

# Ansible Tower 소개

(IT Automation & Configuration Management)

# Application Performance Management

# Ansible 배경

**KHAN**  
a p m  
g b w

# ANSIBLE은 누구에게 필요한 도구 일까요?



- IT 운영 부담을 줄이려고 하는 개발팀, 시스템 운영팀, 네트워크 운영팀, DEVOPS 팀
  - 아직 Ansible을 사용하고 있지 않으세요?
  - IT 관련 신청 및 접수가 많아서 힘들어요 ?
  - IT 운영이 특정 인력에 의존적이신가요?
  - 애플리케이션 배포와 인프라 변경을 자주하시나요?
  - 운영에 대한 커뮤니케이션 도구가 필요해요?

# 주 52 시간 근무와 IT Automation

- 주 52 시간 근무로 “워라벨” 실현 가능할까?



줄어드는 근무시간에 대한  
근본적 해결 위한

업무 생산성  
강화 방안은?

주 52시간 시대에 효과적으로 대응하기 위해서는  
근무시간 단축 뿐만 아니라 생산성 향상을  
IT 자동화 노력이 필수적임

## 인프라 운영의 과제

- 대규모 서버의 관리에서 감사와 로깅
- 권한 위임 및 관리
- 인프라 운영을 위한 조직 체계 반영
- 멀티 플랫폼 관리 번잡함
- 서버, 네트워크 그리고 클라우드 인프라 관리
- 구성 변경 시 유연성 결여

# 왜 운용 자동화 시스템이 필요한가요?



# Ansible 을 이용한 IAC 구현

- 시스템 마다 절차와 관리도구가 다르고 수작업에 의한 운영
- Infrastructure As Code 를 통한 시스템 운영 환경의 변화

## 업무 별로 독립적인 운영 환경

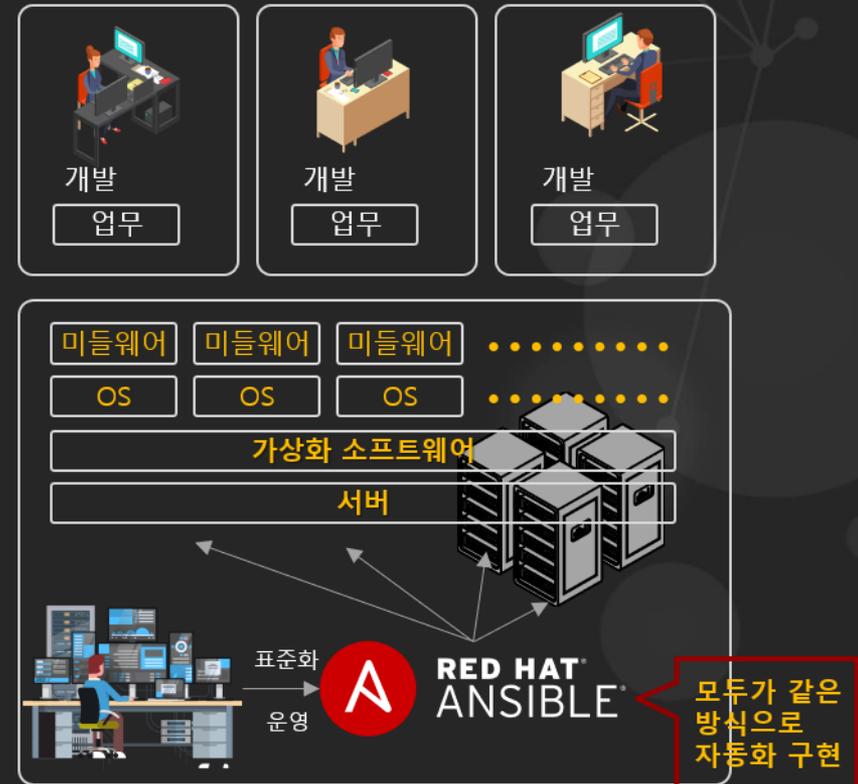


Infrastructure As Code 를 통한 표준화



운영 환경 표준화와 자동화

## 표준화와 자동화된 운영환경



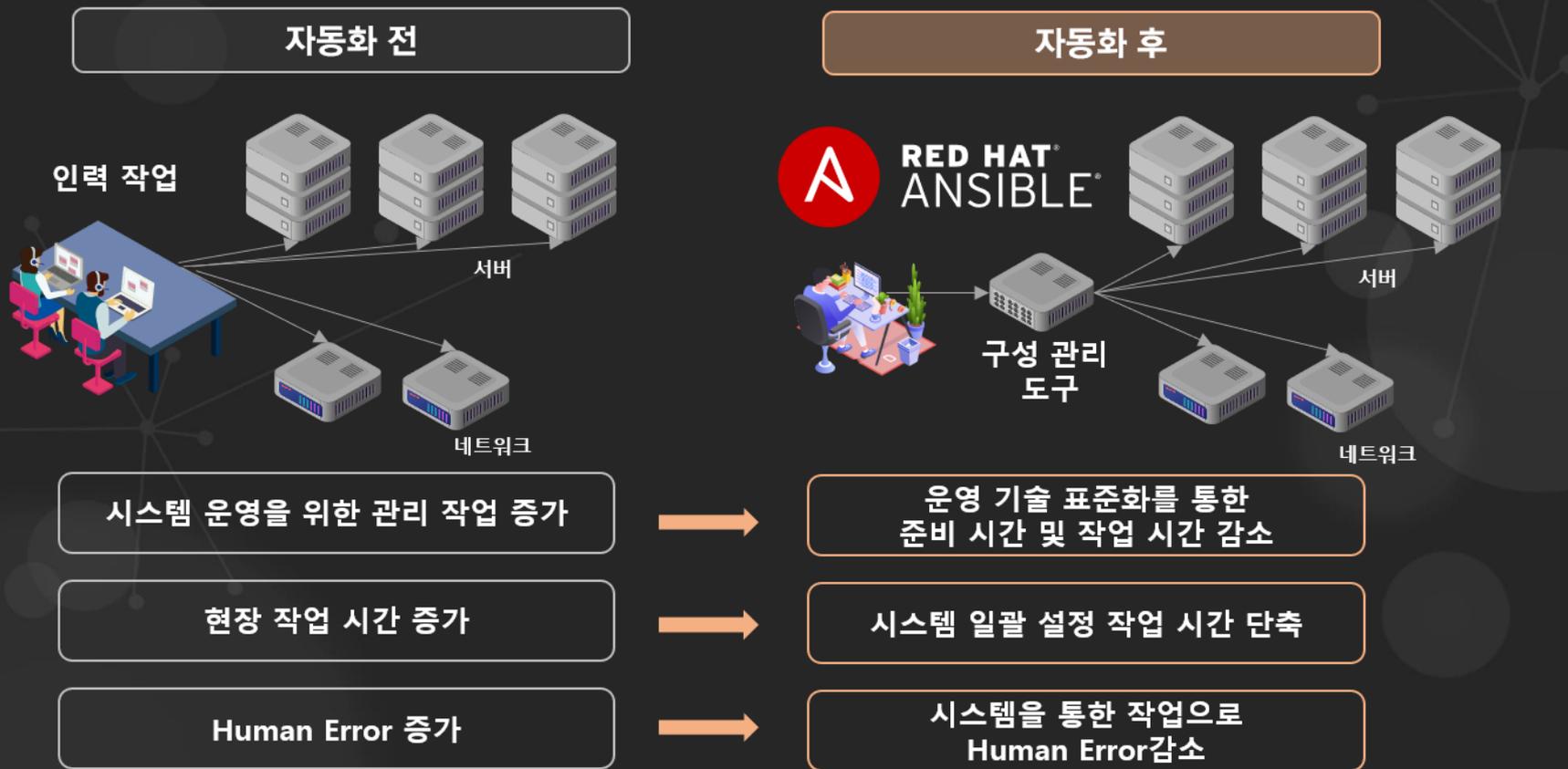
# Application Performance Management

## Ansible Tower 소개

KHAN  
a p m  
g b w

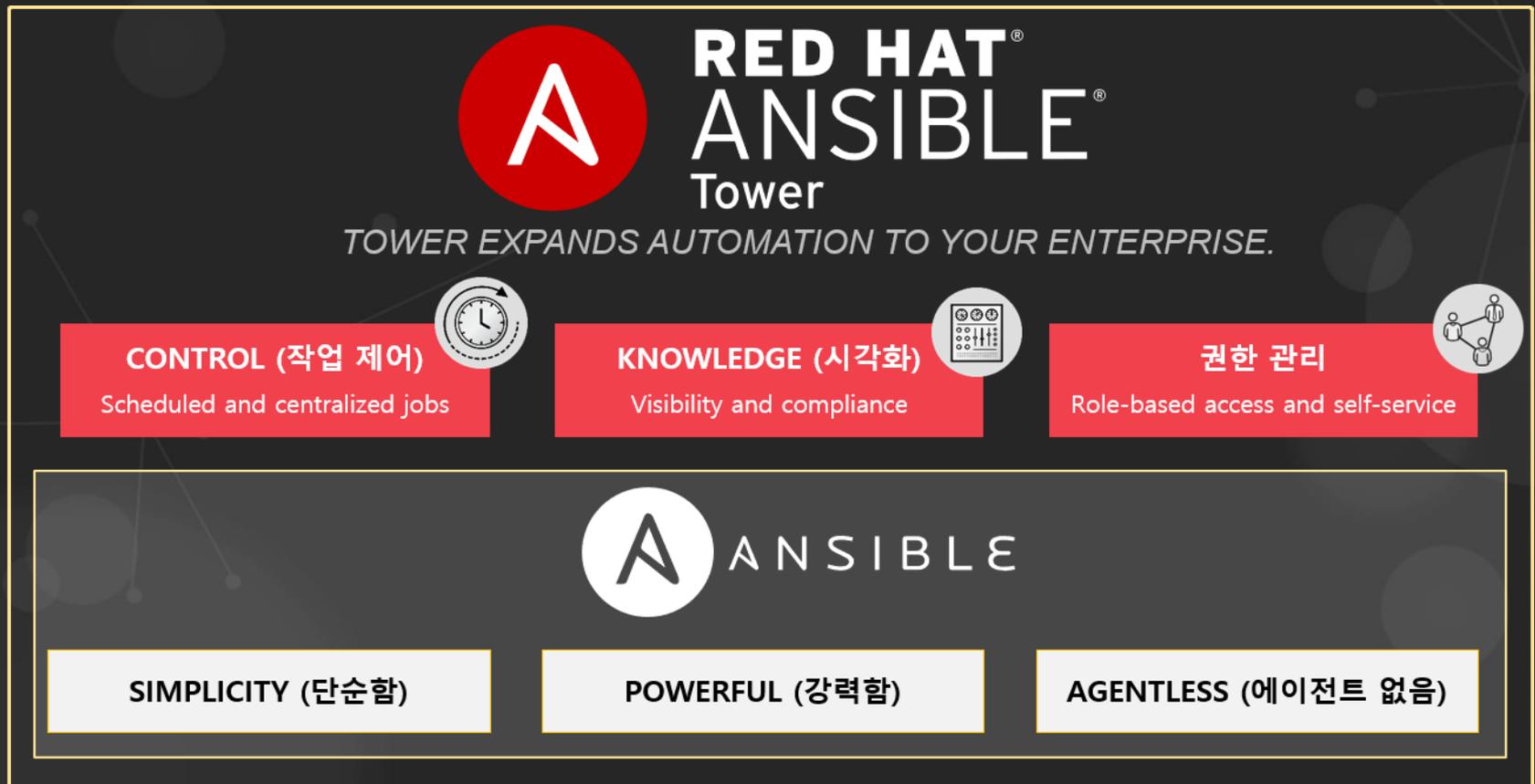
# Ansible 을 통한 IT 인프라 운영 자동화

- IT 인프라의 대규모화, 가상화 및 고도화에 따라 IT 장비에 대한 환경설정 및 정보 취합이 복잡하고 어려움
- 작업 계획시간과 현장 작업 시간의 증가와 휴먼 에러의 증가



