Red Hat Enterprise Linux 8

RHEL 8의 cloud-init 구성 및 관리

Red Hat Enterprise Linux 클라우드 인스턴스 초기화 자동화

Last Updated: 2023-09-19

Red Hat Enterprise Linux 8 RHEL 8의 cloud-init 구성 및 관리

Red Hat Enterprise Linux 클라우드 인스턴스 초기화 자동화

법적 공지

초록

cloud-init 패키지를 사용하여 RHEL의 여러 클라우드 인스턴스를 효율적으로 생성할 수 있습니다. 이렇게 하면 다양한 클라우드 플랫폼에서 RHEL을 일관되고 반복할 수 있습니다. 다음 장에서는 다음에 대한 정보를 제공합니다. cloud-init 작동 방식 cloud-init를 사용하여 클라우드 인스턴스 시작 방법 cloud-init Red Hat에서 지원하는 클라우드 사용

차례

보다 포괄적 수용을 위한 오픈 소스 용어 교체 3

RED HAT 문서에 관한 피드백 제공 4

1장. 퍼블릭 클라우드 플랫폼에 RHEL 소개 5
1.1. 퍼블릭 클라우드에서 RHEL을 사용할 때의 이점 5
1.2. RHEL의 퍼블릭 클라우드 사용 사례 6
1.3. 퍼블릭 클라우드로 마이그레이션할 때 빈번한 우려 사항 6
1.4. 퍼블릭 클라우드 배포를 위한 RHEL 가져오기 7
1.5. RHEL 클라우드 인스턴스 생성 방법 8

2장. CLOUD-INIT 소개 9
2.1. CLOUD-INIT 구성 9
2.2. CLOUD-INIT는 단계별로 작동합니다. 10
2.3. CLOUD-INIT 모듈은 단계에서 실행됩니다. 10
2.4. CLOUD-INIT는 사용자 데이터, 메타데이터 및 공급 업체 데이터를 기반으로 작동합니다. 11
2.5. CLOUD-INIT가 클라우드 플랫폼 식별 11
2.6. 추가 리소스 12

3장. CLOUD-INIT에 대한 RED HAT 지원 13
3.1. CLOUD-INIT 중요한 디렉터리 및 파일 13
3.2. CLOUD-INIT를 사용하는 RED HAT 제품 13
3.3. RED HAT은 이러한 CLOUD-INIT 모듈 지원 14
3.4. 기본 CLOUD.CFG 파일 17
3.5. CLOUD.CFG.D 디렉토리 19
3.6. 기본 05_LOGGING.CFG 파일 20
3.7. CLOUD-INIT /VAR/LIB/CLOUD 디렉터리 레이아웃 21

4장. CLOUD-INIT 구성 23
4.1. NOCLOUD 데이터 소스에 CLOUD-INIT를 포함하는 가상 머신 생성 23
4.2. CLOUD-INIT를 사용하여 클라우드 사용자 암호 만료 25
4.3. CLOUD-INIT로 기본 사용자 이름 변경 26
4.4. CLOUD-INIT를 사용하여 루트 암호 설정 26
4.5. CLOUD-INIT를 사용하여 RED HAT 서브스크립션 관리 27
4.6. CLOUD-INIT를 사용하여 사용자 및 사용자 옵션 추가 29
4.7. CLOUD-INIT를 사용하여 첫 번째 부팅 명령 실행 30
4.8. CLOUD-INIT를 사용하여 SUDOERS 추가 31
4.9. CLOUD-INIT를 사용하여 정적 네트워킹 구성 설정 32
4.10. CLOUD-INIT를 사용하여 ROOT 사용자만 구성 33
4.11. CLOUD-INIT에서 CONTAINER-STORAGE-SETUP으로 스토리지 설정 34
4.12. CLOUD-INIT로 시스템 로케일 변경 35
4.13. CLOUD-INIT 및 쉘 스크립트 36
4.14. CLOUD-INIT에서 구성 파일 업데이트 방지 36
4.15. CLOUD-INIT를 실행한 후 KVM 게스트 이미지에서 생성된 VM 수정 37
4.16. CLOUD-INIT가 실행된 후 특정 데이터 소스의 VM 수정 38
4.17. CLOUD-INIT 문제 해결 39

보다 포괄적 수용을 위한 오픈 소스 용어 교체

Red Hat은 코드, 문서 및 웹 속성에서 문제가 있는 언어를 교체하기 위해 최선을 다하고 있습니다. 먼저 마스터(master), 슬레이브(slave), 블랙리스트(blacklist), 화이트리스트(whitelist) 등 네 가지 용어를 교체하고 있습니다. 이러한 변경 작업은 작업 범위가 크므로 향후 여러 릴리스에 걸쳐 점차 구현할 예정입니다. 자세한 내용은 CTO Chris Wright의 메시지 를 참조하십시오.

RED HAT 문서에 관한 피드백 제공

문서에 대한 피드백에 감사드립니다. 어떻게 개선할 수 있는지 알려주십시오.

특정 문구에 대한 의견 제출

1. **Multi-page HTML** 형식으로 설명서를 보고 페이지가 완전히 로드된 후 오른쪽 상단 모서리에 **피드백** 버튼이 표시되는지 확인합니다.
2. 커서를 사용하여 주석 처리할 텍스트 부분을 강조 표시합니다.
3. 강조 표시된 텍스트 옆에 표시되는 **피드백 추가** 버튼을 클릭합니다.
4. 의견을 추가하고 **제출** 을 클릭합니다.

Jira를 통해 피드백 제출 (등록 필요)

1. Jira 웹 사이트에 로그인합니다.
2. 상단 탐색 모음에서 **생성** 을 클릭합니다.
3. **Summary** (요약) 필드에 설명 제목을 입력합니다.
4. **Description** (설명) 필드에 개선을 위한 제안을 입력합니다. 문서의 관련 부분에 대한 링크를 포함합니다.
5. 대화 상자 하단에서 **생성** 을 클릭합니다.

1장. 퍼블릭 클라우드 플랫폼에 RHEL 소개

퍼블릭 클라우드 플랫폼은 컴퓨팅 리소스를 서비스로 제공합니다. 온프레미스 하드웨어를 사용하는 대신 RHEL(Red Hat Enterprise Linux) 시스템을 퍼블릭 클라우드 인스턴스로 포함한 IT 워크로드를 실행할 수 있습니다.

퍼블릭 클라우드 플랫폼의 RHEL에 대한 자세한 내용은 다음을 참조하십시오.

- 퍼블릭 클라우드에서 RHEL을 사용할 때의 이점
- RHEL의 퍼블릭 클라우드 사용 사례
- 퍼블릭 클라우드로 마이그레이션할 때 빈번한 우려 사항
- 퍼블릭 클라우드 배포를 위한 RHEL 가져오기
- RHEL 클라우드 인스턴스 생성 방법

1.1. 퍼블릭 클라우드에서 RHEL을 사용할 때의 이점

퍼블릭 클라우드 플랫폼에 있는 클라우드 인스턴스로의 RHEL은 RHEL 온-프레미스 물리적 시스템 또는 VM(가상 머신)에 비해 다음과 같은 이점이 있습니다.

- **유연하고 세분화된 리소스 할당**
  RHEL의 클라우드 인스턴스는 클라우드 플랫폼에서 VM으로 실행되며, 이는 일반적으로 클라우드 서비스 공급자가 관리하는 원격 서버 클러스터를 의미합니다. 따라서 특정 유형의 CPU 또는 스토리지와 같은 인스턴스에 하드웨어 리소스를 할당하면 소프트웨어 수준에서 발생하며 쉽게 사용자 지정할 수 있습니다.

  로컬 RHEL 시스템과 비교하여 물리적 호스트의 기능에 의해 제한되지도 않습니다. 대신 클라우드 공급자가 제공하는 선택 사항에 따라 다양한 기능을 선택할 수 있습니다.

- **공간 및 비용 효율성**
  클라우드 워크로드를 호스팅하기 위해 온-프레미스 서버를 소유할 필요가 없습니다. 이렇게 하면 물리적 하드웨어와 관련된 공간, 전원 및 유지 관리 요구 사항을 방지할 수 있습니다.

  대신, 퍼블릭 클라우드 플랫폼에서는 클라우드 인스턴스를 사용하기 위해 클라우드 공급자를 직접 지불합니다. 비용은 일반적으로 인스턴스에 할당된 하드웨어 및 이를 사용하는 시간을 기반으로 합니다. 따라서 요구 사항에 따라 비용을 최적화할 수 있습니다.

- **소프트웨어 제어 구성**
  클라우드 인스턴스의 전체 구성은 클라우드 플랫폼에 데이터로 저장되며 소프트웨어에 의해 제어됩니다. 따라서 인스턴스를 쉽게 생성, 제거, 복제 또는 마이그레이션할 수 있습니다. 클라우드 인스턴스는 클라우드 공급자 콘솔에서도 원격으로 작동하며 기본적으로 원격 스토리지에 연결됩니다.

  또한 언제든지 클라우드 인스턴스의 현재 상태를 스냅샷으로 백업할 수 있습니다. 나중에 스냅샷을 로드하여 인스턴스를 저장된 상태로 복원할 수 있습니다.

- **호스트 및 소프트웨어 호환성 분리**
  로컬 VM과 유사하게 클라우드 인스턴스의 RHEL 게스트 운영 체제는 가상화된 커널에서 실행됩니다. 이 커널은 호스트 운영 체제와 인스턴스에 연결하는 데 사용하는 *클라이언트* 시스템과 다릅니다.

  따라서 모든 운영 체제를 클라우드 인스턴스에 설치할 수 있습니다. 즉, RHEL 퍼블릭 클라우드 인스턴스에서 로컬 운영 체제에서 사용할 수 없는 RHEL별 애플리케이션을 실행할 수 있습니다.

또한 인스턴스의 운영 체제가 불안정해지거나 손상되는 경우에도 클라이언트 시스템은 어떤 방식으로든 영향을 받지 않습니다.

