

IDG Deep Dive

컨테이너 관리의 정석 쿠버네티스의 이해와 활용

‘쿠버네티스(Kubernetes)’가 거침없이 질주하고 있다. 쿠버네티스는 컨테이너 일정 관리부터 컨테이너 간 서비스 검색, 시스템의 부하 분산, 룰링 업데이트/롤백, 고가용성 등을 지원하는 오픈스택레이션 툴이다. 컨테이너 원천 기술을 가진 도커의 ‘스웜(Swarm)’을 가볍게 제압하고, 이제는 기업의 60%가 사용하는 사실상의 표준 컨테이너 툴이 됐다. 340억 달러, 우리 돈 38조 원에 달하는 IBM의 레드햇 인수도 그 이면에는 쿠버네티스가 자리 잡고 있다. 오늘날 기업 IT 인프라에서 쿠버네티스가 중요한 이유와 구축 방법을 살펴본다. 관리와 보안을 도와줄 유용한 툴과 주요 클라우드 업체의 쿠버네티스 서비스도 심층 분석한다.

※ Tech Trend

미안, 리눅스! 이제 주인공은 ‘쿠버네티스’야
“최신 1.12부터 구버전까지” 쿠버네티스 컨테이너 버전별 변천사

※ HowTo

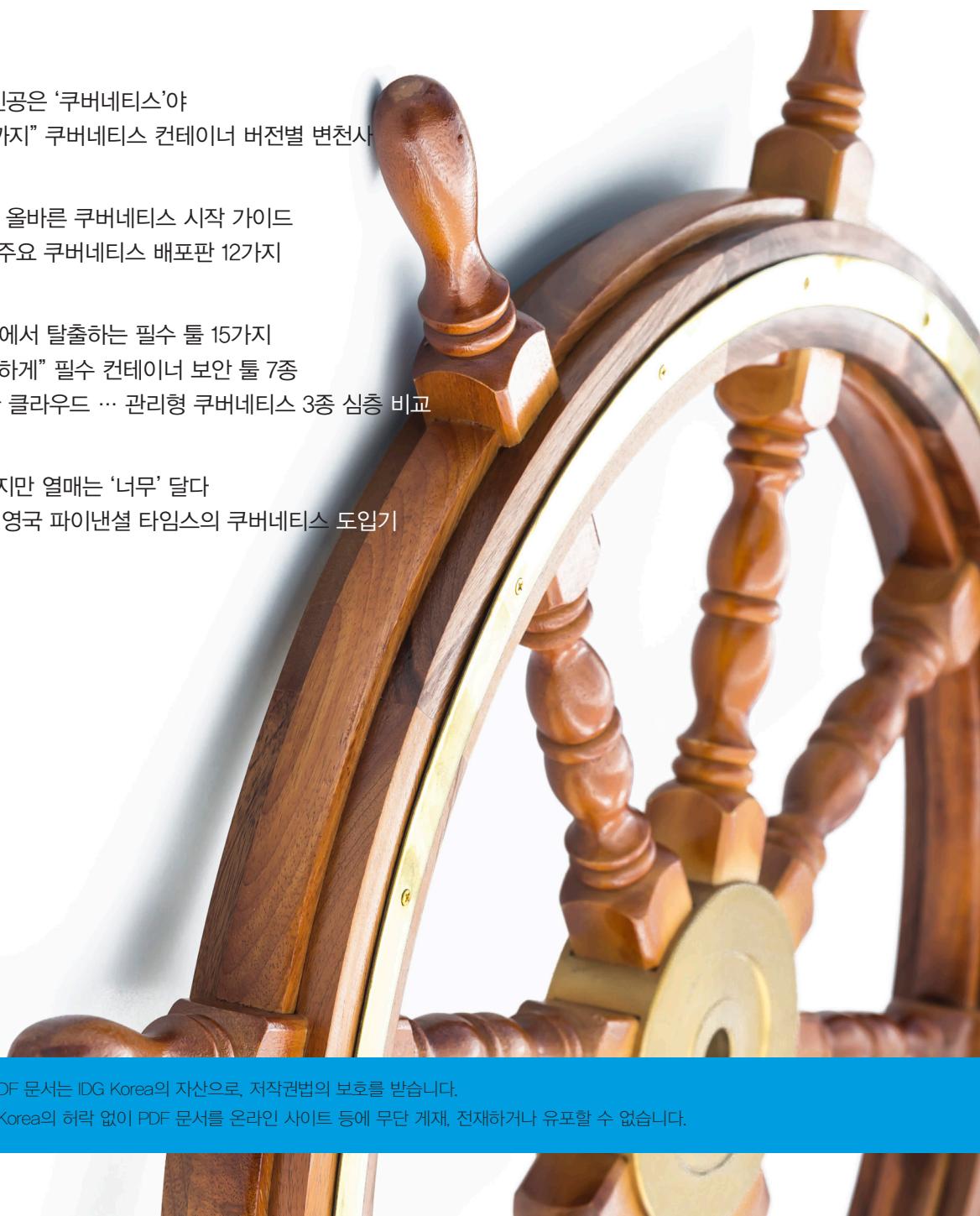
“배포판부터 예제까지” 올바른 쿠버네티스 시작 가이드
컨테이너 혁명 이끄는 주요 쿠버네티스 배포판 12가지

※ Tech Solution

쿠버네티스 ‘관리 지옥’에서 탈출하는 필수 툴 15가지
“쿠버네티스를 더 안전하게” 필수 컨테이너 보안 툴 7종
AWS vs. 애저 vs. 구글 클라우드 … 관리형 쿠버네티스 3종 심층 비교

※ Column

쿠버네티스, 고통은 쓰지만 열매는 ‘너무’ 달다
“서버 비용 80% 절감” 영국 파이낸셜 타임스의 쿠버네티스 도입기



무단 전재
재배포 금지

본 PDF 문서는 IDG Korea의 자산으로, 저작권법의 보호를 받습니다.

IDG Korea의 허락 없이 PDF 문서를 온라인 사이트 등에 무단 게재, 전재하거나 유포할 수 없습니다.

미안, 리눅스! 이제 주인공은 ‘쿠버네티스’야

Tech
Trend

Matt Asay | InfoWorld

운영체제의 시대가 가고 있다. 개발자나 클라우드에 있어 운영체제인 ‘리눅스’가 더는 중요하지 않게 된 것이다. 이런 주장의 근거는 (일어날 수도 있었지만) 결국 일어나지 않은 어떤 사건이다. IBM이 340억 달러를 제시하며 인수하겠다고 제안한 기업이 캐노니컬(Canonical)이 아니라 레드햇(Redhat)이었던 것이다. 캐노니컬은 유명 리눅스 배포판인 우분투(Ubuntu) 개발사다. 설립자인 마크 셔틀워스는 매각에 관심이 없다고 말해 왔지만 실제로 IBM이 340억 달러를 내밀었다면 어떻게 됐을까? 필자는 받아들였을 가능성이 크다고 생각한다.

결과적으로 현실에서는 인수 제안이 캐노니컬에 가지 않았고 당분간은 그럴 일이 없을 것이다. 이유는 앞서 언급한 것처럼 이제는 IT 산업이 운영체제 자체를 가치 있게 생각하지 않기 때문이다. 아니, IT 산업이 가치 있게 생각하는 ‘새로운 운영체제’가 있기 때문이라는 말이 더 정확할 것이다. 바로 쿠버네티스(Kubernetes)다.

우리는 지금 쿠버네티스의 세계에 살고 있다

레드몽크(RedMonk)의 애널리스트 스티븐 오그레디가 이를 절묘하게 표현했다.

“지금은 쿠버네티스의 세상이고 우리는 그 안에 살고 있다. 조금이라도 의심이 든다면 IBM의 행보를 떠올려보자. IBM은 무려 340억 달러를 들여 레드햇과 쿠버네티스 기반의 오픈시프트(OpenShift)를 인수했다. 그래도 쿠버네티스의 세상이라는 것을 의심하겠나?”

2018년 한 해 동안 오픈소스를 둘러싼 인수합병 금액은 600억 달러에 달했다. 대부분이 쿠버네티스와 관련된 것이었다. 물론 레드햇 리눅스의 가치를 과소평가할 수는 없다. 그 자체로 오랫동안 엔터프라이즈 리눅스의 표준으로 대접을 받았다. 그러나 IBM이 레드햇을 인수한 것은 엔터프라이즈 리눅스 때문이 아니다. 대신 IBM이 그렇게 많은 돈을 들여 손에 넣고 싶었던 것은 쿠버네티스로 구동하는 클라우드에 대한 실마리였다.

지난 몇 년 동안 레드햇의 쿠버네티스 기반 오픈시프트 서비스 매출은 계속해서 증가했다. 현재 레드햇 매출의 64%는 RHEL(Red Hat Enterprise Linux)이지만 이 사업의 성장률은 8%

에 불과하다. 반면 오픈시프트는 이보다 몇 배 빠르게 성장하고 있다. 특히 레드햇은 오픈시프트 구독 요금제에 RHEL을 추가했다. 이를 통해 RHEL 매출을 보호하고 오히려 더 성장시킨 것이다. 매우 영리하고 자연스러운 전략이다. 레드햇의 CEO 짐 화이트허스트는 최근 레드햇 실적 발표 행사에서 다음과 같이 말했다.

“많은 빅데이터 워크로드가 리눅스에서 실행된다. 인공지능(AI) 워크로드도, 데브옵스 같은 플랫폼도 예외 없이 리눅스를 기반으로 운영된다. 즉, 새로 만들어지는 워크로드 대부분이 리눅스와 연관된 것이다”

이 발언을 자세히 들여다보면 핵심은 리눅스가 아니다. 일종의 통로 역할을 할 뿐이다. 대신 진정한 운영체제 역할을 하는 것이 바로 쿠버네티스다

쿠버네티스는 새로운 엔터프라이즈 리눅스

레드햇의 대니얼 리크는 여기서 한 발 더 나갔다. 쿠버네티스는 새로운 엔터프라이즈 리눅스라고 선언했다.

“AWS, 마이크로소프트 애저, 구글 클라우드가 새로운 EMC, HP, 시스코, 씬, 오라클이라면, 쿠버네티스는 새로운 RHEL이다. RHEL은 수직적으로 결합된 메인프레임, 유닉스 시스템에 대한 오픈소스 대안을 제공하고, 기업에 오픈소스 생태계와 하이브리드 인프라를 선택할 수 있는 자유를 제공했다. 마찬가지로 새로운 RHEL인 쿠버네티스는 수직적으로 결합된 클라우드에 대한 대안을 제공하고, 기업에 클라우드에 대한 선택권을 넓혀준다. 이러한 맥락에서 쿠버네티스는 새로운 RHEL, 새로운 운영체제다.”

이제 IBM이 캐노니컬이 아니라 레드햇을 인수한 이유도 더 명확해진다. 클라우드 마켓(Cloud Market)의 자료를 보면, 우분투는 AWS에서 가장 인기 있는 운영체제 인스턴스다. 그

러나 이것은 클라우드에 대한 선택권 측면에서 큰 의미가 없다. 만약 이것이 의미가 있었다면 캐노니컬이 레드햇보다 수십 배는 더 가치가 있었을 것이다. AWS 측면에서 봐도 리눅스 이미지를 지원하는 데 따른 매출은 그렇게 크지 않다. 오히려 개발자에게 더 높은 가치를 만들어내는 가상화와 컨테이너 주도 서비스에서 훨씬 큰 매출이 나온다.

결국 업계는 새로운 운영체제, 쿠버네티스로 눈을 돌리고 있다. 물론 낡은 리눅스나 심지어 오픈스택으로 구축된 낡은 클라우드를 지원하는 것은 여전히 돈이 된다. 그러나 쿠버네티스가 수천



억 달러 가치를 가진 마당에 이런 '푼돈'에 집착하는 것은 빠른 기술 흐름에 도태되기 딱 좋은 태도다.

쿠버네티스 역시 언젠가는 사라진다

쿠버네티스의 공동 창시자인 브랜든 번즈는 리눅스가 점점 더 그 중요성을 잃어갈 것으로 전망한다.

