**아마존 클라우드에 대한 자료 조사**

**Knight76.tistory.com**

1. **아마존 AWS**(Amazon Web Services) **소개**



AWS(Amazon Web Services)는 1995년 인터넷 서점으로 시작된 아마존 웹 서비스(Amazon.com)가 운영하는 클라우드 컴퓨팅 플랫폼을 구성하는 분산 컴퓨팅 서비스의 집합이다. AWS 서비스는 9년 전, 2002년 7월 16일에 서비스를 시작했으며, 당시에는 개발자가 사용할 수 있는 제한된 기능만을 제공했다. 아마존(Amazon) 뉴스 자료에 따르면, 2007년에 33만명이 사용하고 있다고 발표했는데, 현재는 훨씬 그 이상의 사용자가 쓰고 있을 것이다.

또한, 2008년 1월 아마존 뉴스 자료에 따르면 Amazon.com보다 Amazon AWS 서비스가 훨씬 대역폭(Bandwidth)을 많이 사용하고 있다고 그림 1 과 같이 발표했다. 엄청난 트래픽이 AWS에서 발생된 것을 확인할 수 있다.

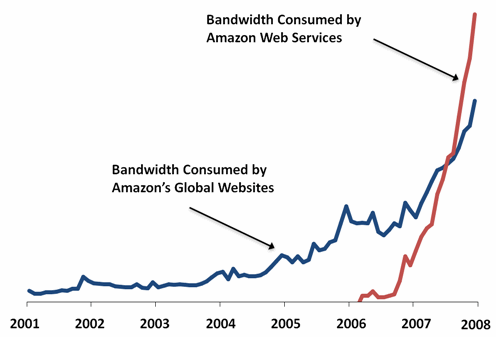


그림 1 Amazon.com과 Amazon AWS 대역폭 사용 비교

현재, AWS 서비스는 웹 호스팅 및 컴퓨팅 자원을 서비스하는 EC2(Elastic Compute Cloud), 스토리지 서비스인 S3(Simple Storage Service), Queue 서비스를 위한 SQS(Simple Queue Service), 데이터베이스 서비스를 위한 SimpleDB 와 RDS(Relational Database System)등 다양한 서비스를 제공하고 있다. 이는 모두 초기 서비스인 AWS를 기반으로 운영하고 있다. 개발자들은 AWS 웹 페이지와 SOAP(Simple Object Access Protocol), REST(Representational State Transfer)와 같은 API를 제공하여 클라우드 서비스를 여러 언어에 맞춰 연동 및 사용할 수 있는 편의를 제공한다.

AWS를 이용하면, 시스템 엔지니어 도움 없이 바로 서버를 설치하고 사용하는 방식을 채용하고 있기 때문에 속도나 인건비측면에서 훌륭한 성과를 얻을 수 있다. 또한, 웹 서비스를 최초 구축하고 나서, 서버를 유지하기 위한 고정 비용이 사용량에 맞춰진 비용 구조로 변경되기 때문에 전체적인 인건비를 낮출 수 있다.

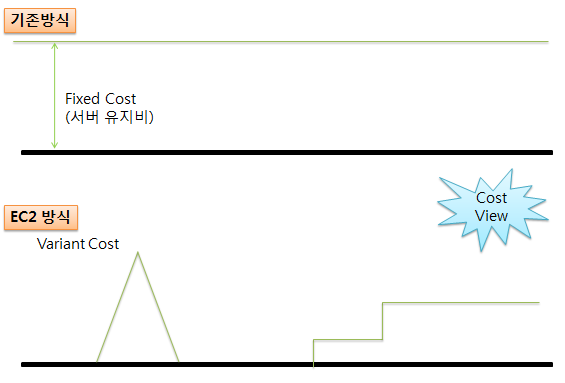


그림 2 서버 유지 비용 비교

그러나 Amazon AWS가 항상 저렴한 것은 아니다. 트래픽이 증가하게 되어 사용량이 높아지면, 그에 대한 비용이 서버 유지비보다 더 많이 나갈 수 있다.

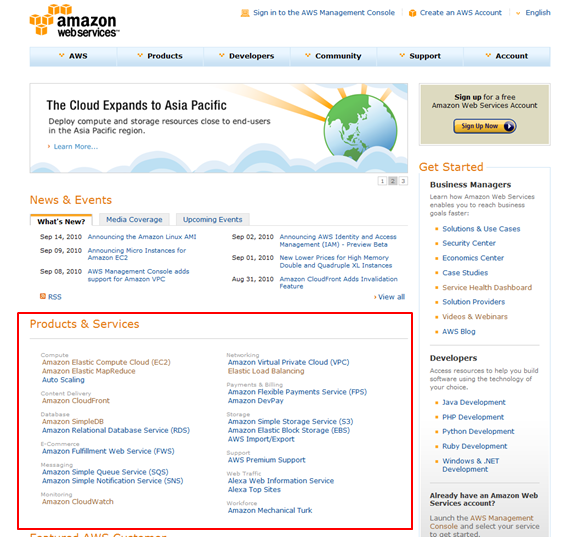


그림 3 AWS Web Page

최근에 대중들에게 아마존 AWS 서비스가 알려진 계기는 세계의 비리를 폭로한 위키리크스 사건이었다. 위키리크스 웹이 아마존 AWS 서비스를 사용하고 있었는데, 도덕적이고 법률적인 문제가 발생하자, 아마존 측은 카피라이트(copyright) - 콘텐츠 사용에 대한 약관(term)을 위배했다고 주장하고 위크리크스 웹을 강제로 내렸다.