추가 리소스

- 퍼블릭 클라우드란 무엇입니까?
- 하이퍼스케일러(Hyperscaler)란?
- 클라우드 컴퓨팅 유형
- RHEL의 퍼블릭 클라우드 사용 사례
- 퍼블릭 클라우드 배포를 위한 RHEL 가져오기

1.2. RHEL의 퍼블릭 클라우드 사용 사례

퍼블릭 클라우드에 배포하면 많은 이점이 있지만 모든 시나리오에서 가장 효율적인 솔루션은 아닙니다. RHEL 배포를 퍼블릭 클라우드로 마이그레이션할지 여부를 평가하고 있는 경우 사용 사례가 퍼블릭 클라우드의 이점을 활용할지 여부를 고려하십시오.

유용한 사용 사례

- 퍼블릭 클라우드 인스턴스 배포는 배포의 활성 컴퓨팅 성능을 유연하게 늘리고 줄이는 데 매우 효과적이며 확장 및 축소 라고도 합니다. 따라서 다음 시나리오에서는 퍼블릭 클라우드에서 RHEL을 사용하는 것이 좋습니다.
  - 최대 워크로드 및 낮은 일반 성능 요구 사항이 있는 클러스터입니다. 필요에 따라 확장 및 축소하면 리소스 비용 측면에서 매우 효율적일 수 있습니다.
  - 클러스터를 신속하게 설정하거나 확장합니다. 이렇게 하면 로컬 서버를 설정하는 데 드는 초기 비용이 발생하지 않습니다.
- 클라우드 인스턴스는 로컬 환경에서 발생하는 작업의 영향을 받지 않습니다. 따라서 백업 및 재해 복구에 사용할 수 있습니다.

잠재적으로 문제가 있는 사용 사례

- 조정할 수 없는 기존 환경을 실행하고 있습니다. 기존 배포의 특정 요구에 맞게 클라우드 인스턴스를 사용자 정의하는 것은 현재 호스트 플랫폼과 비교하여 비용 효율적이지 않을 수 있습니다.
- 예산에 하드 제한으로 작업하고 있습니다. 로컬 데이터 센터에서 배포를 유지 관리하면 일반적으로 퍼블릭 클라우드보다 유연성이 떨어지지만 최대 리소스 비용을 제어할 수 있습니다.

다음 단계

- 퍼블릭 클라우드 배포를 위한 RHEL 가져오기

추가 리소스

- 애플리케이션을 클라우드로 마이그레이션해야 합니까? 여기 결정 방법이 있습니다.

1.3. 퍼블릭 클라우드로 마이그레이션할 때 빈번한 우려 사항

RHEL 워크로드를 로컬 환경에서 퍼블릭 클라우드 플랫폼으로 이동하면 변경 사항에 대한 우려가 발생할 수 있습니다. 다음은 가장 일반적으로 묻는 질문입니다.

RHEL이 로컬 가상 머신과 다른 방식으로 작동합니까?

대부분의 경우 퍼블릭 클라우드 플랫폼의 RHEL 인스턴스는 온프레미스 서버와 같은 로컬 호스트의 RHEL 가상 머신과 동일하게 작동합니다. 주요 예외 사항은 다음과 같습니다.

- 퍼블릭 클라우드 인스턴스는 프라이빗 오케스트레이션 인터페이스 대신 클라우드 리소스를 관리하기 위해 공급자별 콘솔 인터페이스를 사용합니다.
- 중첩된 가상화와 같은 특정 기능이 제대로 작동하지 않을 수 있습니다. 배포에 특정 기능이 중요한 경우 선택한 퍼블릭 클라우드 공급자와 사전에 기능의 호환성을 확인하십시오.

내 데이터가 로컬 서버와 달리 퍼블릭 클라우드에서 안전하게 유지됩니까?

RHEL 클라우드 인스턴스의 데이터는 소유권에 있으며 퍼블릭 클라우드 공급자는 이에 액세스할 수 없습니다. 또한 주요 클라우드 공급자는 전송 시 데이터 암호화를 지원하므로 가상 머신을 퍼블릭 클라우드로 마이그레이션할 때 데이터의 보안이 향상됩니다.

RHEL 퍼블릭 클라우드 인스턴스의 일반적인 보안은 다음과 같이 관리됩니다.

- 클라우드 하이퍼바이저의 보안을 담당하는 퍼블릭 클라우드 공급자
- Red Hat은 인스턴스의 RHEL 게스트 운영 체제의 보안 기능을 제공합니다.
- 클라우드 인프라의 특정 보안 설정 및 관행을 관리합니다.

내 지역이 RHEL 퍼블릭 클라우드 인스턴스의 기능에 미치는 영향은 무엇입니까?

지리적 위치에 관계없이 퍼블릭 클라우드 플랫폼에서 RHEL 인스턴스를 사용할 수 있습니다. 따라서 온-프레미스 서버와 동일한 리전의 인스턴스를 실행할 수 있습니다.

그러나 물리적으로 멀리 있는 지역에 인스턴스를 호스트하면 작동 시 대기 시간이 길어질 수 있습니다. 또한 퍼블릭 클라우드 공급자에 따라 특정 리전에서 추가 기능을 제공하거나 비용 효율성을 높일 수 있습니다. RHEL 인스턴스를 만들기 전에 선택한 클라우드 공급자에 사용할 수 있는 호스팅 리전의 속성을 검토하십시오.

1.4. 퍼블릭 클라우드 배포를 위한 RHEL 가져오기

퍼블릭 클라우드 환경에 RHEL 시스템을 배포하려면 다음을 수행합니다.

1. 요구 사항과 현재 출시에 따라 사용 사례에 맞는 최적의 클라우드 공급자를 선택합니다. 현재 RHEL 인스턴스 실행에 대해 인증된 클라우드 공급자는 다음과 같습니다.
   - AWS(Amazon Web Services)
     - 자세한 내용은 Amazon Web Services에 RHEL 8 배포를 참조하십시오.
   - GCP(Google Cloud Platform)
     - 자세한 내용은 Google Cloud Platform에 RHEL 8 배포를 참조하십시오.
   - Microsoft Azure
     - 자세한 내용은 Microsoft Azure에 RHEL 8 배포를 참조하십시오.

2. 선택한 클라우드 플랫폼에 RHEL 클라우드 인스턴스를 생성합니다. 자세한 내용은 RHEL 클라우드 인스턴스 생성 방법을 참조하십시오.

3. RHEL 배포를 최신 상태로 유지하려면 RHUI(Red Hat Update Infrastructure)를 사용하십시오.

추가 리소스

- RHUI 문서
- Red Hat Open Hybrid Cloud

1.5. RHEL 클라우드 인스턴스 생성 방법

퍼블릭 클라우드 플랫폼에 RHEL 인스턴스를 배포하려면 다음 방법 중 하나를 사용할 수 있습니다.

RHEL의 시스템 이미지를 생성하여 클라우드 플랫폼으로 가져옵니다.

- 시스템 이미지를 생성하려면 RHEL 이미지 빌더를 사용하거나 이미지를 수동으로 빌드할 수 있습니다.
- 이 방법은 기존 RHEL 서브스크립션을 사용하며 BYOOS(사용자 *서브스크립션*)라고도 합니다.
- 연간 서브스크립션 비용을 미리 지불하면 Red Hat 고객 할인 혜택을 누릴 수 있습니다.
- 고객의 고객 서비스는 Red Hat에서 제공합니다.
- 효과적으로 여러 이미지를 생성하기 위해 **cloud-init** 툴을 사용할 수 있습니다.

클라우드 공급자 마켓플레이스에서 직접 RHEL 인스턴스를 구입하십시오.

- 서비스 사용에 대한 시간당 요금을 지불하십시오. 따라서,이 방법은 또한 *지불*(PAYG)이라고합니다.
- 고객 서비스는 클라우드 플랫폼 공급자가 제공합니다.

추가 리소스

- 골든 이미지란 무엇입니까?

2장. CLOUD-INIT 소개

cloud-init 는 시스템을 부팅하는 동안 클라우드 인스턴스 초기화를 자동화하는 소프트웨어 패키지입니다. 다양한 작업을 수행하도록 **cloud-init** 를 구성할 수 있습니다. **cloud-init** 에서 수행할 수 있는 몇 가지 샘플 작업은 다음과 같습니다.

- 호스트 이름 구성
- 인스턴스에 패키지 설치
- 스크립트 실행
- 기본 VM(가상 머신) 동작 비활성화

cloud-init 구성을 위해 이미지를 가져오는 위치는 사용 방법에 따라 다릅니다.

- **cloud-init** 패키지는 Red Hat Customer Portal 에서 다운로드한 KVM 게스트 이미지에 설치됩니다. 인스턴스를 시작하면 **cloud-init** 가 활성화됩니다. Red Hat 고객 포털에서 다운로드한 KVM 게스트 이미지는 RHV(Red Hat Virtualization), RHOSP(Red Hat OpenStack Platform) 및 Red Hat OpenShift Virtualization과 함께 사용하기 위한 것입니다.
- Red Hat 고객 포털에서 RHEL ISO 이미지를 다운로드하여 사용자 지정 게스트 이미지를 생성할 수도 있습니다. 이 경우 게스트 이미지에 **cloud-init** 패키지를 직접 설치해야 합니다.
- 클라우드 공급자가 있는 이미지를 사용하려는 경우(예: AWS 또는 Azure) Red Hat 이미지 빌더를 사용하여 이미지를 생성합니다. 이미지 빌더 이미지는 특정 클라우드 공급자에 사용하도록 사용자 지정할 수 있습니다. AMI, VHD 및 qcow2에는 이미 설치된 **cloud-init** 가 포함되어 있습니다. 이미지 빌더 에 대한 정보는 사용자 지정된 RHEL 시스템 이미지 구성을 참조하십시오.

대부분의 클라우드 플랫폼은 **cloud-init** 를 지원하지만 구성 절차 및 지원되는 옵션은 다릅니다. 또는 NoCloud 환경에 대해 **cloud-init** 를 구성할 수 있습니다.

하나의 VM에서 **cloud-init** 를 구성한 다음 해당 VM을 추가 VM 또는 VM 클러스터의 템플릿으로 사용할 수 있습니다.

특정 Red Hat 제품(예: Red Hat Virtualization)은 해당 제품과 함께 사용할 수 있도록 **cloud-init** 를 구성하기 위한 절차에 대해 문서화되어 있습니다.