"쿠버네티스와 쿠버네티스 API는 일종의 포식스(Posix, 유닉스에 기반을 둔 표준 운영체제 인터페이스)와 같다. 리눅스 시스템에서 실행되는 모든 프로그램은 포식스 API를 거쳐 실행되지만 사람들은 이를 크게 신경 쓰지 않았다. 운영체제 안에서 아들을 익혔고, 포식스 스레드(POSIX Thread) 같은 것도 사용했지만 깊게 생각하지 않았던 것이다. 쿠버네티스 역시 이렇게 발전할 것이다. 즉 배후로 스며들고 궁극적으로 사라질 것이다. 쿠버네티스 그 자체는 중요하고 유용하며 우리가 작업하는 모든 것의 중추지만, 정작 우리는 다른 고차원적 추상화에 집중하느라 이를 크게 생각하지 않는 것이다"

정리하면 쿠버네티스는 일종의 운영체제가 될 것이다. 한때 모든 관심의 중심이었던 그 리눅스처럼 말이다. **ITWORLD**



IT 트렌드 종합 정보센터 IDG Tech Library

IDG Tech Library는 IDG 글로벌 네트워크를 통해 축적된 전문 정보를 재구성하여 최신 기술의 기본 개념부터 현황, 전략 및 도입 가이드까지 다양한 프리미엄 IT 정보를 제공합니다. Computer World, Info World, CIO, Network World 등의 세계적 IT 유명 매체의 심도 깊은 정보를 무료로 만나보세요

IDG Deep Dive, Tech Focus, Summary, World Update 등의 다양한 콘텐츠를 제공 받을 수 있습니다.



“최신 1.12부터 구버전까지” 쿠버네티스 컨테이너 버전별 변천사

Tech
Trend

Serdar Yegulalp | InfoWorld

컨테이너 오케스트레이션 시스템 쿠버네티스의 최신 버전은 안정화 버전(GA) 기준 1.12다. 큐블릿(Kubelets)을 위한 TLS 클라이언트 인증서 프로비저닝을 자동화하는 큐블릿 TLS 부트스트랩이 추가됐다. 또한 마이크로소프트 애저 가상머신 확장 집합에서 컨테이너 클러스터 자동 확장도 지원한다. 쿠버네티스 소스는 쿠버네티스 공식 깃허브 리포지토리의 릴리스 페이지(<https://github.com/kubernetes/kubernetes/releases>)에서 다운로드할 수 있다. 쿠버네티스 배포판을 공급하는 다양한 업체가 제공하는 업그레이드 방식을 통해서도 내려 받을 수 있다.

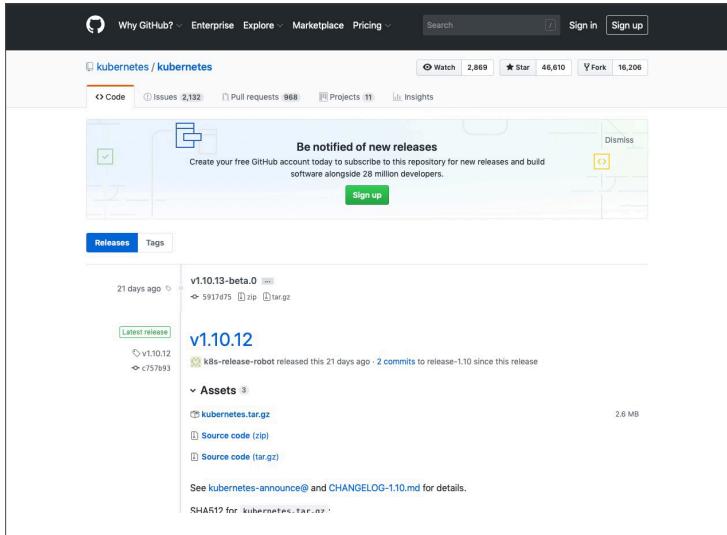
현재 버전: 쿠버네티스 1.12의 새로운 기능

2018년 9월 말에 출시된 쿠버네티스 1.12는 일반 안정화 버전에 큐블릿 TLS 부트스트랩이 추가됐다. 큐블릿 TLS 부트스트랩은 큐블릿 또는 쿠버네티스 노드에서 실행되는 주 에이전트가 API를 통해 TLS 클라이언트 인증서를 요청한다. 이를 통해 TLS로 보호되는 클러스터에 자동으로 통합할 수 있다. 이 프로세스가 자동화되면 기본적으로 더 높은 보안 수준에서 클러스터를 구성할 수 있다. 또한, 쿠버네티스 1.12는 마이크로소프트 애저의 가상머신 확장 집합(VMSS)을 지원한다. VMSS를 이용하면 일정에 따라 또는 수요에 맞춰 자동으로 확장/축소되는 VM 그룹을 설정할 수 있다.

이밖에 쿠버네티스 1.12의 새로운 기능은 다음과 같다.

- **볼륨 스냅샷 및 복원 기능(알파)**
- **팟 자동 확장을 위한 맞춤 메트릭(베타).** 팟을 확장할 때 맞춤 조건 또는 다른 기준을 사용할 수 있다. 예를 들어 애플리케이션 관리 전략의 일부로 특정 쿠버네티스 배포 전용 리소스를 추적해야 하는 경우에 이 기능이 유용하다.
- **수직 팟 확장(베타).** 폐기 비용이 비싼 팟의 관리 효율성을 높이기 위해 팟의 리소스 제한이 팟 수명 주기에 따라 바꿔도록 설정하는 기능이다. 기존 스케줄링에서는 관리하기가 쉽지 않았던 팟을 전략적으로 처리할 수 있어, 그동안 쿠버네티스 사용자의 기능 추가 단골 요구 중 하나였다.

화면 | 쿠버네티스의 깃허브 리포지토리



한 '상태'와 '범위'를 정의하는 새로운 방법인 서브리소스(Subresources, <https://github.com/kubernetes/community/blob/master/contributors/design-proposals/api-machinery/customresources-subresources.md>)를 이용하면 한 클러스터 안에서 모니터링과 고가용성 프레임워크를 통합할 수 있다.

이밖에 쿠버네티스 1.11의 다른 새로운 기능은 다음과 같다.

- 1.10부터 도입된 코어DNS(CoreDNS)는 이제 클러스터 DNS 부가기능으로 사용할 수 있으며 kubeadm 관리 툴에 기본으로 적용됐다.
- 쿠빌렛(Kubelet) 구성 변경사항은 이제 클러스터를 다운시키지 않고도 실시간 클러스터에서 실행할 수 있다.
- CSI(Container Storage Interface)는 로우 블록 볼륨을 지원하며 쿠빌렛 플러그인 등록 시스템과 통합된다.
- 영구 볼륨의 크기 조정, 노드의 최대 볼륨 카운트 지정, 사용 중인 저장소 객체가 제거되지 않도록 보호하는 기능 등 저장소 관련해 많은 부분이 개선됐다.

이전 버전: 쿠버네티스 1.10의 새로운 점

쿠버네티스 1.10 프로덕션 버전은 2018년 3월에 공개됐다. 그전까지는 쿠버네티스 바이너리(Binary)를 재컴파일해야 했지만 이제는 CSI(쿠버네티스 1.9 당시 알파, <http://blog.kubernetes.io/2018/01/introducing-container-storage-interface.html>)가 포함돼 쿠버네티스에 볼륨 플러그인을 쉽게 추가할 수 있게 됐다. 쿠버네티스에서 공통 유지보수와 관리 작업에 사용하던 Kubectl CLI는 이제 클라우드 업체와 액티브 딜렉터리 등 서드파티 서비스 인증을 담당(<https://github.com/kubernetes/features/issues/541>)하는 바이너리 플러그인을 지원한다.

이전 버전: 쿠버네티스 1.11의 새 기능

2018년 7월 초에 공개된 쿠버네티스 1.11에는 IPVS(IP Virtual Server, <https://github.com/kubernetes/features/issues/265>)가 추가됐다. iptables 시스템이 사용하는 것보다 덜 복잡한 인커널(In-kernel) 기술을 이용해 고성능 클러스터 로드밸런싱을 제공한다. 쿠버네티스는 결국 IPVS를 기본 로드밸런서(Load Balancer)로 사용할 것으로 보이는데, 현재 까지는 옵션이다.

커스텀 리소스 정의는 표준화를 준수하면서 쿠버네티스에 사용자 정의 구성 변경사항을 적용하는 방법이다. 점점 기능이 발전해 이제는 커스텀 리소스 세트를 다른 세트로 쉽게 전환할 수 있게 됐다. 또

'비공유 저장소(Non-shared storage, <https://github.com/kubernetes/features/issues/121>)' 또는 로컬 저장소 볼륨을 상시 쿠버네티스 볼륨으로 마운트 할 수 있는 기능도 베타가 됐다. 이제 상시 볼륨 API는 사용 중인 상시 볼륨이 삭제되지 않도록 추가 확인을 수행한다. 쿠버네티스의 네이티브 DNS 제공자는 모듈식 아키텍처를 가진 CNCF 관리형 DNS 프로젝트인 코어DNS(<https://kubernetes.io/docs/tasks/administer-cluster/coredns>)로 대체할 수 있다. 단, 쿠버네티스 클러스터가 첫 번째 구성인 경우에만 가능하다.

이전 버전: 쿠버네티스 1.9의 새로운 점

쿠버네티스 1.9는 2017년 12월에 공개됐다. 가장 눈에 띄는 것은 프로덕션 릴리즈 상태인 앱스 워크로드(Apps Workloads) API다. 이를 이용하면 상시 상태가 필요한 앱의 작업 부하를 정의할 수 있다. 앱스 워크로드 API는 총 4개다.

- **Deployment.** ReplicaSet을 포함해 작동 중인 애플리케이션에 대해 원하는 상태를 설명하는 기본적인 수단이다.
- **ReplicaSet.** Deployment의 앱 정의에 충족하는 실행 컨테이너 인스턴스('replicas')가 있는지 확인한다.
- **Daemonset.** 로깅 또는 모니터링 솔루션처럼 다른 앱이 실행하는 대상에 상관없이 연속적으로 작동하는 앱을 배포한다.
- **StatefulSet.** 컨테이너를 다시 시작하더라도 상시 상태가 필요한 작업 부하에 사용된다. 컨테이너용 네트워크 식별 또는 컨테이너가 시작하고 정지하는 순서 등을 정할 때도 쓸 수 있다.



마이크로소프트가 윈도우에 도커(Docker) 컨테이너 지원을 추가한 후 쿠버네티스처럼 도커를 사용하던 다른 앱도 이를 따르고 있다. 쿠버네티스 1.9는 현재 임시로 윈도우 서버에서 쿠버네티스를 사용(<https://kubernetes.io/docs/getting-started-guides/windows>)할 수 있도록 지원한다. 윈도우 서버에서 쿠버네티스를 테스트하려면 윈도우 서버 2016과 도커 1.12가 필요하다. 단, 쿠버네티스 제어 영역은 리눅스에서만 작동한다. 즉, 리눅스 컨트롤러에서 컨테이너가 윈도우 서버에서 작동하도록 설정하는 것은 가능하지만, 리눅스 대신 윈도우 서버를 컨트롤러로 사용할 수는 없다.

쿠버네티스의 가장 중요한 기능은 개발 초기부터 명확했다. 바로 애플리케이션으로부터 리소스를 추상화하는 것이다. 그러나 안타깝게도 컨테이너 저장소는 제대로 된 표준이 없었고, 쿠버네티스를 포함한 거의 모든 컨테이너 솔루션이 이를 자체 방식으로 처리했다.

이에 따라 CNCF(Cloud Native Computing Foundation)

의 하위 그룹인 CNCF SWG(Storage Working Group, <https://github.com/cncf/wg-storage>)가 컨테이너 클러스터 저장소용 표준인 CSI 표준을 만들었다. 쿠버네티스 1.9에는 CSI 플러그인의 알파 버전이 포함돼 있어 저장소 볼륨 플러그인을 쿠버네티스 자체와 완전히 독립적으로 개발(<https://github.com/kubernetes/features/issues/178>)할 수 있다. 이 프로젝트는 여전히 초기 단계이지만 방향이 적절하다는 것은 두말할 나위가 없다.

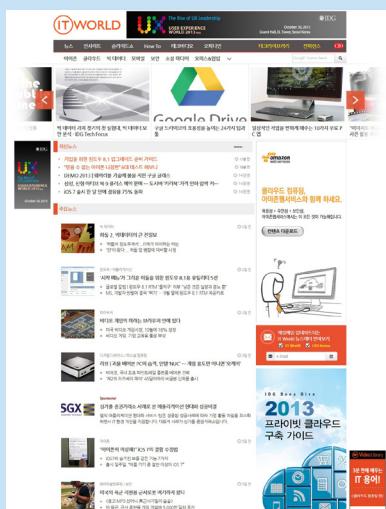
이밖에 쿠버네티스 1.9의 새 기능은 다음과 같다.

- **알파 버전의 하드웨어 가속.** 이를 이용하면 GPU를 하나의 리소스로 사용할 수 있고 머신러닝(ML) 작업 부하를 더 잘 관리할 수 있다.
- **IPv6 주소 지정 알파 지원**
- **더 신속한 CRD(Custom Resource Definition) 데이터 검증**(<https://kubernetes.io/docs/concepts/api-extension/custom-resources>). CRD를 이용하면 새 버전의 쿠버네티스가 나와도 호환성을 해치지 않으면서 설치할 수 있다. 사용자 정의도 기존의 것 그대로 적용할 수 있다.