그림 4 위키리크스 사태를 풍자한 Time지

기존 호스팅 업체나, 다른 클라우드 서비스와 차별화 된 아마존 AWS의 강점은 개발자의 고민이 무엇인지 명확히 알았다는 점, 명확한 과금 정책, 복잡하지 않은 플랫폼 서비스(PAAS), 사용하기 쉬운 UI(User Interface)와 API를 들 수 있겠다. 아마존의 AWS의 대표 서비스는 EC2와 S3로 생각된다. EC2는 개발자가 언제든지 내가 원하는 스펙의 하드웨어를 설정할 수 있으며, 대용량 파일을 마음대로 올릴 수 있고, 개발자 중심의 편한 아키텍처를 제공한다는 점을 매력으로 꼽을 수 있다. AWS의 장점을 하나의 그림을 표현하자면 그림 5과 같을 것이다.

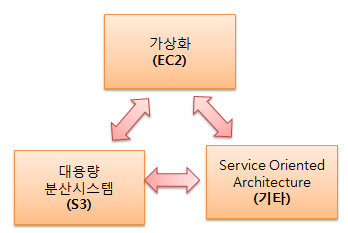


그림 5 AWS의 특징

필자는 2009년 런던 Qcon(InfoQ Conference)에 참석했다. 당시 컨퍼런스의 분위기는 클라우드에 대한 붐이 있었던 시기여서, 2008~2009년에 가장 인기가 좋은 클라우드라는 주제를 가지고 발표가 많이 진행되었다. 당시 필자는 아직까지는 실제 클라우드 서비스가 한국에까지 영향을 미치지 않았기 때문에 과연 이런 서비스가 잘될까 라는 생각을 했다. 그러나 점차 대용량 메일과 파일 공유 등 언제 어디서든 인터넷이 되는 곳이면 내가 저장한 파일들을 다운받을 수 있는 니즈(Need)들이 많아졌고, 네이버의 N드라이브(Ndrive) 서비스가 2009년 말 런칭하는 것을 보면서 우리나라도 클라우드 서비스의 니즈가 더 커질 수 있지 않을까 생각하게 되었다. (실제로 N드라이브는 필자인 나도 유용하게 사용하고 있는 서비스이다)

현재 국내에서는 아직 아마존의 AWS와 비슷하게 클라우드 서비스를 진행하는 곳은 없는 것 같다. 다만 아마존의 AWS S3와 같이 개인이 대용량 파일을 업로드/다운로드 할 수 있는 대용량 스토리지 분산 시스템 서비스들은 있다. 네이버의 N드라이브가 단연 대표적이라 할 수 있고, 다음의 클라우드, KT의 U클라우드 서비스가 있다. 특별히 KT의 U클라우드는 AWS 서비스와 비슷하게 가려는 움직임이 있다.

얼마 전까지 AWS는 아시아 한 곳(싱가폴), 미국 두 곳(동부, 서부), 유럽 한 곳 이렇게 배치해왔는데 최근에는 일본어 서비스로 일본 AWS 서비스를 실시하며 동아시아권에 대한 공격적인 마케팅을 진행하고 있다.



그림 6 AWS 일본어 페이지 소개

아마존 AWS는 클라우드 서비스의 독보적이고, 후발 주자들이 따라 할 수 밖에 없는 운영 아키텍처를 가지고 있다. KT의 클라우드 서비스가 이 AWS의 아키텍처와 AWS UI를 따라 하고 있는 실정임을 안다면, 클라우드에 관심있는 개발자는 AWS 아키텍처가 많은 도움이 될 것이다. 또한 AWS 서비스의 API는 오랜 운영을 통해 얻은 노하우가 담겨 때문에 스스로 공부하기에도 의미 있을 것이라고 본다. 아마존의 AWS 서비스들이 API 인터페이스를 제공하여 3rd party 업체들이 아마존과 협업하는 생태계를 만들면서 훌륭한 결과물이 나오고 있다는 점은 매우 주목할 만 하다.

또한 AWS 서비스에서 연동할 수 있는 오픈 소스들의 규모가 커지는 것도 지켜볼 필요가 있다. 예를 들어 Whirr라는 아파치 프로젝트는 AWS 서비스와 연동해서 Hadoop, Cassandra, Cloudera’s Distribution for Haddop, Zookeep, HBase등을 설치/설정/실행을 가능하게 해준다.

1. **Amazon AWS 아키텍처**

아래 그림 7은 Amazon AWS 아키텍처라고 검색하면 잘 나오는 그림이다. 모든 서비스가 있지는 않지만, 한 눈에 파악할 수 있는 그림이다.

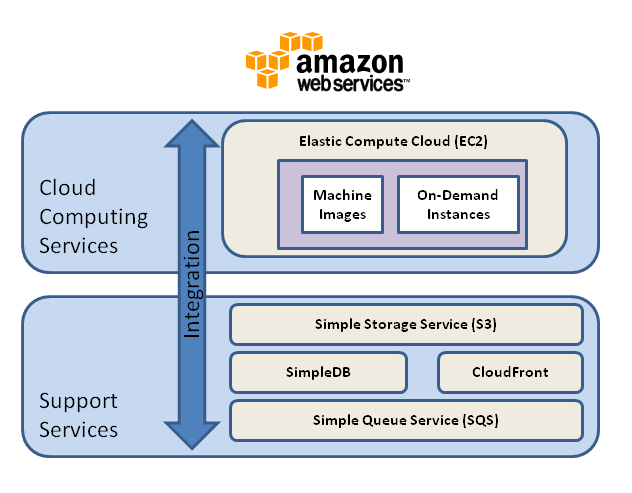


그림 7 간략한 AWS 서비스 구조도

사용자가 Amazon Machine Image(AMI)이라는 데이터, 어플리케이션, 설정 등을 S3 위에 이미지 형태로 업로딩 하면, EC2가 버추얼머신(VM) 환경에서 AMI를 구동하는 형태로 사용이 가능하다. 또한 AMI 형태 외에 On-Demand 인스턴스가 있는데, 많은 양을 처리해야 할 때 혹은 필요할 때만 사용할 수 있다. 이 외에도 아마존에서 남는 자원을 사용하는 Spot Instance와 특정 기간동안 예약해서 사용할 수 있는 Reserved Instance를 제공하고 있다.