이 문서는 여러 위치에 있는 **cloud-init** 설명서를 참조합니다. **cloud-init** 에 대한 전체 정보는 참조된 **cloud-init** 설명서를 참조하십시오.

사전 요구 사항

- Red Hat 고객 포털 계정에 등록합니다.

2.1. CLOUD-INIT 구성

cloud-init 는 YAML 형식의 파일 지침을 사용하여 작업을 수행합니다. YAML 파일 내에 지침을 제공하여 **cloud-init** 에서 수행할 초기 구성을 결정합니다. 인스턴스가 부팅되면 **cloud-init** 서비스가 시작되고 지침을 검색하고 실행합니다. **cloud-init** 구성을 기반으로 첫 번째 부팅 중 또는 이후 VM 부팅 시 완료된 작업.

/etc/cloud/cloud.cfg 파일을 구성하고 **/etc/cloud/cloud.cfg.d/** 디렉터리에 지시문을 추가하여 작업을 정의합니다.

- **cloud.cfg** 파일에는 사용자 액세스 및 인증 및 시스템 정보와 같은 지시문이 포함되어 있습니다. 파일에는 **cloud-init** 에 대한 기본 및 선택적 모듈도 포함되어 있습니다. 이 모듈은 **cloud-init** 초기

화 단계, 구성 단계 및 최종 단계를 포함하는 세 단계로 순서대로 실행됩니다. **cloud.cfg** 파일 내에서 세 단계의 모듈은 **cloud_init_modules**,**cloud_config_modules**, **cloud_final_modules** 에 각각 나열됩니다.

- **cloud.cfg.d** 디렉토리는 **cloud-init** 에 대한 지시문을 추가할 수 있는 위치입니다. **cloud.cfg.d** 디렉토리에 지시문을 추가하는 경우 일반적으로 ***.cfg** 이라는 파일에 추가하고 파일 맨 위에 **#cloud-config** 를 항상 포함합니다.

2.2. CLOUD-INIT는 단계별로 작동합니다.

cloud-init 는 시스템을 부팅하는 동안 5단계로 작동합니다. 이러한 단계는 **cloud-init** 가 실행 중인지 여부와 다른 작업 중에서 데이터 소스를 찾을 위치를 결정합니다. 단계는 다음과 같습니다.

1. **systemd** 서비스를 통해 **cloud-init** 생성기 단계에서는 부팅 시 **cloud-init** 실행 여부를 결정합니다.
2. 로컬 단계에서 **cloud-init** 는 로컬 데이터 소스를 찾아 네트워크 구성을 적용합니다.
3. 네트워크 단계 중에 **cloud-init** 은 사용자 데이터를 처리하고 **cloud.cfg** 파일의 **cloud_init_modules** 에 나열된 모듈을 실행합니다. **cloud_init_modules** 섹션에 모듈을 활성화, 비활성화 또는 추가할 수 있습니다.
4. 구성 단계에서 **cloud-init** 는 **cloud.cfg** 파일의 **cloud_config_modules** 에 나열된 모듈을 실행합니다. **cloud_config_modules** 섹션에 모듈을 활성화, 비활성화 또는 추가할 수 있습니다.
5. 마지막 단계에서 **cloud-init** 는 **cloud.cfg** 파일의 **cloud_final_modules** 에 포함된 항목을 실행할 수 있습니다. 일반적으로 시스템을 부팅한 후 실행할 패키지 설치를 포함할 수 있으며 구성 관리 플러그인 및 사용자 스크립트를 포함할 수도 있습니다. **cloud_final_modules** 섹션에 모듈을 활성화, 비활성화 또는 추가할 수 있습니다.

5개의 부팅 단계는 **cloud-init** 문서 섹션 Boot Stage에 설명되어 있습니다.

2.3. CLOUD-INIT 모듈은 단계에서 실행됩니다.

cloud-init 가 실행되면 다음 세 단계에서 순서대로 **cloud.cfg** 내에서 모듈을 실행합니다.

1. 네트워크 단계(**cloud_init_modules**)
2. 구성 단계(**cloud_config_modules**)
3. 마지막 단계 (**cloud_final_modules**)

VM에서 **cloud-init** 가 처음 실행되면 각 단계에서 구성한 모든 모듈이 실행됩니다. 이후 **cloud-init** 실행 시 단계 내에서 모듈이 실행되는지 여부는 개별 *모듈의 모듈 빈도* 에 따라 달라집니다. 일부 모듈은 **cloud-init** 가 실행될 때마다 실행됩니다. 일부 모듈은 인스턴스 ID가 변경되어도 **cloud-init** 가 처음 실행되는 경우에만 실행됩니다.

참고

인스턴스 ID는 인스턴스를 고유하게 식별합니다. 인스턴스 ID가 변경되면 **cloud-init** 에서 인스턴스를 새 인스턴스로 처리합니다.

가능한 *모듈 빈도* 값은 다음과 같습니다.

- **인스턴스당** 모듈은 인스턴스를 처음 부팅할 때 실행됩니다. 예를 들어 인스턴스를 복제하거나 저장된 이미지에서 새 인스턴스를 생성하면 인스턴스별로 지정된 모듈이 다시 실행됩니다.
- **한 번만** 실행하면 모듈이 한 번만 실행됩니다. 예를 들어 인스턴스를 복제하거나 저장된 이미지에서 새 인스턴스를 생성하는 경우 한 번에 지정된 모듈이 해당 인스턴스에서 다시 실행되지 않습니다.
- **항상** 모듈은 모든 부팅 시 실행됩니다.

참고

모듈을 구성하거나 명령줄을 사용하여 모듈의 빈도를 재정의할 수 있습니다.

2.4. CLOUD-INIT는 사용자 데이터, 메타데이터 및 공급 업체 데이터를 기반으로 작동합니다.

cloud-init 는 사용자 데이터, 메타데이터 및 공급 업체 데이터를 사용하고 작동합니다.

- 사용자 데이터에는 **cloud.cfg** 파일 및 **cloud.cfg.d** 디렉터리에 지정하는 지시문이 포함되어 있습니다. 예를 들어 사용자 데이터에 실행할 파일, 설치할 패키지 및 쉘 스크립트가 포함될 수 있습니다. **cloud-init** 에서 허용하는 사용자 데이터 유형에 대한 정보는 **cloud-init** 문서 섹션 User-Data Formats 를 참조하십시오.
- 메타데이터에는 특정 데이터 소스와 관련된 데이터가 포함됩니다. 예를 들어 메타데이터에는 서버 이름 및 인스턴스 ID가 포함될 수 있습니다. 특정 클라우드 플랫폼을 사용하는 경우 플랫폼에서는 인스턴스에서 사용자 데이터 및 메타데이터를 찾을 위치를 결정합니다. 플랫폼에 HTTP 서비스에 메타데이터 및 사용자 데이터를 추가해야 할 수 있습니다. 이 경우 **cloud-init** 가 HTTP 서비스의 메타데이터와 사용자 데이터를 사용합니다.
- 공급업체 데이터는 조직(예: 클라우드 공급자)에서 선택적으로 제공하며, 이미지가 실행되는 환경에 맞게 이미지를 사용자 지정할 수 있는 정보를 포함합니다. **cloud-init** 는 메타데이터를 읽고 시스템을 초기화한 후 선택적 공급업체 데이터 및 사용자 데이터에 적용됩니다. 기본적으로 공급업체 데이터는 첫 번째 부팅 시 실행됩니다. 공급업체 데이터 실행을 비활성화할 수 있습니다.
메타데이터에 대한 설명은 **cloud-init** 문서 섹션 인스턴스 메타데이터, 데이터 소스 목록의 데이터 소스, 벤더 데이터에 대한 자세한 내용은 Vendor Data 를 참조하십시오.

2.5. CLOUD-INIT가 클라우드 플랫폼 식별

cloud-init 는 **ds-identify** 스크립트를 사용하여 클라우드 플랫폼을 식별하려고 합니다. 이 스크립트는 인스턴스를 처음 부팅할 때 실행됩니다.

datasource 지시문을 추가하면 **cloud-init** 가 실행되는 시간을 절약할 수 있습니다. **/etc/cloud/cloud.cfg** 파일 또는 **/etc/cloud/cloud.cfg.d** 디렉터리에 지시문을 추가합니다. 예를 들어 다음과 같습니다.

```
datasource_list:[Ec2]
```

클라우드 플랫폼에 지시어를 추가하는 것 외에도 메타데이터 URL과 같은 구성 세부 정보를 추가하여 **cloud-init** 를 추가로 구성할 수 있습니다.

```
datasource_list: [Ec2]
datasource:
  Ec2:
    metadata_urls: ['http://169.254.169.254']
```

cloud-init 가 실행된 후에는 플랫폼에 대한 자세한 정보를 제공하는 로그 파일(**run/cloud-init/ds-identify.log**)을 볼 수 있습니다.

추가 리소스

- 데이터 소스
- 어떤 데이터 소스를 사용하고 있습니까?
- 사용자 데이터를 디버깅하려면 어떻게 해야 합니까?

2.6. 추가 리소스

- cloud-init의 업스트림 문서

3장. CLOUD-INIT에 대한 RED HAT 지원

이 장에서는 **cloud-init** 에 대한 Red Hat 지원에 대해 설명합니다. 여기에는 Red Hat이 지원하는 cloud-init , **cloud-init** 모듈, 기본 디렉터리 및 파일을 사용하는 Red Hat 제품에 대한 정보가 포함되어 있습니다.

3.1. CLOUD-INIT 중요한 디렉터리 및 파일

다음 표에는 중요한 디렉터리와 파일이 포함되어 있습니다. 이러한 디렉터리 및 파일을 검토합니다. 다음과 같은 작업을 수행할 수 있습니다.

- **cloud-init**구성
- **cloud-init** 를 실행한 후 구성에 대한 정보 검색
- 로그 파일 검사
- 템플릿 찾기

시나리오 및 데이터 소스에 따라 구성에 중요한 추가 파일과 디렉터리가 있을 수 있습니다.