# Ansible Tower 는

- Ansible Tower는 다양한 관리 기능을 제공하고
- 관리자는 Ansible Tower 통해 Ansible을 실행



**RED HAT<sup>®</sup>**  
**ANSIBLE<sup>®</sup>**  
Tower

*TOWER EXPANDS AUTOMATION TO YOUR ENTERPRISE.*

**CONTROL (작업 제어)**  
Scheduled and centralized jobs

**KNOWLEDGE (시각화)**  
Visibility and compliance

**권한 관리**  
Role-based access and self-service

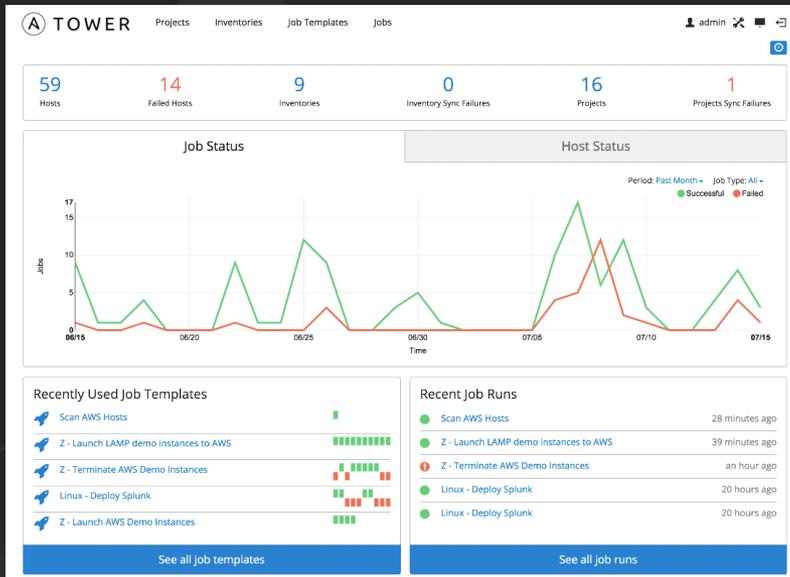
**ANSIBLE**

**SIMPLICITY (단순함)**

**POWERFUL (강력함)**

**AGENTLESS (에이전트 없음)**

# ANSIBLE TOWER



- 역할 기반 접근 제어를 통해 안전하고 효율적인 관리
- 간단한 조작으로 애플리케이션 배포
- Ansible 실행 결과는 서버에서 중앙 집중으로 감사/추적을 용이하게 제공
- Ansible 자동화, 관리 및 모니터링
  - 웹UI / API 이용

# ANSIBLE TOWER 적용 방안

- 배치 작업 및 반복 작업
- 실시간 작업 실행
- 역할 기반 접근 제어
- 대시 보드 제공
- SSH키에 대한 안전한 공유
- 감사 추적
- 권한 설정
- Restful API를 통한 관리 플랫폼 통합

# ANSIBLE TOWER



**ACCESS CONTROL**  
역할 기반ACL,  
LDAP과의 연계

**카탈로그 관리**  
Playbook의 종류와 대상 자원을 그래픽으로 관리

**DELEGATION OF CREDENTIALS**  
작업 실행의 권한 관리

**PUSH-BUTTON LAUNCH**  
작업 실행을 한 번의 클릭으로 시작

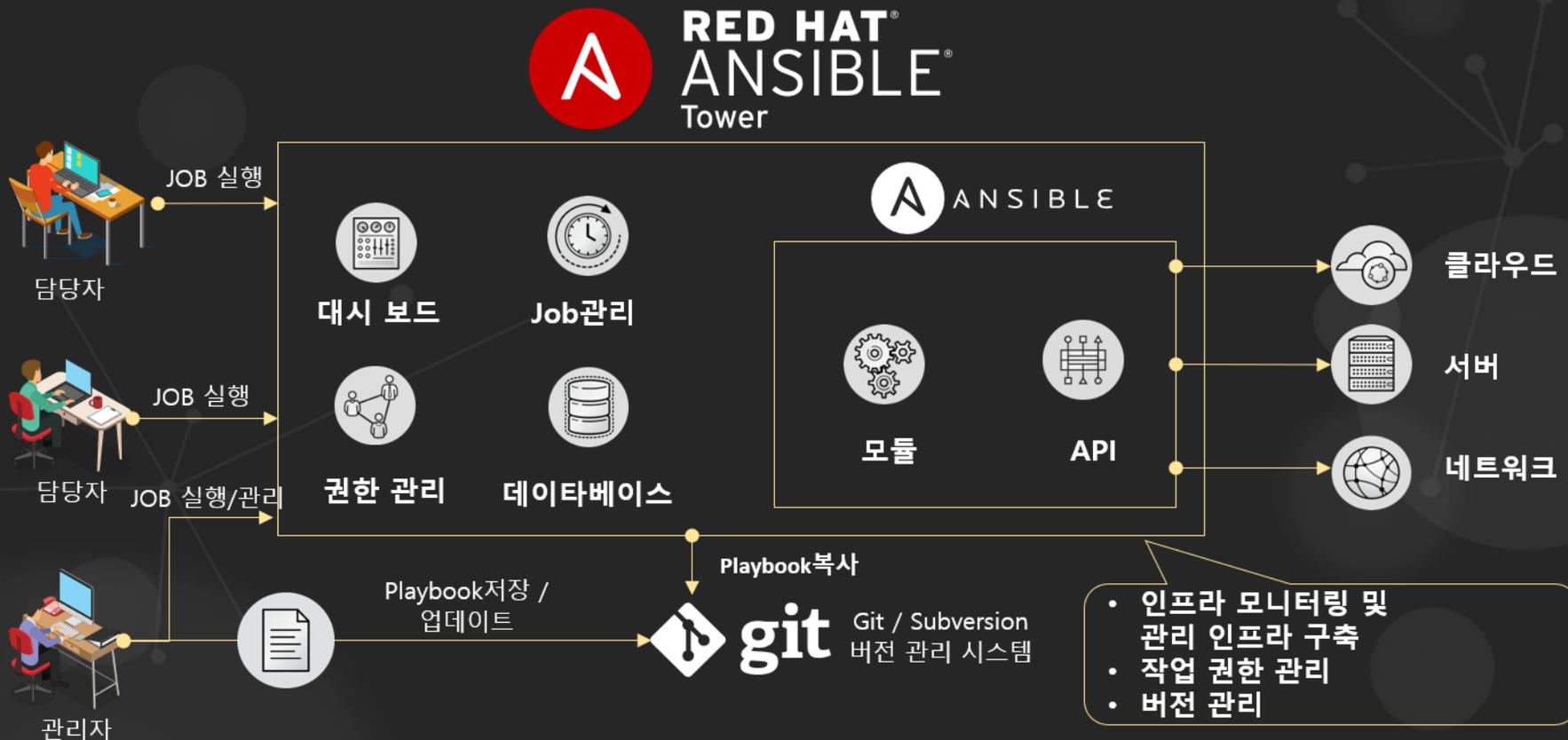
**AUDITING**  
Ansible작업 실행 기록  
를 드릴 다운 모니터

**API & CLI**  
RESTful API 를 제공하기 때문에  
외부에서API연계 가능.  
Tower CLI 를 통해 다른 도구와 통합

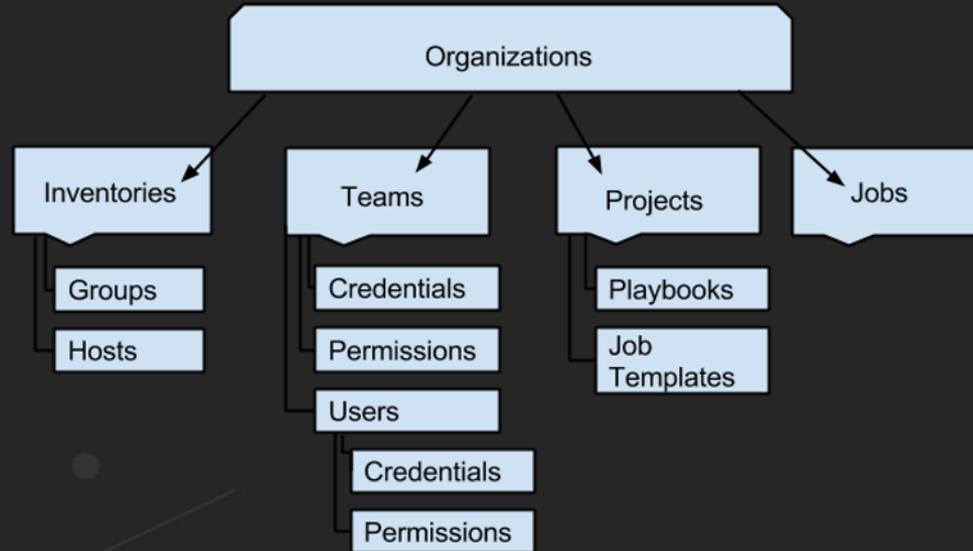
**JOB SCHEDULING**  
각종 작업 예약  
및 자동 실행  
상태 목록



# 권한 관리, 인프라 자동화 모니터링



# Ansible 계층 구조



항목		해설
Organizations		4 가지 항목을 관리하는 상위 계층
	Inventories	Playbook를 실행하는 대상 호스트 / 그룹 관리
	Teams	팀 단위로 접근을 제한하는 단위. Tower에서 각 서버에 연결하기위한 인증 정보, Inventory / Job Template에 대한 액세스 권한 관리
	Projects	Ansible Playbook 및 작업 템플릿을 관리하는 단위. Playbook 저장할 SCM, Git / Subversion / Mercurial / Manual (Tower 서버의 파일 관리)에서 선택 또한 Playbooks 실행 서버와 Job Template을 만들고 관리한다.
	Jobs	Ansible Playbook 실행 기록

## Application Performance Management

# Ansible Tower 세부 기능 소개

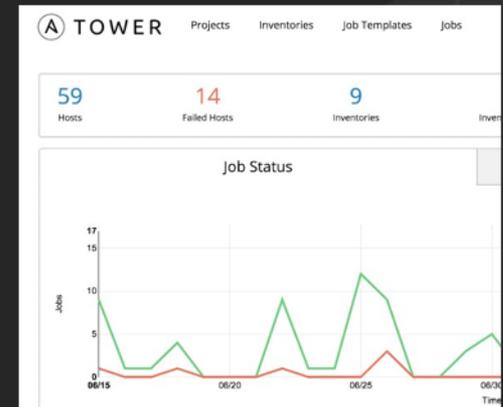
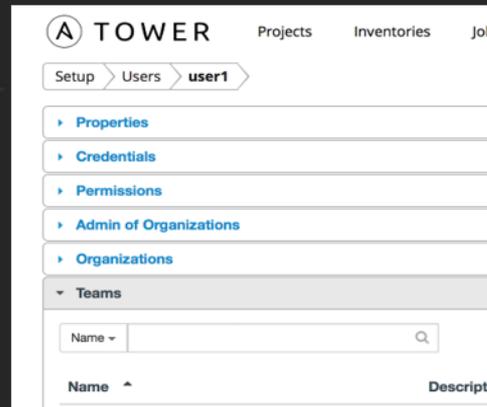
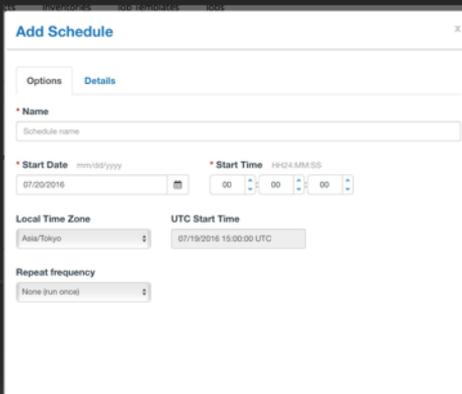
**KHAN**  
[ a p m ]  
[ g b w ]



