때 이 이미지로 부팅되고, Uyuni에서 AutoYaST 구성으로 로드하며, AutoYaST 프로파일에 따라 SUSE Linux Enterprise Server를 설치합니다.

ISO 이미지를 사용하려면 시스템을 부팅하고 프롬프트에 `autoyast`를 입력하십시오(AutoYaST 부팅을 위한 레이블을 `autoyast` 상태로 두었다고 가정함). `Enter` 키를 눌러 AutoYaST 설치를 시작합니다.

KIWI에 대한 자세한 정보는 <http://doc.opensuse.org/projects/kiwi/doc/>에서 참조하십시오.

8.2.3. Cobbler를 사용한 Red Hat ISO 이미지 빌드

자세한 내용은 [client-configuration:autoinst-cdrom.pdf](#)에서 확인할 수 있습니다.

8.3. 자동 설치 가능한 배포판

자동 설치 프로세스에서는 여러 파일을 사용하여 설치를 시작합니다. 이러한 파일에는 Linux 커널, 초기 RAM 디스크 및 설치 모드에서 운영 체제를 부팅하기 위해 필요한 기타 파일이 포함됩니다.

Uyuni은(는) `{systemitem}mgradm` 도구를 사용하여 설치 파일을 소스에서 서버 컨테이너로 복사합니다.

필요한 파일을 DVD 이미지에서 추출할 수 있습니다. 관련 정보는 [ISO 이미지 기반 배포](#)를 참조하십시오.

또는 `tftpboot-installation` 패키지를 설치할 수 있습니다. 관련 정보는 [RPM 패키지 기반 배포](#)를 참조하십시오.

또한 해당 파일과 운영 체제 버전이 동일한 Uyuni 서버에서 동기화된 기본 채널이 있어야 합니다.

파일이 준비되고 기본 채널이 동기화되면 배포를 선언해야 합니다. 이 작업은 기본 채널에 설치 파일을 연결합니다. 배포는 1개 이상의 설치 프로파일에 의해 참조될 수 있습니다. 관련 정보는 [자동 설치 가능 배포판 선언](#)을 참조하십시오.

8.3.1. ISO 이미지 기반 배포

이 방법에서는 클라이언트에 설치하려는 운영 체제의 설치 미디어가 있는 경우를 가정합니다. 일반적으로 Linux 커널, `initrd` 파일 및 설치 모드에서 운영 체제를 부팅하는 데 필요한 기타 파일을 포함하는 DVD `.iso` 이미지입니다.

절차: 설치 미디어에서 파일 임포트

1. `mgradm`을 사용하여 ISO 이미지에서 설치 데이터를 가져옵니다

```
# mgradm distribution copy <image_name>.iso <image_name>
```

2. `mgradm`이 보고한 배포 경로를 기록해둡니다. 이 정보는 배포를 Uyuni(으)로 선언할 때 필요합니다.

8.3.1.1. 배포 자동 감지 및 등록

`mgradm`은 배포 이름을 자동으로 감지하여 이를 서버에 등록할 수 있습니다. 제공되는 ISO 이미지에는 `.treeinfo` 파일이 포함되어 있어야 합니다.

절차: 자동 감지 및 등록을 사용한 배포 파일 가져오기

1. `mgradm` 사용:

```
# mgradm distribution copy --api-user <username> --api-password
<password> <image_name>.iso
```

8.3.2. RPM 패키지 기반 배포

이 방법은 SUSE 시스템에 적용됩니다. 설치 시스템에서 미리 패키지화된 파일을 사용하므로 설치 미디어에서 컨텐트를 임포트하는 것보다 단순합니다.

절차: 설치 패키지에서 파일 추출

1. Uyuni 서버에서 이름이 `tftpboot-installation`으로 시작하는 패키지를 설치합니다. 정확한 이름은 `zypper se tftpboot-installation` 명령으로 확인할 수 있습니다.
2. 다음 명령을 사용하여 패키지를 다른 루트에 설치하여 다시 시작할 필요가 없도록 할 수 있습니다.

```
# mkdir /opt/tftpinstall
# zypper --installroot /opt/tftpinstall install tftpboot-
installation-SLE-Micro-5.5-x86_64
```

3. 명령어 `ls -d /opt/tftpinstall/usr/share/tftpboot-installation/*`로 설치 파일을 찾습니다.
4. `mgradm`을 사용하여 설치 파일을 복사합니다.

```
# mgradm distribution copy /opt/tftpinstall/usr/share/tftpboot-
installation/SLE-Micro-5.5-x86_64 SLE-Micro-5.5-x86_64
```

5. `mgradm` 도구가 보고한 배포 경로를 기록해둡니다. 이 정보는 Uyuni에 배포를 선언할 때 필요합니다.
6. `mgradm` 도구가 완료되면 `/opt/tftpinstall` 디렉토리를 제거할 수 있습니다.

이 절차를 통해 Uyuni 서버와 버전이 동일한 운영 체제를 설치하도록 준비할 수 있습니다. 클라이언트에 다른 운영 체제 또는 버전을 설치하려면 해당 배포에서 `tftpboot-installation-*` 패키지를 수동으로 가져와야 합니다. Uyuni의
→→→→→ 입력 상자에서 이름이 `tftpboot-installation`으로 시작하는 패키지를 검색한 후 패키지의 세부 정보를 확인합니다. 해당 정보는 `/var/spacewalk/` 아래의 로컬 경로에 표시됩니다.

8.3.3. 자동 설치 가능한 배포판 선언

자동 설치 파일을 추출한 후의 단계는 자동 설치 가능한 배포판을 선언하는 것입니다.

절차: 자동 설치 가능한 배포판 선언

1. Uyuni Web UI에서 **시스템 > 자동 설치 > 배포**로 이동합니다.
2. →→→→→을 클릭하고 다음 필드를 입력합니다.
 - →→→→→ 필드에서 자동 설치 가능한 배포를 식별할 수 있는 이름을 입력합니다.
 - →→→→→ 필드에서 Uyuni 서버에 저장된 설치 미디어의 경로를 입력합니다.
 - 일치하는 →→→→→을 선택합니다. 설치 미디어와 일치해야 합니다.
 - →→→→→→→→→을 선택합니다. 설치 미디어와 일치해야 합니다.
 - 선택 사항: 이 배포를 부팅할 때 사용할 커널 옵션을 지정하십시오. 여러 가지 방식으로 커널 옵션을 제공할 수 있습니다. 배포에 일반적으로 사용되는 옵션만 여기에 추가하십시오.
3. **[자동 설치 가능한 배포 생성]**을 클릭합니다.

준비한 설치 파일에는 설치에 필요한 패키지가 포함되지 않을 수 있습니다. 포함되지 않은 경우, `useonlinerepo=1`을 →→→→→ 필드에 추가합니다.

패키지 리포지토리에는 서명되지 않을 수 있는 메타데이터가 포함되어 있습니다. 메타데이터가 서명되지 않은 경우 `insecure=1`을 →→→→→ 필드에 추가하거나 **Client-configuration > Autoinstall-ownpgpkey**에서의 설명과 같이 자체 GPG 키를 사용합니다.

이러한 커널 옵션은 예를 들어 전체 DVD가 아닌 "online installer" ISO 이미지를 사용하거나 `tpboot-`

installation 패키지를 사용할 때 필요합니다.

시스템, 자동 설치, 배포판으로 이동하여 자동 설치 가능한 배포판을 관리합니다.

8.4. 자동 설치 프로파일

운영 체제가 설치되는 방법은 자동 설치 프로파일에 따라 다릅니다. 예를 들어, 설치 프로그램에 전달한 추가 커널 파라미터를 지정할 수 있습니다.

프로파일에서 가장 중요한 부분은 "자동 설치 파일"입니다. 수동으로 설치하는 경우 파티셔닝 및 네트워크 정보, 사용자 상세 정보와 같은 정보를 설치 프로그램에 제공해야 합니다. 자동 설치 파일은 이러한 정보를 스크립트된 형식으로 제공하는 방법입니다. 이러한 유형의 파일을 종종 "답변 파일"이라고도 합니다.

Uyuni에서 설치할 클라이언트의 운영 체제에 따라 두 가지 유형의 프로파일을 사용할 수 있습니다.

- SUSE Linux Enterprise 또는 openSUSE 클라이언트에 대해서는 AutoYaST를 사용하십시오.
- Red Hat Enterprise Linux 클라이언트에 대해서는 Kickstart를 사용하십시오.

운영 체제가 다양한 클라이언트를 설치하려는 경우 AutoYaST 및 Kickstart 프로파일을 모두 사용할 수 있습니다.

- 프로파일을 선언하는 방법은 [프로파일 선언](#)에서 확인할 수 있습니다.
- AutoYaST 프로파일에 대한 설명은 [AutoYaST 프로파일](#)을 참조하십시오.
- Kickstart 프로파일에 대한 설명은 [Kickstart 프로파일](#)을 참조하십시오.