ITWORLD



테크놀로지 및 비즈니스 의사 결정을 위한 최적의 미디어 파트너



기업 IT 책임자를 위한 글로벌 IT 트렌드와 깊이 있는 정보

ITWorld의 주 독자층인 기업 IT 책임자들이 원하는 정보는 보다 효과적으로 IT 환경을 구축하고 IT 서비스를 제공하여 기업의 비즈니스 경쟁력을 높일 수 있는 실질적인 정보입니다.

ITWorld는 단편적인 뉴스를 전달하는 데 그치지 않고 업계 전문가들의 분석과 실제 사용자들의 평가를 기반으로 한 깊이 있는 정보를 전달하는 데 주력하고 있습니다. 이를 위해 다양한 설문조사와 사례 분석을 진행하고 있으며, 실무에 활용할 수 있고 자료로서의 가치가 있는 내용과 형식을 지향하고 있습니다.

특히 IDG의 글로벌 네트워크를 통해 확보된 방대한 정보와 전세계 IT 리더들의 경험 및 의견을 통해 글로벌 IT의 표준 패러다임을 제시하고자 합니다.

“배포판부터 예제까지” 올바른 쿠버네티스 시작 가이드



HowTo

Serdar Yegulalp | InfoWorld

도 든 혁신에는 항상 새로운 문제가 뒤따르기 마련이다. 컨테이너(container)도 마찬가지다. 이를 이용하면 편리하고 이식 가능한 형태로 애플리케이션을 패키징할 수 있지만 대규모 컨테이너를 관리하는 일은 좋게 말해 ‘매우’ 까다롭다.

이 문제를 해결하기 위해 구글은 쿠버네티스(Kubernetes)를 만들었다. 전체 클러스터에서 컨테이너를 관리하는 단일 프레임워크를 제공한다. 쿠버네티스의 기능은 보통 ‘오퍼스트레이션(orchestration)’이라는 용어로 표현하는데, 여기에는 컨테이너 일정 관리, 컨테이너 간 서비스 검색, 시스템 전반의 부하 분산, 롤링 업데이트/롤백, 고가용성 등이 포함된다. 여기서는 간단한 쿠버네티스 실행 예제를 통해 쿠버네티스가 어떻게 동작하는지 살펴보자.

쿠버네티스 배포판 사용하기

쿠버네티스는 리눅스 컨테이너 관리를 위해 태어났다. 쿠버네티스 1.5부터는 윈도우 서버 컨테이너도 지원하지만 쿠버네티스 제어 영역은 여전히 리눅스에서 실행해야 한다. 물론 가상화 덕분에 모든 플랫폼에서 쿠버네티스를 사용할 수 있다.

자체 하드웨어 또는 가상머신(VM)에서 쿠버네티스를 실행할 때, 가장 일반적인 방법은 패키징된 쿠버네티스 배포판을 다운로드하는 것이다. 배포판에는 일반적으로 업스트림 쿠버네티스 비트와 완전한 배포에 필요한 다른 요소, 예를 들면 컨테이너 레지스트리, 네트워킹, 스토리지, 보안, 로깅, 모니터링, 지속 통합 파이프라인 등이 들어있다. 쿠버네티스 배포판 대부분은 아마존 EC2(Amazon EC2), 애저 버추얼 머신(Azure Virtual Machines), 구글 컴퓨터 엔진(Google Compute Engine), 오픈스택(OpenStack) 등 모든 가상머신 인프라에 설치해 사용할 수 있다.

쿠버네티스 배포판은 수십 가지다. 캐노니컬 쿠버네티스(Canonical Kubernetes), 클라우드 파운드리 컨테이너 런타임(Cloud Foundry Container Runtime), 메소스피어 쿠버네티스 서비스(Mesosphere Kubernetes Service), 오라클 리눅스 컨테이너 서비스(Oracle Linux Container Services), 피보탈 컨테이너 서비스(Pivotal Container Service), 랜처(Rancher), 레드햇 오픈시프트(Red Hat OpenShift), 수세 CaaS 플랫폼(Suse CaaS Platform) 등이다. 캐노니컬, 레드햇, 수세 배포판은 쿠버네티스와 리눅스 배포판을 함께 제공하므로 운영체제에서 쿠버네

티스를 설정하는 번거로움이 없다. 다운로드하고 설치하는 프로세스뿐만 아니라 구성, 관리 작업도 일부 줄일 수 있다.

다른 방법은 일반적인 리눅스 배포판에서 쿠버네티스를 실행하는 방법이다. 단, 관리 부담이 크고 손이 많이 간다. 예를 들어 레드햇 엔터프라이즈 리눅스의 패키지 리포지토리에는 쿠버네티스가 포함돼 있지만, 레드햇조차 이 쿠버네티스를 테스트 용도로만 사용할 것을 권장한다. 현재 오픈시프트는 자체 네이티브 오케스트레이터로 쿠버네티스를 사용하므로, 레드햇 사용자라면 오픈시프트 PaaS로 쿠버네티스를 쓰는 것 이 더 합리적이다.

화면 | 캐노니컬의 MicroK8s

The screenshot shows the official MicroK8s website. At the top, there's a navigation bar with links for CANONICAL, MicroK8s (highlighted in orange), Docs, Discourse, Code, and Bugs. Below the header, the main title is "Kubernetes for workstations and appliances". A subtext below it says "A single package of k8s that installs on 44+ flavours of Linux. Made for developers and great for appliances." To the right, there's a "certified" badge with a Kubernetes logo. On the left, there's a "Get it from the Snap Store" button. At the bottom, there's a "FEATURES" section with several items listed.

FEATURES					
Istio	Local storage	Dashboard	Ingress		
GPGPU bindings	Local registry	Metrics	DNS		
Daily builds	Updates	Upgrades	Conformance		

많은 기존 리눅스 배포판이 쿠버네티스와 다른 소프트웨어 스택을 설정하는 별도의 툴을 제공한다. 예를 들어 우분투는 클라우드 인스턴스와 베어메탈 인스턴스에서 모두 쿠버네티스 업스트림 버전을 배포할 수 있는 컨저-업(conjure-up, <https://kubernetes.io/docs/getting-started-guides/ubuntu>)을 지원한다. 캐노니컬도 스냅(Snap) 패키지 시스템 (<https://snapcraft.io>)을 통해 설치하는 쿠버네티스 버전인 MicroK8s(<https://microk8s.io>)를 제공한다.

컨테이너 혁명 이끄는 주요 쿠버네티스 배포판 12가지

Serdar Yegulalp | InfoWorld

쿠버네티스(Kubernetes)는 대규모 컨테이너 오케스트레이션(orchestration)이 필요한 상황이라면 최우선으로 검토해야 할 프로젝트다. 그러나 설치와 구성이 까다롭기로 유명하므로, 혼자서 하기보다는 쿠버네티스를 포함된 통합 컨테이너 솔루션을 이용하는 것이 좋다. 현재 가장 널리 사용하는 통합 쿠버네티스 솔루션 12가지를 정리했다.

코어OS 텍토닉

코어OS(CoreOS)는 컨테이너 중심의 리눅스 배포판과 ‘엔터프라이즈급 쿠버네티스’ 배포판을 제공하는데, 이 둘을 합친 것이 코어OS 텍토닉(Tectonic, <https://coreos.com/tectonic>)이다. 특히 코어OS 운영체제인 컨테이너 리눅스(Container Linux)가 컨테이너화된 구성 요소의 모음으로 제공되므로, 실행 중인 애플리케이션을 중단하지 않고도 OS에 대한 자동 업데이트가 가능하다. 단 1번 클릭으로 쿠버

화면 | 코어OS 텍토닉

The screenshot shows the CoreOS Tectonic website. The main headline is "We're integrating Tectonic with Red Hat OpenShift". Below it, a subtext says "We're bringing the best of Tectonic to Red Hat OpenShift to build the most secure, hybrid Kubernetes application platform." At the bottom, there are two sections: "Simplify Kubernetes deployments" and "Provide a common application platform across cloud providers".

네이티스도 업데이트할 수 있다. 코어OS 텍토닉은 아마존 웹 서비스(AWS), 마이크로소프트 애저, 베어 메탈에서 사용할 수 있다.

캐노니컬 쿠버네티스 배포판

우분투 리눅스 개발사 캐노니컬(Canonical)은 자체 쿠버네티스 배포판(<https://www.ubuntu.com/kubernetes>)을 제공한다. 이 배포판의 장점은 그 바탕에 우분투 리눅스 배포판이 있다는 것이다. 캐노니컬은 자사의 솔루션이 모든 클라우드와 온프레미스 환경에서 작동한다고 설명한다. 유료 고객은 캐노니컬 엔지니어가 제공하는 원격 관리 서비스(<https://www.ubuntu.com/kubernetes/managed>)를 받을 수 있다.

도커 커뮤니티 에디션/도커 엔터프라이즈

도쿄야말로 진정한 컨테이너라고 생각하는 이가 많을 것이다. 2014년 이래 도커는 자체 클러스터링과 오케스트레이션 시스템 ‘도커 스웜(Docker Swarm)’을 개발해왔고 이는 최근까지 쿠버네티스의 대항마였다. 그러나 2017년 10월 도커는 전격적으로 쿠버네티스 지원을 발표했다. 즉, 도커 커뮤니티 에디션(Docker Community Edition, <https://www.docker.com/kubernetes>)과 도커 엔터프라이즈(Docker Enterprise, <https://www.docker.com/kubernetes>) 모두에 쿠버네티스를 추가하겠다는 것이다. 간단히 말해 스웜보다 쿠버네티스가 더 뛰어나다는 것을 인정한 셈이다. 그러나 도커에는 여전히 ‘스웜 모드’가 포함돼 있다. 방화벽 뒤의 로컬 애플리케이션 등 특정 목적에는 스웜 모드가 더 유용하다.

헬피오 쿠버네티스 서브스크립션

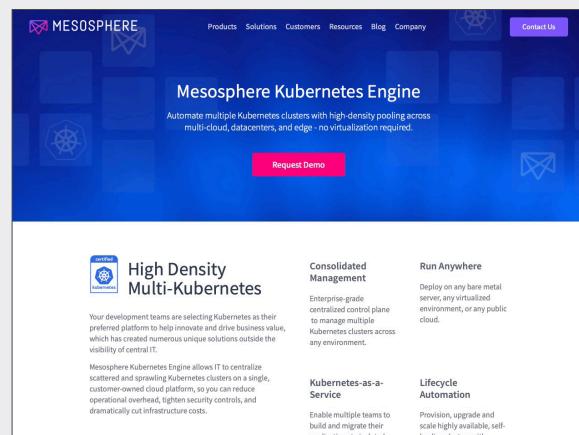
쿠버네티스를 처음 만든 크레이그 맥루키, 조 베다가 창업한 기업 헬피오(Heptio)의 첫 작품으로, 서비스 이름은 헬피오 쿠버네티스 서브스크립션(Heptio Kubernetes Subscription, <https://heptio.com/products/kubernetes-subscription>)이다. 24시간 헬피오의 유료 지원이 제공되며, 월 2,000달러에서 시작한다. 헬피오의 장점은 업체에 구속되지 않는 엔터프라이즈급 쿠버네티스라는 것이다. 퍼블릭 클라우드나 온프레미스 하드웨어에서 실행할 수 있다.

메소피어 DC/OS

메소피어(Mesosphere, <https://mesosphere.com/solutions/kubernetes>) DC/OS는 아파치 메소(Apache Meso)를 이용해 클러스터를 단일 리소스로 바꾼 후 다시 동적으로 분할해 복수의 애플리케이션이 사용할 수 있도록 지원하는

데, 이 DC/OS 상의 애플리케이션 패키지 중 쿠버네티스가 포함(<https://github.com/mesosphere/dcos-kubernetes-quickstart>)돼 있다. 이를 이용하면 DC/OS 클러스터에서 쿠버네티스를 설치, 실행, 업데이트할 수 있다. DC/OS를 쿠버네티스 배포판으로 볼 수 있는지는 논란의 여지가 있다. 그러나 메소피어의 작동방식은 철저하게 쿠버네티스의 그것을 따른다. 기존 쿠버네티스 툴과의 호환성도 매우 높다.