## 작업 제어(Control)

## 권한 관리(Delegate)

## 시각화(Knowledge)



- 스케줄에 따른 작업
- 중앙 통제 작업 / Batch 작업 실행
- Ad-hoc Command 실행
- 작업 추상화

- 사용자 인증 기능 (로그인)
- 사용자/팀 별 권한 설정

- 대시 보드
- Job의 실행 결과
- Job Template 실행 결과
- 업데이트 실행
- Inventory/Host 실행 결과

# Ansible Tower - 주요 기능 (1)



## Restful API

- 외부 Restful API를 통해 Ansible Tower 기능을 수행



## 알림 기능

- Job 실행/playbook 업데이트 알림 기능
- Email/ Slack/ Twilio/ PagerDuty/ HipChat/ Webhook



## 비밀번호 관리

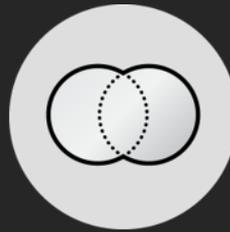
- 단 방향 패스워드 관리

# Ansible Tower - 주요 기능 (2)



## 점검용 Job

- 외부 Restful API를 통해 Ansible Tower 기능을 수행



## Fact Caching

- 수집 된 fact를 지정된 시간 동안 유지
- fact 수집을 여러 번 실행해서 처리 시간을 단축.



## Activity Stream

- 전체 작업에 대한 실행 기록의 시계열 표시

## Ansible Tower – 기타 기능

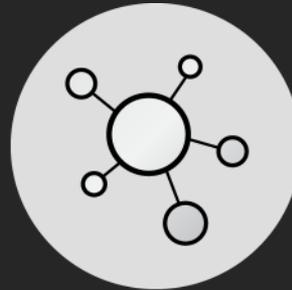
- Active / Passive 구성
- 화면 사용자 지정
- SAML / RADIUS 인증
- LDAP 지원
- 멀티테넌트 지원 ( 복수 Organization 지원 )
- Surveys
  - 사용자 Job 실행시에 특정 변수의 값을 선택하는 대화 만드는 기능
- Scan Job
  - Server Status / Service / package 정보를 조회하는 특수 Job
- System Tracking
  - 과거의 결과와 비교하여 차이를 확인하는 기능
- Notification
  - Job 실행 또는 Playbook 업데이트 등의 Action 을 e-mail, SMS Chat tool로 통지
- Activity Stream
  - Ansible Tower의 모든 실행 기록을 시간순으로 표시하는 기능
- etc...

# Ansible 사용 대상



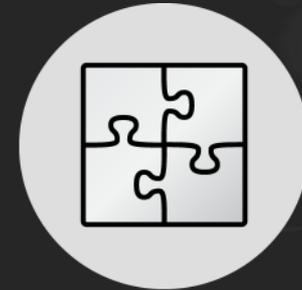
## 프로비저닝

- 패키지 설치
- 설정 변경
- 파일 전송



## 오케스트레이션

- 여러 기기 제품에 대해 자동으로 주문 실행
- 서버 / 라우터 / 네트워크 스위치 / 방화벽 / Load Balancer / Storage / Database / Cloud 등 ...



## 구성 관리

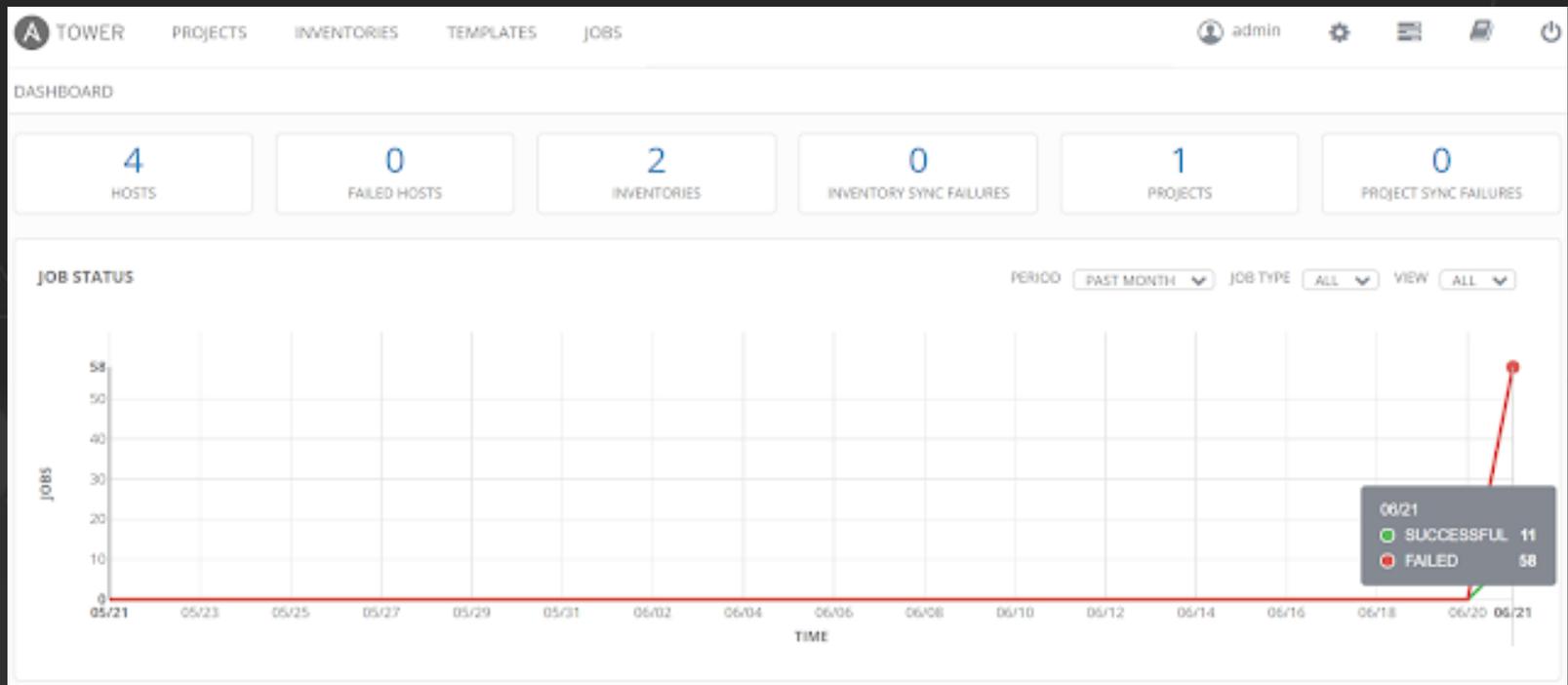
- 상태 확인
- Batch 처리
- 업데이트 실행
- 보안 패치 확인

# Ansible Tower 3.1.3 – Pre requisite

- OS
  - Red Hat Enterprise Linux 7.4 or later 64-bit
  - CentOS 7.4 or later 64-bit
  - Ubuntu 14.04 LTS 64-bit
  - Ubuntu 16.04 LTS 64-bit
- 메모리 : 2GB 이상 (권장 4GB 이상)
- 디스크 용량 : / var /에 20GB 이상의 여유 공간
- 자세한 내용은 아래의 URL에서 확인
  - [http://docs.ansible.com/ansible-tower/latest/html/installandreference/requirements\\_refguide.html](http://docs.ansible.com/ansible-tower/latest/html/installandreference/requirements_refguide.html)