표 3.1. cloud-init 디렉터리 및 파일

디렉터리 또는 파일	설명
/etc/cloud/cloud.cfg	**cloud.cfg** 파일에는 기본 **cloud-init** 구성이 포함되어 있으며 각 모듈이 실행되는 단계에서 알 수 있습니다.
/etc/cloud/cloud.cfg.d	**cloud.cfg.d** 디렉토리는 **cloud-init** 에 대한 지시문을 추가할 수 있는 위치입니다.
/var/lib/cloud	**cloud-init** 가 실행되면 **/var/lib/cloud** 에 디렉터리 레이아웃이 생성됩니다. 레이아웃에는 인스턴스 구성에 대한 세부 정보를 제공하는 디렉터리 및 파일이 포함되어 있습니다.
/usr/share/doc/cloud-init/examples	**예제** 디렉터리에는 여러 예제가 포함되어 있습니다. 이를 사용하여 자체 지시문을 모델링할 수 있습니다.
/etc/cloud/templates	이 디렉터리에는 특정 시나리오에 대해 **cloud-init** 에서 활성화할 수 있는 템플릿이 포함되어 있습니다. 템플릿은 활성화의 방향을 제공합니다.
/var/log/cloud-init.log	**cloud-init.log** 파일은 디버깅에 유용한 로그 정보를 제공합니다.
/run/cloud-init	**/run/cloud-init** 디렉터리에는 데이터 소스 및 **ds-identify** 스크립트에 대한 기록된 정보가 포함되어 있습니다.

3.2. CLOUD-INIT를 사용하는 RED HAT 제품

다음 Red Hat 제품을 통해 **cloud-init** 를 사용할 수 있습니다.

- **Red Hat Virtualization** VM에 **cloud-init** 를 설치한 후 템플릿을 생성하고 해당 템플릿에서 생성된 모든 VM에 대해 **cloud-init** 함수를 활용할 수 있습니다. VM과 함께 **cloud-init** 사용 방법에 대한 정보는 Using Cloud-Init to Automate the Configuration of Virtual Machines 를 참조하십시오.
- **Red Hat OpenStack Platform. cloud-init** 를 사용하여 OpenStack의 이미지를 구성할 수 있습니다. 자세한 내용은 인스턴스 및 이미지 가이드 를 참조하십시오.
- **Red Hat CloudForms. cloud-init** 를 사용하여 Red Hat CloudForms의 VM을 프로비저닝할 수 있습니다. 자세한 내용은 가상 머신 및 인스턴스 프로비저닝에 대한 사용자 지정 템플릿 을 참조하십시오.
- **Red Hat Satellite.** Red Hat Satellite에서 **cloud-init** 를 사용할 수 있습니다. 자세한 내용은 Red Hat Virtualization에서 Cloud-init 이미지 준비를 참조하십시오.
- **Red Hat OpenShift.** OpenShift용 VM을 생성할 때 **cloud-init** 를 사용할 수 있습니다. 자세한 내용은 가상 머신 생성을 참조하십시오.

3.3. RED HAT은 이러한 CLOUD-INIT 모듈 지원

Red Hat은 대부분의 **cloud-init** 모듈을 지원합니다. 개별 모듈에는 여러 구성 옵션이 포함될 수 있습니다. 다음 표에는 Red Hat이 현재 지원하는 모든 **cloud-init** 모듈과 기본 모듈 빈도가 나열되어 있습니다. 이러한 모듈에 대한 완전한 설명 및 옵션은 cloud-init 설명서의 모듈 섹션을 참조하십시오. https://cloudinit.readthedocs.io/en/latest/topics/modules.html#modules

표 3.2. 지원되는 cloud-init 모듈

cloud-init 모듈	설명	기본 모듈 빈도
bootcmd	부팅 프로세스 초기에 명령 실행	항상 당
ca_certs	CA 인증서 추가	인스턴스당
debug	디버깅을 지원하기 위해 내부 정보의 출력 활성화 또는 비활성화	인스턴스당
disable_ec2_metadata	AWS EC2 메타데이터 활성화 또는 비활성화	항상 당
disk_setup	간단한 파티션 테이블 및 파일 시스템 설정	인스턴스당
final_message	**cloud-init** 가 완료되면 출력 메시지를 지정합니다.	항상 당
foo	예, 모듈 구조(Modules는 아무 작업도 수행하지 않음)	인스턴스당
growpart	사용 가능한 디스크 공간을 채우기 위해 파티션의 크기 조정	항상 당

cloud-init 모듈	설명	기본 모듈 빈도
keys_to_console	콘솔에 쓸 수 있는 지문과 키를 제어할 수 있습니다.	인스턴스당
Eventory	기록 클라이언트 설치 및 구성	인스턴스당
로케일	시스템 로케일을 설정하고 시스템 전체에 적용합니다.	인스턴스당
m-02-ive	**mcollective**설치, 구성 및 시작	인스턴스당
Migrator	이전 버전의 **cloud-init** 를 최신 버전으로 이동	항상 당
mounts	마운트 지점 및 스왑 파일 구성	인스턴스당
phone_home	부팅 완료 후 원격 호스트에 데이터 게시	인스턴스당
power_state_change	모든 설정 모듈이 실행된 후 완전히 종료 및 재부팅	인스턴스당
Puppet	puppet 설치 및 구성	인스턴스당
resizefs	파티션에서 사용 가능한 모든 공간을 사용하도록 파일 시스템의 크기 조정	항상 당
resolv_conf	**resolv.conf**를 설정합니다.	인스턴스당
rh_subscription	Red Hat Enterprise Linux 시스템 등록	인스턴스당
rightscale_userdata	**cloud-init**에 rightScale 설정 후크 지원 추가	인스턴스당
rsyslog	**rsyslog**를 사용하여 원격 시스템 로깅 구성	인스턴스당
runcmd	임의의 명령 실행	인스턴스당
salt_minion	Salt **minion 설치, 구성 및**시작	인스턴스당
scripts_per_boot	부팅 스크립트당 실행	항상 당
scripts_per_instance	인스턴스 스크립트당 실행	인스턴스당

cloud-init 모듈	설명	기본 모듈 빈도
scripts_per_once	스크립트를 한 번 실행	한 번당
scripts_user	사용자 스크립트 실행	인스턴스당
scripts_vendor	벤더 스크립트 실행	인스턴스당
seed_random	임의의 초기 데이터 제공	인스턴스당
set_hostname	호스트 이름과 FQDN(정규화된 도메인 이름) 설정	항상 당
set_passwords	사용자 암호를 설정하고 SSH 암호 인증을 활성화하거나 비활성화합니다.	인스턴스당
ssh_authkey_fingerprints	사용자 SSH 키의 지문을 기록합니다.	인스턴스당
ssh_import_id	SSH 키 가져오기	인스턴스당
ssh	SSH 및 호스트 및 권한 있는 SSH 키 구성	인스턴스당
timezone	시스템 시간대 설정	인스턴스당
update_etc_hosts	업데이트 **/etc/hosts**	항상 당
update_hostname	호스트 이름 및 FQDN 업데이트	항상 당
users_groups	사용자 및 그룹 구성	인스턴스당
write_files	임의의 파일 쓰기	인스턴스당
yum_add_repo	시스템에 yum 리포지토리 설정 추가	항상 당

다음 표에는 Red Hat이 현재 지원하지 않는 모듈이 나열되어 있습니다.

표 3.3. 지원되지 않는 모듈

module
apt_configure
apt_pipeline

module
Byobu
chef
emit_upstart
grub_dpkg
ubuntu_init_switch

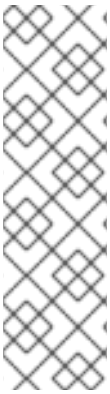
3.4. 기본 CLOUD.CFG 파일

/etc/cloud/cloud.cfg 파일에는 **cloud-init** 의 기본 구성으로 구성된 모듈이 나열됩니다.

파일의 모듈은 **cloud-init** 의 기본 모듈입니다. 환경에 대한 모듈을 구성하거나 필요하지 않은 모듈을 제거할 수 있습니다. **cloud.cfg** 에 포함된 모듈은 파일에 나열되므로 반드시 아무 것도 수행하지 않습니다. **cloud-init** 단계 중 하나에서 작업을 수행하려는 경우 개별적으로 구성해야 합니다.

cloud.cfg 파일은 개별 모듈 실행에 대한 chronology를 제공합니다. Red Hat이 추가 추가 모듈을 지원하는 한 **cloud.cfg** 에 모듈을 추가할 수 있습니다.

RHEL(Red Hat Enterprise Linux)에 대한 파일의 기본 내용은 다음과 같습니다.



참고

- 모듈은 **cloud.cfg** 에서 지정한 순서대로 실행됩니다. 일반적으로 이 명령은 변경하지 않습니다.
- **cloud.cfg** 지시문은 사용자 데이터로 재정의할 수 있습니다.
- **cloud-init** 를 수동으로 실행하는 경우 명령줄 옵션으로 **cloud.cfg** 를 덮어쓸 수 있습니다.
- 각 모듈에는 특정 정보를 추가할 수 있는 자체 구성 옵션이 포함되어 있습니다.

```
users: 1
 - default

disable_root: 1 2
ssh_pwauth:   0 3

mount_default_fields: [~, ~, 'auto', 'defaults,nofail,x-systemd.requires=cloud-init.service', '0', '2'] 4
ssh_deletekeys:   1 5
ssh_genkeytypes: ['rsa', 'ecdsa', 'ed25519'] 6
syslog_fix_perms: ~ 7
disable_vmware_customization: false 8

cloud_init_modules: 9
```

```
 - disk_setup
 - migrator
 - bootcmd
 - write-files
 - growpart
 - resizefs
 - set_hostname
 - update_hostname
 - update_etc_hosts
 - rsyslog
 - users-groups
 - ssh

cloud_config_modules: 10
 - mounts
 - locale
 - set-passwords
 - rh_subscription
 - yum-add-repo
 - package-update-upgrade-install
 - timezone
 - puppet
 - chef
 - salt-minion
 - mcollective
 - disable-ec2-metadata
 - runcmd

cloud_final_modules: 11
 - rightscale_userdata
 - scripts-per-once
 - scripts-per-boot
 - scripts-per-instance
 - scripts-user
 - ssh-authkey-fingerprints
 - keys-to-console
 - phone-home
 - final-message
 - power-state-change

system_info:
  default_user: 12
    name: cloud-user
    lock_passwd: true
    gecos: Cloud User
    groups: [adm, systemd-journal]
    sudo: ["ALL=(ALL) NOPASSWD:ALL"]
    shell: /bin/bash
  distro: rhel 13
  paths:
    cloud_dir: /var/lib/cloud 14
    templates_dir: /etc/cloud/templates 15
  ssh_svcname: sshd 16

# vim:syntax=yaml
```