화면 | 메소피어 DC/OS



미란티스 클라우드 플랫폼

미란티스(Mirantis) 클라우드 플랫폼(<https://www.mirantis.com/software/mcp>)은 오픈스택, 쿠버네티스 또는 이 둘을 일명 ‘애자일 인프라 플랫폼’으로 통합한다. 간단히 말하면, 가상머신과 컨테이너, 베어 메탈 서버의 오케스트레이션을 위한 단일 통합 솔루션이다. 플랫폼에 배치된 애플리케이션은 ‘데브옵스 스타일’로 수명주기에 걸쳐 관리할 수 있다. 이 때 구성 관리 툴은 솔트(Salt)를 사용하며, 애플리케이션이 정확하게 배치될 수 있도록 통합 CI/CD를 지원한다. 미란티스 클라우드 플랫폼을 이용하면 베어메탈과 오픈스택 클러스터, 퍼블릭 클라우드 등에서 직접 쿠버네티스를 실행할 수 있다.

플랫폼9 매니지드 쿠버네티스

쿠버네티스 배포판 대부분은 쿠버네티스를 횡단, 종단으로 관리하는 것에 초점을 맞추고 있다. 반면 플랫폼9 매니지드 쿠버네티스(Platform9 Managed Kubernetes, <https://platform9.com/managed-kubernetes>)는 로컬 베어메탈, 원격 퍼블릭 클라우드 등 다양한 환경에서 실행하는 것을 목

표로 한다. 플랫폼9의 엔지니어가 서비스 형태로 원격 관리하는 방식도 있어서, 매니지드 쿠버네티스에 대한 업데이트를 6주 정도에 한 번씩 배포한다. 이 서비스에는 플랫폼9의 피션(Fission, <https://platform9.com/fission>) 프로젝트가 통합돼 있다. 이 프로젝트는 서비스 컴퓨트(serverless compute) 또는 서비스로서의 함수(FaaS) 시스템이라고도 하는데, 대부분의 프로그래밍 언어와 컨테이너화된 런타임을 지원한다.

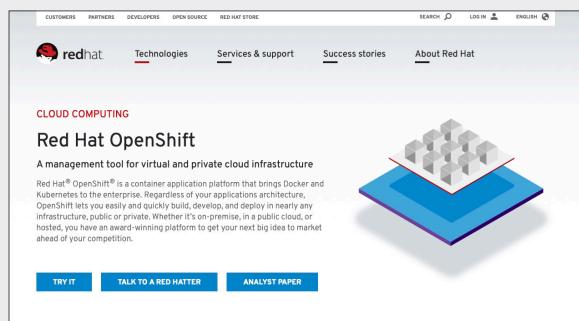
랜처 2.0

랜처 랩스는 쿠버네티스를 컨테이너 관리 플랫폼 내부에 통합했는데, 이를 간단히 랜처(rancher, <https://rancher.com/products/rancher>)라고 부른다. 랜처 2.0은 다른 쿠버네티스 배포판에 비해 높은 수준에서 작동한다. 즉, 리눅스 호스트, 도커 컨테이너, 쿠버네티스 노드 위에 자리 잡고 위치나 인프라와 관계없이 통합 관리할 수 있다. 심지어 쿠버네티스 클라우드 서비스에서 쿠버네티스 클러스터를 관리하는 것도 가능하다. 랜처는 쿠버네티스 클러스터를 설치하고 쿠버네티스를 특정 환경에 맞추는 번거로운 과정을 상당 부분 없애면서도, 이러한 맞춤 설정이 쿠버네티스의 업그레이드에 방해가 되지 않도록 설계됐다. 빠르게 개발하고 끊임없이 업데이트하는 인프라를 관리해야 한다면 눈여겨볼 만하다.

레드햇 오픈시프트

레드햇 오픈시프트(Red Hat OpenShift, <https://www.redhat.com/en/technologies/cloud-computing/openshift>)는 레드햇의 PaaS 서비스다. 본래 레드햇은 애플리케이션을 패키지하는데 헤로쿠(Heroku) 빌드팩(buildpack) 같은 ‘카트리지(cartridge)’를 사용했지만, 도커가 등장하면서 이 새로운 컨

화면 | 레드햇 오픈시프트



테이너 이미지와 런타임 표준을 활용하기 위해 오픈소스를 완전히 뜯어고쳤다. 이 과정에서 쿠버네티스를 표준 오픈스택레이션 기술로 채택했다. 오픈소스는 PaaS의 모든 요소에 추상화와 자동화를 제공한다. 쿠버네티스 관리는 여전히 쉽지 않지만 오픈소스를 이용하면 작업 상당 부분을 덜어낼 수 있다.

스택큐브

스택큐브(Stackube, <https://github.com/openstack/stackube>)는 쿠버네티스 중심의 오픈스택(OpenStack) 배포판이다. 보통 오픈스택은 컴퓨터 노드의 프로비저닝과 관리를 위해 노바(Nova)를 사용하지만 스택큐브는 노바 대신 쿠버네티스를 쓴다. 스택큐브의 장점은 사용하는 컨테이너 런타임에 따라 다양한 멀티테넌시를 제공하는 것이다. ‘소프트’ 한 멀티테넌시를 위해서라면 도커를, 더 강력한 리소스 분배가 필요하다면 하이파바이저(hypervisor)급 하이퍼컨테이너(HyperContainer)를 선택할 수 있다.

서비스 플랫폼으로서의 수세 클라우드

유럽에서 큰 인기인 리눅스 배포판 수세(SUSE)도 수세 CaaS(SUSE CaaS, <https://www.suse.com/products/caas-platform>) 플랫폼을 제공한다. 개념적으로는 코어OS 텍토닉과 비슷하다. 컨테이너를 실행하는 베어메탈 ‘마이크로’ OS, 컨테이너 오픈스택레이션 시스템인 쿠버네티스, 내장 이미지 레지스트리, 클러스터 구성 툴 등을 포함한다. 수세 CaaS 플랫폼은 로컬 베어메탈은 물론 퍼블릭 클라우드에서 사용할 수 있으며, 복수의 클라우드와 데이터센터에 걸쳐 컨테이너를 실행하는 것도 가능하다.

텔레큐브

텔레큐브(Telekube, <https://gravitational.com/telekube>)는 텔레포트(Teleport) SSH 서버를 만든 그레이비테이셔널(Gravitational)이 개발했다. 로컬 또는 원격 클러스터에서 실행하는 실제 업무용 시스템에 적합한 쿠버네티스 배포판이다. 사설 SaaS 플랫폼 역할을 하거나 쿠버네티스를 여러 지역 또는 호스팅 업체를 통해 서비스로 실행할 때 유용하다. 단, 텔레큐브 상의 애플리케이션은 ‘번들’로 패키지화한 후 쿠버네티스 클러스터로 배포해 쿠버네티스 상의 컨테이너에서 실행할 수 있도록 미리 작업해야 한다.

쿠버네티스 클라우드 서비스 사용하기

쿠버네티스는 다른 클라우드에서 표준 서비스로 사용할 수 있지만, 구글 클라우드 플랫폼(Google Cloud Platform, GCP)에서는 네이티브 기능으로 지원한다. GCP에서 쿠버네티스를 실행하는 방법은 2가지다. 가장 편리한 것은 쿠버네티스 명령줄 터미널을 관리하는 구글 쿠버네티스 엔진(<https://cloud.google.com/kubernetes-engine/docs/quickstart>)을 사용하는 것이다.

또는, 구글 컴퓨터 엔진을 이용해 수동으로 컴퓨팅 클러스터를 설정하고 쿠버네티스를 배포하는 방법이 있다. 더 많은 수작업이 필요하지만 구글 쿠버네티스 엔진에서는 불가능한 맞춤 설정이 가능하다. 컨테이너를 처음 접한다면 컨테이너 엔진을 추천한다. 쿠버네티스에 어느 정도 익숙해지고 쿠버네티스 맞춤 버전이나 자체 수정판 등 더 고급 기능을 사용하고 싶을 때 두 번째 방법을 쓸 것을 권한다.

아마존의 경우 본래 EC2에 컴퓨팅 클러스터를 배포하려면 쿠버네티스를 실행해야 했다. 지금도 옵션으로 이 방법을 사용할 수 있다. 그러나 현재는 쿠버네티스용 엘라스틱 컨테이너 서비스(Elastic Container Service for Kubernetes, EKS, <https://aws.amazon.com/eks>)를 별도로 제공한다. EKS에서는 아마존이 제어부를 관장하므로, 사용자는 원하는 구성으로 사용할 컨테이너 배포에만 집중할 수 있다. 또한 EKS는 쿠버네티스의 표준 업스트림 에디션을 사용한다. 그래서 유용한 기능 중 한 가지가 쿠버네티스와 나머지 AWS 서비스 간의 통합이다. AWS 서비스는 EKS에서 쿠버네티스 네이티브 맞춤 리소스 정의(Custom Resource Definitions)로 표시되며, AWS 또는 쿠버네티스의 어느 부분을 변경해도 연결이 끊기지 않는다.

많은 쿠버네티스 배포판이 AWS를 비롯한 여러 환경을 설정하는 세부 가이드를 함께 제공한다. 예를 들어 레드햇 오픈시프트는 대화형 설치 프로그램이나 스크립트(https://docs.openshift.com/enterprise/3.0/install_config/install/quick_install.html) 또는 코드로서의 인프라(infrastructure-as-code) 프로비저닝 툴 테라폼(Terraform, <https://www.terraform.io>)을 사용해 하나 또는 여러 개의 호스트에 쿠버네티스를 설치할 수 있다. 또는 쿠버네티스 캡스(Kops, <https://kubernetes.io/docs/getting-started-guides/kops>) 툴을 사용하면 구글 클라우드 엔진, VM웨어 v스피어는 물론 다른 클라우드를 지원하도록 일반 VM의 클러스터를 프로비저닝 할 수 있다.

マイクロソフト 애저는 애저 쿠버네티스 서비스(<https://docs.microsoft.com/en-us/azure/aks/intro-kubernetes>)를 통해 쿠버네티스를 지원한다. 애저가 쿠버네티스 마스터 노드를 관리하고, 사용자는 리소스 매니저(Resource Manager) 템플릿 또는 테라폼을 통해 클러스터를 만들 수 있다. 마스터 노드와 에이전트 노드를 모두 제어하고 싶다면, 애저 가상 머신에 쿠버네티스 배포판을 설치해야 한다. 이처럼 AKS를 사용하면 비용 측면에서 장점이 있다. 마스터 노드에는 비용이 청구되지 않으므로 에이전트에 대한 비용만 내면 되기 때문이다.

클라우드든 아니든 다양한 환경에서 기본 쿠버네티스 클러스터를 신속하게 프로비저닝하고 싶다면 쿠버네티스 애니웨어(Kubernetes Anywhere, <https://github.com/kubernetes/kubernetes-anywhere>) 프로젝트를 눈여겨볼 필요가 있다. 이 스크립트는 구글 컴퓨터 엔진,

마이크로소프트 애저, VM웨어 v스피어(v센터 필요), 오픈스택에서 작동하며 설정의 일정 부분을 자동화한다.

미니큐브로 로컬에서 쿠버네티스 실행하기

개발용 시스템 등 로컬 환경에서만 쿠버네티스를 실행하고 다른 부가 기능이 필요 없다면 새로운 대안이 있다. 쿠버네티스 개발팀이 제공하는 미니큐브(Minikube, <https://github.com/kubernetes/minikube>)다. 이를 이용하면 원하는 가상화 호스트에 싱글 노드 쿠버네티스 클러스터를 배포할 수 있다. 몇 가지 제약이 있기는 하지만 맥OS, 리눅스 또는 윈도우에서 손쉽게 사용할 수 있는 것이 큰 장점이다.

쿠버네티스 예제 앱 실행하기

여기까지 잘 따라왔다면 컨테이너를 배포하고 관리한 준비를 마친 것이다. 지금부터는 컨테이너 기반 예제 앱 몇 개를 이용해 간단한 컨테이너 운영을 체험해 보자. 먼저 미니큐브다. 헬로 미니큐브(Hello Minikube) 자습서(<https://kubernetes.io/docs/tutorials/hello-minikube>)를 통해 싱글 노드 쿠버네티스에서 간단한 Node.js 앱이 포함된 도커 컨테이너를 만드는 방법을 배울 수 있다. 일단 감을 잡고 나면 자체 컨테이너를 넣어서 배포 절차를 익히는 것도 가능하다.

다음 단계는 프로덕션에서 사용하는 애플리케이션과 비슷한 예제 애플리케이션(<https://kubernetes.io/docs/samples>)을 배포하면서 포드(pod, 애플리케이션을 구성하는 하나 이상의 컨테이너), 서비스(포드의 논리적 집합), 복제본 집합(머신 장애 시 자가 치유 제공), 배포(애플리케이션 버전 관리) 같은 고급 쿠버네티스 개념에 익숙해지는 것이다.