# Ansible Tower - 샘플 작업 실행

- Ansible Tower에 로그인
- 샘플 실행

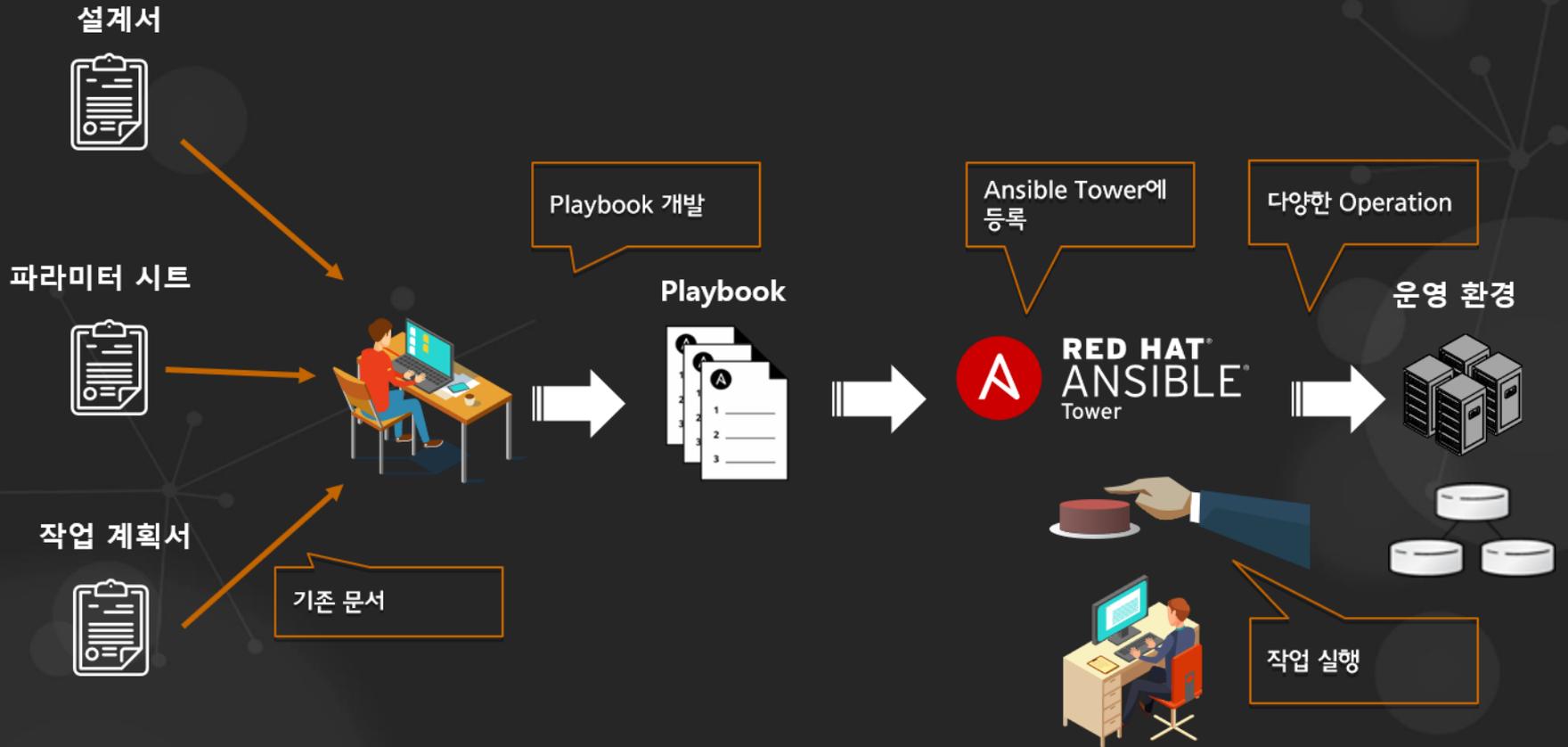


Application Performance Management

# Ansible Tower Use Case

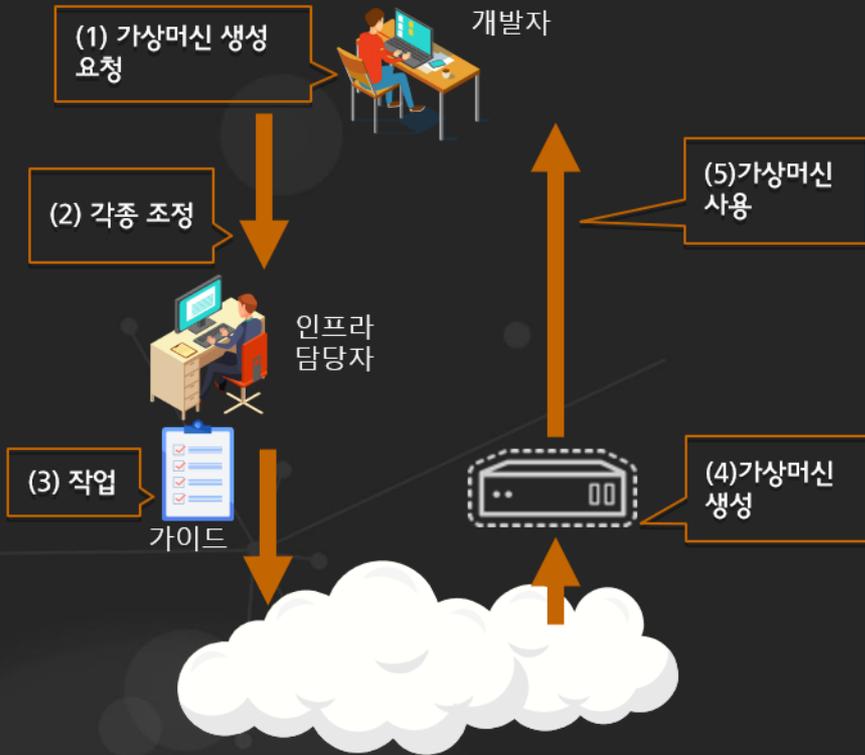
**KHAN**  
a p m  
g b w

# Ansible 에서의 자동화의 흐름



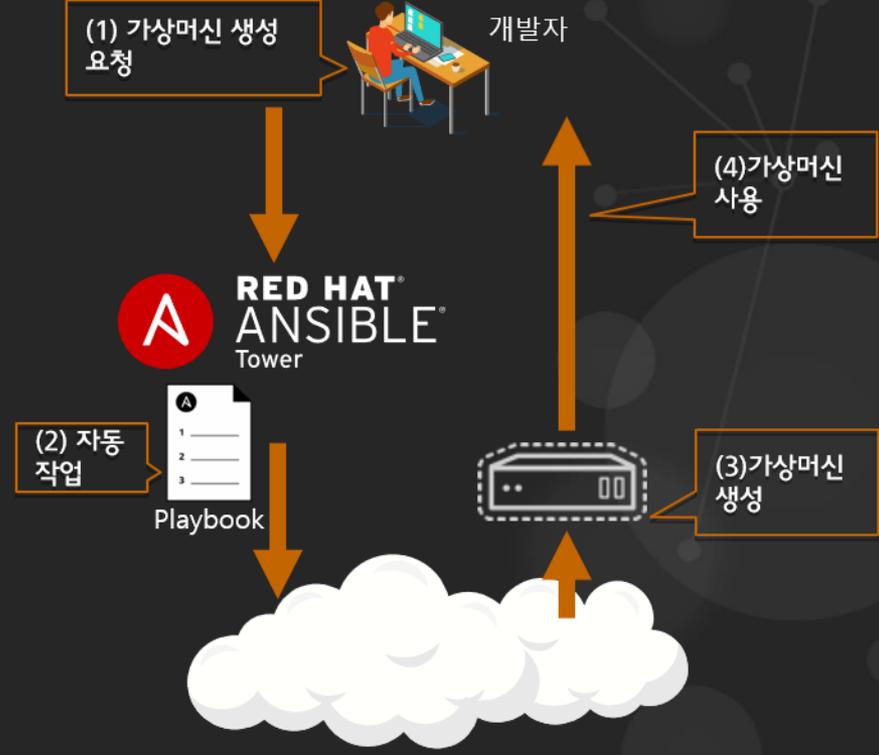
# Ansible 도입 방안 – 가상화

## Before Ansible



- ✓ 사람에 의한 조정과 작업
- ✓ Human Error 와 품질의 차이

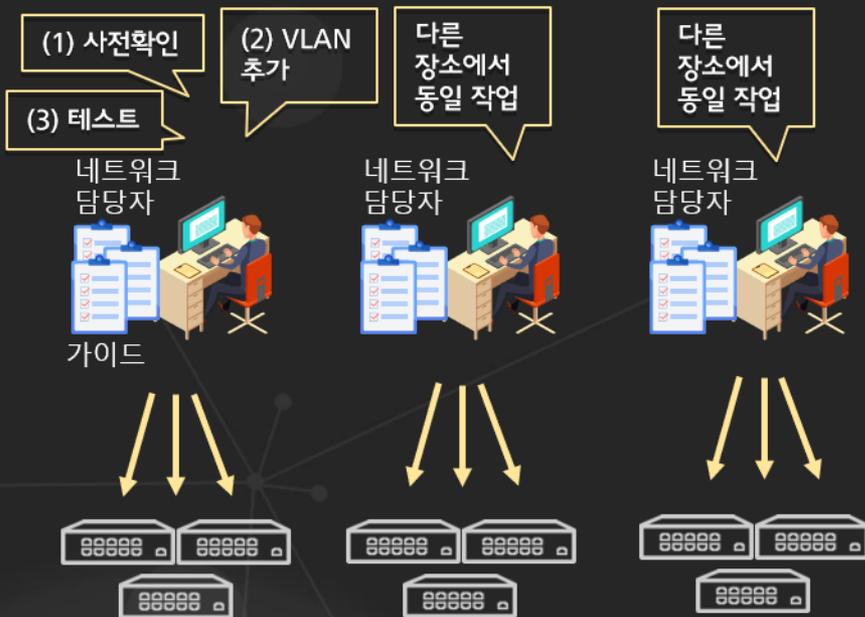
## After Ansible



- ✓ 테스트까지도 자동화
- ✓ 일정한 품질 유지

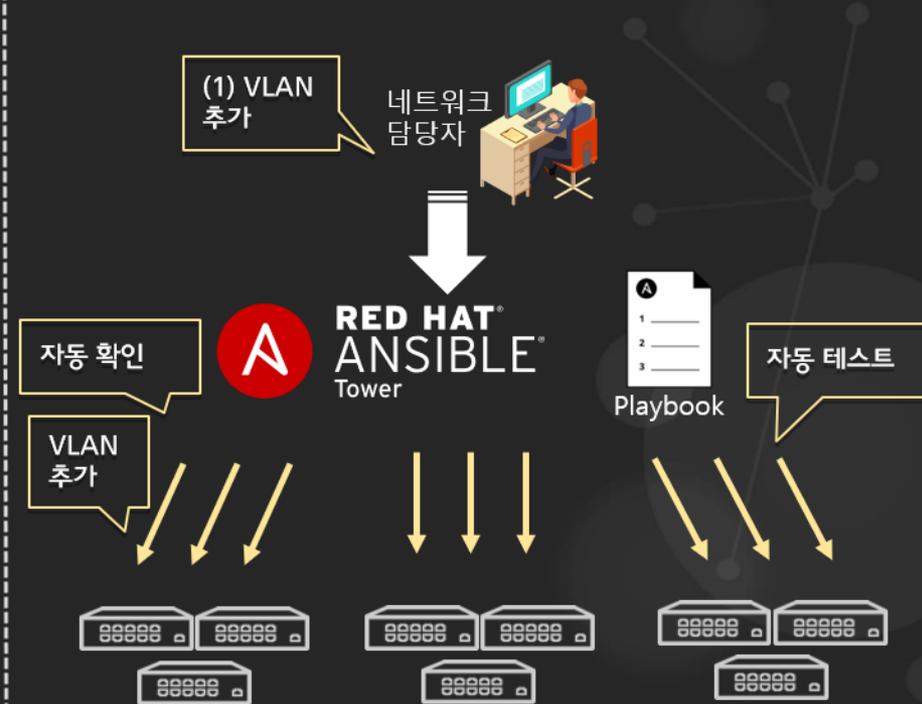
# Ansible 도입 방안 – 네트워크 관리

## Before Ansible



- ✓ 동일한 작업을 반복 수행, 횟수 만큼 시간 소요
- ✓ 제한적인 테스트와 품질의 차이

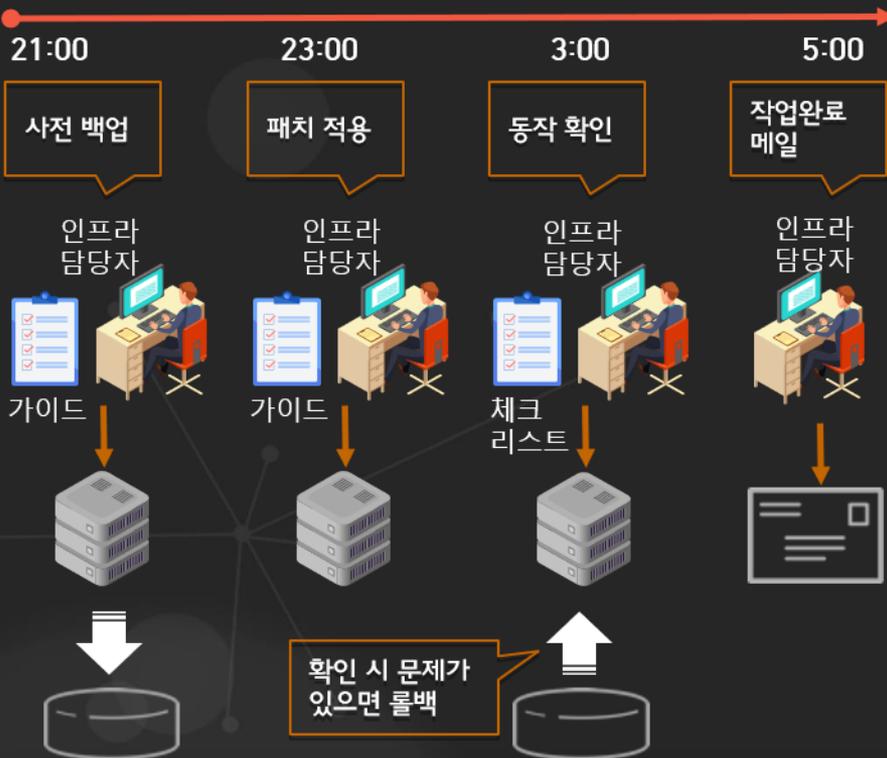
## After Ansible



- ✓ 하나를 자동화하면 반복적인 작업 가능
- ✓ 자동으로 확인하고 테스트
- ✓ 일정한 품질 유지

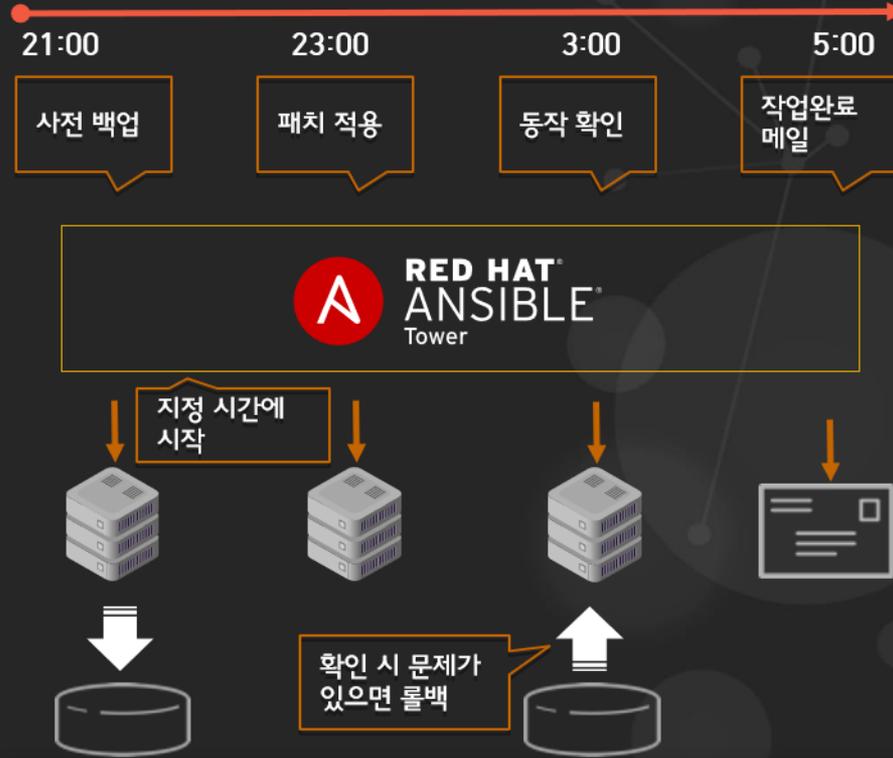
# Ansible 도입 방안 – 야간 패치 작업

## Before Ansible



- ✓ 작업하는 동안 상주 지원하며 시간 소요