프로파일에 포함된 자동 설치 파일에는 변수 및 코드 조각이 포함될 수 있습니다. 변수 및 코드 조각에 대한 설명은 [템플릿 구문](#)을 참조하십시오.

8.4.1. 프로파일 선언

자동 설치 파일 및 배포를 준비했으면 프로파일을 생성하여 Uyuni 서버에서 자동 설치를 관리할 수 있습니다. 프로파일은 선택한 이 배포판을 설치하는 방법을 결정합니다. 프로파일을 만드는 한 가지 방법은 AutoYaST 또는 Kickstart 파일을 업로드하는 것입니다. 또는 Kickstart의 경우에만 Web UI 마법사를 사용할 수 있습니다.

절차: 업로드하여 자동 설치 프로파일 생성

1. Uyuni Web UI에서 시스템, 자동 설치, 프로파일로 이동합니다.
2. **[Kickstart/AutoYaST 파일 업로드]**를 클릭합니다.
3. **→** 필드에 프로파일의 이름을 입력합니다. 공백을 사용하면 안 됩니다.
4. **→** **→** **→** 필드에서 이 프로파일에 사용할 자동 설치 가능한 배포판을 선택합니다.
5. **→** **→** 필드에서 이 프로파일에 사용할 가상화 유형을 선택하거나, 이 프로파일을 사용하여 새로운 가상 시스템을 생성하지 않으려면 **→**을 선택합니다.
6. 자동 설치 파일의 내용을 **→** **→** 필드에 복사하거나 **→** **→** **→** **→** 필드를 사용하여 직접 파일을 업로드합니다.

여기에 포함되는 세부 정보에 대한 자세한 설명은 [AutoYaST 프로파일](#) 또는 [Kickstart 프로파일](#)을 참조하십시오.

7. [생성]을 클릭하여 프로파일을 생성합니다.

절차: 마법사로 Kickstart 프로파일 생성

1. Uyuni Web UI에서 **시스템** > **자동 설치** > **프로파일**로 이동합니다.
2. **[Kickstart 프로파일 생성]**을 클릭합니다.
3. **필드**에 프로파일의 이름을 입력합니다. 공백을 사용하면 안 됩니다.
4. **필드**에서 이 프로파일에 사용할 기본 채널을 선택합니다. 이 필드는 사용 가능한 배포에서 채워집니다. 필요한 기본 채널을 사용할 수 없는 경우 배포를 올바르게 생성하였는지 확인하십시오.
5. **필드**에서 이 프로파일에 사용할 가상화의 유형을 선택합니다. 가상화가 없는 경우 **--**을 선택합니다.
6. **[다음]**을 클릭합니다.
7. **--**에서 Uyuni 서버에 설치된 설치 미디어의 경로를 입력합니다.
8. **[다음]**을 클릭합니다.
9. 클라이언트에서 루트 사용자에게 비밀번호를 제공합니다.
10. **[완료]**를 클릭합니다.
11. 새 프로파일의 상세 정보를 검토하고 필요에 따라 사용자 정의합니다.

자동 설치 프로파일을 생성할 때 **--**의 확인란을 선택할 수 있습니다. 이 설정은 Uyuni가 특정 기본 채널과 연결된 최신 배포를 자동으로 선택하도록 허용합니다. 나중에 새 배포를 추가하는 경우 Uyuni는 최근에 생성되거나 수정된 것을 사용합니다.

--을 변경하려면 일반적으로 프로파일 부트로더 및 파티션 옵션을 변경해야 합니다. 이렇게 하면 사용자 정의가 덮어쓰기될 수 있습니다. 신규 또는 변경된 설정을 저장하기 전에 **--** 탭으로 이동하여 확인하십시오.

배포판 및 프로파일의 커널 옵션은 결합되어 있습니다.

시스템 > **자동 설치** > **프로파일**로 이동해 편집하려는 프로파일의 이름을 클릭하여 자동 설치 프로파일의 상세 정보 및 설정을 변경할 수 있습니다. 또는 **시스템** > **시스템 목록**으로 이동하여 프로비저닝 할 클라이언트를 선택한 후 **프로비저닝** > **자동 설치** 하위 탭으로 이동합니다.

8.4.2. AutoYaST 프로파일

AutoYaST 프로파일은 프로파일을 식별하는 **--**, 자동 설치 가능한 배포판을 가리키는 **--**, 여러 옵션과 가장 중요한 AutoYaST 설치 파일로 구성됩니다.

AutoYaST 설치 파일은 AutoYaST 설치 프로그램에 방향을 제공하는 XML 파일입니다. AutoYaST에서는 이 파일을 "제어 파일"이라고 부릅니다. AutoYaST 설치 파일의 전체 구문은 <https://doc.opensuse.org/projects/autoyast/#cha-configuration-installation-options>를 참조하십시오.

SUSE는 사용자 정의 파일의 시작점으로 사용할 수 있는 AutoYaST 설치 파일의 템플릿을 제공합니다. 이 템플릿은 AutoYaST 디렉토리의 <https://github.com/SUSE/manager-build-profiles>에서 확인할 수 있습니다. 이러한 각 프로파일은 사용하기 전 일부 변수를 설정해야 합니다. 필요한 변수는 스크립트에 포함된 README 파일에서 확인합니다. AutoYaST 스크립트에서 변수 사용에 대한 자세한 설명은 [변수](#)를 참조하십시오.

다음은 Uyuni로 설치하기 위한 AutoYaST 설치 파일에서 가장 중요한 부분입니다.

- <add-on>을 사용하면 설치에 하위 채널을 추가할 수 있습니다. 예제는 <https://doc.opensuse.org/projects/autoyast/#Software-Selections-additional>을 참조하십시오.
- <general>\$SNIPPET('spacewalk/sles_no_signature_checks')</general>은 서명 확인을 비활성화합니다.
- <software>를 사용하면 Unified Installer에 제품을 지정할 수 있습니다.
 - "<software>" 예제가 포함된 <https://doc.opensuse.org/projects/autoyast/#CreateProfile-Software>를 참조하십시오.
- <init-scripts config:type="list">\$SNIPPET('spacewalk/minion_script')</init-scripts> 을 사용하면 클라이언트가 Uyuni를 Salt client로 등록할 수 있습니다.

AutoYaST에 대한 자세한 내용은 <https://doc.opensuse.org/projects/autoyast/>를 참조하십시오.

Salt를 기반으로 하는 AutoYaST의 최신 대안은 Yomi입니다. Yomi에 대한 자세한 내용은 **Specialized-guides** > **Salt**에서 확인할 수 있습니다.

8.4.3. 킥스타트 프로파일

Kickstart 프로파일은 매우 다양한 구성 옵션을 제공합니다. 이러한 프로파일을 생성하려면 해당 프로파일을 업로드하거나 전용 마법사를 사용합니다.

Kickstart 프로파일을 사용하면 파일 보관 목록을 사용할 수 있습니다. Kickstart를 사용하여 다시 설치할 클라이언트에 사용자 정의 구성 파일이 여러 개 있는 경우 해당 파일을 목록으로 저장하고 사용할 Kickstart 프로파일과 이 목록을 연결할 수 있습니다.

절차: 파일 보관 목록 생성

1. Uyuni Web UI에서 **시스템** > **자동 설치** > **파일 보관**으로 이동하여 **[파일 유지 목록 생성]**을 클릭합니다.
2. 적절한 레이블을 입력하고 저장하려는 모든 파일 및 디렉토리의 절대 경로를 나열합니다.
3. **[목록 생성]**을 클릭합니다.
4. Kickstart 프로파일에 파일 보관 목록을 포함시킵니다.
5. **시스템** > **자동 설치** > **프로파일**로 이동한 후 편집할 프로파일을 선택하고 **시스템 세부 정보** > **파일 보관** 하위 탭으로 이동하여 포함할 파일 보관 목록을 선택합니다.



파일 유지 목록은 전체 크기가 1 MB로 제한됩니다. /dev/hda1 및 /dev/sda1과 같은 특수 장치는 보관할 수 없습니다. 파일 및 디렉토리 이름만 사용하십시오. 정규식 와일드카드는 사용할 수 없습니다.

Kickstart에 대한 자세한 설명은 Red Hat 문서를 참조하십시오.

8.4.4. 템플릿 구문

설치 파일의 일부는 설치 중에 대체됩니다. 변수는 단일 값으로 대체되며 코드 조각은 텍스트의 전체 섹션으로 대체됩니다. 이스케이프된 기호 또는 섹션은 대체되지 않습니다.

Cheetah라는 템플릿 엔진을 사용하면 Cobbler가 해당 대체를 수행할 수 있습니다. 이 방식을 통해 각각에 대한 프로파일을 수동으로 생성할 필요 없이 대량의 시스템을 다시 설치할 수 있습니다.