워드프레스(WordPress)/마이SQL 샘플 애플리케이션(<https://kubernetes.io/docs/tutorials/stateful-application/mysql-wordpress-persistent-volume>)을 보자. 각각을 쿠버네티스에 배포해 실행할 수 있지만 여기서는 주목해야 할 다른 내용도 있다. 즉 프로덕션 수준의 쿠버네티스 애플리케이션에 사용되는 주요 개념을 한눈에 파악할 수 있다. 영구 볼륨을 설정해 애플리케이션 상태를 보존하는 방법(<https://kubernetes.io/docs/concepts/storage/persistent-volumes>), 서비스를 사용해 포드를 서로에게 노출하거나 혹은 외부에 노출하는 방법(<https://kubernetes.io/docs/concepts/services-networking/service>), 애플리케이션 암호와 API 키를 안전하게 저장하는 방법(<http://kubernetes.io/docs/user-guide/secrets>) 등이 대표적이다.

화면 | 쿠버네티스 워드프레스/마이SQL 예제 앱

The screenshot shows the Kubernetes Tutorials page. The main content area displays the title 'Example: Deploying WordPress and MySQL with Persistent Volumes'. Below the title, there is a detailed description of the tutorial, mentioning the use of Minikube, PersistentVolumes, and PersistentVolumeClaims. A warning note states that this deployment is not suitable for production use cases due to its single instance nature. At the bottom, a note specifies that the files provided are for Kubernetes version 1.9 and later.

위브웍스(Weaveworks)에는 스톡 숍(Stock Shop, <https://github.com/microservices-demo/microservices-demo>)이라는 예제 앱이 있다. 이 앱은 마이크로서비스 패턴을 사용해 쿠버네티스에서 애

플리케이션을 구성하는 방법을 보여준다. 스톡 속은 그 기반 기술인 노드js, 고(Go) 키트, 스프링 부트(Spring Boot)에 익숙한 사람에게 유용하다. 워드프레스/마이SQL 애플리케이션 예제를 따라하다 보면 '내 요구 사항에 꼭 맞는 사전 제작된 쿠버네티스 앱이 있지 않을까'라는 생각이 들 것이다. 실제로 있다. 쿠버네티스에는 쿠버네티스 애플리케이션에 대한 패키징과 버전 관리, 공유 등을 지원하는 애플리케이션 정의 시스템 '헬름(Helm, <https://helm.sh>)'이 있다.

쿠버네티스로 컨테이너 관리하기

쿠버네티스는 강력한 추상화(abstractions)를 통해 컨테이너 관리를 간소화한다. 동시에 레이블, 네임스페이스(<https://kubernetes.io/docs/tasks/administer-cluster/namespaces-walkthrough>) 등의 메커니즘을 통해 포드와 서비스, 배포를 분리하고 높은 유연성도 제공한다. 예를 들어 앞서 살펴본 예제 중 하나를 선택해 여러 네임스페이스에서 다양한 인스턴스를 설정하면 각 네임스페이스의 구성요소를 상호 독립적으로 변경할 수 있다. 이렇게 하면 지정된 네임스페이스에 따라 전체 포드로 업데이트를 배포(<https://kubernetes.io/docs/tutorials/kubernetes-basics/update-intro>)하는 것도 가능하다.

다음 단계는 쿠버네티스의 인프라 관리 기능을 배울 차례다. 퍼펫(Puppet, <https://forge.puppet.com/puppetlabs/kubernetes>)에는 쿠버네티스에서 리소스를 만들고 수정하는 모듈이 있다. 해시코프(HashiCorp)의 테라폼(<https://www.terraform.io/docs/providers/kubernetes/index.html>)도 쿠버네티스에 대한 지원을 강화하고 있다. 단, 주의할 것은 이런 리소스 관리 툴마다 기본 전제가 크게 다르다는 점이다. 예를 들어 퍼펫과 테라폼은 기본적으로 각각 가변성 인프라와 불변성 인프라를 사용(<https://blog.gruntwork.io/why-we-use-terraform-and-not-chef-puppet-ansible-saltstack-or-cloudformation-7989dad2865c>)한다. 이런 개발 철학과 동작 방식의 차이를 이해하지 못하면 쿠버네티스 관리가 더 어려울 수도 있다. 

쿠버네티스 ‘관리 지옥’에서 탈출하는 필수 툴 15가지

**Tech
Solution**

Serdar Yegulalp | InfoWorld

컨테이너 애플리케이션을 대규모로 배포하는 방법으로 쿠버네티스(Kubernetes)가 확산하고 있다. 이미 표준이 됐다고 말하는 사람도 많다. 쿠버네티스를 이용하면 어렵고 복잡한 컨테이너 배포 작업을 간편하게 처리할 수 있지만, 안타깝게도 쿠버네티스 자체가 쓰기 힘들다는 평가를 듣는다. 너무 복잡하고 혼란스러워 쿠버네티스를 관리하는 것이 더 큰 일이라는 불만이 나올 정도다. 물론 쿠버네티스 사용자가 늘어나고 기능이 개선되면서 이런 혼란스러움도 점차 정리되고 있다. 그러나 개선 작업이 너무 느리다고 느끼는 일부 사용자는 직접 대안을 만들어 공유하고 있다. 그 결과가 여기서 소개하는 15개 프로젝트다.

‘비트나미 캐빈’, iOS와 안드로이드용 쿠버네티스 대시보드

오늘날 모바일 인터페이스 지원은 모든 웹 애플리케이션과 서비스의 기본 중 기본이다. 캐빈(Cabin, <https://github.com/bitnami-labs/cabin>)을 이용하면 쿠버네티스 관리자가 iOS나 안드로이드 스마트폰에서 쿠버네티스 대시보드에 접속할 수 있다. 모바일로 접속한 후에는 헬름 차트와 대규모 배포, 팟로그 읽기, 쿠버네티스 기반 웹 앱 접속 등 쿠버네티스 대시보드 기능 대부분을 사용할 수 있다.

‘캐지’, 쿠버네티스 배포 정의 간소화

쿠버네티스에 대한 가장 많은 불만은 매니페스트(manifest) 혹은 애플리케이션 정의가 너무 복잡하고 장황하다는 것이다. 이는 결국 코드 작성과 관리의 어려움으로 이어진다. 그래서 대안으로 등장한 것이 캐지(Kedge, <https://github.com/kedgeproject/kedge>)다. 캐지는 더 단순하고 간결한 구문을 제공한다. 간단한 쿠버네티스 정의 파일을 입력하면 캐지가 쿠버네티스의 완전한 대체재로 변신한다. 비슷한 기능의 코키 쇼트(Koki Short)와 달리, 캐지는 선언 파일에 모듈러 구문을 사용하지 않는다. 단지 애플리케이션 정의를 일반적인 단축키로 줄일 뿐이다.

‘코키 쇼트’, 관리 가능한 쿠버네티스 매니페스트

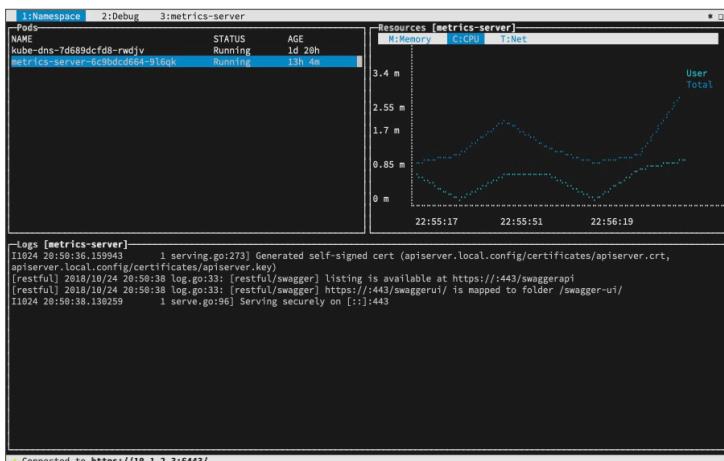
코키 쇼트(Koki Short, <https://github.com/koki/short>)는 쿠버네티스 내에서 사용하는 애플리케이션 정의 방법이나 매니페스트를 개선하는 프로젝트다. 캐지와 마찬가지로 쿠버네티

스 팟을 간결하게 기술할 수 있도록 지원한다. 쇼트를 사용하지 않은 완전한 구문으로 전환할 수 있고 다시 돌리는 것도 가능하다. 또한, 쇼트는 캐지와 달리 모듈식으로 구성돼 한 곳에서 선언한 쇼트의 세부 내용을 다른 곳에서 재사용할 수 있다. 공통 요소를 가진 많은 팟을 간결하게 정의할 수 있다.

‘캡스’, 쿠버네티스 클러스터용 명령줄 작업

쿠버네티스 팀이 개발한 캡스(Kops, <https://github.com/kubernetes/kops>)를 사용하면 명령줄에서 쿠버네티스 클러스터를 관리할 수 있다. AWS와 GCE 기반의 클러스터를 지원하며 현재는 VM웨어 v스피어를 비롯한 다른 환경에서 쓸 수 있도록 개발하고 있다. 캡스는 설정, 해체 프로세스를 자동화하는 것 외에 다른 형태의 자동화에도 유용하다. 예를 들어 테라폼(Terraform)을 사용해 클러스터를 재배포하도록 구성할 수 있다.

화면 | 쿠버네티스용 터미널 콘솔, ‘큐브박스’



‘큐브박스’, 쿠버네티스용 터미널 콘솔

쿠버네티스용 고급 터미널 콘솔인 큐브박스(Kubebox, <https://github.com/astefanutti/kubebox>)는 단순히 보기 좋은 쿠버네티스와 API용 셸을 제공하는 데 그치지 않고 메모리와 CPU 사용률, 팟 목록, 실행 중인 로그를 인터랙티브하게 보여준다. 무엇보다 좋은 점은 리눅스, 윈도우, 맥OS에 따라 별도의 애플리케이션을 제공한다는 것이다.

‘큐브-몽키’, 쿠버네티스용 카오스 몽키

가장 확실한 스트레스 테스트 방법은 무작위로 이 것저것을 중단해 보는 것이다. 카오스 엔지니어링 툴

인 넷플릭스 카오스 몽키(Chaos Monkey, <https://github.com/Netflix/chaosmonkey>)의 개념도 이와 같다. 이미 업무에서 사용하고 있는 가상머신과 컨테이너를 무작위로 다운시킨 후 더 탄력적인 시스템을 만들 수 있도록 지원한다. 큐브-몽키(Kube-monkey, <https://github.com/asobti/kube-monkey>)은 이와 같은 개념을 쿠버네티스 클러스터 스트레스 테스트에 적용했다. 사용자가 지정하는 클러스터의 팟을 무작위로 중단한다. 특정 시간에만 작동하도록 더 상세하게 설정할 수도 있다.

‘큐브-ps1’, 똑똑한 쿠버네티스 명령 프롬프트

큐브-ps1(Kube-ps1, <https://github.com/jonmosco/kube-ps1>)은 (당연하게도!) 쿠버네티스용 1세대 소니 플레이스테이션 에뮬레이터가 아니다(재치있는 생각이라는 것은 인정한다). 배시(Bash)에 간단한 기능을 추가한 것으로, 현재 쿠버네티스의 현황과 네임스페이스를 프롬프트에 표시해준다. 큐브-ps1은 큐브 셸(Kube-shell)에 포함된 많은 추가 요소 중 하나다. 큐브-셸(Kube-shell)의 다른 기능은 필요 없고 스마트 프롬프트만 사용하고 싶다면 큐브-ps1이 정답이다.

‘큐브-프롬프트’, 대화형 쿠버네티스 클라이언트

쿠버네티스 CLI를 변경하는, 소소하지만 유용한 툴이다. 큐브-프롬프트(Kube-prompt, <https://github.com/c-bata/kube-prompt>)를 이용하면 다양한 명령을 대화형 세션에 입력할 수 있다. 모든 명령어 앞에 kubectl를 입력할 필요가 없으며, 각 명령에 따라 상황을 인식해 자동으로 명령어를 완성해 준다.

‘큐브-셸’, 쿠버네티스 CLO용 셸

쿠버네티스 명령줄은 강력하다. 그러나 다른 명령줄 앱이 그렇듯, 사용해야 하는 옵션이 장황한 것이 문제다. 큐브 셸(Kube-shell, <https://github.com/cloudnativelabs/kube-shell>)은 표준 쿠버네티스 명령줄에 명령 자동 완성과 자동 제안 기능을 제공한다. 서비스명 같이 쿠버네티스 서버가 지원하는 내용도 제안해준다. 이밖에 강화된 명령 히스토리 기능과 vi 스타일 편집 모드, 사용자와 네임스페이스, 클러스터, 다른 설치 사양 세부 사항에 대한 실행 정보 등도 제공한다.