- ✓ 오류 확인 및 롤백에 대한 위험요소
- ✓ 품질의 차이

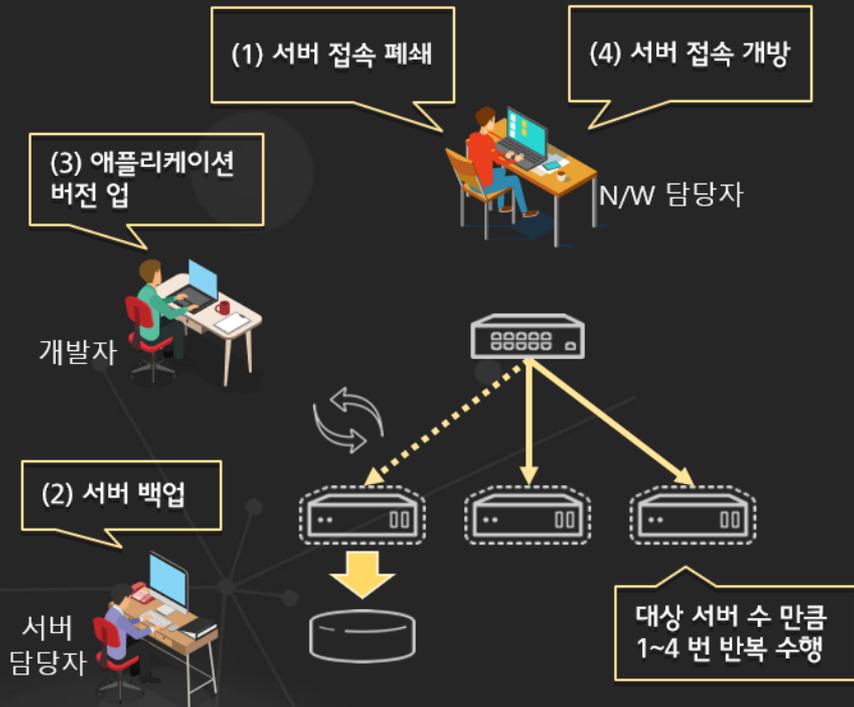
## After Ansible



- ✓ 비상주 지원과 유비보수 시간 단축
- ✓ Human Error 차단
- ✓ 일정한 품질 유지

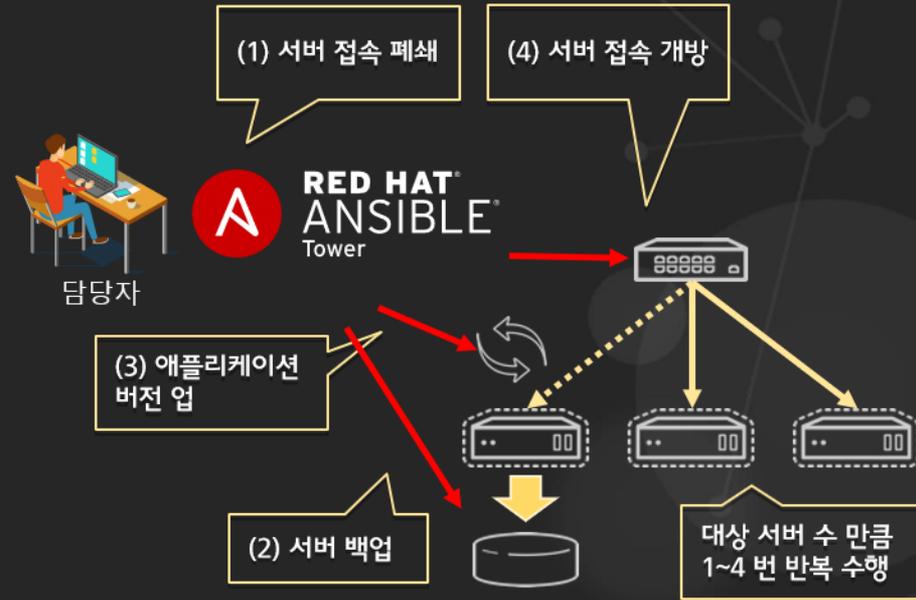
# Ansible 도입 방안 - 애플리케이션 릴리즈

## Before Ansible



- ✓ 작업하는 동안 상주 지원하며 시간 소요
- ✓ 작업 관련자 전원이 일정을 맞추어 참여
- ✓ 서버 대수가 많을 수록 시간과 작업량 증가

## After Ansible



- ✓ 비상주 지원과 유비보수 시간 단축
- ✓ Human Error 차단
- ✓ 서버 대수와 상관 없이 일정한 품질 유지

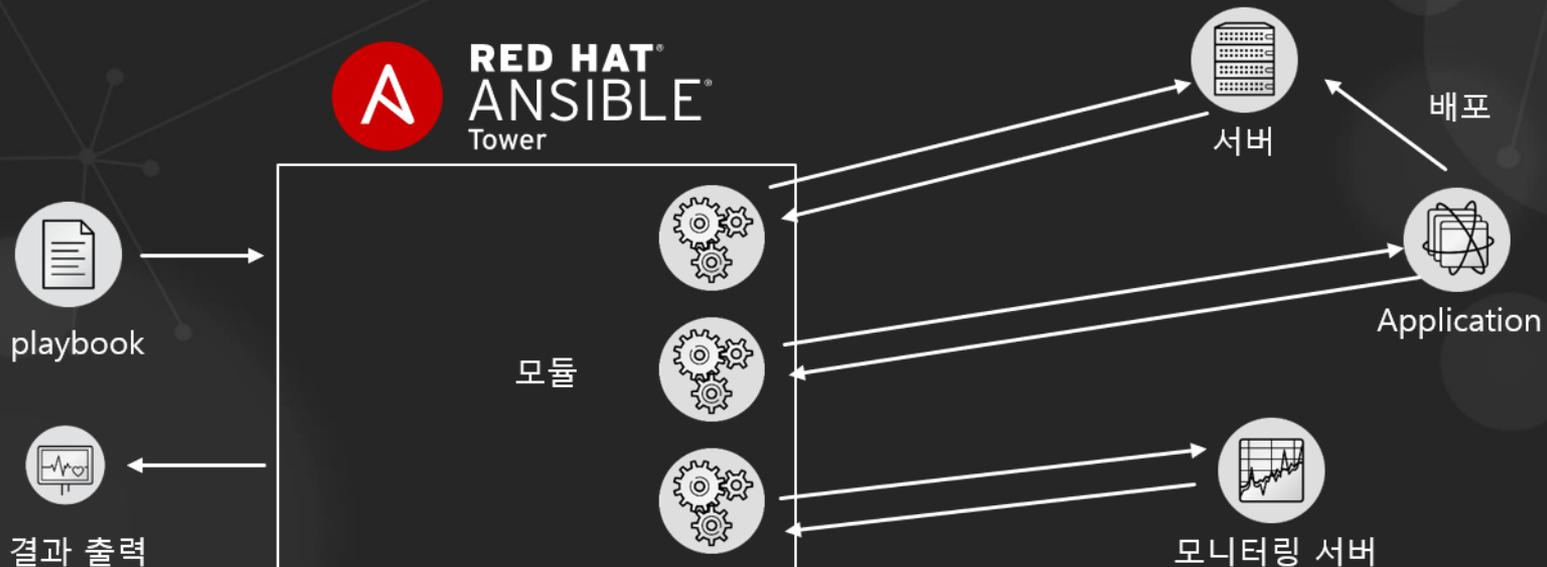
# Server구축 및 모니터링 설정 자동화

## Before Ansible

- 수작업으로 인하여 작업 시간이 소요
- 사람에 의한 모니터링으로 Human Error

## After Ansible

- Server 구축( Cloud / 가상화 환경에서 배포)을 Ansible 로 실행
- 모니터링 툴 설정도 Ansible 로 동시에 구성하여, 오류 제거와 작업 시간 단축



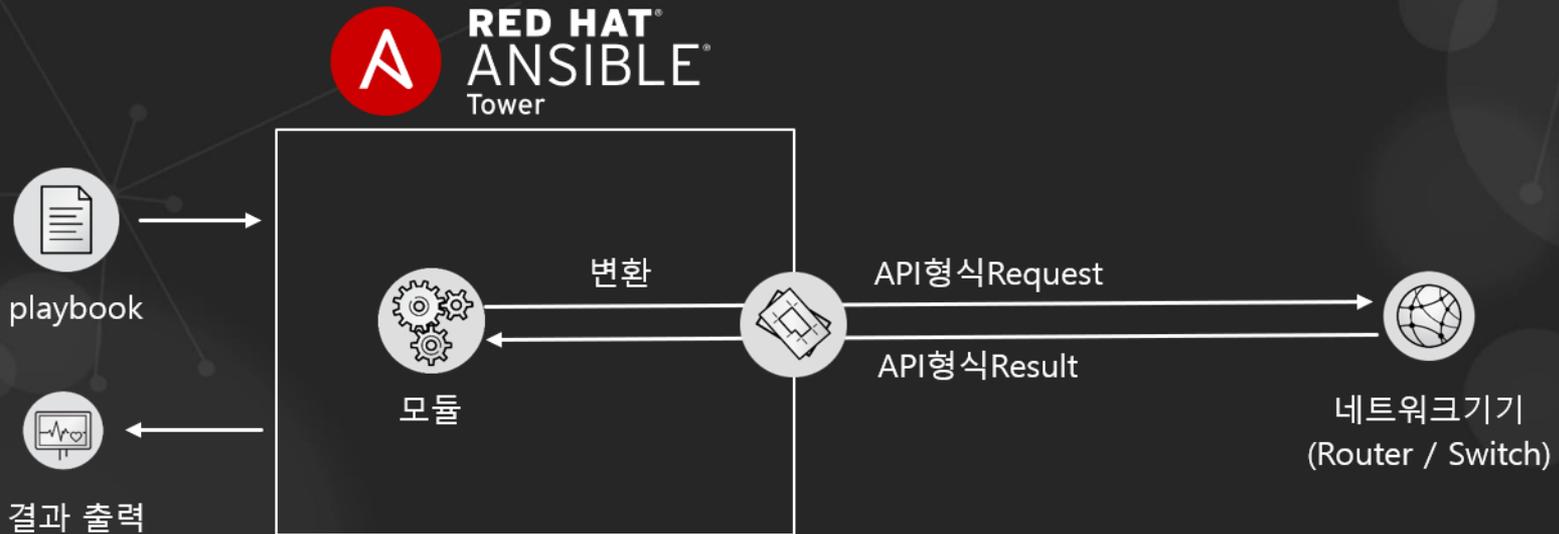
# Network 장비 관리

## Before Ansible

- CLI 커멘드로 NW장비 설정 자동화는 어려움
- CLI의 조작 방법(설정 서식 등)이 OS Ver에 따라 변경 됨
- API를 사용하기에도 커멘드가 많으며 서식이 복잡함