Uyuni Web UI에서 자동 설치 변수 및 코드 조각을 생성할 수 있습니다. 프로파일의 **변수**, **코드 조각**, **이스케이핑** 탭을 사용하면 대체 결과를 확인할 수 있습니다.

- 변수에 대한 설명은 [변수](#)를 참조하십시오.
- 코드 조각에 대한 설명은 [코드 조각](#)을 참조하십시오.
- 이스케이프 기호 또는 텍스트 블록에 대한 설명은 [이스케이핑](#)을 참조하십시오.

8.4.4.1. 변수

자동 설치 변수를 사용해 값을 Kickstart 및 AutoYaST 프로파일로 대신할 수 있습니다. 변수를 정의하려면 프로파일에서 **변수** 탭으로 이동하여 텍스트 상자에서 name=value 쌍을 생성합니다.

예를 들어, 클라이언트의 IP 주소를 가진 변수와 게이트웨이의 주소를 가진 변수를 생성할 수 있습니다. 그러면 이러한 변수를 동일한 프로파일에서 설치된 모든 클라이언트에 대해 정의할 수 있습니다. 이 작업을 수행하려면 **텍스트** 상자에 다음 줄을 추가합니다.

```
ipaddr=192.168.0.28
gateway=192.168.0.1
```

변수를 사용하려면 프로파일에서 \$ 기호를 앞에 추가하여 변수를 대체합니다. 예를 들어, Kickstart 파일의 **network** 부분은 다음과 같습니다.

```
network --bootproto=static --device=eth0 --onboot=on --ip=$ipaddr \
--gateway=$gateway
```

\$ipaddr은 192.168.0.28로, \$gateway는 192.168.0.1로 확인됩니다.

설치 파일에서 변수는 계층 구조를 사용합니다. 시스템 변수는 프로파일 변수에 우선하고, 프로파일 변수는 배포 변수에 우선합니다.

8.4.4.2. 코드 조각

Uyuni는 미리 정의된 다양한 코드 조각과 함께 제공됩니다. 시스템 > 자동 설치 > 자동 설치 조각으로 이동하여 기존 조각의 목록을 확인합니다.

자동 설치 파일에 \$SNIPPET() 매크로를 삽입하여 코드 조각을 사용합니다. 예를 들어, Kickstart에서는 다음과 같습니다.

```
$SNIPPET('spacewalk/rhel_register_script')
```

또는 AutoYaST에서는 다음과 같습니다.

```
<init-scripts config:type="list">
$SNIPPET('spacewalk/sles_register_script')
</init-scripts>
```

매크로는 Cobbler로 구문 분석되고 코드 조각의 컨텐트로 대체됩니다. 또한 나중에 자동 설치 파일에서 사용할 자체 코드 조각을 저장할 수 있습니다. [조각 생성]을 클릭하여 새 코드 조각을 생성하십시오.

다음 예에서는 일반 하드 드라이브 파티션 구성에 대한 Kickstart 코드 조각을 다음과 같이 설정합니다.

```
clearpart --all
part /boot --fstype ext3 --size=150 --asprimary
part / --fstype ext3 --size=40000 --asprimary
part swap --recommended

part pv.00 --size=1 --grow

volgroup vg00 pv.00
logvol /var --name=var vgname=vg00 --fstype ext3 --size=5000
```

예를 들어, 다음과 같이 코드 조각을 사용합니다.

```
$SNIPPET('my_partition')
```

8.4.4.3. 이스케이핑

자동 설치 파일에 `$(example)`와(과) 같은 셸 스크립트 변수가 포함된 경우 백슬래시를 사용하여 컨텐트를 이스케이프해야 합니다. `\$(example)`. `$` 기호를 이스케이프하면 템플릿 엔진이 기호를 내부 변수로 평가하지 못하게 됩니다.

코드 조각이나 스크립트와 같은 텍스트 블록은 `\#raw` 및 `\#end raw` 지시어로 래핑하여 이스케이프할 수 있습니다. 예:

```
#raw
#!/bin/bash
for i in {0..2}; do
echo "$i - Hello World!"
done
#end raw
```

기호 다음에 공백이 있는 모든 줄은 주석으로 취급되므로 평가되지 않습니다. 예:

```
# 특정 세션 시작(주석임)
echo "Hello, world"
# 특정 세션 종료(주석임)
```

8.5. 무인 프로비저닝

API 호출을 사용하여 MAC 주소에 의해 식별되는 클라이언트와 자동 설치 프로파일 사이의 관계를 선언할 수 있습니다. 다음에 시스템이 재부팅되면 지정된 프로파일에 따라 설치가 시작됩니다.

절차: 수동으로 선언된 프로파일에서 다시 설치

1. Uyuni 서버의 명령 프롬프트에서 `system.createSystemRecord` API를 호출합니다. 이 예에서 `name`을 클라이언트의 이름으로 바꾸고, `<profile>`을 프로파일 레이블로 바꾸며, `<iface>`를 `eth0`과 같이 클라이언트의 인터페이스 이름으로 바꾸고, `<hw_addr>`을 `00:25:22:71:e7:c6`과 같은 하드웨어 주소로 바꿉니다.

```
$ spacecmd api -- --args '[ "<name>", "<profile>", "", "", \
[ {"name": "<iface>", "mac": "<hw_addr>" } ]' \
system.createSystemRecord
```

2. 클라이언트의 전원을 켭니다. 네트워크에서 부팅되며 설치에 맞는 올바른 프로파일이 선택됩니다.

이 명령을 통해 Cobbler에서 시스템 레코드가 생성됩니다. 커널 옵션, 클라이언트의 IP 주소, 도메인 이름과 같은 추가 파라미터도 지정할 수 있습니다. 자세한 설명은 API 설명서에서 `createSystemRecord` ↗을 참조하십시오.

8.6. 자체 GPG 키 사용

자동 설치에서 사용할 리포지토리에 서명되지 않은 메타데이터가 있는 경우 일반적으로 `insecure=1` 커널 파라미터를 자동 설치 가능한 배포판 옵션으로 사용하고 AutoYaST 설치 파일에서 `spacewalk/sles_no_signature_checks` 코드 조각을 사용해야 합니다.

더 안전한 방법은 자체 GPG 키를 제공하는 것입니다.



이 방법은 SUSE 클라이언트에만 해당합니다.

절차: 자체 GPG 키 포함

1. GPG 키를 생성합니다.
2. 이 키를 사용하여 패키지의 메타데이터에 서명합니다.
3. 설치 미디어의 초기 RAM 디스크에 추가합니다.
4. To copy GPG keys to the container use `mgradm gpg add`. This command copies the GPG key and adds it to the customer keyring.
5. Run command `mgradm restart`. This command creates a unique keyring using the SUSE and the customer keyring.



새 GPG 키를 사용하여 메타데이터에 서명한 후에는 이미 온보딩된 클라이언트는 새 키에 대해 알 수 없습니다. 이상적으로는 메타데이터에 서명한 후 클라이언트를 등록해야 합니다.

그러한 리포지토리를 사용하는 클라이언트를 이미 온보딩한 경우 해결 방법은 해당 클라이언트에서 GPG 키 확인을 비활성화하는 것입니다.

Chapter 9. 시각화

Uyuni를(를) 사용하여 가상화된 클라이언트를 관리할 수 있습니다. 이 설치 유형에서는 Uyuni 서버에 가상 호스트가 설치되어 원하는 수의 가상 게스트를 관리합니다. 원하는 경우 여러 가상 호스트를 설치하여 게스트 그룹을 관리할 수 있습니다.

가상화 클라이언트가 보유한 기능의 범위는 사용자가 선택하는 타사 가상화 공급자에 따라 달라집니다.

Xen 및 KVM 호스트 및 게스트는 Uyuni에서 직접 관리할 수 있습니다. 이를 통해 AutoYaST 또는 Kickstart를 사용해 호스트 및 게스트를 자동 설치하고 Web UI에서 게스트를 관리할 수 있습니다.

VMWare vSphere를 포함한 VMWare와 Nutanix AHV의 경우 Uyuni는 가상 호스트 관리자(VHM)를 설정해 VM을 제어할 것을 사용자에게 요구합니다. 이를 통해 사용자는 호스트 및 게스트에 대한 제어권을 얻게 됩니다. 하지만 Xen 및 KVM에서보다는 더 제한적입니다. Uyuni은(는) VMWare vSphere 또는 Nutanix AHV에서 VM을 생성하거나 편집할 수 없습니다.

다른 타사 가상화 공급자는 Uyuni가 직접 지원하지 않습니다. 하지만 공급자가 VM에 대한 JSON 구성 파일을 엑스포트하도록 허용하는 경우 이 구성 파일을 Uyuni로 업로드하여 VHM으로 관리할 수 있습니다.

VHM을 사용해 가상화를 관리하는 방법에 대한 자세한 내용은 [Client-configuration > Vhm](#)을 참조하십시오.