EC2는 컴퓨터, S3는 디스크, AMI는 운영체제 또는 운영환경이라고 보면 되는데, 필요에 따라 EC2의 파워를 조절할 수 있어서 연산 작업 등의 파워가 필요한 작업 시 사용한 만큼의 비용만 지불하는 종량제 컴퓨팅 서비스이다.

이렇게 EC2와 S3를 하나로 묶어 사용하는 것을 인스턴스(Instance, 실행단위)라고 부르는데, 하나의 인스턴스는 1.7GHz의 인텔 제온 프로세서(32bit)를 탑재하고 1.7GB의 메인 메모리와 160GB의 하드디스크, 250Mbps의 네트워크 대역을 확보한 서버 컴퓨터와 성능이 비슷하다.

아마존 AWS 서비스 아키텍처는 다음과 같이 그림 8과 같이 표현할 수 있다. 자세한 내용은 이어서 뒷장에서 설명한다.

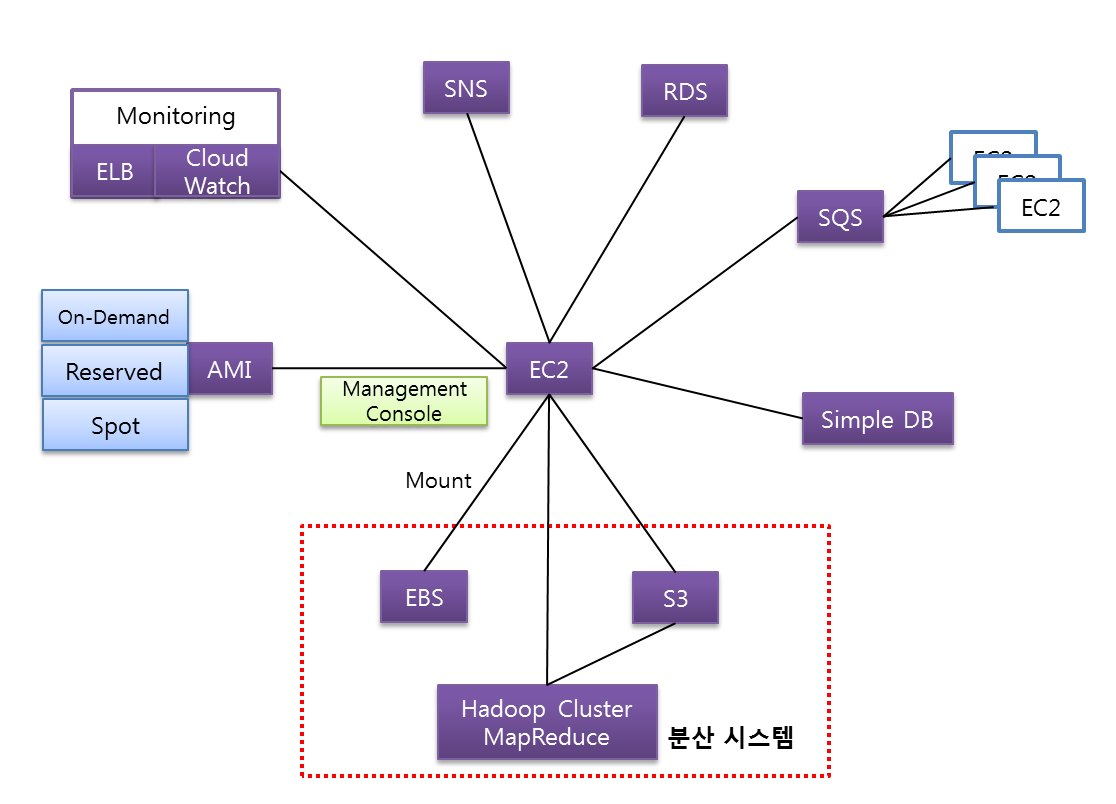


그림 8 AWS 서비스 아키텍처

AWS 서비스는 Management Console이라는 GUI 인터페이스, 개발자들이 쉽게 사용할 수 있는 API 인터페이스를 제공하여 편의를 제공하고 있다.

* 1. **인증 서비스**

AWS 서비스를 사용하기 위해서는 아마존 AWS에 가입을 하고 가입한 이메일로 메일이 날아온다.