## After Ansible

- Ansible 모듈은 실제로 동작하는 NetConf 등의 API를 사용
- API Ver 지정할 수 있어 OS Version가 바뀌어도 계속해 사용
- API 형식으로 커멘드를 변환하는 것은 Ansible Module에서 처리



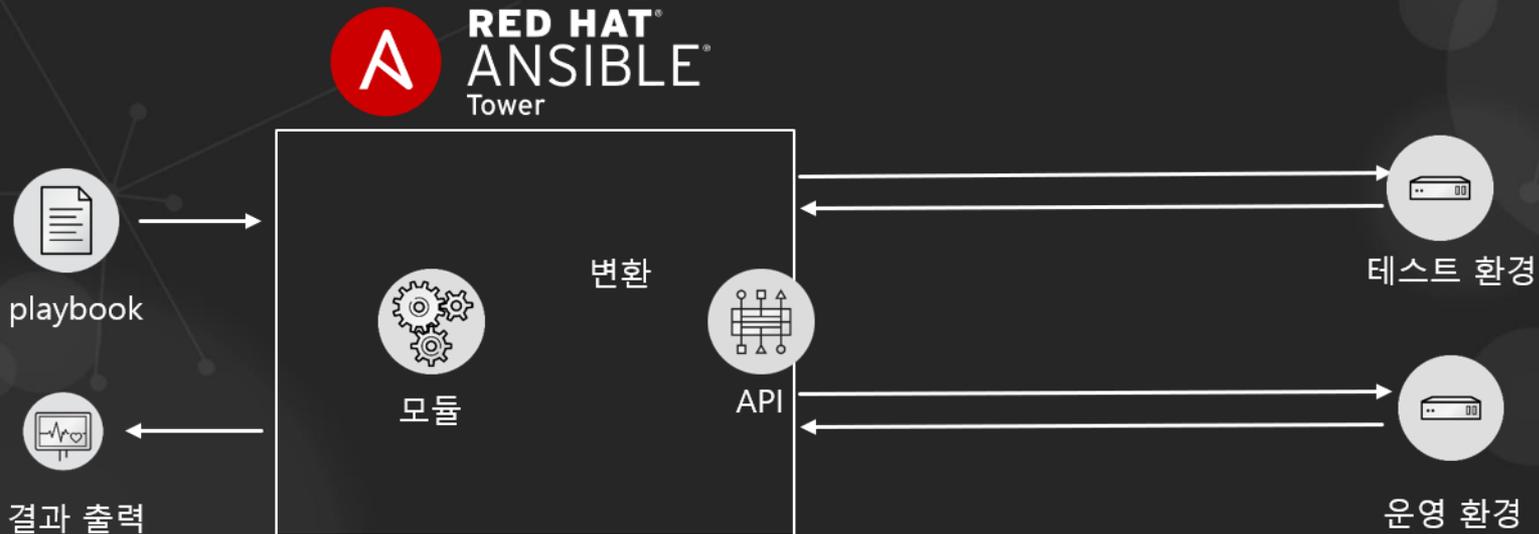
# 작업 계획서 대체

## Before Ansible

- 테스트 환경에서 구축 및 테스트 후, 작업계획서를 작성해 리뷰의 뒤, 순서서를 보면서 실전 환경의 구축
- Human Error
- 인건비가 많아지고 민첩성이 떨어짐

## After Ansible

- Ansible 의playbook 을 사용하여 작업계획서 내용을 Code 로 작성
- Human Error 제거 및 신속한 배포가 가능
- 오류가 발생해도 신속하게 다시 실행할 수
- 작업계획서를 playbook로 대체



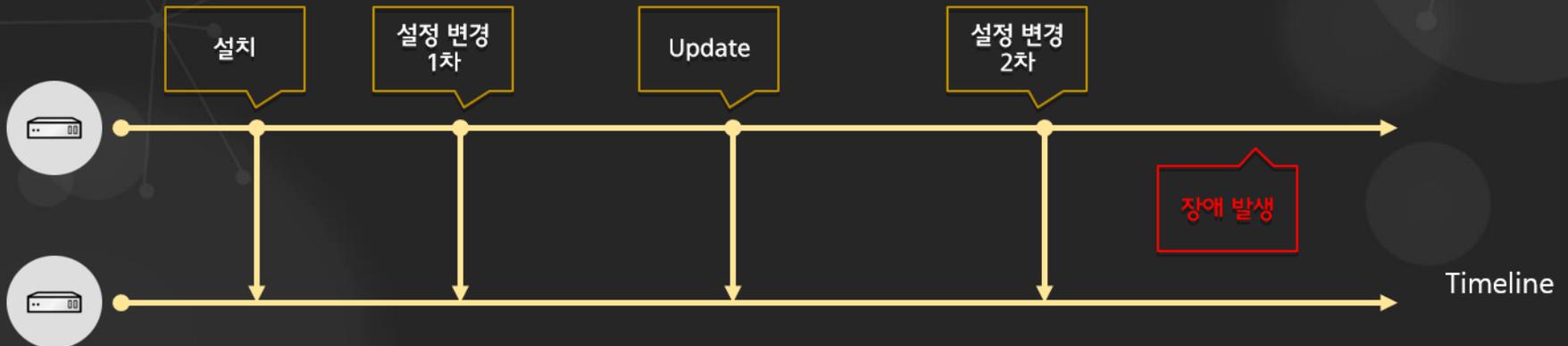
# 변경 이력 관리

## Before Ansible

- 기존 운영 환경에서는 모든 작업 이력을 관리하기 어려움
- 여러 사용자가 로그인하고 root 계정을 공유

## After Ansible

- Ansible Tower로 사용자 접근 및 권한 관리
- 설정 변경 시 작업자와 작업 내용을 시계열로 저장하여 확인 가능
- 장애 발생시 직전의 변경(문제가 발생이 원인이 될 수 있는) 내용을 정확하게 파악



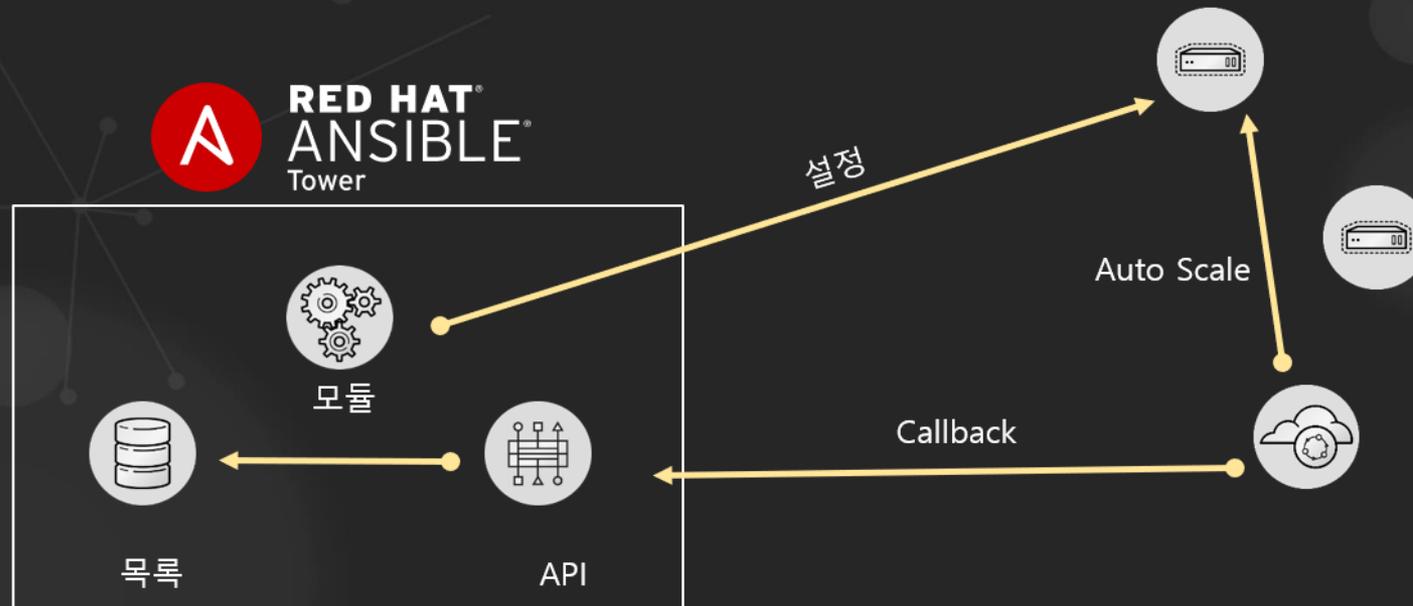
# Cloud 환경에서 Auto-Scale 기능과 연계

## Before Ansible

- Auto-Scale 기능으로 추가된 Host에 대해서 한정적인 설정만
- Auto-Scale 에 세부 내용은 수작업으로 조작

## After Ansible

- Ansible Tower를 이용하여 Auto-Scale로 노드의 작성 완료를 Callback을 인식하여 자동으로 Inventory 에 추가
- 자동으로 Auto-Scale에 의한 확장/축소



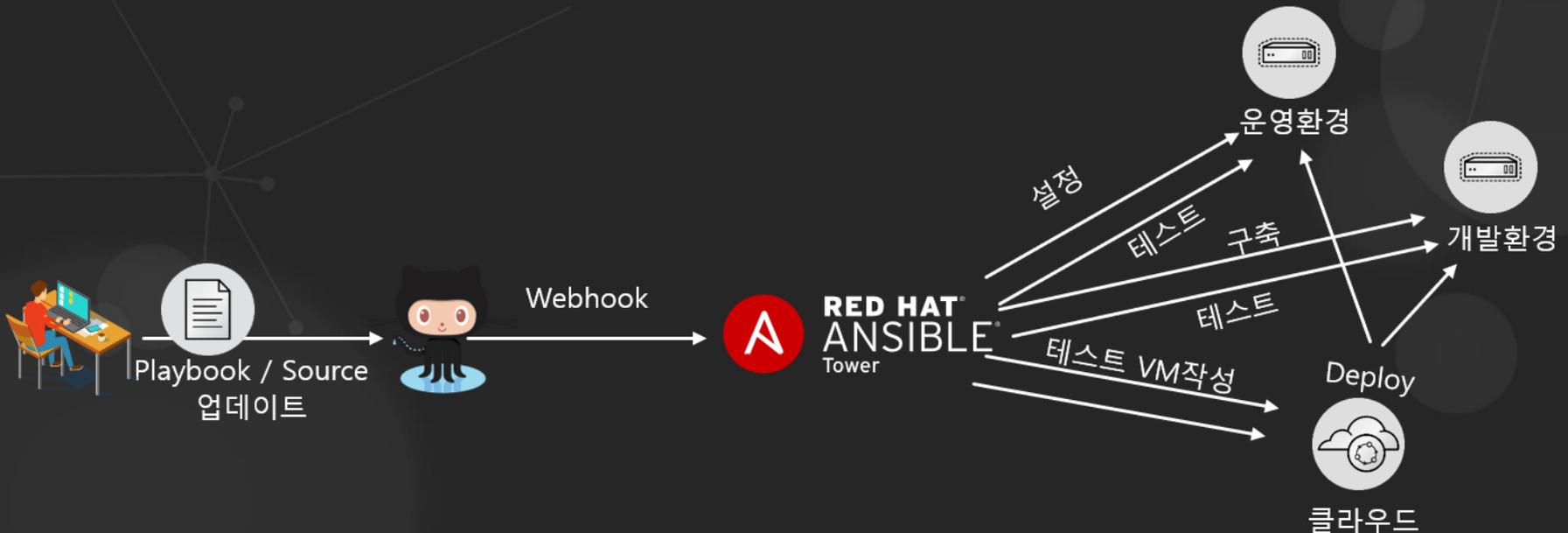
# DevOps 구현

## Before Ansible

- 소스 변경 트리거에 의해 자동 배포하기 위해서는 버전 관리 시스템 이외에, 작업 관리, 구성 관리 도구, 설정 확인 도구 등 여러 제품을 조합하여 개발해야함