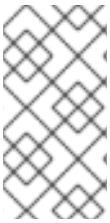
1. 시스템의 기본 사용자를 지정합니다. 자세한 내용은 사용자 및 그룹을 참조하십시오.
2. root 로그인을 활성화하거나 비활성화합니다. 자세한 내용은 인증 키 를 참조하십시오.
3. **ssh** 가 암호 인증을 수락하도록 구성되었는지 여부를 지정합니다. 자세한 내용은 암호 설정을 참조하십시오.
4. 마운트 지점을 설정합니다. 6개의 값이 포함된 목록이어야 합니다. 자세한 내용은 마운트 를 참조하십시오.
5. 기본 호스트 SSH 키를 제거할지 여부를 지정합니다. 자세한 내용은 호스트 키 를 참조하십시오.
6. 생성할 키 유형을 지정합니다. 자세한 내용은 호스트 키 를 참조하십시오. RHEL 8.4 및 이전 버전의 경우 이 행의 기본값은 ~ 입니다.
7. **cloud-init** 는 여러 부팅 단계에서 실행됩니다. **cloud-init** 가 모든 단계를 로그 파일에 기록할 수 있도록 이 옵션을 설정합니다. **usr/share/doc/cloud-init/examples** 디렉터리의 **cloud-config.txt** 파일에서 이 옵션에 대한 자세한 정보를 찾습니다.
8. VMware vSphere 사용자 정의 활성화 또는 비활성화
9. 이 섹션의 모듈은 부팅 프로세스 초기에 **cloud-init** 서비스가 시작될 때 실행되는 서비스입니다.
10. 이러한 모듈은 초기 부팅 후 **cloud-init** 구성 중에 실행됩니다.
11. 이러한 모듈은 구성이 완료된 후 **cloud-init** 의 최종 단계에서 실행됩니다.
12. 기본 사용자에 대한 세부 정보를 지정합니다. 자세한 내용은 사용자 및 그룹을 참조하십시오.
13. 배포를 지정합니다.
14. **cloud-init**- 특정 하위 디렉터리가 포함된 기본 디렉터리를 지정합니다. 자세한 내용은 디렉터리 레이아웃 을 참조하십시오.
15. 템플릿이 있는 위치를 지정합니다.
16. SSH 서비스의 이름

추가 리소스

- 구성 파일은 어디에 있습니까?
- 모듈

3.5. CLOUD.CFG.D 디렉토리

cloud-init 는 사용자가 제공하고 구성하는 지시문에 따라 작동합니다. 일반적으로 이러한 지시문은 **cloud.cfg.d** 디렉터리에 포함됩니다.



참고

cloud.cfg 파일에 사용자 data 지시문을 추가하여 모듈을 구성할 수 있지만, **cloud.cfg** 파일을 그대로 두는 것이 좋습니다. 지시문을 **/etc/cloud/cloud.cfg.d** 디렉토리에 추가하십시오. 이 디렉토리에 지시문을 추가하면 향후 수정 및 업그레이드를 더 쉽게 수행할 수 있습니다.

지시문을 추가하는 방법은 여러 가지가 있습니다. **#cloud-config** 제목을 포함하는 ***.cfg** 이라는 파일에 지시문을 포함할 수 있습니다. 일반적으로 디렉터리에는 여러 ***cfg** 파일이 포함됩니다. 지시문을 추가하는 다른 옵션도 있습니다(예: 사용자 데이터 스크립트를 추가할 수 있습니다. 자세한 내용은 User-Data Formats 를 참조하십시오.

추가 리소스

- 구성 파일은 어디에 있습니까?
- 클라우드 구성 예

3.6. 기본 05_LOGGING.CFG 파일

05_logging.cfg 파일은 **cloud-init** 에 대한 로깅 정보를 설정합니다. **/etc/cloud/cloud.cfg.d** 디렉터리에는 추가하는 다른 **cloud-init** 지시문과 함께 이 파일이 포함되어 있습니다.

cloud-init 는 기본적으로 **05_logging.cfg** 에서 로깅 구성을 사용합니다. RHEL(Red Hat Enterprise Linux)에 대한 파일의 기본 내용은 다음과 같습니다.

```
## This yaml formatted config file handles setting
## logger information.  The values that are necessary to be set
## are seen at the bottom.  The top '_log' are only used to remove
## redundancy in a syslog and fallback-to-file case.
##
## The 'log_cfgs' entry defines a list of logger configs
## Each entry in the list is tried, and the first one that
## works is used.  If a log_cfg list entry is an array, it will
## be joined with '\n'.
_log:
 - &log_base |
   [loggers]
   keys=root,cloudinit

   [handlers]
   keys=consoleHandler,cloudLogHandler

   [formatters]
   keys=simpleFormatter,arg0Formatter

   [logger_root]
   level=DEBUG
   handlers=consoleHandler,cloudLogHandler

   [logger_cloudinit]
   level=DEBUG
   qualname=cloudinit
   handlers=
   propagate=1

   [handler_consoleHandler]
   class=StreamHandler
   level=WARNING
   formatter=arg0Formatter
   args=(sys.stderr,)
```

```
    [formatter_arg0Formatter]
    format=%(asctime)s - %(filename)s[%(levelname)s]: %(message)s

    [formatter_simpleFormatter]
    format=[CLOUDINIT] %(filename)s[%(levelname)s]: %(message)s
  - &log_file |
    [handler_cloudLogHandler]
    class=FileHandler
    level=DEBUG
    formatter=arg0Formatter
    args=('/var/log/cloud-init.log',)
  - &log_syslog |
    [handler_cloudLogHandler]
    class=handlers.SysLogHandler
    level=DEBUG
    formatter=simpleFormatter
    args=("/dev/log", handlers.SysLogHandler.LOG_USER)

log_cfgs:
# Array entries in this list will be joined into a string
# that defines the configuration.
#
# If you want logs to go to syslog, uncomment the following line.
# - [ *log_base, *log_syslog ]
#
# The default behavior is to just log to a file.
# This mechanism that does not depend on a system service to operate.
 - [ *log_base, *log_file ]
# A file path can also be used.
# - /etc/log.conf

# This tells cloud-init to redirect its stdout and stderr to
# 'tee -a /var/log/cloud-init-output.log' so the user can see output
# there without needing to look on the console.
output: {all: '| tee -a /var/log/cloud-init-output.log'}
```

추가 리소스

- 로깅

3.7. CLOUD-INIT /VAR/LIB/CLOUD 디렉터리 레이아웃

cloud-init 를 처음 실행하면 인스턴스 및 **cloud-init** 구성에 대한 정보가 포함된 디렉터리 레이아웃이 생성됩니다.

디렉터리에는 **/scripts/vendor** 와 같은 선택적 디렉터리가 포함될 수 있습니다.

다음은 **cloud-init** 의 샘플 디렉터리 레이아웃입니다.

```
/var/lib/cloud/
    - data/
      - instance-id
      - previous-instance-id
      - previous-datasource
      - previous-hostname
```

```
    - result.json
    - set-hostname
    - status.json
  - handlers/
  - instance
    - boot-finished
    - cloud-config.txt
    - datasource
    - handlers/
    - obj.pkl
    - scripts/
    - sem/
    - user-data.txt
    - user-data.txt.i
    - vendor-data.txt
    - vendor-data.txt.i
  - instances/
     f111ee00-0a4a-4eea-9c17-3fa164739c55/
       - boot-finished
       - cloud-config.txt
       - datasource
       - handlers/
       - obj.pkl
       - scripts/
       - sem/
       - user-data.txt
       - user-data.txt.i
       - vendor-data.txt
       - vendor-data.txt.i
  - scripts/
    - per-boot/
    - per-instance/
    - per-once/
    - vendor/
  - seed/
  - sem/
    - config_scripts_per_once.once
```

추가 리소스

- 디렉터리 레이아웃

4장. CLOUD-INIT 구성

이 장에는 **cloud-init** 에 대한 가장 일반적인 구성 작업의 예가 포함되어 있습니다.

cloud-init 구성에 cloud.cfg 파일 및 **cloud.cfg**. **d** 디렉터리에 지시문을 추가해야 할 수 있습니다. 또는 특정 데이터 원본에 사용자 데이터 파일 및 메타데이터 파일과 같은 파일에 지시문을 추가해야 할 수 있습니다. 데이터 원본에 지시문을 HTTP 서버에 업로드해야 할 수 있습니다. 데이터 소스의 요구 사항을 확인하고 그에 따라 지시문을 추가합니다.

4.1. NOCLOUD 데이터 소스에 CLOUD-INIT를 포함하는 가상 머신 생성

cloud-init 가 포함된 새 VM(가상 머신)을 생성하려면 다음 절차를 참조하십시오. 이 절차에서는 **meta-data** 및 **user-data** 파일을 생성합니다.

- **meta-data** 파일에는 인스턴스 세부 정보가 포함되어 있습니다.
- **user-data** 파일에는 사용자를 생성하고 액세스 권한을 부여하는 정보가 포함되어 있습니다.

그런 다음 이러한 파일을 새 ISO 이미지에 포함하고 KVM 게스트 이미지에서 생성한 새 VM에 ISO 파일을 연결합니다. 이 시나리오에서는 데이터 소스가 NoCloud입니다.

절차

1. **cloudinitiso** 라는 디렉터리를 만들고 이 디렉터리로 이동합니다.