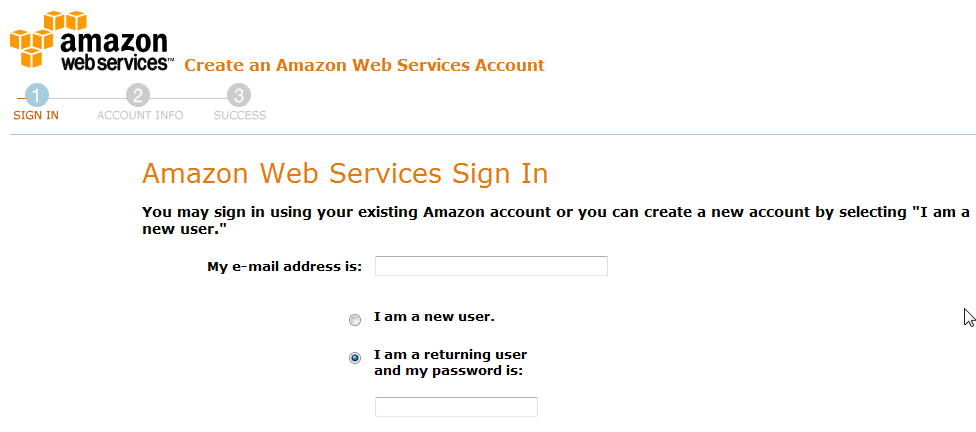


그림 9 가입 화면

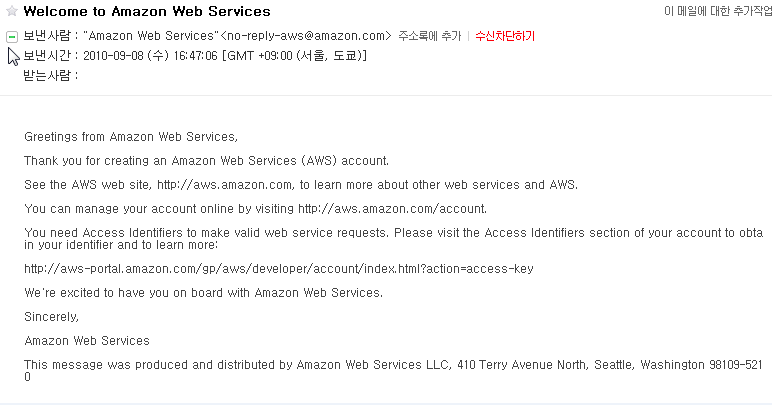


그림 10 환영 메일

AWS 서비스의 EC2 서비스를 사용하기 위해서 다시 사용에 대한 등록(Sign up)을 해야 한다.

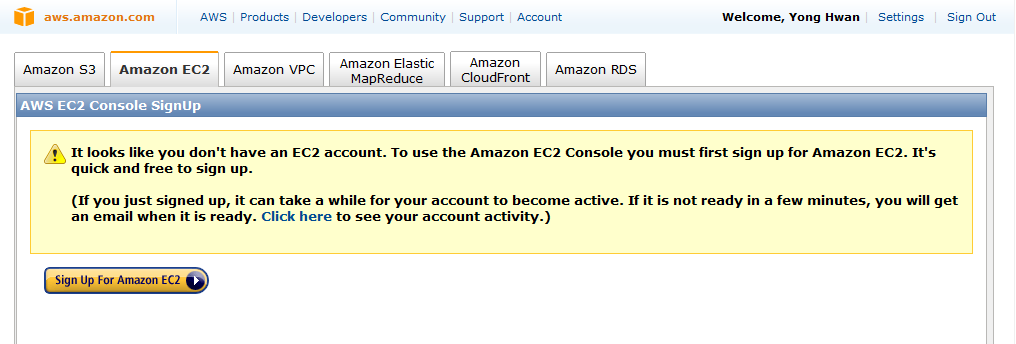


그림 11 EC2 첫 화면

전화 인증과 신용카드인증을 한 후에 서비스 사용이 가능하다. 모든 단계가 끝나면, 그림 12와 같이 화면이 나온다.

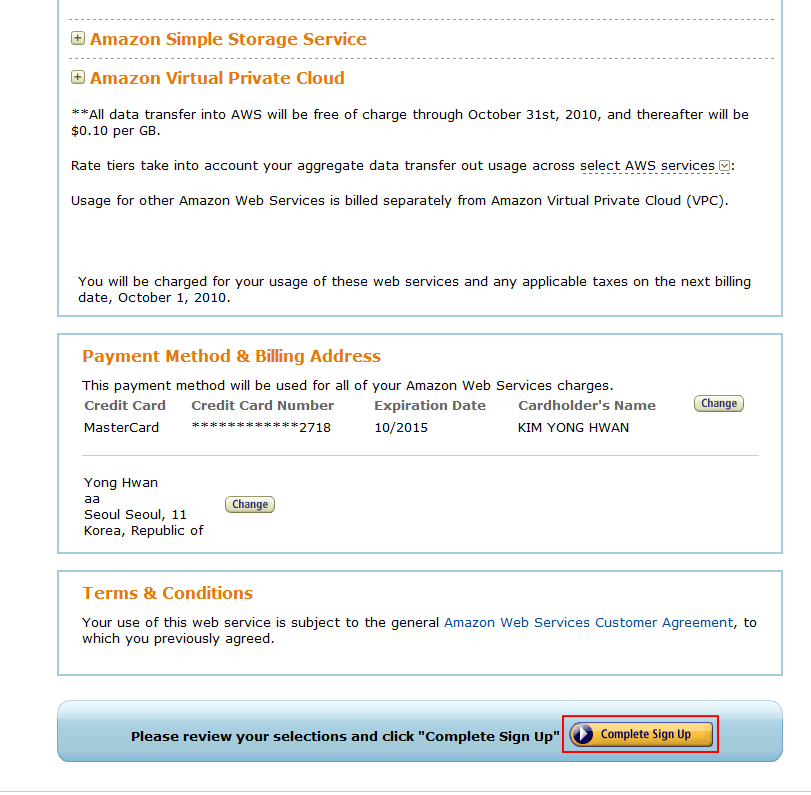


그림 12 최종 인증

AWS 웹 서비스인 Management Console를 이용할 때는 가입한 이메일으로 인증을 한다. API를 사용하는 인증은 서비스마다 다르다. 인증방식을 모아두면 표 1과 같다. Simple DB와 같이 PaaS 플랫폼 서비스를 API 로 사용될 때 사용하는 방식은 Access Credentials이다.

| 구성 요소 | 방식 |
| --- | --- |
| Access Credentials | Access Keys (Access Key ID, Secret Access Key)  X.509 인증서  Key Pairs ID |
| Sign-In Credentials | Email address, password  AWS Multi-Factor Authentication Device |
| Account Identifiers | AWS Account ID  Canonical User ID |

표 1 인증 방식

예를 들어 EC2 서비스를 통해서 서버를 구축했다고 가정했을 때, 터미널 접근을 위해서 SSH를 이용하는데, key pair 파일들을 이용하면 쉽게 접근할 수 있다.

그림 13과 같이 EC2를 생성하고 나면 SSH 클라이언트(Client)로 어떻게 접근할 수 있는지 쉽게 설명이 되어 있다. 그대로 따라 하면 쉽게 터미널에 접근해서 원하는 작업을 할 수 있다.