‘큐브스파이’, 쿠버네티스 리소스에 대한 실시간 모니터링

풀루미(Pulumi)의 큐브스파이(Kubespy, <https://github.com/pulumi/kubespy>)는 쿠버네티스 리소스의 변화를 실시간으로 추적해 텍스트 기반 대시보드로 제공하는 진단 툴이다. 예를 들어 팟이 부팅되는 동안 팟 정의가 Etcd에 작성되고 팟이 노드에서 실행되도록 예약되고 팟을 생성하는 노드의 큐블릿에 이어 팟이 ‘실행 중’으로 표시되기까지 팟의 상태 변경(<https://blog.pulumi.com/kubespy-and-the-lifecycle-of-a-kubernetes-pod-in-four-images>)을 바로 볼 수 있다. 큐브스파이는 독립형 바이너리 혹은 Kubectl의 플러그인으로 실행할 수 있다.

AWS용 쿠버네티스 인그레스 컨트롤러

쿠버네티스는 인그레스(Ingress, <https://kubernetes.io/docs/concepts/services-networking/ingress>) 서비스를 통해 클러스터에 대한 외부 로드밸런싱과 네트워크 서비스를 지원한다. AWS도 로드밸런싱 기능을 지원하지만 이들 서비스를 자동으로 쿠버네티스 기능과 연동하지는 못하는데, 이 틈을 메우는 것이 바로 AWS용 쿠버네티스 인그레스 컨트롤러(Kubernetes Ingress Controller for AWS, <https://github.com/zalando-incubator/kube-ingress-aws-controller>)다.

이 컨트롤러는 클러스터내 개별 인그레스 객체의 AWS 리소스를 자동으로 관리한다. 예를 들어 새 인그레스 리소스에 대한 로드 밸런스를 생성하고, 제거한 리소스는 로드 밸런서에서 삭제한다. 이 과정에서 클러스터 오류를 막기 위해 AWS 클라우드포메이션을 이용한다. 이밖에 SSL 인증서와 EC2 오토 스케일링 그룹 등 클러스터 내에서 사용되는 다른 요소도 자동으로 관리한다.

‘큐브-옵스-뷰’, 멀티 쿠버네티스 클러스터용 대시보드

쿠버네티스는 일반적인 모니터링 용도의 대시보드를 지원한다. 그러나 일부 쿠버네티스 사

화면 | 멀티 쿠버네티스 클러스터용 대시보드, ‘큐브-옵스-뷰’



자체 툴 가운데 하나로, 쿠버네티스 앱의 지속적 배포를 지원한다. 소스 코드를 변경하면 스캐폴드가 자동으로 변경을 감지하고 빌드와 배포 프로세스를 트리거하며, 오류가 발생할 경우 알려준다. 스캐폴드는 온전히 클라이언트에서 실행된다. 기존 CI/CD 파이프라인에서 사용 가능하며 구글 바젤(Bazel)을 비롯한 몇몇 외부 빌드 툴과 통합할 수도 있다.

‘스턴’, 쿠버네티스 로그 분석기

배의 꼬리 부분, 즉 ‘선미’를 의미하는 그 ‘스턴(stern, <https://github.com/wercker/stern>)’이다. 이를 이용하면 쿠버네티스의 팟과 컨테이너 같은 리소스에서 테일 명령으로 컬러 코드 결과물을 만들 수 있다. 즉, 이들 리소스의 모든 결과물을 단일 스트림으로 한눈에 볼 수 있다. 컬러 코딩도 지원해 여러 스트림을 구분하기도 쉽다.

‘테레사’, 쿠버네티스 기반의 간단한 PaaS

테레사(Teresa, <https://github.com/luizalabs/teresa>)는 쿠버네티스에서 실행하는 간단한 PaaS형 애플리케이션 배포 시스템이다. 팀으로 구성된 사용자가 애플리케이션을 배포, 관리할 수 있다. 정해진 애플리케이션을 사용하는 신뢰할 수 있는 사용자 그룹이 업무를 더 쉽게 처리할 수 있다. 이 과정에서 쿠버네티스를 직접 다룰 필요가 없다는 것이 가장 큰 장점이다. ITWORLD

용자는 여기에 만족하지 못하고 더 직관적으로 데이터를 볼 수 있는 대안을 찾고 있다. 그런 노력 중 하나가 ‘큐브-옵스-뷰(Kube-ops-view, <https://github.com/hjacobs/kube-ops-view>)’다. 여러 쿠버네티스 클러스터의 현황을 한눈에 볼 수 있다. CPU와 메모리 사용량, 팟 상황 등 여러 클러스터의 상태를 확인할 수 있다. 단, 명령은 실행할 수는 없으며, 온전히 보여주는 역할만 한다. 그러나 이 시각화가 매우 직관적이고 효과적이어서 운영센터 상황판 모니터에 안성맞춤이다.

‘스캐폴드’, 쿠버네티스를 위한 반복적 개발

스캐폴드(Skaffold, <https://github.com/Google-ContainerTools/skaffold>)는 구글의 쿠버네티스용

“쿠버네티스를 더 안전하게” 필수 컨테이너 보안 툴 7종

**Tech
Solution**

Serdar Yegulalp | InfoWorld

도 커 컨테이너를 이용하면 소프트웨어 개발자가 애플리케이션을 더 빨리 만들고 더 유연하게 배포할 수 있다. 보안에도 이점이 있다. 소프트웨어를 더 안전하게 만드는 방법의 하나가 바로 컨테이너다. 컨테이너가 애플리케이션 보안을 강화하는 이유는 여러 가지다. 컨테이너에 내장된 소프트웨어 구성 요소의 자동 분석 기능, 컨테이너 클러스터와 여러 애플리케이션 버전을 아우르는 대응 정책, 취약성 데이터 추적과 관리 기술 등이 대표적이다. 하지만, 이런 기능과 기술은 기본일 뿐이다. ‘그 이상’의 보안, 예를 들어 높은 수준의 보안 모니터링과 대응을 원한다면 서드파티 툴을 활용해야 한다.

아포레토

아포레토(Aporeto, <https://www.aporeto.com/container>)는 런타임 보호에 집중한 솔루션이다. 쿠버네티스 워크로드를 보호하는 마이크로서비스 보안 제품과, 분산 애플리케이션을 보호하는 클라우드 네트워크 방화벽 시스템으로 구성됐다. 아포레토는 온-프레미스는 물론 구글 쿠버네티스 엔진 기반의 쿠버네티스의 워크로드를 보호한다. 구체적인 가격은 계좌를 등록(<https://console.aporeto.com/register>)한 후 요청하면 확인할 수 있다. 30일간 무료로 사용 할 수 있다.

아쿠아 컨테이너 시큐리티 플랫폼

아쿠아 컨테이너 시큐리티 플랫폼(Aqua Container Security Platform, <https://www.aqua-sec.com/products/aqua-container-security-platform>)은 리눅스 컨테이너와 윈도우 컨테이너에 대한 컴플라이언스 및 런타임 보안을 제공하는 엔드투엔드 컨테이너 보안 관리자다. 이를 이용하면 보안 정책과 위험 대응 프로파일을 애플리케이션에 적용하고, 이를 다시 다른 애플리케이션 빌드 파이프라인에 연결할 수 있다. CI/CD 툴과 통합(<https://www.aquasec.com/use-cases/devsecops-automation>)하는 것도 가능하다.

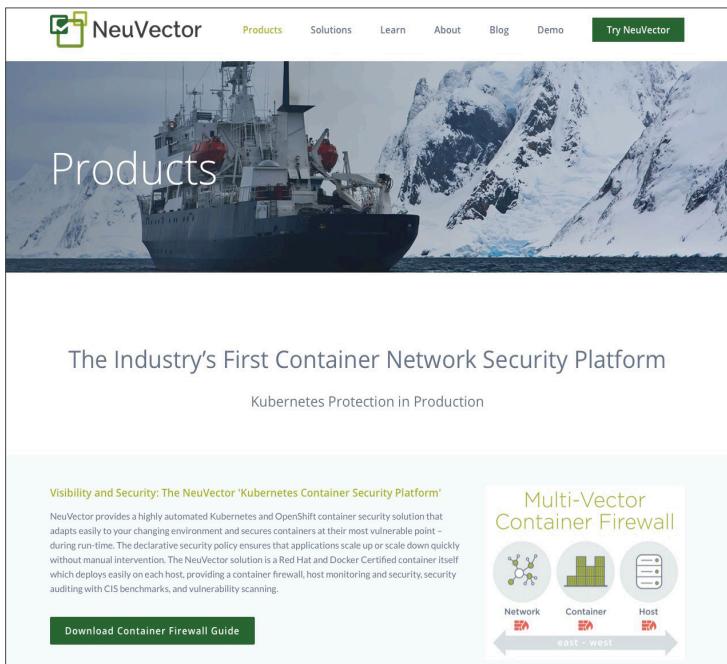
아쿠아 플랫폼은 하시코프 볼트(Hasicorp Vault) 같은 비밀 관리 도구와 연동되며, 소프트웨어 구성 요소에서 메타데이터에 접근할 수 있는 그라파스(Grafeas, <https://www.infoworld.com/article/3230462/security/what-is-grafeas-better-auditing-for-containers.html>) API를 지원한다. 온프레미스 또는 클라우드에 설치해 사용할 수 있다. 무료 평가판이나 오픈

소스는 없다.

아토믹 시큐어드 도커

아토믹 시큐어드 도커(Atomic Secured Docker, <https://www.atomicorp.com/features-atomic-secured-docker-kernel>)는 우분투, 센트OS, 레드햇 엔터프라이즈 리눅스의 대체 커널이다. 사이버 공격을 막기 위한 강화된 메모리 사용 권한 등을 지원한다. 주로 아토믹코프(Atomicorp)의 보안 커널 제품에서 따온 기능인데, 컨테이너 브레이크아웃 보호같이 도커용으로 특별히 설계된 것도 있다. 업체에서 직접 구매(<https://www.atomicorp.com/pricing>)할 수 있으며, AWS 호스팅 센트OS, 애저 호스팅 센트OS 및 우분투 버전은 AWS나 애저 마켓에서 살 수 있다.

화면 | 쿠버네티스 클러스터 보안 툴, '뉴벡터'



뉴벡터

뉴벡터(NeuVector, <https://neuvectr.com/runtime-container-security>)는 쿠버네티스 클러스터 보안 툴이다. 레드햇 오픈시프트(OpenShift), 도커 엔터프라이즈 에디션(Docker Enterprise Edition) 같은 기존 쿠버네티스 관리 솔루션과 연동되며, 젠킨스(Jenkins) 플러그인이 있어, 모든 배포 단계에서 애플리케이션을 보호할 수 있다. 뉴벡터는 기존 코드를 수정하지 않고 쿠버네티스 클러스터에 컨테이너로 배치된다. 클러스터에 뉴벡터를 추가하면 호스팅된 모든 컨테이너를 검색해 동작 명세를 보여주는 맵이 생성된다. 애플리케이션 추가, 삭제로 인한 모든 변경 사항을 탐지하므로, 컨테이너 삭제나 새로운 취약성 같은 위협 요소에 실시간 대응할 수 있다. 뉴벡터의 가격은 도커 호스트 수에 따라 달라지며, 연 9,950 달러부터 시작한다. 무료 체험 버전(<https://neuvec-tor.com/try-neuvector>)도 있다.

시스딕 시큐어

시스딕 시큐어(Sysdig Secure, <https://sysdig.com/product/secure>)는 컨테이너 런타임 환경을 모니터링하는 포렌식 보고서 툴이다. 시스딕 모니터(Sysdig Monitor) 등 다른 시스딕 툴과 함께 작동하도록 설계됐다. 애플리케이션, 컨테이너, 호스트 또는 네트워크 작업별로 환경을 설정해 적용할 수 있으며, 모든 이벤트는 호스트나 컨테이너 또는 쿠버네티스 같은 오케스트레이션 툴을 통해 확인할 수 있다. 컨테이너의 명령 명세를 기록, 검토하고 클러스터 전반의 포렌식을 마치 트위스트락(Twistlock)의 '사건 탐색기'(incident explorer)와 유사한 방식으로 기록, 재생할 수 있다. 유료(<https://sysdig.com/pricing>)이며 클라우드 버전과 온-프레미스 버전도 있다.

테너블닷아이오 컨테이너 시큐리티

테너블닷아이오 컨테이너 시큐리티(Tenable.io Container Security, <https://www.tenable.com/products/tenable-io/container-security>)는 데브옵스 팀에 컨테이너 보안에 대한 가시성을 제공하는 데 초점을 맞춘 툴이다. 빌드 과정에서 악성코드, 보안 취약점, 정책 컴플라이언스 등을 중심으로 컨테이너 이미지를 검사한다. 위험을 발견하면 해당 문제의 특성과 문제가 발생한 정확한 위치를 알려준다. 예를 들어 다중 레이어 이미지 중 특정 레이어에 문제가 있다고 알려주므로 신속하게 대응할 수 있다.