9.1. 가상화 호스트 관리

시작하기 전에 가상화 호스트로 사용할 클라이언트에 **시스템 유형**이 할당되어 있는지 확인합니다. **시스템** > **시스템 목록**으로 이동하여 가상화 호스트로 사용할 클라이언트의 이름을 클릭합니다. **시스템 유형**이 목록에 없으면 **선택** 버튼을 클릭하여 초기화합니다. 자세한 내용은 [client-configuration:virt-xenkvm.pdf](#)에서 확인할 수 있습니다.

클라이언트에 **시스템 유형**이 있는 경우 클라이언트의 시스템 정보 페이지에서 **호스트** 탭을 제공합니다. **호스트** 탭을 통해 가상 게스트를 생성 및 관리하고 저장소 풀 및 가상 네트워크를 관리할 수 있습니다.

9.2. 가상 게스트 생성

Uyuni Web UI에서 가상화 호스트에 가상 게스트를 추가할 수 있습니다.

절차: 가상 게스트 생성

1. Uyuni Web UI에서 **시스템** > **시스템 목록**으로 이동하여 가상화 호스트의 이름을 클릭하고, **호스트** 탭으로 이동합니다.
2. **선택** 섹션에서 다음과 같은 상세 정보를 입력합니다.
 - **하위 탭**에서 **[게스트 생성]**을 클릭합니다.
 - **필드**에서 게스트의 이름을 입력합니다.
 - **하이퍼바이저** 필드에서 사용할 하이퍼바이저를 선택합니다.
 - **완전 가상화** 또는 **반가상화**를 선택합니다.
 - **디스크 크기** 필드에 게스트 디스크 크기의 상한을 MiB 단위로 입력합니다.
 - **vCPU 수**에서 게스트를 위한 vCPU의 수를 입력합니다.

- 필드에서 게스트에서 사용할 에뮬레이트된 CPU 아키텍처를 선택합니다. 선택한 아키텍처는 기본적으로 가상 호스트와 일치합니다.
 - 필드에서 게스트를 설치하는 데 사용할 자동 설치 도구를 선택합니다. 자동 설치를 사용하지 않으려면 이 필드를 공백으로 두십시오.
3. 섹션에서 클라이언트와 함께 사용할 가상 디스크의 상세 정보를 입력합니다. URL 필드에 운영 체제 이미지의 경로를 반드시 입력해야 합니다. 입력하지 않으면 게스트가 빈 디스크로 생성됩니다.
 4. 섹션에서 클라이언트와 함께 사용할 가상 네트워크 인터페이스의 상세 정보를 입력합니다. MAC 주소를 생성하려면 MAC 필드를 공백으로 두십시오.
 5. 섹션에서 클라이언트와 함께 사용할 그래픽 드라이버의 상세 정보를 입력합니다.
 6. 생성할 게스트의 일정을 잡고 [생성]을 클릭하여 게스트를 생성합니다.
 7. 새로운 가상 게스트는 생성되자마자 시작합니다.

Uyuni Web UI 내에서 pacemaker 클러스터에도 가상 게스트를 추가할 수 있습니다.

절차: 클러스터 관리형 가상 게스트 생성

1. 다음을 추가하여 클러스터 노드 중 하나에서 절차를 따르십시오.
 - 필드가 선택되어 있는지 확인합니다.
 - VM 필드에 게스트 구성이 저장될 모든 클러스터 노드가 공유하는 폴더의 경로를 입력합니다.
 - 모든 클러스터 노드에서 공유하는 저장소 폴더에 모든 디스크가 있는지 확인합니다.

클러스터에서 관리하는 가상 게스트는 라이브 마이그레이션이 가능합니다.

9.3. Xen 및 KVM을 이용한 가상화

Xen 및 KVM 가상화 클라이언트는 Uyuni에서 직접 관리할 수 있습니다.

시작하려면 Uyuni 서버에서 가상 호스트를 설정해야 합니다. 그런 다음 추가 가상 호스트 및 가상 게스트에 대해 AutoYaST 또는 Kickstart를 사용하여 자동 설치를 설정할 수 있습니다.

이 섹션에는 가상 게스트를 설치한 후 관리하는 방법에 대한 정보도 포함되어 있습니다.

9.3.1. 호스트 설정

VM 호스트에서 Xen 또는 KVM을 설정하는 방법은 연결된 게스트에서 사용할 운영 체제에 따라 다릅니다.

- SUSE 운영 체제와 관련해서는 <https://documentation.suse.com/sles/15-SP4/html/SLES-all/book-virtualization.html>에서 제공되는 SLES 가상화 가이드를 참조하십시오.
- Red Hat Enterprise Linux 운영 체제의 경우 해당 버전의 Red Hat 문서를 참조하십시오.

수식을 사용하면 호스트를 초기화하는 데 유용합니다. 자세한 내용은 [client-configuration:virt-xenkm.pdf](#)에서 확인할 수 있습니다.

9.3.1.1. 배경 정보

{systemitem}libvirt를 사용하여 게스트를 설치 및 관리합니다. 호스트에 libvirt-daemon 패키지가 설치되어 있어야 합니다. 대부분의 경우 기본 설정으로 충분하며 조정할 필요가 없습니다. 그러나 게스트가 루트가 아닌 사용자로 VNC 콘솔에 액세스하려면 몇 가지 구성을 변경해야 합니다. VNC 콘솔을 설정하는 방법에 대한 자세한 내용은 사용 중인 운영 체제의 설명서를 참조하십시오.

Uyuni 서버에 부트스트랩 스크립트가 필요합니다. 부트스트랩 스크립트에는 호스트의 활성화 키가 포함되어야 합니다. 또한 추가 보안을 위해 GPG 키를 포함하는 것이 좋습니다. 부트스트랩 스크립트 생성에 대한 자세한 내용은 **Client-configuration > Registration-bootstrap**에서 확인할 수 있습니다.

부트스트랩 스크립트가 준비되면 이를 사용하여 Uyuni 서버에 호스트를 등록합니다. 클라이언트 등록에 대한 자세한 내용은 **Client-configuration > Registration-overview**에서 확인할 수 있습니다.

9.3.1.2. 초기화

수식을 사용하여 다음과 같이 호스트를 초기화합니다.

절차: 초기화

1. Uyuni Web UI에서 호스트의 페이지로 이동하고 탭을 클릭합니다.
2. 수식을 선택하고 [저장]을 클릭합니다.
3. 하위 탭을 클릭합니다.
4. 설정을 확인하고 [수식 저장]을 클릭합니다.
5. 변경 사항을 적용하려면 Highstate를 적용합니다.
6. salt-minion 서비스를 다시 시작해 새 구성을 활성화합니다.

```
systemctl restart salt-minion
```

9.3.2. VM 게스트 자동 설치

AutoYaST 또는 Kickstart를 사용해 Xen 및 KVM 게스트를 자동으로 설치하고 등록할 수 있습니다.

게스트를 등록할 VM 호스트와 각 게스트에 대한 활성화 키가 필요합니다. 활성화 키에는 및 권한이 있어야 합니다. 활성화 키 생성에 대한 자세한 내용은 **Client-configuration > Activation-keys**에서 확인할 수 있습니다.

설치 후 Uyuni(으)로 게스트를 자동으로 등록하려면 부트스트랩 스크립트를 생성해야 합니다. 부트스트랩 스크립트 생성에 대한 자세한 내용은 **Client-configuration > Registration-bootstrap**에서 확인할 수 있습니다.

9.3.2.1. 자동 설치 가능한 배포판 생성

Uyuni에서 클라이언트를 자동 설치하려면 VM 호스트에 자동 설치 가능한 배포를 생성해야 합니다. 마운트한 로컬 또는 원격 디렉토리나 루프 마운트 ISO 이미지에서 배포를 사용할 수 있는 상태로 만들 수 있습니다.

자동 설치 가능 배포의 구성은 게스트에서 Red Hat Enterprise Linux 또는 SUSE 운영 체제를 사용하는지 여부에 따라

다릅니다. Red Hat Enterprise Linux 설치용 패키지는 연결된 기본 채널에서 가져옵니다. SUSE 시스템 설치용 패키지는 자동 설치 가능 배포에서 가져옵니다. 따라서 SUSE 시스템의 경우 자동 설치 가능 배포는 완전한 설치 소스여야 합니다.

표 49. 자동 설치 가능한 배포의 경로

운영 체제 유형	커널 위치	initrd 위치
Red Hat Enterprise Linux	images/pxeboot/vmlinuz	images/pxeboot/initrd.img
SUSE	boot/<arch>/loader/initrd	boot/<arch>/loader/linux

모든 경우 기본 채널이 자동 설치 배포와 일치하는지 확인하십시오.

시작하기 전에 VM 호스트에서 사용할 수 있는 설치 미디어가 있는지 확인합니다. 원격 리소스, 로컬 디렉토리 또는 루프 마운트된 ISO 이미지일 수 있습니다. 또한 모든 파일과 디렉토리를 전 세계에서 읽을 수 있는지도 확인해야 합니다.