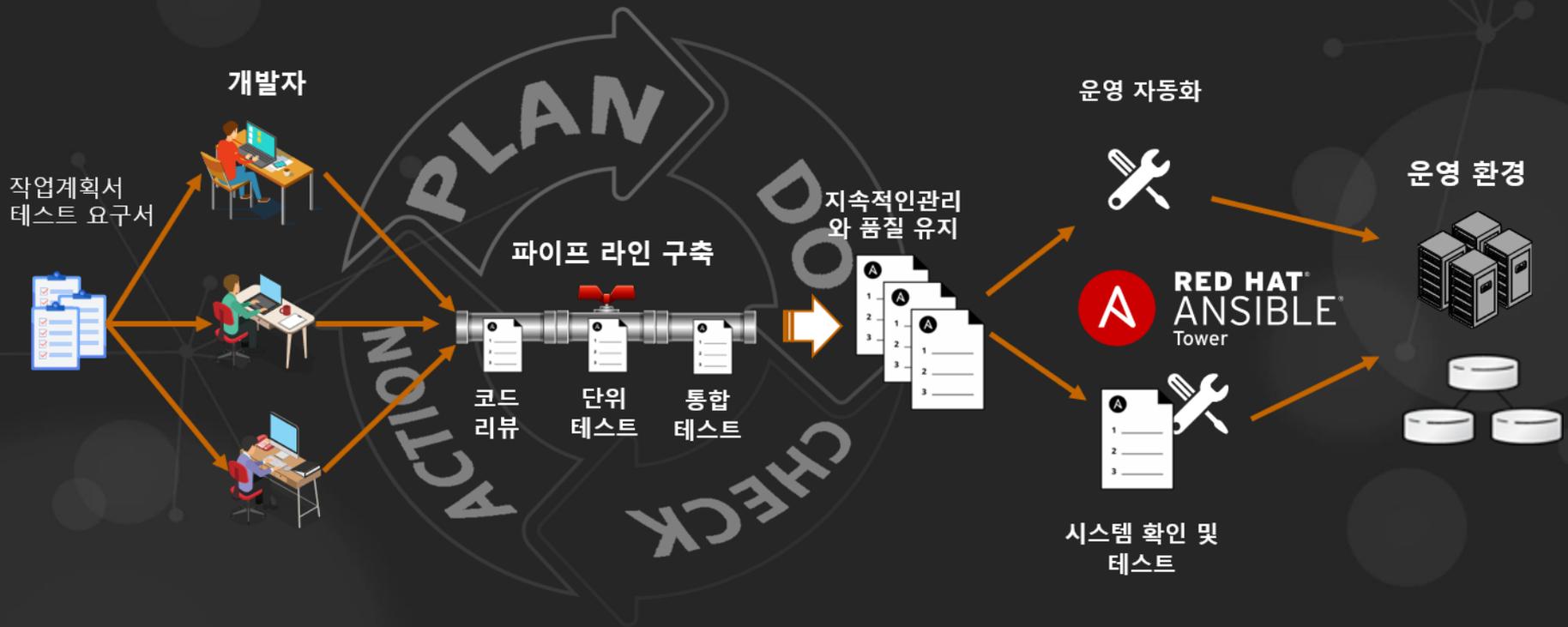
## After Ansible

- Ansible Tower를 사용해, Version 관리와 배포까지 자동화
- 예를 들면, Playbook 업데이트 → 변경 알림 → 테스트 환경 배포 → 설정 → 구성 확인 → 테스트 → 운영 환경 배포 → 설정 → 테스트까지 자동화



# Ansible 도입 방안 – 인프라 CI 구현

- 산출물( Playbook )의 품질을 관리하고 지속적인 관리
  - Infrastructure as Code 구현
  - 각종 CI /CD 도구와 함께 적용



# Application Performance Management

# Ansible 적용 효과

KHAN

apm

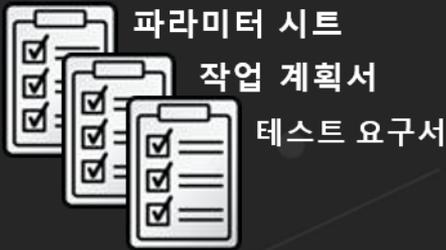
gbw

# Ansible 자동화 추진 방안

- Start Small Think Big

단순한 작업들을 대체  
(개발이 쉬운 부분부터 일부분 자동화)

대체



Playbook



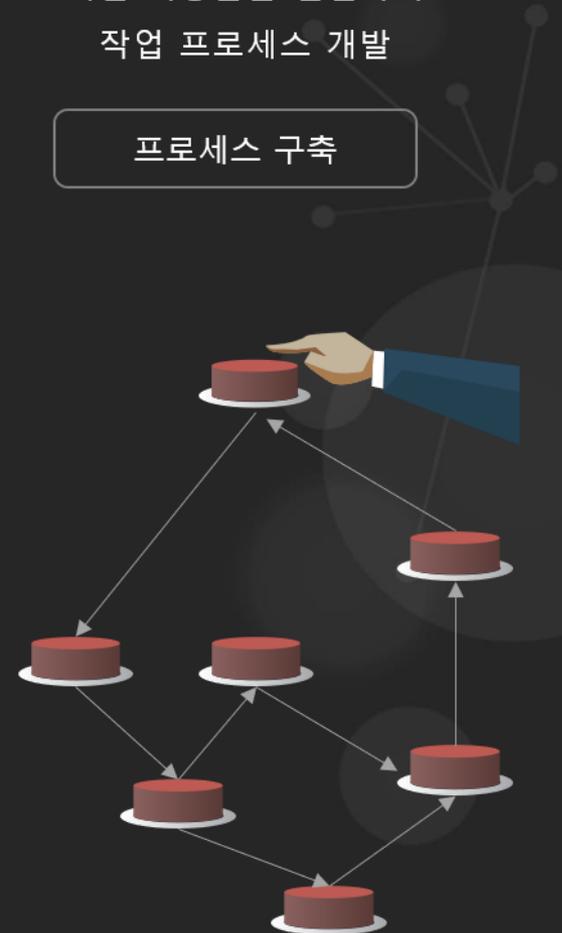
자동화를 통하여  
운영 작업 위탁 ( 버튼 )

기능화 (서비스화)