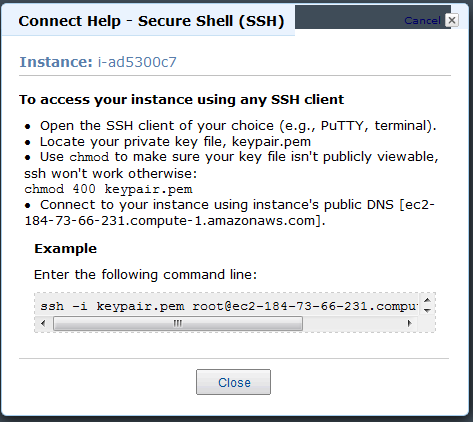


그림 13 SSH 연결 방법 설명

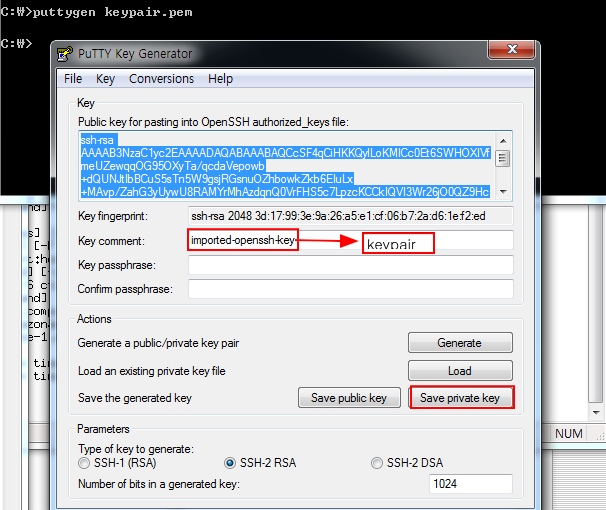


그림 14 Putty를 이용한 연결 방법



그림 15 EC2 서비스에 접속한 화면

* 1. **EC2 와 AMI**

EC2는 Elastic Compute Cloud의 약자이며, S3와 함께 AWS의 핵심 서비스 중의 하나이다. 유연성과 서버 운영에 대한 상당한 노하우와 안정성이 필요한 중요 서비스이다. 우리의 환경과 EC2를 비교하자면, 서비스 운영을 위한 받게 되는 일반 서버와 비교할 수 있다. 추가되는 개념은 사용자의 니즈(Need)에 맞는 컴퓨팅 자원(Computing Resource)과 운영체제를 개발자가 선택하도록 하고 그 비용을 청구받도록 한다. 게다가 EC2는 한 곳에 모여있지 않고 유럽, 미국 동부, 미국 서부, 싱가폴, 일본 등 5개 지역(Region)으로 나누어서 관리되고 있다. Region에는 다시 Zone으로 분류되어 관리되고 있다.

EC2 서비스를 시작하기 위해서는 어떤 운영체제를 사용할지를 먼저 결정한다. 이를 AMI (Machine Image)라고 불리 우는데, EC2에 동작할 수 있는 다양한 운영체제를 아마존 AWS에서 선택한다. 아마존과 파트너 사의 도움으로 만들어진 윈도우 서버, 오픈 솔라리스, 리눅스 배포판등 다양하게 지원되고 있으며, 필요하다면 개발자가 직접 만든 운영체제와 어플리케이션(Application)을 부팅 이미지로 만들고 이를 S3에 올려 AMI와 연동하여 설치할 수 있도록 되어 있다. 아파치 Http 서버, 데이터 베이스, WAS, 개발환경, Hadoop등 다양한 애플리케이션을 운영체제와 함께 설치할 수 있다. 커스텀(Custom) 이미지를 생성해서 타인에게도 공유할 수 있고, 돈을 받고 팔 수도 있다.

우리가 하고 있는 일반적인 웹 서비스 개발을 위한 프로세스와 아마존 AWS를 이용한 방식을 비교하면, 그림 16과 같다. 기존의 방식과 명확히 다른 방법을 사용하고 있는데, 가상화 기술을 적용하는 방식을 사용하며 무엇보다 시스템 엔지니어의 지원 여부없이 서버를 바로 사용할 수 있다는 점이 가장 중요하다.