절차: 자동 설치 가능한 배포 생성

1. Uyuni Web UI에서 **시스템 > 자동 설치 > 배포판**으로 이동한 후 **[배포 생성]**을 클릭합니다.
2. **설치 채널** 섹션에서 다음 파라미터를 사용합니다.
 - 설치 채널에 배포판의 고유 이름을 입력합니다. 글자, 숫자, 하이픈(-), 마침표(.) 및 줄(_)만 사용하고, 이름을 구성하는 문자는 다섯 개 이상이어야 합니다.
 - 필드에 설치 원본의 절대 경로를 입력합니다.
 - 설치 채널에서 설치 소스와 일치하는 채널을 선택합니다. 이 채널은 비SUSE 설치에 대해 패키지 소스로 사용됩니다.
 - 설치 채널에서 설치 소스와 일치하는 운영 체제 버전을 선택합니다.
 - 필드에는, 설치를 위해 부팅할 때 커널로 전달될 옵션을 입력합니다. `install=` 파라미터와 `self_update=0` 파라미터는 기본적으로 추가됩니다.
 - 설치된 시스템을 처음 부팅할 때 커널로 전달될 옵션을 입력합니다.
3. **[자동 설치 가능한 배포판 생성]**을 클릭하여 저장합니다.

자동 설치 가능한 배포판을 생성했으면 **시스템 > 자동 설치 > 배포판**으로 이동하여 편집할 배포판을 선택하고 편집할 수 있습니다.

9.3.2.2. 자동 설치 프로파일 생성 및 업로드

자동 설치 프로파일에는 시스템을 설치하는 데 필요한 모든 설치 및 구성 데이터가 포함되어 있습니다. 설치 완료 후 실행될 스크립트도 포함할 수 있습니다.

Kickstart 프로파일은 Uyuni Web UI에서 **시스템 > 자동 설치 > 프로파일**로 이동하여 **[새 Kickstart 파일 생성]**을 클릭하고 표시되는 프롬프트에 따라 생성할 수 있습니다.

AutoYaST 또는 Kickstart 자동 설치 프로파일을 수동으로 생성할 수도 있습니다. SUSE는 사용자 정의 파일의 시작점으로 사용할 수 있는 AutoYaST 설치 파일의 템플릿을 제공합니다. 템플릿은 <https://github.com/SUSE/manager-build-profiles>에서 확인할 수 있습니다.

AutoYaST를 사용하여 SLES를 설치하는 경우 다음 코드 조각도 포함해야 합니다.

```
<products config:type=\list>
<listentry>SLES</listentry>
</products>
```

- AutoYaST에 대한 설명은 [client-configuration:autoinst-profiles.pdf](#)를 참조하십시오.
- Kickstart에 대한 자세한 설명은 [client-configuration:autoinst-profiles.pdf](#)를 참조하거나 해당 설치에 대한 Red Hat 문서를 참조하십시오.

절차: 자동 설치 프로파일 업로드

1. Uyuni Web UI에서 **시스템** > **자동 설치** > **프로파일**로 이동하여 **[Kickstart/AutoYaST 파일 업로드]**를 클릭합니다.
2. **···** **···** **···** **···** **···** **···** 섹션에서 다음 파라미터를 사용합니다.
 - **···** 필드에 프로파일의 고유 이름을 입력합니다. 글자, 숫자, 하이픈(-), 마침표(.) 및 줄(_)만 사용하고, 이름을 구성하는 문자는 일곱 개 이상이어야 합니다.
 - **···** **···** **···** 필드에서 앞서 생성한 자동 설치 가능한 배포를 선택합니다.
 - **···** **···** **···** 필드에서 해당되는 게스트 유형(예: KVM **···** **···**)을 선택합니다. 여기에서 Xen **···** **···**를 선택하지 마십시오.
 - 옵션: 자동 설치 프로파일을 수동으로 생성하려면 **···** **···** 필드에 프로파일을 직접 입력할 수 있습니다. 파일을 이미 생성한 경우 **···** **···** 필드를 공백으로 두십시오.
 - **···** **···** **···** 필드에서 **[파일 선택]**을 클릭하고 시스템 대화 상자를 사용해 업로드할 파일을 선택합니다. 파일 업로드가 완료되면 파일 이름이 **···** **···** 필드에 표시됩니다.
 - 업로드한 파일의 내용이 **···** **···** 필드에 표시됩니다. 편집하려면 직접 할 수 있습니다.
3. **[생성]**을 클릭하여 변경 사항을 저장하고 프로파일을 보관합니다.

자동 설치 프로파일을 생성했으면 **시스템** > **자동 설치** > **프로파일**로 이동해 편집하려는 프로파일을 선택하여 편집할 수 있습니다. 원하는 대로 변경한 후 **[생성]**을 클릭하여 설정을 저장합니다.



기존 Kickstart 프로파일의 **···** **···**을 변경하면 부트로더 및 파티션 옵션이 수정되어 사용자 정의 설정이 덮어쓰기될 수도 있습니다. **···** 탭을 주의 깊게 검토하여 이 설정을 확인한 후에 변경하십시오.

9.3.2.3. 게스트를 자동으로 등록

VM 게스트를 자동으로 설치하는 경우 Uyuni에 등록되지 않습니다. 게스트가 설치되자마자 자동으로 등록되게 하려면 부트스트랩 스크립트를 호출하고 게스트를 등록하는 자동 설치 프로파일에 섹션을 추가할 수 있습니다.

이 섹션에서는 부트스트랩 스크립트를 기존 AutoYaST 프로파일에 추가하는 것에 관한 지침을 제공합니다.

부트스트랩 스크립트 생성에 대한 자세한 내용은 **Client-configuration > Registration-bootstrap**에서 확인할 수 있습니다. Kickstart에 대해 이 작업을 하는 방법에 대한 지침은 해당 설치에 대한 Red Hat 설명서를 참조하십시오.

절차: 부트스트랩 스크립트를 AutoYaST 프로파일에 추가

1. 등록하려는 VM 게스트에 대한 활성화 키를 부트스트랩 스크립트가 포함하는지, 이 키가 `/srv/www/htdocs/pub/bootstrap_vm_guests.sh`의 호스트에 있는지 확인합니다.
2. Uyuni Web UI에서 시스템 > 자동 설치 > 프로파일로 이동하여 이 스크립트를 연결할 AutoYaST 프로파일을 선택합니다.
3. → 필드에서 이 코드 조각을 파일 끝의 종료 태그 `</profile>` 바로 앞에 추가합니다. 아래 코드 조각의 예시 IP 주소 192.168.1.1을 Uyuni 서버의 올바른 IP 주소로 바꿉니다.

```

<scripts>
  <init-scripts config:type="list">
    <script>
      <interpreter>shell</interpreter>
      <location>
        http://192.168.1.1/pub/bootstrap/bootstrap_vm_guests.sh
      </location>
    </script>
  </init-scripts>
</scripts>

```

4. 업데이트를 클릭하여 변경 사항을 저장합니다.



AutoYaST 프로파일에 `<scripts>` 섹션이 이미 포함되어 있는 경우 이 섹션을 추가하지 마십시오. 기존 `<scripts>` 섹션 안에 부트스트랩 코드 조각을 배치합니다.

9.3.2.4. VM 게스트 자동 설치

모든 설정이 완료되면 VM 게스트 자동 설치를 시작할 수 있습니다.



각 VM 호스트는 한 번에 한 게스트만 설치할 수 있습니다. 두 건 이상의 자동 설치 일정을 잡는 경우 이전 설치가 완료되기 전에 다음번 설치가 시작되지 않도록 시간을 안배해야 합니다. 다른 게스트 자동 설치가 아직 실행 중일 때 게스트 설치가 시작되면 실행 중인 설치가 취소됩니다.

1. Uyuni Web UI에서 시스템 > 개요로 이동하여 게스트를 설치하려는 VM 호스트를 선택합니다.
2. → 템과 ←←←←← 헤더 템으로 이동합니다.
3. 사용하려는 자동 설치 프로파일을 선택하고, 게스트에 고유한 이름을 지정합니다.
4. 해당되는 경우 프록시를 선택하고 일정을 입력합니다.
5. 게스트의 하드웨어 프로파일 및 구성 옵션을 변경하려면 [고급 옵션]을 클릭합니다.
6. [자동 설치 일정 잡기 후 완료]를 클릭하여 완료합니다.

9.3.3. VM 게스트 관리

Uyuni Web UI를 사용하여 종료, 재시작, CPU 및 메모리 할당 조정 등 작업과 같은 VM 게스트를 관리할 수 있습니다.

이를 위해서는 Uyuni 서버에 Xen 또는 KVM VM 호스트가 등록되어 있어야 하며 호스트에서 libvirtd 서비스가 실행 중이어야 합니다.

Uyuni Web UI에서 **시스템** > **시스템 목록**으로 이동하여 관리하려는 게스트의 VM 호스트를 클릭합니다. ⏪ 템으로 이동하여 이 호스트에 등록된 모든 게스트를 확인하고 관리 기능에 액세스합니다.