테너블닷아이오 컨테이너 시큐리티는 일반적인 CI/CD 빌드 시스템과 컨테이너 이미지 레지스트리에서 작동한다. 모든 컨테이너 이미지의 현재 상태와 정책 적용 상태, 저장소 동작을 보여주는 대시보드를 별도로 제공한다. 가격 정책(<https://www.tenable.com/buy>)은 별도로 요청해야 확인할 수 있다. 60일 무료 체험판(<https://www.tenable.com/try>)도 있다.

트위스트락

트위스트락(Twistlock, <https://www.twistlock.com>)은 도커 엔터프라이즈 같은 주요 컨테이너 제품이 다루지 않는 여러 추가 보안 제어 기능을 제공한다. 주요 기능은 다음과 같다.

- 컨테이너에 HIPAA 및 PCI 규칙을 적용하기 위한 컴플라이언스 컨트롤
- 젠킨스 등 빌드 툴에 대한 컴플라이언스 경고
- 클라우드 네이티브 애플리케이션 방화벽
- 유효하거나 유효하지 않은 컨테이너 동작 분석 결과에 따른 런타임 공격 보호
- 쿠버네티스의 CIS 벤치마크 지원(<https://www.cisecurity.org/benchmark/kubernetes>). 이를 통해 현재의 쿠버네티스 상태와 일반적인 쿠버네티스 보안 기준을 비교할 수 있다.

현재 트위스트락 최신 버전은 2018년 8월에 나온 2.5다. 런타임 오버헤드를 줄이는 새로운 포렌식 기법이 추가돼, 컨테이너 외부에 사전 및 사후 컨테이너 상태 정보를 저장할 수 있다. 이밖에 네임스페이스(namespace), 팟(pods), 컨테이너 매핑용 실시간 시각화 툴이 개선됐고, 서버리스 컴퓨팅 시스템을 위한 보안 기능도 추가됐다. 트위스트락은 유료 엔터프라이즈 에디션 (<https://www.twistlock.com/get-twistlock>)만 제공한다. (IT)WORLD

AWS vs. 애저 vs. 구글 클라우드 관리형 쿠버네티스 3종 심층 비교

**Tech
Solution**

Eric Johanson | InfoWorld

분명 쿠버네티스는 가장 뜨거운 주제다. 구글이 처음 만들었고 현재는 CNCF(Cloud Native Computing Foundation)가 개발을 이끌고 있다. 이 오픈소스 프로젝트는 컨테이너 오케스트레이션 전쟁에서 승리했다. 메소스피어(Mesosphere)와 도커(Docker) 등 경쟁사 조차 쿠버네티스를 채택했고, 오픈시프트(OpenShift)와 클라우드 파운드리(Cloud Foundry) 등 선도 PaaS 스택도 쿠버네티스를 포함하는 추세다. 또한, 현재 모든 주요 클라우드 업체가 쿠버네티스를 지원한다.

하지만 그렇다고 해서 모든 쿠버네티스 제품이 같거나 비슷한 것은 아니다. 관리형 쿠버네티스의 핵심 구성 요소를 분석하고 AECSK(Amazon Elastic Container Service for Kubernetes), AKS(Azure Kubernetes Service), 구글 KE(Google Kubernetes Engine) 등 주요 클라우드 제공업체의 플랫폼을 테스트, 비교해 보자.

쿠버네티스 클러스터 설정하기

이번 테스트에서 3사의 서비스 모두 클러스터(Cluster) 구성에 문제가 없었다. 그러나 구성에 필요한 작업의 번거로움에서 차이가 있었다. 예를 들어, 아마존 EKS는 클러스터를 생성하는 추가 단계가 필요 하지만 마이크로소프트의 AKS와 구글 KE는 몇 개의 간단한 명령으로 수분 안에 클러스터를 구성해 운영할 수 있었다. 또 AWS에서는 AWS IAM(Identity and Access Management)을 이용한 통합 인증을 위해 헤픽토(Heptio) 등 별도 패키지를 설치해야 했다.

다양한 쿠버네티스 배치를 관리하는 것도 차이가 있었다. 일단 기본적으로 kubectl 명령줄 툴이 필요하다. 많은 서비스가 기본적으로 기존 파일에 kubeconfig 컨텍스트를 추가하는 명령을 제공하므로 여러 업체의 클러스터를 쉽게 전환할 수 있었다. 반면 아마존 EKS는 수동 작업이었다. 상황에 따라 신속하게 클러스터를 생성하고 삭제해야 한다면 시간이 더 걸릴 수 있다. 정리하면 설정 측면에서 구글 KE와 AKS는 비슷하지만 아마존 EKS는 더 많은 단계가 필요했다. 아마존은 현재



클러스터 생성 속도를 높이는 개선 조치(<https://aws.amazon.com/ko/blogsopensource/eksctl-eks-cluster-one-command>)를 진행중이다.

쿠버네티스 대시보드 비교

네이티브 쿠버네티스 대시보드는 모든 쿠버네티스 서비스에 사용할 수 있는 단순한 웹 기반 UI이다. 배치, 팟(pods), 서비스에 대한 몇몇 지표를 제공하며 클러스터를 관리할 수 있다. 이런 대시보드는 있으면 좋지만 꼭 필요한 것은 아니다. 대시보드에서 하는 모든 것을 명령줄에서도 할 수 있기 때문이다.

아마존 EKS와 AKS에서 대시보드 사용 방법은 매우 간단했다. 필자는 이를 이용해 클러스터를 로컬로 호스팅하고 필자의 기기에서 프록시를 시작하며 대시보드의 로컬 호스트 버전을 탐색할 수 있었다. 하지만 가장 뛰어난 대시보드는 단연 구글 KE였다. 필자는 구글의 UI가 아주 마음에 들었다. 쿠버네티스를 개발한 것이 구글이라는 점을 생각하면, 당연할 수도 있다. 구글의 UI는 네이티브 쿠버네티스 대시보드의 업그레이드 버전처럼 보였다. 동작이 일관되고 매끄럽게 작동하며 유용한 정보를 제공한다. 둘러보기도 쉽다. 또한 즉시 사용할 수 있어 대시보드 설정을 건드릴 필요가 없었다. AKS도 대시보드를 내장하고 있지만 탐색이 덜 직관적이고 익숙해지는 데 시간이 좀 필요했다.

아마존 EKS는 3사의 서비스 중 유일하게 대시보드를 기본으로 제공하지 않았다. 로컬이 아니라 외부 인스턴스(아마존 EC2, 구글 클라우드 엔진, 애저 컴퓨트)에서 클러스터를 운영할 계획이라면 미리 고려해야 할 부분이다. 실제로 필자가 그랬다. 아마존 EKS 대시보드를 만들려면 API 서버를 노출하거나 기기 호스팅 쿠버네티스에서 프록시에 대한 외부 액세스가 필요한데, 둘 다 불가능했다. 인스턴스에 대한 PEM 키 액세스가 있었기 때문에 로컬 프록시를 추가 구성 없이 노출할 수 없었던 것이다. 결국 필자는 대시보드 구축을 포기했다. 명령줄에서 모든 작업을 할 수 있으므로 UI를 설정하느라 시간 낭비할 필요가 없다고 판단했다. 그러나 모든 사용자

가 필자와 같은 상황은 아닐 것이다.

화면 | 구글 쿠버네티스 엔진

The screenshot shows the Google Cloud Kubernetes Engine dashboard. At the top, there's a navigation bar with links for Google Cloud services like Compute Engine, Storage, and Container Registry. Below that is a search bar and a sign-in button. The main content area is titled 'KUBERNETES ENGINE' and includes a sub-section 'Google Cloud에서 제공하는 Kubernetes에서 컨테이너 애플리케이션 배포, 관리, 확장하기'. It features a large image of a cargo ship being loaded by a crane, symbolizing container management. Below the image, there's a section titled '규모에 맞춘 컨테이너 애플리케이션 관리' with a brief description of Google Kubernetes Engine's capabilities. Further down, there's a section titled '신속한 개발 및 반복 작업 가능' with a diagram showing a pipeline of blue rectangular boxes connected by arrows, representing CI/CD pipelines. The overall interface is clean and modern, designed for cloud-native application management.

쿠버네티스 대시보드는 결국 개인적인 필요에 의한 것이다. 관리형 쿠버네티스로 전환하면서 모든 서비스가 원활하게 작동하도록 하고 싶다면 내장형 대시보드와 사용 편의성 등을 고려했을 때 구글 KE가 더 좋은 선택이다. 단, 쿠버네티스를 쓰는 목적은 대부분 워크플로를 자동화하는 것으로, 쿠버네티스를 실제 업무용 시스템에 배치할 때쯤 되면 작업 대부분이 스크립트화될 가능성이 크다. 즉 대시보드가 전혀 필요 없게 될 수도 있다. 실제로 유럽 입자 물리학 연구소(CERN)는 쿠버네티스 클러스터 210여 개를 구성해 사용하고 있지만, 그들이 자체 워크플로를 위해 대시보드를 구성했다는 이야기는 듣지 못했다.

화면 | 아마존 EKS

Amazon Elastic Container Service for Kubernetes(EKS)는 사용자를 위해 아래 AWS 기용 영역 전체에서 Kubernetes 관리 인프라를 운영하여 단일 장애 지점을 제거합니다. Amazon EKS는 공인 Kubernetes 준수 서비스이므로 노드와 Kubernetes 커뮤니티의 기준 도구 및 플러그인을 사용할 수 있습니다. 표준 Kubernetes 환경에서 실행되는 애플리케이션은 완벽하게 호환되며 Amazon EKS로 송금 마이그레이션할 수 있습니다.

Amazon EKS는 모든 AWS 고객에게 정식 버전으로 제공됩니다.

아침

관리할 컨트롤 팔레트에 없음

기본적으로 안전

작업자 노드와 관리형 컨트롤 팔레트 간에 일관화된 보안 통신 채널이 자동으로 설정되므로 Amazon EKS에서 실행되는 고객의 인프라는 기본적으로 안전합니다.

화면 | 애저 쿠버네티스 서비스

이미 Azure를 사용하고 계십니까? 지금 AKS를 체험해 보세요.

Kubernetes 관리, 배포 및 운영 간소화

작업자 노드를 프로비저닝하고 이를 제공된 Amazon EKS 엔드포인트에 연결하기만 하면 됩니다.

체험 시작 >

체험 시작을 통해 AKS를 체험해 보세요.

Kubernetes Service 살펴보기: 가격 정보 | 설명서 | 업데이트 | 비디오 | 전자책

말됨 | 이 만화 가이드를 통해 재미있게 Kubernetes 핵심 개념 살펴보기 >

간편해진 Kubernetes

AKS(Azure Kubernetes Service)를 사용하여 자신 있고 신속하게 Kubernetes 배포 및 운영을 단순화하고 애플리케이션 인프라의 규모를 동적으로 조정하는 방법을 알아보세요.

09-18-2018 02 min, 10 sec

쿠버네티스 클러스터 스케일링

편리한 확장과 축소, 즉 스케일링은 쿠버네티스의 가장 중요한 장점 중 하나다. 스케일링 작업은 명령줄, 대시보드를 이용해 쉽게 처리할 수 있다. 단 필자가 테스트한 3개 쿠버네티스 서비스 중 구글 K8s만 클러스터 자동 스케일링을 지원했다. 이 기능을 이용하면 팟에 리소스가 고갈됐을 때 쿠버네티스가 팟을 자동으로 확대하도록 설정할 수 있다. 반대로 노드가 충분히 활용되지 못하면 이를 축소하고 팟을 다른 노드로 옮긴다. 이와 같은 자동 스케일링은 짧은 프로세스를 운영할 때 특히 유용하다.

아마존 EKS와 AKS도 노드가 자동 확대/축소 그룹으로 구동한다고 주장한다. 그러나 쿠버네티스가 그 정책을 인식하지 못하므로 결국은 사용자가 수동으로 설정해야 한다. 이 구성이 아주 번거운 것은 아님지만 추가 설정과 관리가 필요한 것이 분명하다. 필자는 결국 마스터 노드에 대한 배치로 자동 확대/축소를 설정한 후 정책을 구성했다. 그러나 개인적인 경험상 이런 식으로 서드파티 서비스를 유지하는 것은 위험할 수 있다. 아마존의 자동 확대/축소(<https://github.com/kubernetes/autoscaler/blob/master/cluster-autoscaler/cloudprovider/aws/README.md>), 애저의 클러스터 자동 확대/축소(<https://github.com/kubernetes/autoscaler/blob/master/cluster-autoscaler/cloudprovider/azure/README.md>) 관련 더 자세한 내용은 깃허브에서 확인할 수 있다.