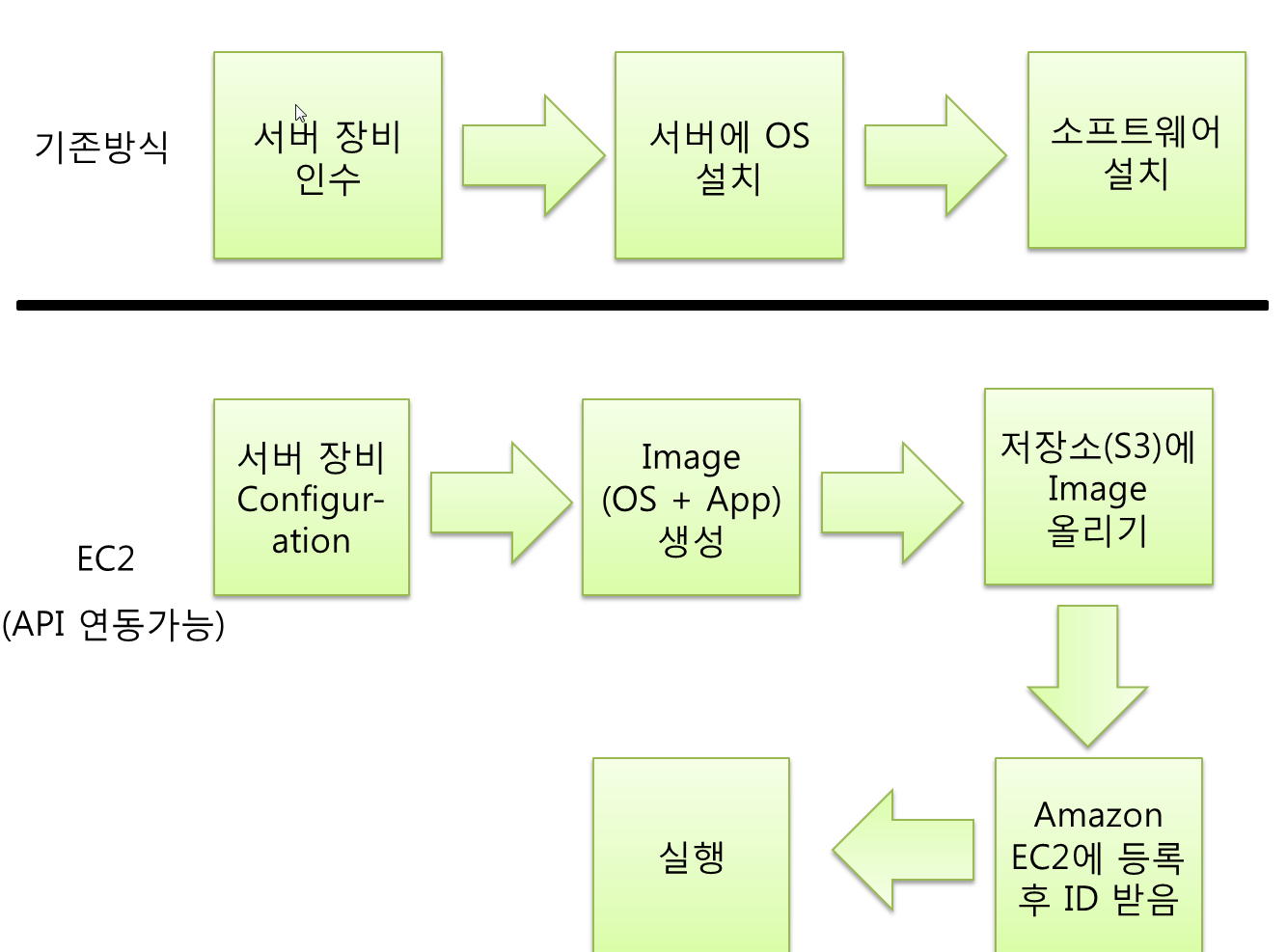


그림 16 EC2 사용 방법 (기존과 비교)

일반적으로 사용량만큼만 사용하는 On-Demand Instance를 선택한다. 언제든지 인스턴스를 종료시킬 수 있고, 다시 생성할 수 있는 동적인(Dynamic) 환경에서 편하게 사용할 수 있다.

연 단위(per year)로 길게 운영되는 경우에는 일정한 비용을 내서 운영할 수 있는데, 이 방식은 Reserved Instance라고 한다. Reserved Instance를 사용하다가 어느 기간 동안 해당 인스턴스를 쓰지 않을 때에는 월세를 받듯이 다른 사용자에게 빌려줄 수 있는데, 이를 Spot Instance라고 한다. 이런 것을 서로 판매할 수 있는 마켓이 AWS에 존재한다.

이런 구성이 가능한 것이 가상화(Virtualization) 기술이다. 가상화 솔루션은 여러 개의 CPU를 가지고 있는 중형 서버에 여러 개의 서로 다른 OS 인스턴스를 설치하여 저렴하게 서버를 운영할 수 있도록 도와준다.

AMI는 Xen 기반으로 되어 있다. Xen은 MS의 Hyper-V, SpringSource 을 인수한 VMWare사의 가상화 솔루션과 함께 전세계적으로 사용하는 가상화 기술 중 하나이다. Xen은 반가상화 (para-virtualization)기법을 사용하는데, 운영체제가 직접 하드웨어를 제어하지 않고 중간 레이어(Layer)인 ‘하이퍼 바이저(Hypervisor)’에게 요청하고 하이퍼 바이저는 하드웨어에 요청해서 응답 받는 수직 구조를 가지고 있다. 뒤에서 설명하게 될 전가상화 방식에 비해서 호출이 적게 일어나기 때문에 성능은 상대적으로 높다. 그러나 Xen위에 동작하는 운영체제는 커널의 일부 소스가 수정되어야 하는 단점이 있고, Xen이 보편화되어서 수정된 운영체제를 쉽게 구할 수 있다.

한편 전가상화(full-virtualization)기법을 사용하면, 운영체제의 커널 수정없이 에뮬레이터 돌리듯이 작동시킬 수 있다. 운영체제가 하드웨어(CPU의 Virtualization Techonology)에 요청을 하면, 중간 레이어인 ‘하이퍼 바이저’에 하드웨어 제어 요구를 하고 ‘하이퍼 바이저’는 다시 하드웨어에 요구하는 Z 방식의 호출구조가 성능을 떨어뜨릴 수 있다.

따라서, 반가상화 기법을 사용하는 Xen을 이해하면, EC2의 UI나 EC2가 어떻게 AMI와 연동되는 부분을 쉽게 사용할 수 있다.

운영체제 이미지(AMI)를 선택한 후에는 컴퓨팅 자원를 선택한다. 컴퓨팅 자원을 EC2 인스턴스라고 표현하는데, 선택할 수 있는 타입(Type)은 총 10가지이다. 개발자는 자신이 개발할 서비스의 특성을 파악하여 CPU 또는 메모리(Memory) 중심으로 시스템 사양을 인스턴스별로 정할 수 있다. 한번 설정되어 운영 중에서는 EC2 컴퓨팅 인스턴스를 변경할 수 없다. 서버를 반납(terminated)시키고 EC2 인스턴스를 생성해야 한다.