Web UI를 사용한 VM 게스트 관리에 대한 자세한 내용은 **Reference** > **Systems**에서 확인할 수 있습니다.

Chapter 10. 가상 호스트 관리자

가상 호스트 관리자(VHM)는 다양한 클라이언트 유형에서 정보를 수집하는 데 사용됩니다.

VHM을 사용해 개인 또는 공용 클라우드 인스턴스를 수집하여 가상화 그룹으로 편성할 수 있습니다. 가상화 클라이언트를 이렇게 편성한 상태에서 Taskomatic은 Uyuni Web UI에 표시할 클라이언트의 데이터를 수집합니다. VHM을 통해 가상화 클라이언트에서 일치하는 구독을 사용할 수도 있습니다.

Uyuni 서버에서 VHM을 생성한 후 이를 사용해 사용 가능한 공용 클라우드 인스턴스를 목록에 포함할 수 있습니다. VHM을 사용해 Kubernetes로 생성한 클러스터를 관리할 수도 있습니다.

- Amazon Web Services에서 VHM을 사용하는 방법에 대한 자세한 내용은 [Client-configuration > Vhm-aws](#)를 참조하십시오.
- Microsoft Azure에서 VHM을 사용하는 방법에 관한 자세한 내용은 [Client-configuration > Vhm-azure](#)를 참조하십시오.
- Google Compute Engine에서 VHM을 사용하는 방법에 대한 자세한 내용은 [Client-configuration > Vhm-gce](#)를 참조하십시오.
- Nutanix에서 VHM을 사용하는 방법에 대한 자세한 내용은 [Client-configuration > Vhm-nutanix](#)를 참조하십시오.
- VMWare vSphere에서 VHM을 사용하는 방법에 대한 자세한 내용은 [Client-configuration > Vhm-vmware](#)를 참조하십시오.
- 다른 호스트에서 VHM을 사용하는 방법에 대한 자세한 내용은 [Client-configuration > Vhm-file](#)을 참조하십시오.

10.1. 가상 호스트 관리자 및 Amazon Web Services

가상 호스트 관리자(VHM)를 사용하여 Amazon Web Services(AWS)에서 인스턴스를 수집할 수 있습니다.

VHM을 통해 Uyuni는 클러스터에 대한 정보를 획득하고 보고할 수 있습니다. VHM에 대한 자세한 내용은 [Client-configuration > Vhm](#)을 참조하십시오.

10.1.1. Amazon EC2 VHM 생성

가상 호스트 관리자(VHM)는 Uyuni 서버에서 실행됩니다.

Uyuni 서버에 virtual-host-gatherer-libcloud 패키지를 설치했는지 확인하십시오.

절차: Amazon EC2 VHM 생성

1. Uyuni Web UI에서 **시스템 > 가상 호스트 관리자**로 이동합니다.
2. **[생성]**을 클릭하고 드롭다운 메뉴에서 Amazon EC2를 선택합니다.
3. Amazon EC2 >> >> >> >> 섹션에서 다음과 같은 파라미터를 사용합니다.
 - >> 필드에 VHM의 사용자 정의 이름을 입력합니다.
 - >> > ID 필드에서 Amazon이 제공하는 액세스 키 ID를 입력합니다.

- 필드에 Amazon 인스턴스에 연결된 비밀 액세스 키를 입력합니다.
- 필드에 사용할 지역을 입력합니다.
- 필드에 VM이 있는 영역을 입력합니다. 이 영역은 작업과 일치하는 구독에 필요합니다. 지역 및 영역 설정에 대한 자세한 설명은 [client-configuration:virtualization.pdf](#)를 참조하십시오.

4. [생성]을 클릭하여 변경 사항을 저장하고 VHM을 생성합니다.

5. 페이지에서 새 VHM을 선택합니다.

6. 페이지에서 [데이터 새로 고침]을 클릭하여 새 VHM을 목록에 포함합니다.

어떤 객체 및 리소스가 목록에 포함되었는지 확인하려면 **시스템**, **시스템 목록**, **가상 시스템**으로 이동하십시오.

Amazon 공용 클라우드에서 실행되는 인스턴스는 뒤에 17개의 16진 숫자가 나오는 형식으로 Uyuni 서버에 UUID를 보고합니다.

I1234567890abcdef0

10.1.2. 가상 호스트 관리자에 대한 AWS 권한

보안상의 이유로 항상 수행할 작업에 가능한 최소 권한을 부여해야 합니다. 사용자에게 과도한 권한이 있는 액세스 키를 사용하여 AWS에 연결하는 것은 권장되지 않습니다.

SUSE Manager가 AWS에서 필요한 정보를 수집하기 위해서는 VHM에 EC2 인스턴스 및 주소를 설명할 수 있는 권한이 필요합니다. 이러한 권한을 부여할 수 있는 한 가지 방법은 이 작업과 관련된 새 IAM 사용자(ID 및 액세스 관리)를 생성하고 다음과 같이 정책을 생성한 후 사용자에게 연결하는 것입니다.

```
{
  "버전": "2012-10-17",
  "설명": [
    {
      "효과": "허용",
      "동작": [
        "ec2:DescribeAddresses",
        "ec2:DescribeInstances"
      ],
      "리소스": "*"
    }
  ]
}
```

특정 영역에 대한 액세스를 제한하여 권한을 자세하게 제한할 수 있습니다. 자세한 내용은 https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ExamplePolicies_EC2.html#iam-example-read-only를 참조하십시오.

10.2. 가상 호스트 관리자 및 Azure

가상 호스트 관리자(VHM)을 사용하여 Microsoft Azure에서 인스턴스를 수집할 수 있습니다.

VHM을 통해 Uyuni는 가상 머신에 대한 정보를 획득하고 보고할 수 있습니다. VHM에 대한 자세한 내용은 **Client-configuration** > **Vhm**을 참조하십시오.

10.2.1. 전제 조건

사용자가 생성하는 VHM은 Azure VM에 액세스하려면 올바른 권한이 할당되어 있어야 합니다.

Azure 계정에 구독 관리자로 로그인하여 Azure 사용자 계정 및 응용 프로그램이 올바른 그룹에 있는지 확인하십시오. 응용 프로그램이 속한 그룹에 따라 응용 프로그램의 역할이 결정되고, 이에 따라 권한도 결정됩니다.

10.2.2. Azure VHM 생성

가상 호스트 관리자(VHM)는 Uyuni 서버에서 실행됩니다.

Uyuni 서버에 virtual-host-gatherer-libcloud 패키지를 설치했는지 확인하십시오.

절차: Azure VHM 생성

1. Uyuni Web UI에서 **시스템** > **가상 호스트 관리자**로 이동합니다.
2. **[생성]**을 클릭하고 드롭다운 메뉴에서 Azure를 선택합니다.
3. Azure 설정 섹션에서 다음과 같은 파라미터를 사용합니다.
 - 사용자 정의 이름 필드에 VHM의 사용자 정의 이름을 입력합니다.
 - ID 필드에서 Azure > > 페이지에서 찾은 구독 ID를 입력합니다.
 - > ID 필드에 응용 프로그램 등록 시 수집한 응용 프로그램 ID를 입력합니다.
 - > ID 필드에서 응용 프로그램을 등록할 때 수집한 Azure에서 제공한 테넌트 ID를 입력합니다.
 - > 페일드에 Azure 인스턴스에 연결된 비밀 키를 입력합니다.
 - > 페일드에 VM이 있는 영역을 입력합니다. 예를 들어, 서유럽의 경우 westeurope를 입력합니다. 이 키는 작업과 일치하는 등록을 위해 필요합니다.
4. **[생성]**을 클릭하여 변경 사항을 저장하고 VHM을 생성합니다.
5. > > > 페이지에서 새 VHM을 선택합니다.
6. > > 페이지에서 **[데이터 새로 고침]**을 클릭하여 새 VHM을 목록에 포함합니다.

어떤 객체 및 리소스가 목록에 포함되었는지 확인하려면 **시스템** > **시스템 목록** > **가상 시스템**으로 이동하십시오.

10.2.3. 권한 할당

권한이 올바르게 설정되지 않으면 virtual-host-gatherer를 실행할 때 다음과 같은 오류를 수신할 수 있습니다.

일반 오류: [AuthorizationFailed] 객체 ID가 'object_ID'인 'client_name' 클라이언트에 '/subscriptions/not-very-secret-subscription-id' 범위에 걸쳐 'Microsoft.Compute/virtualMachines/read' 작업을 수행할 권한이 없거나 이 범위가 잘못되었습니다. 액세스가 최근에 허용되었다면 인증서를 새로 고침하시기 바랍니다.