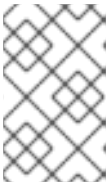
```
$ mkdir cloudinitiso
$ cd cloudinitiso
```

2. **meta-data** 라는 파일을 만듭니다. 파일에 다음 정보를 추가합니다.

```
instance-id: citest
local-hostname: citest-1
```

3. **user-data** 라는 파일을 만듭니다. 파일에 다음 정보를 포함합니다.

```
#cloud-config
password: cilogon
chpasswd: {expire: False}
ssh_pwauth: True
ssh_authorized_keys:
  - ssh-rsa AAA...fhHQ== sample@redhat.com
```



참고

user-data 파일의 최종 행은 SSH 공개 키를 참조합니다. **~/.ssh/id_rsa.pub** 에서 SSH 공개 키를 찾습니다. 이 샘플 프로시저를 시도할 때 공개 키 중 하나를 포함하도록 행을 수정합니다.

4. **genisoimage** 명령을 사용하여 **user-data** 및 **meta-data** 가 포함된 ISO 이미지를 생성합니다.

```
# genisoimage -output ciiso.iso -volid cidata -joliet -rock user-data meta-data

I: -input-charset not specified, using utf-8 (detected in locale settings)
```

```
Total translation table size: 0
Total rockridge attributes bytes: 331
Total directory bytes: 0
Path table size(bytes): 10
Max brk space used 0
183 extents written (0 MB)
```

5. Red Hat Customer Portal에서 **/var/lib/libvirt/images** 디렉토리로 KVM 게스트 이미지를 다운로드합니다.

6. **virt-install** 명령을 사용하여 KVM 게스트 이미지에서 새 VM을 생성합니다. 생성한 ISO 이미지를 이미지에 대한 첨부 파일로 포함합니다.

```
virt-install \
   --memory 4096 \
   --vcpus 4 \
   --name mytestcivm \
   --disk /var/lib/libvirt/images/rhel-8.1-x86_64-
kvm.qcow2,device=disk,bus=virtio,format=qcow2 \
   --disk /home/sample/cloudinitiso/ciiso.iso,device=cdrom \
   --os-type Linux \
   --os-variant rhel8.0 \
   --virt-type kvm \
   --graphics none \
   --import
```

7. **cloud-user** 로 이미지에 로그인합니다. 비밀번호는 **cilogon** 입니다.

```
citest-1 login: cloud-user
Password:
[cloud-user@citest-1 ~]$
```

검증

- **cloud-init** 상태를 확인하여 작업이 완료되었는지 확인합니다.

```
[cloud-user@citest-1 instance]$ cloud-init status
status: done
```

- **cloud-init** 는 실행 시 **/var/lib/cloud** 에 cloud-init 디렉터리 레이아웃을 생성하고 지정된 지시문에 따라 특정 디렉터리 콘텐츠를 업데이트하거나 변경합니다.
예를 들어 데이터 소스 파일을 확인하여 데이터 소스가 **NoCloud** 인지 확인할 수 있습니다.

```
$ cd /var/lib/cloud/instance
$ cat datasource
DataSourceNoCloud: DataSourceNoCloud [seed=/dev/sr0][dsmode=net]
```

cloud-init copies user-data into **/var/lib/cloud/instance/user-data.txt**.

```
$cat user-data.txt
#cloud-config
password: cilogon
chpasswd: {expire: False}
```

```
ssh_pwauth: True
ssh_authorized_keys:
  - ssh-rsa AAA...fhHQ== sample@redhat.com
```

다음은 샘플입니다. **cloud-init** 디렉터리 레이아웃에는 훨씬 더 많은 정보가 포함되어 있습니다.

참고

OpenStack의 경우 인스턴스 생성 및 관리에 는 **cloud-init** 를 사용하여 인스턴스를 구성하기 위한 정보가 포함되어 있습니다. 특정 프로시저 에 대한 사용자 지정 인스턴스 생성을 참조하십시오.

추가 리소스

- NoCloud 데이터 소스에 대한 업스트림 문서

4.2. CLOUD-INIT를 사용하여 클라우드 사용자 암호 만료

cloud-user 가 처음 로그인할 때 **cloud-user** 암호를 변경하도록 할 수 있습니다. 다음 절차에 따라 암호를 만료하십시오.

절차

1. 데이터 소스의 요구 사항에 따라 편집하기 위해 user-data 파일을 열거나, 그렇지 않으면 **cloud.cfg.d** 디렉터리에 다음 지시문을 추가합니다.

참고

모든 사용자 지시문에는 파일 상단에 **#cloud-config** 가 포함되어 있어 **cloud-init** 가 파일을 사용자 지시문이 포함된 것으로 인식합니다. **cloud.cfg.d** 디렉터리에 지시문을 포함하는 경우 파일 ***.cfg** 이름을 지정하고 항상 파일 맨 위에 **#cloud-config** 를 포함합니다.

2. **chpasswd: {expire: False}** 행을 **chpasswd: {expire: True}** 로 변경합니다.

```
#cloud-config
password: mypassword
chpasswd: {expire: True}
ssh_pwauth: True
ssh_authorized_keys:
  - ssh-rsa AAA...SDvz user1@yourdomain.com
  - ssh-rsa AAB...QTuo user2@yourdomain.com
```

암호를 지정하지 않으면 암호가 만료되고 **chpasswd** 가 기본 사용자에서 작동하기 때문입니다.

참고

이는 글로벌 설정입니다. chpasswd 를 True 로 설정하면 생성한 모든 사용자가 로그인할 때 암호를 변경해야 합니다.

4.3. CLOUD-INIT로 기본 사용자 이름 변경

기본 사용자 이름을 **cloud-user** 이외의 다른 이름으로 변경할 수 있습니다.

절차

1. 데이터 소스의 요구 사항에 따라 편집하기 위해 **user-data** 파일을 열거나, 그렇지 않으면 **cloud.cfg.d** 디렉터리에 다음 지시문을 추가합니다.

참고

모든 사용자 지시문에는 파일 상단에 **#cloud-config** 가 포함되어 있어 **cloud-init** 가 파일을 사용자 지시문이 포함된 것으로 인식합니다. **cloud.cfg.d** 디렉터리에 지시문을 포함하는 경우 파일 ***.cfg** 이름을 지정하고 항상 파일 맨 위에 **#cloud-config** 를 포함합니다.

2. **user: <username >** 행을 추가하고 **<username>**을 새 기본 사용자 이름으로 바꿉니다.

```
#cloud-config
user: username
password: mypassword
chpasswd: {expire: False}
ssh_pwauth: True
ssh_authorized_keys:
  - ssh-rsa AAA...SDvz user1@yourdomain.com
  - ssh-rsa AAB...QTuo user2@yourdomain.com
```

4.4. CLOUD-INIT를 사용하여 루트 암호 설정

루트 암호를 설정하려면 사용자 목록을 만듭니다.

절차

1. 데이터 소스의 요구 사항에 따라 편집하기 위해 **user-data** 파일을 열거나, 그렇지 않으면 **cloud.cfg.d** 디렉터리에 다음 지시문을 추가합니다.

참고

모든 사용자 지시문에는 파일 상단에 **#cloud-config** 가 포함되어 있어 **cloud-init** 가 파일을 사용자 지시문이 포함된 것으로 인식합니다. **cloud.cfg.d** 디렉터리에 지시문을 포함하는 경우 파일 ***.cfg** 이름을 지정하고 항상 파일 맨 위에 **#cloud-config** 를 포함합니다.

2.

파일의 **chpasswd** 섹션에서 사용자 목록을 만듭니다. 형식은 다음 샘플에 표시됩니다.

참고

공백은 매우 중요합니다. 사용자 목록의 콜론 앞이나 뒤의 공백을 포함하지 마십시오. 공백을 포함하는 경우 암호가 공백으로 설정됩니다.

```
#cloud-config
ssh_pwauth: True
ssh_authorized_keys:
 - ssh-rsa AAA...SDvz user1@yourdomain.com
 - ssh-rsa AAB...QTuo user2@yourdomain.com
chpasswd:
 list: |
   root:myrootpassword
   cloud-user:mypassword
 expire: False
```

참고

이 방법을 사용하여 사용자 암호를 설정하는 경우 이 섹션의 모든 암호를 설정해야 합니다.

4.5. CLOUD-INIT를 사용하여 RED HAT 서브스크립션 관리

rh_subscription 지시문을 사용하여 시스템을 등록할 수 있습니다. 샘플은 다음과 같습니다. 서브스크립션마다 사용자 데이터를 편집할 수 있습니다.

절차

다음 예제에서는 자동 연결 및 서비스 수준 옵션을 사용합니다.

- **rh_subscription** 에서 사용자 이름과 비밀번호 를 추가하고 자동 연결을 **True** 로 설정하고,

서비스 수준을 자체 지원으로 설정합니다.

```
rh_subscription:
  username: sample@redhat.com
  password: 'mypassword'
  auto-attach: True
  service-level: self-support
```

참고

서비스 수준 옵션을 사용하려면 **auto-attach** 옵션을 사용해야 합니다.

다음 예제에서는 **activation-key** 및 **org** 옵션을 사용합니다.

- **rh_subscription** 아래에서 활성화 키 및 조직 번호를 추가하고 자동 연결을 **True** 로 설정합니다.

```
rh_subscription:
  activation-key: example_key
  org: 12345
  auto-attach: True
```

다음 예제에서는 서브스크립션 풀을 추가합니다.

- **rh_subscription** 에서 사용자 이름 , 암호, 풀 번호를 추가합니다.

```
rh_subscription:
  username: sample@redhat.com
  password: 'password'
  add-pool: XYZ01234567
```

참고

이 샘플은 **subscription-manager attach --pool=XYZ01234567** 명령과 동일합니다.

다음 예제에서는 **/etc/rhsm/rhsm.conf** 파일에 서버 호스트 이름을 설정합니다.

- rh_subscription 에서 사용자 이름 , 암호,server-hostname 을 추가하고 자동 연결을 True 로 설정합니다.

```
rh_subscription:
  username: sample@redhat.com
  password: 'password'
  server-hostname: test.example.com
  auto-attach: True
```

4.6. CLOUD-INIT를 사용하여 사용자 및 사용자 옵션 추가

사용자 섹션에서 사용자를 생성하고 설명합니다. 섹션을 수정하여 초기 시스템 구성에 사용자를 더 추가하고 추가 사용자 옵션을 설정할 수 있습니다.

users 섹션을 추가하는 경우 이 섹션에서 기본 사용자 옵션도 설정해야 합니다.