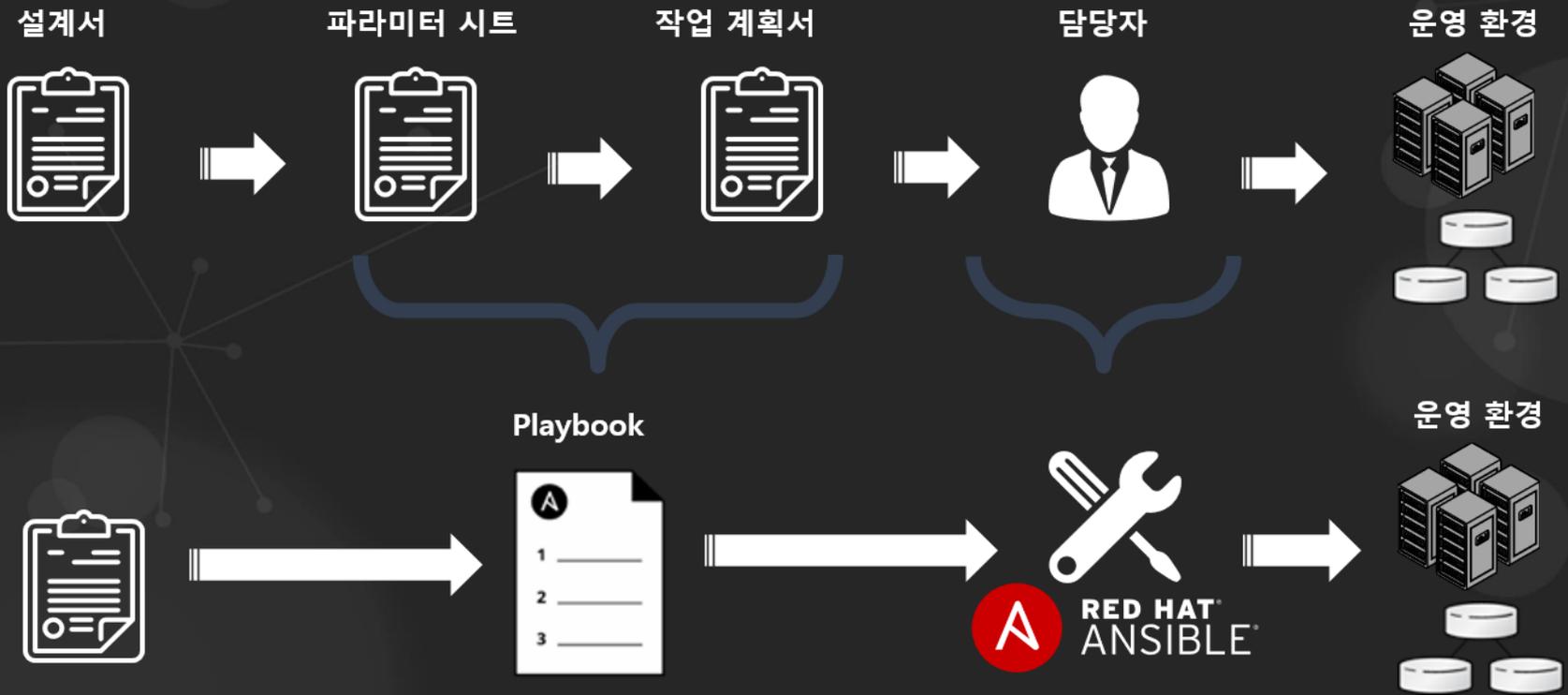
작은 기능들을 연결하여  
작업 프로세스 개발

프로세스 구축



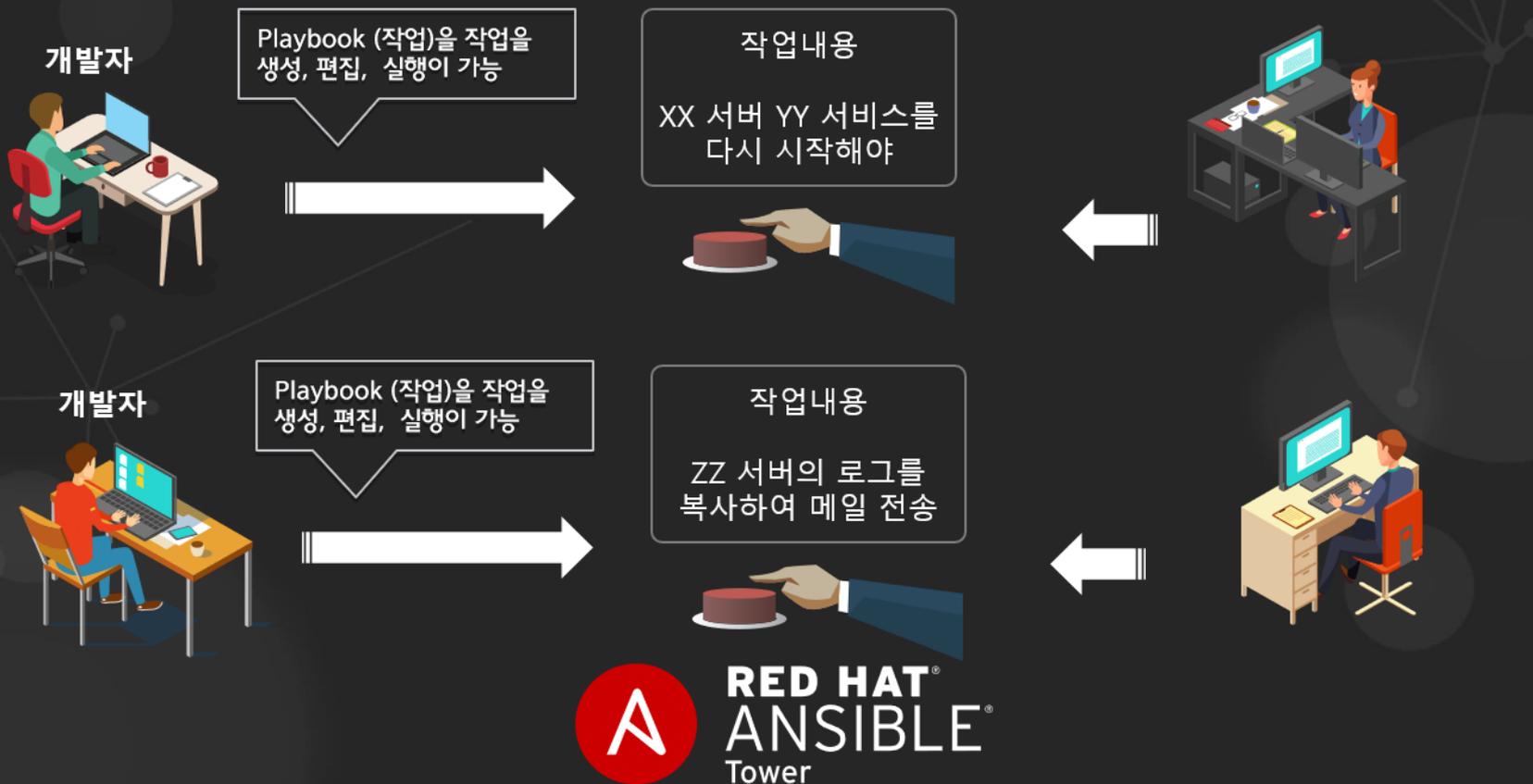
# Ansible 을 통한 작업 계획서 대체

- 현재 운영 작업들을 Playbook 로 정의하여 자동화
  - Playbook 은 실행 가능한 파라미터 시트 겸 가이드



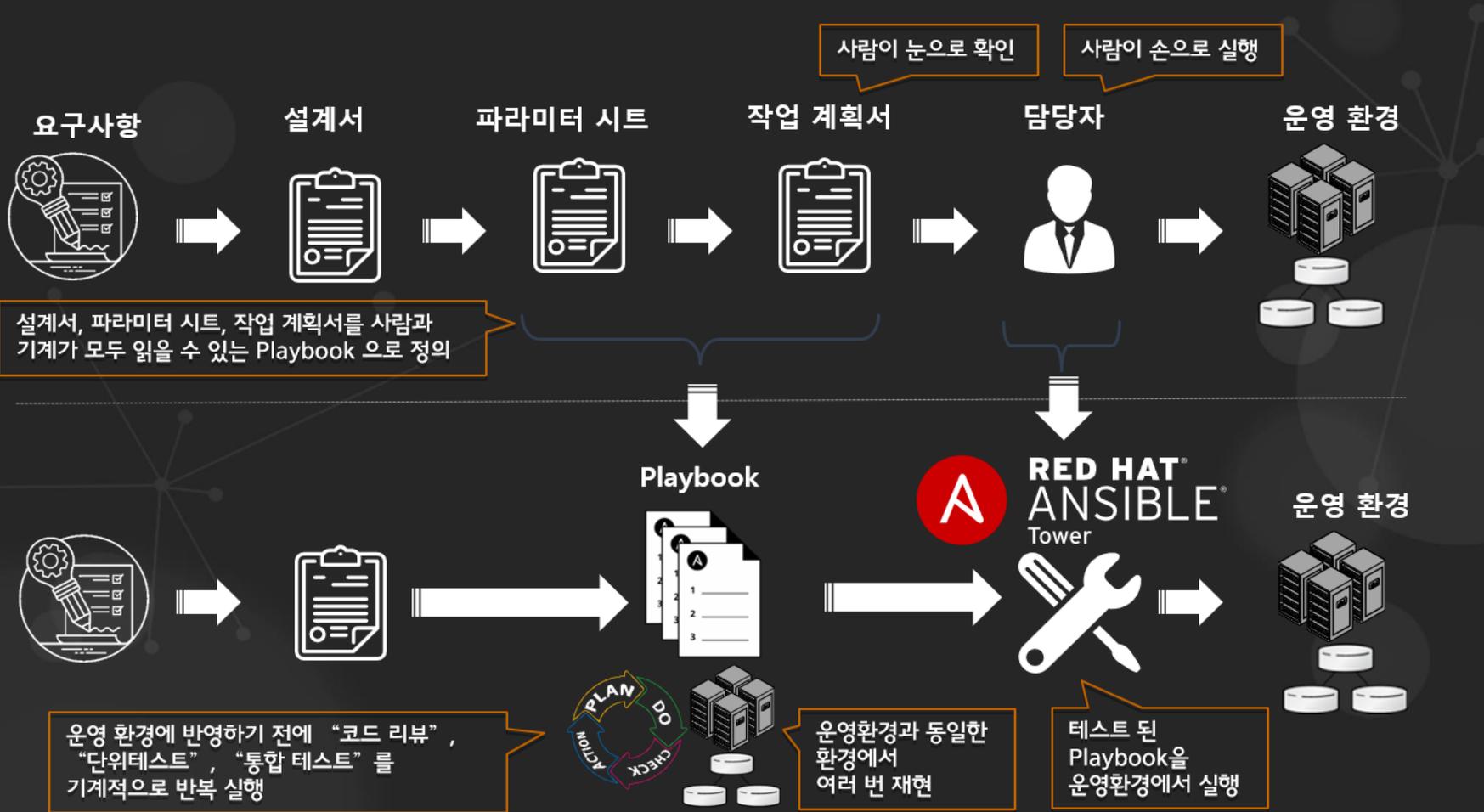
# 운영 작업 위임 (서비스 화)

- 개발된 Playbook 에 대한 실행 권한을 운영자에게 위임
  - 서비스를 재 시작할 수 있는 권한을 운영자에게 위임
  - 서버 로그를 메일로 전달하는 권한을 운영자에게 위임



# Ansible 을 통한 전체 작업의 효율화

- Ansible Playbook으로 Code를 정의하고 검증 후 운영 환경에서 반영



# 애플리케이션을 위한 인프라 제공

- 클라우드 환경에서 앤서블을 통한 인프라 제공의 효과

작업	시간 (분)
가상머신 생성	30
스토리지/네트워킹 추가	30
팀 (작업) 전환 시 대기 시간	120
운영체제 설치	60
설치 후 대기	60
운영체제 구성	90
응용프로그램 설치	90
응용프로그램 구성	45
팀 (작업) 전환 시 대기 시간	120
보안 구성 및 검사	90
실제 작업 시간	5 시간 15분
전체 시간	10 시간

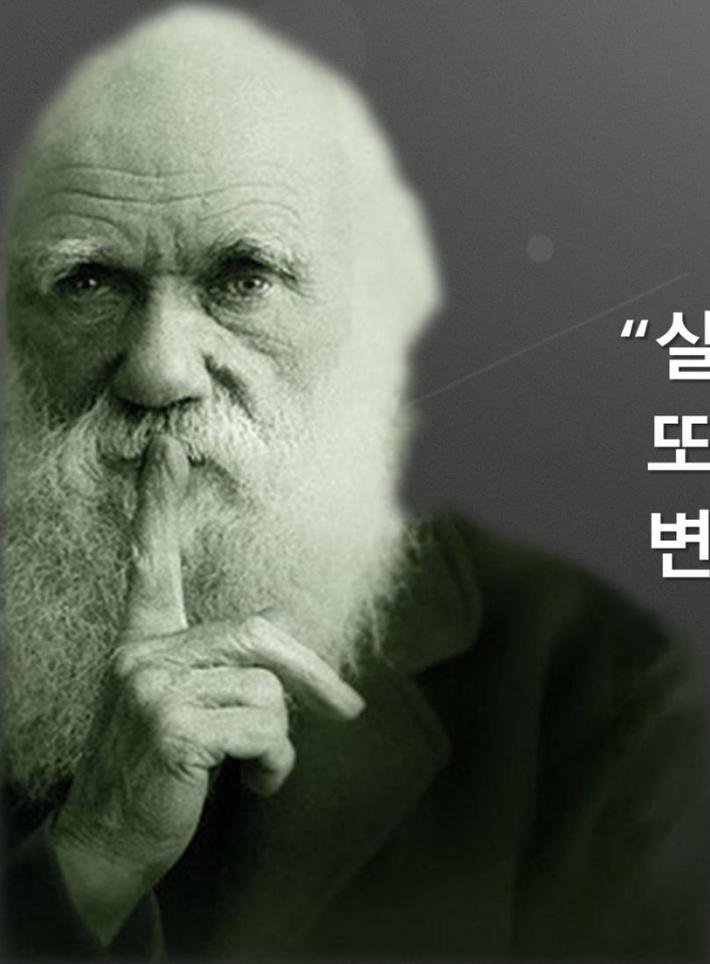
  
 앤서블 오케스트레이션

작업	시간 (분)
가상머신 생성	2
스토리지/네트워킹 추가	3
<del>팀 (작업) 전환 시 대기 시간</del>	<del>120</del>
운영체제 설치	2
<del>설치 후 대기</del>	<del>60</del>
운영체제 구성	1
응용프로그램 설치	2
응용프로그램 구성	1
<del>팀 (작업) 전환 시 대기 시간</del>	<del>120</del>
보안 구성 및 검사	2
실제 작업 시간	0 분
전체 시간	13 분

} 운영 자동화  
 (인프라 프로비저닝)

} 형상 관리  
 + 운영 자동화

} 구성 진단



“살아 남는 종(種)은 강한 종이 아니고,  
또 우수한 종도 아니다.  
변화에 적응하는 종이다.”

- *Charles Darwin, 1809*



감사합니다.



**KHAN**  
[ a p m ]

제품이나 서비스에 관한 문의

콜 센터 : 02-469-5426 ( 휴대폰 : 010-2243-3394 )

전자 메일 : [sales@opennaru.com](mailto:sales@opennaru.com)