인증서가 올바른지 확인하려면 다음과 같이 Uyuni 서버의 프롬프트에서 다음 명령을 실행하십시오.

```
virtual-host-gatherer -i input_azure.json -o out_azure.json -vvv
```

input_azure.json 파일은 다음과 같은 정보를 포함해야 합니다.

```
[  
 {  
   "id": "azure_vhm",  
   "module": "Azure",  
   "subscription_id": "subscription-id",  
   "application_id": "application-id",  
   "tenant_id": "tenant-id",  
   "secret_key": "secret-key",  
   "zone": "zone"  
 }  
 ]
```

10.2.4. Azure UUID

Azure 공용 클라우드에서 실행되는 인스턴스는 다음과 같이 이 UUID를 Uyuni 서버에 보고합니다.

```
13f56399-bd52-4150-9748-7190aae1ff21
```

10.3. 가상 호스트 관리자 및 Google Compute Engine

가상 호스트 관리자(VHM)을 사용하여 Google Compute Engine(GCE)에서 인스턴스를 수집할 수 있습니다.

VHM을 통해 Uyuni는 가상 머신에 대한 정보를 획득하고 보고할 수 있습니다. VHM에 대한 자세한 내용은 **Client-configuration > Vhm**을 참조하십시오.

10.3.1. 전제 조건

사용자가 생성하는 VHM은 GCE VM에 액세스하려면 올바른 권한이 할당되어 있어야 합니다.

Google 클라우드 플랫폼 계정에 관리자로 로그인하여 Cloud Identity and Access Management(IAM) 도구를 사용해 서비스 계정에 적절한 역할이 있는지 확인합니다.

10.3.2. GCE VHM 생성

가상 호스트 관리자(VHM)는 Uyuni 서버에서 실행됩니다.

VHM을 실행하려면 Uyuni 서버가 포트 443을 열어 클라이언트에 액세스해야 합니다.

Uyuni 서버에 virtual-host-gatherer-libcloud 패키지를 설치했는지 확인하십시오.

시작하기 전에 GCE 패널에 로그인하여 인증서 파일을 다운로드합니다. 이 파일을 Uyuni 서버에 로컬로 저장하고 경로를 적어둡니다.

절차: GCE VHM 생성

1. Uyuni Web UI에서 **시스템** > **가상 호스트 관리자**로 이동합니다.
2. **[생성]**을 클릭하고 드롭다운 메뉴에서 Google Compute Engine을 선택합니다.
3. Google Compute Engine >>>>> 섹션에서 다음과 같은 파라미터를 사용합니다.
 - >>> 필드에 VHM의 사용자 정의 이름을 입력합니다.
 - >>>> >> 필드에 서비스 계정에 연결된 이메일 주소를 입력합니다.
 - >>> >> 필드에 Uyuni 서버의 로컬 경로를 GCE 패널에서 다운로드한 키에 입력합니다.
 - >>>> ID 필드에 GCE 인스턴스가 사용하는 프로젝트 ID를 입력합니다.
 - >> 필드에 VM이 있는 영역을 입력합니다. 이 키는 작업과 일치하는 등록을 위해 필요합니다.
4. **[생성]**을 클릭하여 변경 사항을 저장하고 VHM을 생성합니다.
5. >> >>> >>> 페이지에서 새 VHM을 선택합니다.
6. >> >> 페이지에서 **[데이터 새로 고침]**을 클릭하여 새 VHM을 목록에 포함합니다.

어떤 객체 및 리소스가 목록에 포함되었는지 확인하려면 **시스템** > **시스템 목록** > **가상 시스템**으로 이동하십시오.

10.3.3. 권한 할당

권한이 올바르게 설정되지 않으면 virtual-host-gatherer를 실행할 때 다음과 같은 오류를 수신할 수 있습니다.

```
오류: {'domain': 'global', 'reason': 'forbidden', 'message': "Required 'compute.zones.list' permission for 'projects/project-id'"}  
오류: 지정된 인증서를 사용해 Google Compute Engine 공용 클라우드에 연결할 수 없습니다.
```

인증서가 올바른지 확인하려면 다음과 같이 Uyuni 서버의 프롬프트에서 다음 명령을 실행하십시오.

```
virtual-host-gatherer -i input_google.json -o out_google.json -vvv
```

input_google.json 파일은 다음과 같은 정보를 포함해야 합니다.

```
[  
  {  
    "id": "google_vhm",  
    "module": "GoogleCE",  
    "service_account_email": "mail@example.com",  
    "cert_path": "secret-key",  
    "project_id": "project-id",  
    "zone": "zone"  
  }  
]
```

10.3.4. GCE UUID

Google 공용 클라우드에서 실행되는 인스턴스는 다음과 같이 이 UUID를 Uyuni 서버에 보고합니다.

152986662232938449

10.4. Nutanix를 이용한 가상화

가상 호스트 관리자(VHM)를 설정하여 Uyuni에서 Nutanix AHV 가상 머신을 사용할 수 있습니다. 먼저 Uyuni 서버에서 VHM을 설정하고 사용 가능한 VM 호스트를 목록에 포함해야 합니다.

10.4.1. VHM 설정

가상 호스트 관리자(VHM)는 Uyuni 서버에서 실행됩니다.

Uyuni 서버에 virtual-host-gatherer-Nutanix 패키지를 설치했는지 확인하십시오.

VHM을 실행하려면 Uyuni 서버가 포트 9440을 열어 Nutanix Prism Element API에 액세스해야 합니다.

절차: Nutanix VHM 생성

1. Uyuni Web UI에서 시스템 > 가상 호스트 관리자로 이동합니다.
2. [생성]을 클릭하고 Nutanix AHV를 선택합니다.
3. Nutanix AHV 세션에서 다음과 같은 파라미터를 사용합니다.
 - 사용자 정의 이름 필드에 VHM의 사용자 정의 이름을 입력합니다.
 - 도메인 필드에 전체 도메인 이름(FQDN) 또는 호스트 IP 주소를 입력합니다.
 - Prism Element API 포트를 입력합니다(예: 9440).
 - VM 호스트와 연결된 사용자 이름을 입력합니다.
 - VM 호스트 사용자와 연결된 비밀번호를 입력합니다.
4. [생성]을 클릭하여 변경 사항을 저장하고 VHM을 생성합니다.

5. ← ← ← 페이지에서 새 VHM을 선택합니다.
6. ← ← 페이지에서 [데이터 새로 고침]을 클릭하여 새 VHM을 목록에 포함합니다.

어떤 객체 및 리소스가 목록에 포함되었는지 확인하려면 **시스템**, **시스템 목록**, **가상 시스템**으로 이동하십시오.



HTTPS를 사용하여 브라우저에서 Nutanix Prism API 서버에 연결하면 일부 경우 ← ← ← ← 오류가 발생할 수 있습니다. 이 경우 가상 호스트 관리자에서 데이터를 새로 고칠 수 없습니다. Nutanix API 서버에는 유효한 SSL 인증서(자체 서명되지 않음)가 필요합니다. Nutanix SSL 인증서에 사용자 정의 CA 기관을 사용하는 경우 사용자 정의 CA 인증서를 Uyuni 서버의 /etc/pki/trust/anchors에 복사합니다.

명령줄에서 update-ca-certificates 명령을 실행하여 인증서를 다시 신뢰하고 spacewalk 서비스를 다시 시작합니다.

taskomatic은 VHM을 생성하여 구성한 후 데이터 수집을 자동으로 실행합니다. 데이터 수집을 수동으로 수행하려면 **시스템**, **가상 호스트 관리자**로 이동하여 적절한 VHM을 선택한 후 [데이터 새로 고침]을 클릭합니다.

Uyuni는 API를 사용해 VHM에 연결하고 가상 호스트에 대한 정보를 요청할 수 있는 virtual-host-gatherer라고 하는 도구와 함께 제공됩니다. virtual-host-gatherer는 각 모듈이 특정 VHM을 활성화하는 옵션 모듈의 개념을 유지합니다. 이 도구는 Taskomat이 야간에 자동으로 호출합니다. virtual-host-gatherer 도구의 로그 파일은 /var/log/rhn/gatherer.log에 있습니다.

10.5. VMWare를 이용한 가상화

가상 호스트 관리자(VHM)를 설정하여 Uyuni에서 ESXi 및 vCenter를 포함한 VMWare vSphere 가상 머신을 사용할 수 있습니다.

먼저 Uyuni 서버에서 VHM을 설정하고 사용 가능한 VM 호스트를 목록에 포함해야 합니다. 이렇게 하면 Taskomatic이 VM API를 사용해 데이터 수집을 시작할 수 있습니다.

10.5.1. VHM 설정

가상 호스트 관리자(VHM)는 Uyuni 서버에서 실행됩니다.

VHM을 실행하려면 Uyuni 서버가 포트 443을 열어 VMWare API에 액세스해야 합니다.

VMWare 호스트는 액세스 역할 및 권한을 사용해 호스트 및 게스트에 대한 액세스를 제어합니다. VHM에 의해 목록에 포함되었으면 하는 모든 VMWare 객체 또는 리소스에 최소한 ← ← 권한이 있는지 확인합니다. 객체 또는 리소스를 제외하려면 해당 객체 또는 리소스를 no-access로 표시하십시오.