절차

1. 데이터 소스의 요구 사항에 따라 편집하기 위해 user-data 파일을 열거나, 그렇지 않으면 cloud.cfg.d 디렉터리에 다음 지시문을 추가합니다.

참고

모든 사용자 지시문에는 파일 상단에 #cloud-config 가 포함되어 있어 cloud-init 가 파일을 사용자 지시문이 포함된 것으로 인식합니다. cloud.cfg.d 디렉터리에 지시문을 포함하는 경우 파일 *.cfg 이름을 지정하고 항상 파일 맨 위에 #cloud-config 를 포함합니다.

2. users 섹션을 추가하여 사용자를 추가합니다.

- cloud-user 를 사용자가 지정한 다른 사용자와 함께 생성한 기본 사용자로 하려면 섹션의 첫 번째 항목으로 default 를 추가해야 합니다. 첫 번째 항목이 아니면 cloud-user 가 생성되지 않습니다.

- 기본적으로 사용자는 selinux-user 값이 없는 경우 ProfileBundle _u 로 레이블이 지정됩니다.

```
#cloud-config
```

```
users:
  - default
  - name: user2
    gecos: User N. Ame
    selinux-user: staff_u
    groups: users,wheel
    ssh_pwauth: True
    ssh_authorized_keys:
      - ssh-rsa AA..vz user@domain.com
chpasswd:
  list: |
    root:password
    cloud-user:mypassword
    user2:mypassword2
  expire: False
```

참고

- 예에서는 **user2** 사용자를 두 개의 그룹인 **user** 및 **wheel** 에 배치합니다.

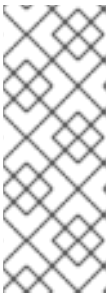
4.7. CLOUD-INIT를 사용하여 첫 번째 부팅 명령 실행

runcmd 및 **bootcmd** 섹션을 사용하여 시작 및 초기화 중 명령을 실행할 수 있습니다.

bootcmd 섹션은 초기화 프로세스 초기에 실행되며 기본적으로 모든 부팅 시 실행됩니다. **runcmd** 섹션은 프로세스 종료 근처에서 실행되며 첫 번째 부팅 및 초기화 중에만 실행됩니다.

절차

1. 데이터 소스의 요구 사항에 따라 편집하기 위해 **user-data** 파일을 열거나, 그렇지 않으면 **cloud.cfg.d** 디렉터리에 다음 지시문을 추가합니다.



참고

모든 사용자 지시문에는 파일 상단에 **#cloud-config** 가 포함되어 있어 **cloud-init** 가 파일을 사용자 지시문이 포함된 것으로 인식합니다. **cloud.cfg.d** 디렉터리에 지시문을 포함하는 경우 파일 ***.cfg** 이름을 지정하고 항상 파일 맨 위에 **#cloud-config** 를 포함합니다.

2. **bootcmd** 및 **runcmd** 에 대한 섹션을 추가합니다. **cloud-init** 를 실행할 명령을 포함합니다.

```
#cloud-config
users:
  - default
  - name: user2
    gecos: User N. Ame
    groups: users
chpasswd:
  list: |
    root:password
    fedora:myfedpassword
    user2:mypassword2
  expire: False
bootcmd:
 - echo New MOTD >> /etc/motd
runcmd:
 - echo New MOTD2 >> /etc/motd
```

4.8. CLOUD-INIT를 사용하여 SUDOERS 추가

users 섹션에 **sudo** 및 **groups** 항목을 추가하여 사용자를 **sudo er**로 구성할 수 있습니다.

절차

1. 데이터 소스의 요구 사항에 따라 편집하기 위해 **user-data** 파일을 열거나, 그렇지 않으면 **cloud.cfg.d** 디렉터리에 다음 지시문을 추가합니다.

참고

모든 사용자 지시문에는 파일 상단에 **#cloud-config** 가 포함되어 있어 **cloud-init** 가 파일을 사용자 지시문이 포함된 것으로 인식합니다. **cloud.cfg.d** 디렉터리에 지시문을 포함하는 경우 파일 ***.cfg** 이름을 지정하고 항상 파일 맨 위에 **#cloud-config** 를 포함합니다.

2. **sudo** 항목을 추가하고 사용자 액세스 권한을 지정합니다. 예를 들어 **sudo: ALL=(ALL) NOPASSWD:ALL** 을 사용하면 사용자가 무제한으로 액세스할 수 있습니다.

3. **groups** 항목을 추가하고 사용자를 포함하는 그룹을 지정합니다.

```
#cloud-config
users:
  - default
  - name: user2
    gecos: User D. Two
    sudo: ["ALL=(ALL) NOPASSWD:ALL"]
```

```
    groups: wheel,adm,systemd-journal
    ssh_pwauth: True
    ssh_authorized_keys:
      - ssh-rsa AA...vz user@domain.com
chpasswd:
  list: |
    root:password
    cloud-user:mypassword
    user2:mypassword2
  expire: False
```

4.9. CLOUD-INIT를 사용하여 정적 네트워킹 구성 설정

metadata에 **network-interfaces** 섹션을 추가하여 **cloud-init** 로 네트워크 구성을 설정할 수 있습니다.

Red Hat Enterprise Linux는 **NetworkManager** 를 통해 기본 네트워킹 서비스를 제공합니다. 이는 사용 가능한 경우 네트워크 장치 및 연결을 켜고 활성 상태로 유지하는 동적 네트워크 제어 및 구성 데몬입니다.

데이터 소스에서 네트워크 구성을 제공할 수 있습니다. 자세한 내용은 **cloud-init** 설명서 섹션을 참조하십시오. **https://cloudinit.readthedocs.io/en/latest/topics/network-config.html#network-configuration-sources**

cloud-init 에 대한 네트워크 구성을 지정하지 않고 네트워크 구성을 비활성화하지 않은 경우 **cloud-init** 에서 연결된 장치에 연결이 있는지 확인합니다. 연결된 장치를 찾으면 인터페이스에서 **DHCP** 요청을 실행하는 네트워크 구성이 생성됩니다. 자세한 내용은 **cloud-init** 설명서 섹션을 참조하십시오.

절차

다음 예제에서는 정적 네트워킹 구성을 추가합니다.

1. 데이터 소스의 요구 사항에 따라 편집하기 위해 **user-data** 파일을 열거나, 그렇지 않으면 **cloud.cfg.d** 디렉터리에 다음 지시문을 추가합니다.

참고

모든 사용자 지시문에는 파일 상단에 **#cloud-config** 가 포함되어 있어 **cloud-init** 가 파일을 사용자 지시문이 포함된 것으로 인식합니다. **cloud.cfg.d** 디렉터리에 지시문을 포함하는 경우 파일 ***.cfg** 이름을 지정하고 항상 파일 맨 위에 **#cloud-config** 를 포함합니다.

2. **network-interfaces** 섹션을 추가합니다.

```
network:
  version: 1
  config:
    - type: physical
      name: eth0
      subnets:
        - type: static
          address: 192.168.1.10/24
          gateway: 192.168.1.254
```

참고

메타데이터에 다음 정보를 추가하여 네트워크 구성을 비활성화할 수 있습니다.

```
network
  config: disabled
```

추가 리소스

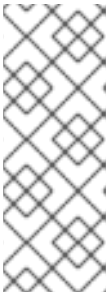
- 네트워크 설정
- NoCloud

4.10. CLOUD-INIT를 사용하여 ROOT 사용자만 구성

root 사용자와 다른 사용자가 없도록 사용자 데이터를 구성할 수 있습니다.

절차

1. 데이터 소스의 요구 사항에 따라 편집하기 위해 **user-data** 파일을 열거나, 그렇지 않으면 **cloud.cfg.d** 디렉터리에 다음 지시문을 추가합니다.



참고

모든 사용자 지시문에는 파일 상단에 **#cloud-config** 가 포함되어 있어 **cloud-init** 가 파일을 사용자 지시문이 포함된 것으로 인식합니다. **cloud.cfg.d** 디렉터리에 지시문을 포함하는 경우 파일 ***.cfg** 이름을 지정하고 항상 파일 맨 위에 **#cloud-config** 를 포함합니다.

2. **users** 섹션에서 사용자 **root** 에 대한 항목을 만듭니다.

다음의 간단한 예제에는 **name** 옵션만 있는 **users** 섹션이 포함되어 있습니다.

```
users:
 - name: root
chpasswd:
 list: |
   root:password
 expire: False
```

3. 선택적으로 **root** 사용자의 **SSH** 키를 설정합니다.

```
users:
 - name: root
   ssh_pwauth: True
   ssh_authorized_keys:
    - ssh-rsa AA..vz user@domain.com
```

4.11. CLOUD-INIT에서 CONTAINER-STORAGE-SETUP으로 스토리지 설정

write_files 모듈 내에서 **container-storage-setup** 유틸리티를 참조하여 스토리지를 설정할 수 있습니다.

절차

1. 데이터 소스의 요구 사항에 따라 편집하기 위해 **user-data** 파일을 열거나, 그렇지 않으면 **cloud.cfg.d** 디렉터리에 다음 지시문을 추가합니다.

참고

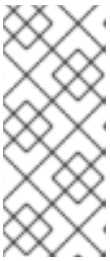
모든 사용자 지시문에는 파일 상단에 **#cloud-config** 가 포함되어 있어 **cloud-init** 가 파일을 사용자 지시문이 포함된 것으로 인식합니다. **cloud.cfg.d** 디렉터리에 지시문을 포함하는 경우 파일 ***.cfg** 이름을 지정하고 항상 파일 맨 위에 **#cloud-config** 를 포함합니다.

2.

container-storage-setup 유틸리티의 경로를 포함하도록 **write_files** 모듈을 추가하거나 수정합니다.

다음 예제에서는 루트 논리 볼륨의 크기를 기본 **3GB**가 아닌 **6GB**로 설정합니다.

```
write_files:
 - path: /etc/sysconfig/docker-storage-setup
   permissions: 0644
   owner: root
   content: |
   ROOT_SIZE=6G
```



참고

RHEL 7.4 이전에는 **container-storage-setup** 이 **docker-storage-setup** 이었습니다. 스토리지에 **OverlayFS**를 사용하는 경우 이제 **RHEL 7.4**부터 강제 모드에서 **SELinux**와 함께 해당 유형의 파일 시스템을 사용할 수 있습니다.