|  |  |
| --- | --- |
| Type | 정보 |
| Standard Instances  (3종류) | -Small Instance: 1.7 GB of memory, 1 EC2 Compute Unit (1 virtual core with 1 EC2 Compute Unit), 160 GB of instance storage, 32-bit platform **(디폴트, 1개의 가상코어는 1.0-1.2Ghz 2007 Xeon 프로세서에 해당)**  -Large Instance: 7.5 GB of memory, 4 EC2 Compute Units (2 virtual cores with 2 EC2 Compute Units each), 850 GB of instance storage, 64-bit platform  -Extra Large Instance: 15 GB of memory, 8 EC2 Compute Units (4 virtual cores with 2 EC2 Compute Units each), 1690 GB of instance storage, 64-bit platform |
| High-CPU Instances  (2종류) | -High-CPU Medium Instance: 1.7 GB of memory, 5 EC2 Compute Units (2 virtual cores with 2.5 EC2 Compute Units each), 350 GB of instance storage, 32-bit platform  -High-CPU Extra Large Instance: 7 GB of memory, 20 EC2 Compute Units (8 virtual cores with 2.5 EC2 Compute Units each), 1690 GB of instance storage, 64-bit platform |
| Micro Instance  (1종류) | Micro Instance 613 MB of memory, up to 2 ECUs (for short periodic bursts), EBS storage only, 32-bit or 64-bit platform |
| High-Memory Instances  (3종류) | -High-Memory Extra Large Instance 17.1 GB memory, 6.5 ECU (2 virtual cores with 3.25 EC2 Compute Units each), 420 GB of local instance storage, 64-bit platform  -High-Memory Double Extra Large Instance 34.2 GB of memory, 13 EC2 Compute Units (4 virtual cores with 3.25 EC2 Compute Units each), 850 GB of local instance storage, 64-bit platform  -High-Memory Quadruple Extra Large Instance 68.4 GB of memory, 26 EC2 Compute Units (8 virtual cores with 3.25 EC2 Compute Units each), 1690 GB of local instance storage, 64-bit platform |
| Cluster Compute Instances  (1종류) | Cluster Compute Quadruple Extra Large 23 GB memory, 33.5 EC2 Compute Units, 1690 GB of local instance storage, 64-bit platform, 10 Gigabit Ethernet |

표 2 EC2 인스턴스 Type

EC2 인스턴스별로 성능테스트 자료는 A Quantitative Analysis of High Performance Computing with Amazon’s EC2 Infrastructure: The Death of the Local Cluster (<http://www.cs.virginia.edu/~humphrey/papers/QuantitativeAnalysisEC2.pdf>)를 참조하면 된다.

AMI을 이용한 운영체제(+애플리케이션) 이미지 설치 및 EC2 연동은 10~20분 사이에 금방 완료가 된다. 굉장히 빠른 속도로 서버를 할당 받을 수 있다.

또한 EC2 인스턴스는 EBS(Elastic block storage) 라는 분산 파일 시스템을 붙여서 백업을 할 수 있다. EC2는 디폴트로 로컬 하드 디스크를 제공하지만, 백업이나 데이터를 이동하는 단위가 대용량이기 때문에, EBS나 S3를 이용하여 가용성을 높일 수 있다. 자세한 내용은 다음 장에서 설명한다.

생성된 EC2 인스턴스는 IP와 퍼블릭(public) DNS명을 받는다. EC2는 기본적인 계정관리, ACL관리, 로드 밸랜서(Load Balancer), 모니터링(Monitoring)이 가능하다.

지금까지 설명한 내용으로 실 사용 예를 보도록 한다. EC2 가입 이후 나오는 대쉬보드(dashboard) 화면은 그림17과 같다.

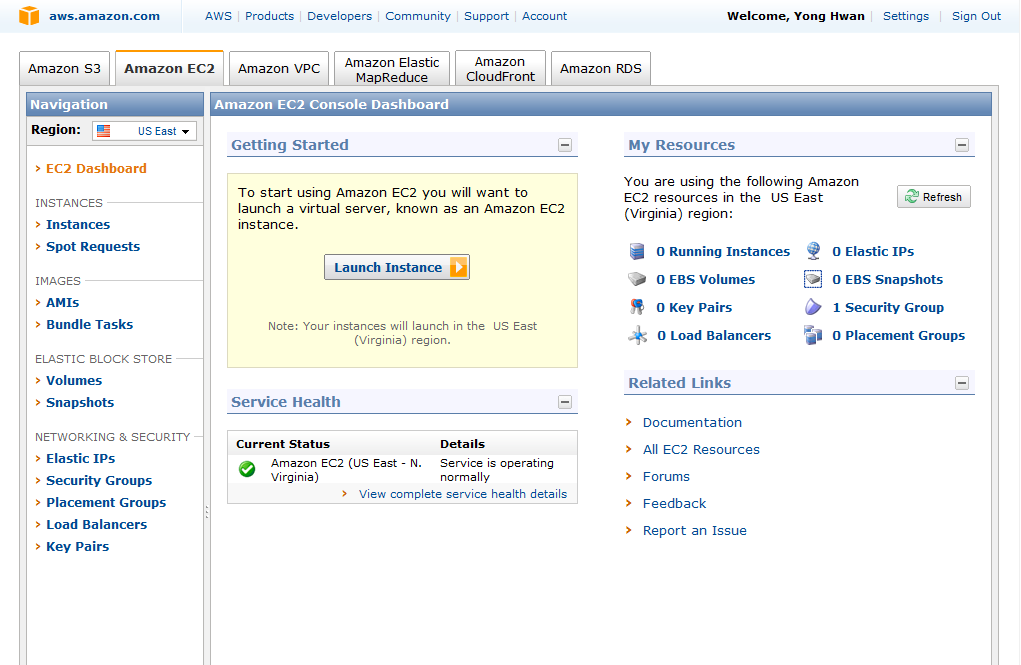


그림 17 EC2 가입 이후의 대쉬보드 화면

EC2 인스턴스를 만드는 위저드(Wizard)의 일부 내용이며, 아래 그림 18은 AMI를 선택하는 화면이다. 누구나 쓸 수 있도록 공개해놓은 커뮤니티(community) AMI를 선택하여 서버를 설치할 수 있다.