쿠버네티스 클러스터를 위한 고가용성

다음은 가용성 비교다. HA(High Availability)의 핵심은 클러스터에서 단일 장애 지점을 없애는 것이고 쿠버네티스 역시 마찬가지다. CNCF의 홍보 대사 루카스 칼드스트롬은 이를 다음과 같이 설명(<http://events17.linuxfoundation.org/events/kubecon-and-cloudnativecon-north-america>)했다.

“일반 클러스터와 여러 마스터, Etcd 복제품이 있는 상황에서 kube-dns를 1개만 구동한다고 가정해보자. 이런 상황에서 kube-dns를 구동하는 마스터에 장애가 발생하면, 모든 서비스의 발견 쿼리가 제대로 처리되지 않는다. 클러스터 전체가 장애를 일으키고 멈춰버린다. 따라서 이런 상황을 피하기 위해서는 클러스터의 핵심 구성 요소가 어디인지 분석한 후 이러한 단일 장애 지점을 없애는 조치를 해야 한다”

그렇다면 HA 관점에서 관리형 쿠버네티스는 현재 어디까지 발전했을까? 구글 KE는 멀티존과 지역 등 2개 모드로 사용할 수 있다. 차이는 지역 클러스터가 3개의 마스터를 생성하지만 멀티존 클러스터는 1개만 생성한다. 예를 들어, 구글 KE에서 미국동부(US-East)를 선택해 지역 클러스터를 운영한다면 US-East-1, US-East-2, US-East-3 등 3개 마스터가 생성된다. 따라서 이 중 하나가 다운돼도 쿠버네티스 제어 영역과 리소스는 영향을 받지 않는다.

아마존 EKS도 구글 KE와 유사한 방식이다. 다양한 구역에서 1개의 HA 마스터와 워커 노드를 제공한다. AKS는 현재 HA를 지원하지 않는 유일한 서비스다. 워커 노드를 사용해 특정 구역에 HA를 구성할 수 있지만, 상대적으로 작업이 복잡하고 HA 마스터 노드와 같은 수준의 민첩성을 제공하지 않는다.

최종 선택 기준은 기업의 요구사항

결국 어떤 쿠버네티스 업체를 선택할지는 기업의 상황과 요건에 달렸다. 쿠버네티스 전용 기능은 여러 업체가 제공하고 있고 대부분 버전 1.10 상태다. 따라서 기업의 필요에 따라 비교해 선택하면 된다.

예를 들어, 필자의 팀은 거의 모든 서비스를 AWS에 맞추어 표준화했다. 도커 이미지는 아마존 ECR에서 호스팅되고 인스턴스는 아마존 EC2에 있으며 소스 코드는 AWS 코드커밋에 있고 호스팅 파일은 아마존 S3에 있다. 특히 필자는 표준화와 자동화를 중요하게 생각하므로, 아마존 EKS를 쓰는 것이 맞다. 반면 여러 업체의 소프트웨어를 사용하고 있다면 필자와 다른 선택을 할 수도 있다. 특히 평가와 테스트 또는 자동 확장/축소 같은 관리 문제가 중요하다면 구글 KE가 안성맞춤이다. 이미 마이크로소프트 제품을 널리 사용하고 있고 클라이언트에서 구동하는 애플리케이션용 애저 서비스가 필요하다면 AKS가 더 합리적이다. [ITWORLD](#)

쿠버네티스, 고통은 쓰지만 열매는 ‘너무’ 달다

Column

Matt Asay | InfoWorld

식 상한 비유지만 쿠버네티스 ‘메시아’가 이 땅에 도래했다. 헵티오(Heptio)의 최신 설문조사 결과(<https://blog.heptio.com/the-results-are-in-the-state-of-k8s-2018-d25e54819416#---96-260>)를 보면, 응답자의 60%가 쿠버네티스(Kubernetes)를 사용하고 있고, 이들 중 절반 이상이 실제 업무용 시스템에 적용했다고 답했다. 즉, 쿠버네티스를 단순히 테스트하는 시기가 끝났음을 알 수 있다.

이번 조사가 시사하는 바는 크다. 쿠버네티스 기반 애플리케이션은 쓰기 편할지 몰라도 쿠버네티스 자체는 관리하기 까다롭기로 유명하기 때문이다. 즉, 진정한 어려움은 컨테이너와 마이크로서비스 기반 아키텍처로 전환하는 것이 아니라 낯선 쿠버네티스 그 자체였다. 쿠버네티스를 실제 업무용 시스템에 적용하려는 기업이 가장 두려워하는 부분이기도 했다.

너무 많아 혼란스러운 쿠버네티스

현재 시장에는 다양한 쿠버네티스 제품과 서비스가 있다. IBM이나 오라클 등 오랜 업력을 가진 기업은 물론, 10개 이상의 스타트업이 있다. 쿠버네티스를 관리하는 CNCF(Cloud Native Computing Foundation)에 따르면, 이 재단의 인증을 받은 서비스 업체가 61개(<https://www.cncf.io/certification/kcsp>)에 달한다. 이들 모두가 자체 쿠버네티스 버전을 판매하는 것은 아니지만, 성능과 호환성을 위해 쿠버네티스를 자사 제품에 통합해 제공한다.

쿠퍼네티스를 지원하는 기업이 많다는 것은 결코 나쁜 것이 아니다. CNCF의 임원 크리스 앤시스직의 말처럼 “선택은 언제나 좋은 것”이기 때문이다. 사용자가 고를 수 있는 대안이 많을수록 사용자에게 전적으로 유리하다. 그러나 선택도 어느 정도여야 한다. 선택할 수 있는 것이 너무 많으면 선택지가 없는 것만큼이나 좋지 않을 수 있다. 특히 모든 업체가 쿠버네티스를 소리 높여 외치면서, 일부 사용자는 이를 일종의 ‘거품’ 혹은 ‘과대광고’로 받아들이고 있다.



쉽지 않아도 쿠버네티스에 주목해야 하는 이유

캐피탈 원(Capital One)의 클라우드 전문가 버나드 골드는 기업 입장에서 쿠버네티스의 어려움에 대해 “쿠버네티스 기반 애플리케이션은 쓰기 편할지 몰라도 쿠버네티스 자체는 쉽지 않다”라고 지적했다. 사실 현실의 ‘쉽지 않음’은 그 이상이다. 헵티오 조사에서 응답자의 46%가 ‘일반적 기술을 가진’ IT 팀이 쿠버네티스를 사용하는 데 어려움이 있다고 답했다. 이런 어려움은 심지어 모든 기업이 같은 업체와 제품을 사용한다고 해도 마찬가지일 수 있다. 쿠버네티스의 ‘미성숙한’ 부분은 여전히 직접 제어해야 하기 때문이다. 골드는 “쿠버네티스는 제품 생태계의 기반이 되는 초기 제품이다. 애플리케이션을 제대로 실행하려면 모든 것을 직접 설치해, 설정하고 정확하게 관리해야 한다”라고 말했다.

“서버 비용 80% 절감” 영국 파이낸셜 타임스의 쿠버네티스 도입기

Charlotte Jee | Computerworld UK

영국 파이낸셜 타임스(FT)의 콘텐츠 플랫폼 팀은 지난 2015년 도커(Docker) 컨테이너를 도입했다. 도커 기술을 비교적 초기에 수용한 기업 중 하나로 꼽힌다. 최근 덴마크 코펜하겐에서 열린 쿠베콘(KubeCon)에서 FT의 운영 및 안정성 부문 기술 디렉터 사라 웰스는 “당시만 해도 도커는 꽤 신생 기술이었다. 각종 요소를 통합하는 작업에 손이 많이 갔다. 특히 컨테이너 오케스트레이션 플랫폼이 필요했는데, 당시만 해도 구매할 수 있는 외부 솔루션이 없었기 때문에 결국 자체 개발해야 했다”라고 말했다.

오케스트레이션의 어려움은 2년여간 FT를 괴롭혔다. 그러다가 마침내 2017년 쿠버네티스에서 문제를 해결하는 희망을 봤다. FT가 쿠버네티스를 선택한 이유는 2가지였다. 오픈소스여서 문서와 정보 공유에 장점이 있었고, 검증된 기술이자 업계에서 새롭게 부상하는 사실상의 표준 기술이라는 점도 고려했다. 그러나 쿠버네티스로 전환하는 실무 작업은 ‘만만하지’ 않았다. 도커 컨테이너 프로젝트에 참여했던 인력 중 상당수가 2016년에 회사를 떠난 것도 한 요인이다.

가장 큰 어려움은 자체 플랫폼에서 쿠버네티스로 이동하는 작업이 동시에 병렬적으로 이루어져야 한다는 것이었다. 스택 두 개를 동기화 상태로 유지하며 양쪽에 변경 사항을 적용하고, 두 플랫폼에서 모든 것을 두 번씩 테스트하는 것은 매우 난해한 작업이었다. 웰스는 “마치 물살이 센 강을 건너면서 말을 바꿔 타는 것과 같았다. 더구나 마이그



파이낸셜 타임스의 운영 및 안정성 부문 기술 디렉터 사라 웰스

레이션을 시작할 당시 새 서비스를 150건가량 새로 시작했고 다른 많은 일이 동시에 진행됐다. 우리는 그중 어떤 것에도 영향을 주지 않으면서 마이그레이션해야 해야 했다”라고 말했다.

FT는 현재 쿠버네티스를 도입해 기존 컨테이너화된 150여 개의 기술 스택을 관리하는 데 사용하고 있다. 쿠버네티스 도입 이후 릴리즈 수는 연간 12건에서 2,200건으로 늘어났고, 비용도 크게 절감할 수 있었다. 웰스는 “사고가 3건 있긴 했지만 전체 서비스에는 영향을 주지 않았다. 전체 스택을 8개의 대형 가상머신(VM)으로 옮겨 AWS 서버 비용을 80% 절감하고 결과적으로 기존 플랫폼보다 훨씬 안정적이고 저렴한 시스템을 확보할 수 있게 됐다. 무엇보다 IT 팀이 단순 기능 업무에서 해방돼 더 중요한 일에 집중할 수 있게 됐다. ‘더 행복한’ IT 팀을 보유할 수 있게 됐다”라고 말했다.

이런 관리 작업이 불가능하지는 않다. 단지 번거롭고 힘들 뿐이다. 이번 헵티오의 조사가 중요한 것은 바로 이 지점이다. 이러한 '힘듦'을 견뎌낼 가치가 충분히 있음을 보여주기 때문이다. 조사결과를 보면, 쿠버네티스를 실제 업무용 시스템에서 사용하는 기업의 63%가 리소스를 더 효과적으로 사용하는 혜택을 즉시 체감했다고 답했다. 58%는 애플리케이션 배포 기간을 줄이는데 도움이 된 것으로 나타났다. '힘듦'을 감수했을 때 누릴 수 있는 실제 장점은 이처럼 명백하다.

그렇다면 이제 쿠버네티스를 시작하는 기업이 악명 높은 사용성의 늪에 빠지지 않고 이런 혜택을 누리려면 어떻게 해야 할까? 이에 대해 골든은 AWS 같은 대형 클라우드 업체를 지목했다. 그는 "이런 업체는 쿠버네티스를 적극적으로 활용하려 노력하고 있고, 복잡한 인프라를 대규모로 운영하는 데 전문성을 갖고 있다"라고 말했다. 즉, 과거에는 쿠버네티스 같은 새 기술이 전통적 업체에 의해 체계화돼 신제품으로 팔리는 것이 일반적이었다. 그러나 이제 기업이 더는 전통적 업체로부터 복잡하고 비싸게 구매할 필요가 없다. AWS 같은 클라우드 업체가 이런 신기술이 더 쉽고 저렴하게 제공하기 때문이다.

이런 변화가 의미하는 것은 명확하다. 2019년이 AWS와 마이크로소프트 애저, 구글 클라우드 플랫폼 등 퍼블릭 클라우드 빅3에게 이정표가 되는 해가 될 것이라는 사실이다. 특히 올해는 이들 3개 업체가 쿠버네티스를 '길들이고 싶어 하는' 기업의 요구에 본격적으로 대응하는 시기가 될 것이다. 다크호스도 있다. 레드햇을 인수한 IBM이다. 오픈시프트(OpenShift) 플랫폼을 이용해 기업이 쿠버네티스 관리를 단순화하는 새로운 대안을 제시할 가능성이 있다. 만약 레드햇이 IBM의 잡다한 제품이나 기술에 발목을 잡히지 않는다면, 2019년에 우리는 레드햇까지 포함한 '쿠버네티스 빅4'를 보게 될 수도 있다. [ITWORLD](#)