4.12. CLOUD-INIT로 시스템 로케일 변경

locale 모듈을 사용하여 시스템 위치를 구성할 수 있습니다.

절차

1.

데이터 소스의 요구 사항에 따라 편집할 메타 데이터 파일을 열거나, 그렇지 않으면 **cloud.cfg** 파일 또는 **cloud.cfg.d** 디렉터리에 다음 지시문을 추가합니다.

2.

위치를 지정하여 **locale** 지시문을 추가합니다. 다음 샘플은 **UTF-8** 인코딩을 사용하여 로케일을 **ja_JP (Japan)**로 설정합니다.

```
#cloud-config
locale: ja_JP.UTF-8
```

추가 리소스

- 로케일

4.13. CLOUD-INIT 및 쉘 스크립트

목록 값 또는 문자열 값을 **bootcmd** 에 추가하거나**cmd**를 실행할 수 있습니다. 사용자 데이터 내에 쉘 스크립트를 제공할 수도 있습니다.

- **bootcmd** 에 목록 값을 사용하거나**cmd**를 실행하는 경우 각 목록 항목이 **execve** 를 사용하여 차례로 실행됩니다.
- 문자열 값을 사용하는 경우 전체 문자열이 쉘 스크립트로 실행됩니다.
- **cloud-init** 를 사용하여 쉘 스크립트를 실행하려면 **cloud-init** 를 **.yaml** 파일로 제공하는 대신 쉘 스크립트(**#!**)를 제공할 수 있습니다.

bootcmd 에 쉘 스크립트를 배치하고**cmd**를 실행하는 방법에 대한 예는 첫 번째 부팅 시 명령 실행을 참조하십시오.

4.14. CLOUD-INIT에서 구성 파일 업데이트 방지

백업 이미지에서 인스턴스를 생성하거나 복원하면 인스턴스 **ID**가 변경됩니다. 인스턴스 **ID**가 변경되면 **cloud-init** 가 구성 파일을 업데이트할 수 있습니다.

다음 절차에 따라 **cloud-init** 가 백업에서 생성하거나 복원할 때 특정 구성 파일을 업데이트하지 않도록 합니다.

절차

1. 편집을 위해 **/etc/cloud/cloud.cfg** 파일을 엽니다.
2. **cloud-init** 가 인스턴스를 복원할 때 업데이트하지 않도록 구성을 주석 처리하거나 제거합니다.

예를 들어 **SSH** 키 파일을 업데이트하지 않으려면 **cloud_init_modules** 섹션에서 **-ssh** 를 제거합니다.

```
cloud_init_modules:
 - disk_setup
 - migrator
 - bootcmd
 - write-files
 - growpart
 - resizefs
 - set_hostname
 - update_hostname
 - update_etc_hosts
 - rsyslog
 - users-groups
 # - ssh
```

검증

cloud-init 가 업데이트된 구성 파일을 확인할 수 있습니다. 이렇게 하려면 **/var/log/cloud/cloud-init.log** 파일을 검사합니다. 업데이트된 파일은 인스턴스 시작 중에 쓰기로 시작하는 메시지로 기록됩니다. 예를 들어 다음과 같습니다.

```
2019-09-03 00:16:07,XXX - util.py[DEBUG]: Writing to /root/.ssh/authorized_keys - wb: [XXX] 554
bytes
2019-09-03 00:16:08,XXX - util.py[DEBUG]: Writing to /etc/ssh/sshd_config - wb: [XXX] 3905 bytes
```

4.15. CLOUD-INIT를 실행한 후 KVM 게스트 이미지에서 생성된 VM 수정

cloud-init 를 다시 실행하기 전에 **cloud-init** 구성을 수정하려면 다음 절차를 사용하십시오. 설치 및 활성화된 **cloud-init** 패키지가 포함된 **VM**을 시작하면 **cloud-init** 가 해당 **VM**의 초기 부팅 시 기본 상태로 실행됩니다.

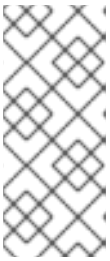
절차

1. **VM**에 로그인합니다.

2. 예를 들어 **/etc/cloud** 디렉토리에서 **cloud.cfg** 파일을 추가하거나 지시문을 **/etc/cloud/cloud.cfg.d** 디렉토리에 추가하십시오.

3. **cloud-init** 가 다시 실행할 수 있도록 **cloud-init clean** 명령을 실행하여 디렉터리를 정리합니다. 다음 명령을 **root**로 실행하여 **VM**을 정리할 수도 있습니다.

```
`rm -Rf /var/lib/cloud/instances/*`
`rm -Rf /var/lib/cloud/instance`
`rm -Rf /var/lib/cloud/data/*`
```



참고

정리된 이미지를 새 이미지로 저장하고 해당 이미지를 여러 VM에 사용할 수 있습니다. 새 VM은 업데이트된 cloud-init 구성을 사용하여 cloud-init 를 실행합니다.

4. cloud-init 를 재실행하거나 VM을 재부팅합니다.

cloud-init 를 다시 실행하여 구성 변경 사항을 구현합니다.

4.16. CLOUD-INIT가 실행된 후 특정 데이터 소스의 VM 수정

cloud-init 를 다시 실행하기 전에 cloud-init 구성을 수정하려면 다음 절차를 참조하십시오. 이 절차에서는 OpenStack을 예제로 사용합니다. 데이터 소스에 따라 수행해야 하는 정확한 단계는 다를 수 있습니다.

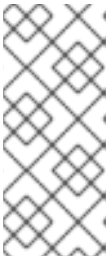
절차

1. OpenStack Platform 인스턴스를 생성하고 시작합니다. OpenStack용 인스턴스 생성에 대한 자세한 내용은 인스턴스 생성을 참조하십시오. 이 예제에서 가상 시스템에는 가상 시스템을 부팅할 때 실행되는 cloud-init 가 포함되어 있습니다.

2. 지시문을 추가하거나 변경합니다. 예를 들어 OpenStack HTTP 서버에 저장된 user-data.file 파일을 수정합니다.

3. 가상 머신을 정리합니다. root로 다음 명령을 실행합니다.

```
`rm -rf /etc/resolv.conf /run/cloud-init`
`userdel -rf cloud-user`
`hostnamectl set-hostname localhost.localdomain`
`rm /etc/NetworkManager/conf.d/99-cloud-init.conf`
```



참고

정리된 이미지를 새 이미지로 저장하고 여러 가상 머신에 해당 이미지를 사용할 수 있습니다. 새 가상 시스템은 업데이트된 **cloud-init** 구성을 사용하여 **cloud-init** 를 실행합니다.

4. **cloud-init** 를 재실행하거나 가상 시스템을 재부팅합니다.

cloud-init를 다시 실행하여 구성 변경 사항을 구현합니다.

4.17. CLOUD-INIT 문제 해결

cloud-init 를 실행한 후 구성 및 로그 파일을 검사하여 인스턴스의 문제를 해결할 수 있습니다. 문제를 확인한 후 인스턴스에서 **cloud-init** 를 다시 실행할 수 있습니다.

cloud-init 명령을 사용하여 명령줄에서 **cloud-init** 를 실행할 수 있습니다. 선택적 인수 및 하위 명령에 대한 설명과 함께 명령 구문을 보려면 **cloud-init --help** 명령을 실행합니다. 기본 구문은 다음과 같습니다.

```
cloud-init [-h] [--version] [--file FILES] [--debug] [--force]
{init,modules,single,query,dhclient-hook,features,analyze,devel,collect-logs,clean,status}
```

다음 절차에서는 **cloud-init** 및 프로그램 재실행을 위한 샘플과 관련된 문제를 식별하기 위한 아이디어를 제공합니다.

절차

1. **cloud-init** 구성 파일을 검토합니다.

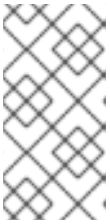
   a. **/etc/cloud/cloud.cfg** 설정 파일을 검사합니다. **cloud_init_modules,cloud_config_modules, cloud_final_modules** 에 포함된 모듈을 확인합니다.

   b. **/etc/cloud/cloud.cfg.d** 디렉토리에서 지시문(***.cfg** 파일)을 확인합니다.

2. 특정 문제에 대한 자세한 내용은 **/var/log/cloud-init.log** 및 **/var/log/cloud-init-output.log** 파일을 검토하십시오. 예를 들어 루트 파티션이 자동으로 확장되지 않은 문제가 발생한 경우 로그

메시지가 **growpart** 인지 확인합니다. 파일 시스템이 확장되지 않은 경우 로그 메시지의 **resizefs** 를 확인합니다. 예를 들어 다음과 같습니다.

```
# grep resizefs /var/log/cloud-init.log
```



참고

growpart 는 **LVM**을 지원하지 않습니다. 루트 파티션이 **LVM**을 기반으로 하는 경우 첫 번째 부팅 시 루트 파티션이 자동으로 확장되지 않습니다.

3. **cloud-init** 를 다시 실행합니다. 다음은 샘플 시나리오입니다. **root**로 명령을 실행합니다.

- **init** 모듈만 사용하여 **cloud-init** 를 다시 실행합니다.

```
/usr/bin/cloud-init -d init
```

- 구성의 모든 모듈로 **cloud-init** 를 다시 실행합니다.

```
/usr/bin/cloud-init -d modules
```

- **cloud-init** 캐시를 삭제하고 부팅 후 **cloud-init** 를 실행하도록 강제 적용합니다.

```
rm -rf /var/lib/cloud/* && /usr/bin/cloud-init -d init
```

- 다음 명령을 실행하여 디렉터리를 정리하고 정리 인스턴스를 시뮬레이션합니다.

```
rm -Rf /var/lib/cloud/instances/*
rm -Rf /var/lib/cloud/instance
rm -Rf /var/lib/cloud/data/*
reboot
```

- 다음 명령을 실행하여 **cloud-init** 를 다시 실행합니다.

```
cloud-init init --local
cloud-init init
```

추가 리소스

- CLI 인터페이스