새 호스트를 Uyuni에 추가하려면 사용자 및 객체에 할당된 역할 및 권한을 Uyuni가 목록에 포함해야 할지 여부를 고려해야 합니다.

사용자, 역할 및 권한에 대한 자세한 내용은 VMWare vSphere 문서(<https://docs.vmware.com/en/VMware-vSphere/index.html>)에서 참조하십시오.

절차: VMWare VHM 생성

1. Uyuni Web UI에서 시스템 > 가상 호스트 관리자로 이동합니다.
2. [생성]을 클릭하고 VMWare 그룹을 선택합니다.
3. VMWare 그룹 설정 섹션에서 다음과 같은 파라미터를 사용합니다.
 - 그룹 필드에 VHM의 사용자 정의 이름을 입력합니다.
 - 그룹 그룹 필드에 전체 도메인 이름(FQDN) 또는 호스트 IP 주소를 입력합니다.
 - 그룹 그룹 필드에 사용할 ESXi API 포트를 입력합니다(예: 443).
 - 그룹 그룹 필드에 VM 호스트와 연결된 사용자 이름을 입력합니다.
 - 그룹 그룹 필드에 VM 호스트 사용자와 연결된 비밀번호를 입력합니다.
4. [생성]을 클릭하여 변경 사항을 저장하고 VHM을 생성합니다.
5. 그룹 페이지에서 새 VHM을 선택합니다.
6. 그룹 페이지에서 [데이터 새로 고침]을 클릭하여 새 VHM을 목록에 포함합니다.

어떤 객체 및 리소스가 목록에 포함되었는지 확인하려면 시스템 > 시스템 목록 > 가상 시스템으로 이동하십시오.



HTTPS를 사용해 브라우저에서 ESXi 서버에 연결하면 때로 그룹 그룹 오류가 로깅될 수 있습니다. 이런 오류가 발생하는 경우 가상 호스트 서버에서 데이터를 새로 고침하는 작업이 실패합니다. 이 문제를 해결하려면 ESXi 서버에서 인증서를 추출하여 /etc/pki/trust/anchors로 복사하십시오. 명령줄에서 update-ca-certificates 명령을 실행하여 인증서를 다시 신뢰하고 spacewalk 서비스를 다시 시작합니다.

taskomatic은 VHM을 생성하여 구성한 후 데이터 수집을 자동으로 실행합니다. 데이터 수집을 수동으로 수행하려면 시스템 > 가상 호스트 관리자로 이동하여 적절한 VHM을 선택한 후 [데이터 새로 고침]을 클릭합니다.

Uyuni는 API를 사용해 VHM에 연결하고 가상 호스트에 대한 정보를 요청할 수 있는 virtual-host-gatherer라고 하는 도구와 함께 제공됩니다. virtual-host-gatherer는 각 모듈이 특정 VHM을 활성화하는 옵션 모듈의 개념을 유지합니다. 이 도구는 Taskomat이 야간에 자동으로 호출합니다. virtual-host-gatherer 도구의 로그 파일은 /var/log/rhn/gatherer.log에 있습니다.

10.5.2. VMWare에서 SSL 오류 문제 해결

VMWare 설치를 구성하는 중에 SSL 오류가 발생한 경우 VMWare에서 CA 인증서 파일을 다운로드하여 Uyuni에서 신뢰해야 합니다.

절차: VMWare CA 인증서 신뢰

1. VMWare 설치에서 CA 인증서를 다운로드합니다. vCenter Web UI에 로그인해 [신뢰할 수 있는 루트 CA 인증서 다운로드]를 클릭하여 다운로드할 수 있습니다.
2. 다운로드한 CA 인증서 파일이 .zip 형식인 경우 아카이브의 압축을 풉니다. 인증서 파일의 확장자는 숫자입니다. 예를 들면 certificate.0과 같습니다.
3. 인증서 파일을 Uyuni 서버로 복사하고 /etc/pki/trust/anchors/ 디렉토리에 저장합니다.
4. 복사한 인증서의 파일 이름 접미사를 .crt 또는 .pem으로 변경합니다.

5. Uyuni 서버의 명령 프롬프트에서 다음과 같이 CA 인증서 레코드를 업데이트합니다.

```
update-ca-certificates
```

10.6. 다른 타사 공급자를 통한 가상화

Xen, KVM 또는 VMware가 아닌 타사 가상화 공급자를 사용하려면 JSON 구성 파일을 Uyuni로 임포트하면 됩니다.

이와 마찬가지로 API에 대한 직접 액세스를 제공하지 않는 VMWare가 설치된 경우 파일 기반 VHM은 몇 가지 기본 관리 기능을 제공합니다.



이 옵션은 virtual-host-gatherer 도구로 생성한 파일을 임포트하기 위한 것입니다.
이 옵션은 수동으로 생성한 파일을 위해 설계된 것은 아닙니다.

절차: JSON 파일 익스포트 및 임포트

1. VM 네트워크에서 virtual-host-gatherer를 실행하여 JSON 구성 파일을 익스포트합니다.
2. 생성한 파일을 Uyuni 서버가 액세스할 수 있는 위치에 저장합니다.
3. Uyuni Web UI에서 **시스템 > 가상 호스트 관리자**로 이동합니다.
4. **[생성]**을 클릭하고 **-->**을 선택합니다.
5. **-->** **-->** **-->** **-->** **-->** **-->** **-->** 섹션에서 다음과 같은 파라미터를 사용합니다.
 - **User** 필드에 VHM의 사용자 정의 이름을 입력합니다.
 - **Url** 필드에 익스포트한 JSON 구성 파일의 경로를 입력합니다.
6. **[생성]**을 클릭하여 변경 사항을 저장하고 VHM을 생성합니다.
7. **-->** **-->** **-->** 페이지에서 새 VHM을 선택합니다.
8. **-->** **-->** 페이지에서 **[데이터 새로 고침]**을 클릭하여 새 VHM을 목록에 포함합니다.

목록 3. 예: 익스포트한 JSON 구성 파일:

```
{
  "examplevhost": {
    "10.11.12.13": {
      "cpuArch": "x86_64",
      "cpuDescription": "AMD Opteron(tm) Processor 4386",
      "cpuMhz": 3092.212727,
      "cpuVendor": "amd",
      "hostIdentifier": "'vim.HostSystem:host-182'",
      "name": "11.11.12.13",
      "os": "VMware ESXi",
      "osVersion": "5.5.0",
      "ramMb": 65512,
    }
  }
}
```

```
        "totalCpuCores": 16,
        "totalCpuSockets": 2,
        "totalCpuThreads": 16,
        "type": "vmware",
        "vms": {
            "vCenter": "564d6d90-459c-2256-8f39-3cb2bd24b7b0"
        }
    },
    "10.11.12.14": {
        "cpuArch": "x86_64",
        "cpuDescription": "AMD Opteron(tm) Processor 4386",
        "cpuMhz": 3092.212639,
        "cpuVendor": "amd",
        "hostIdentifier": "'vim.HostSystem:host-183'",
        "name": "10.11.12.14",
        "os": "VMware ESXi",
        "osVersion": "5.5.0",
        "ramMb": 65512,
        "totalCpuCores": 16,
        "totalCpuSockets": 2,
        "totalCpuThreads": 16,
        "type": "vmware",
        "vms": {
            "49737e0a-c9e6-4ceb-aef8-6a9452f67cb5": "4230c60f-3f98-2a65-f7c3-600b26b79c22",
            "5a2e4e63-a957-426b-bfa8-4169302e4fdb": "42307b15-1618-0595-01f2-427ffccdd88e",
            "NSX-gateway": "4230d43e-aafe-38ba-5a9e-3cb67c03a16a",
            "NSX-l3gateway": "4230b00f-0b21-0e9d-dfde-6c7b06909d5f",
            "NSX-service": "4230e924-b714-198b-348b-25de01482fd9"
        }
    }
}
```

자세한 내용은 Uyuni 서버의 사용자 지정 페이지에서 `virtual-host-gatherer`를 검색하십시오.

man virtual-host-gatherer

이 패키지의 README 파일은 하이퍼바이너의 ‘유형’ 등에 대한 배경 정보를 제공합니다.

/usr/share/doc/packages/virtual-host-gatherer/README.md

사용자 지정 페이지 및 ‘README’ 파일에는 예시 구성 파일도 포함되어 있습니다.

Chapter 11. GNU Free Documentation License

Copyright © 2000, 2001, 2002 Free Software Foundation, Inc. 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA. Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

-
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
 - H. Include an unaltered copy of this License.
 - I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
 - J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
 - K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
 - L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
 - M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
 - N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
 - O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the

Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

Copyright (c) YEAR YOUR NAME.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.

A copy of the license is included in the section entitled "GNU Free Documentation License".