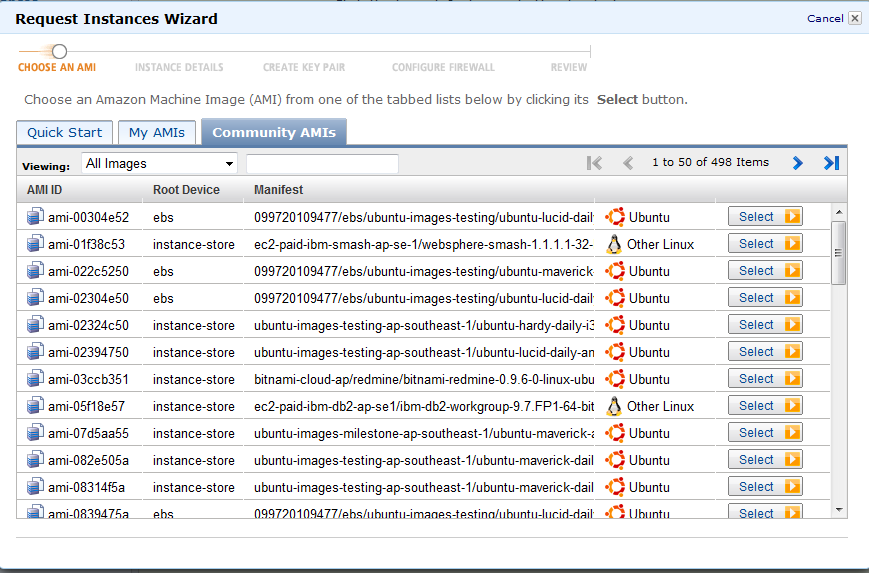


그림 18 AMI 리스트 화면

AMI를 선택한 후에, 위 표 2에서 언급한 컴퓨팅 인스턴스(Computing Instance)를 선택할 수 있다.

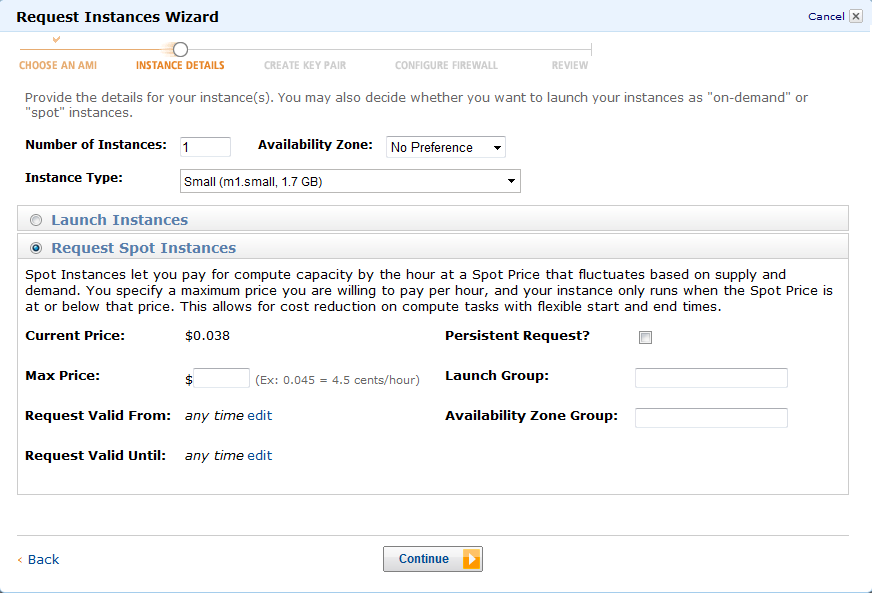


그림 19 위저드 화면 (EC2 Instance 선택)

이렇게 설정한 EC2 인스턴스는 10~20분 사이에 완료가 되고, 그림 20과 같이 내부 팝업이 뜨면서 사용 가능하다라고 화면에 출력된다.

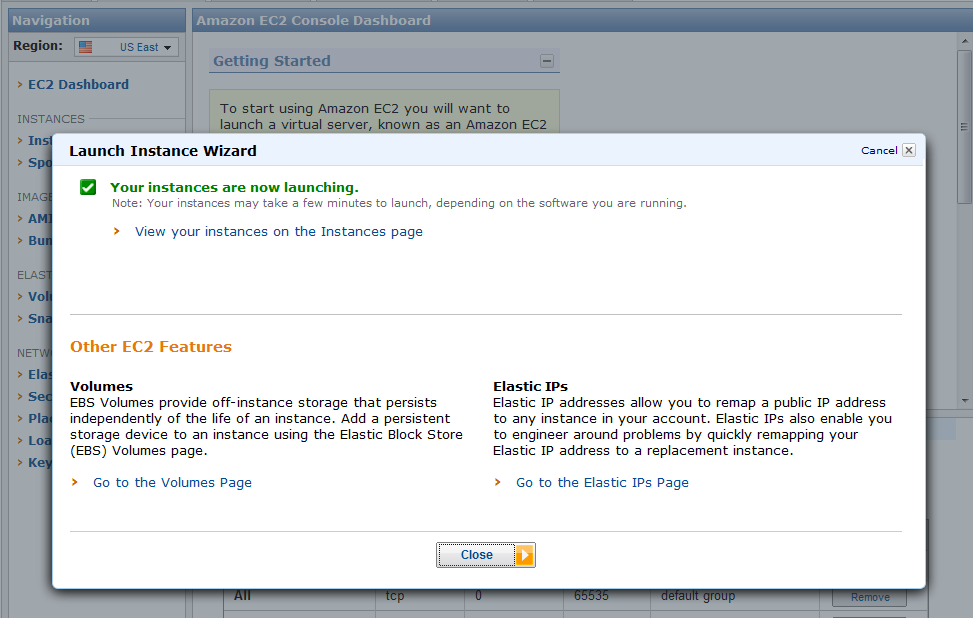


그림 20 위저드 화면 (최종 사용 화면)

이렇게 생성된 EC2 인스턴스를 선택하면, 아래와 같이 인스턴스의 상태정보를 확인할 수 있다. IP나 DNS, 모니터링 여부 등 시스템 설정에서 필요한 정보들을 보여준다.

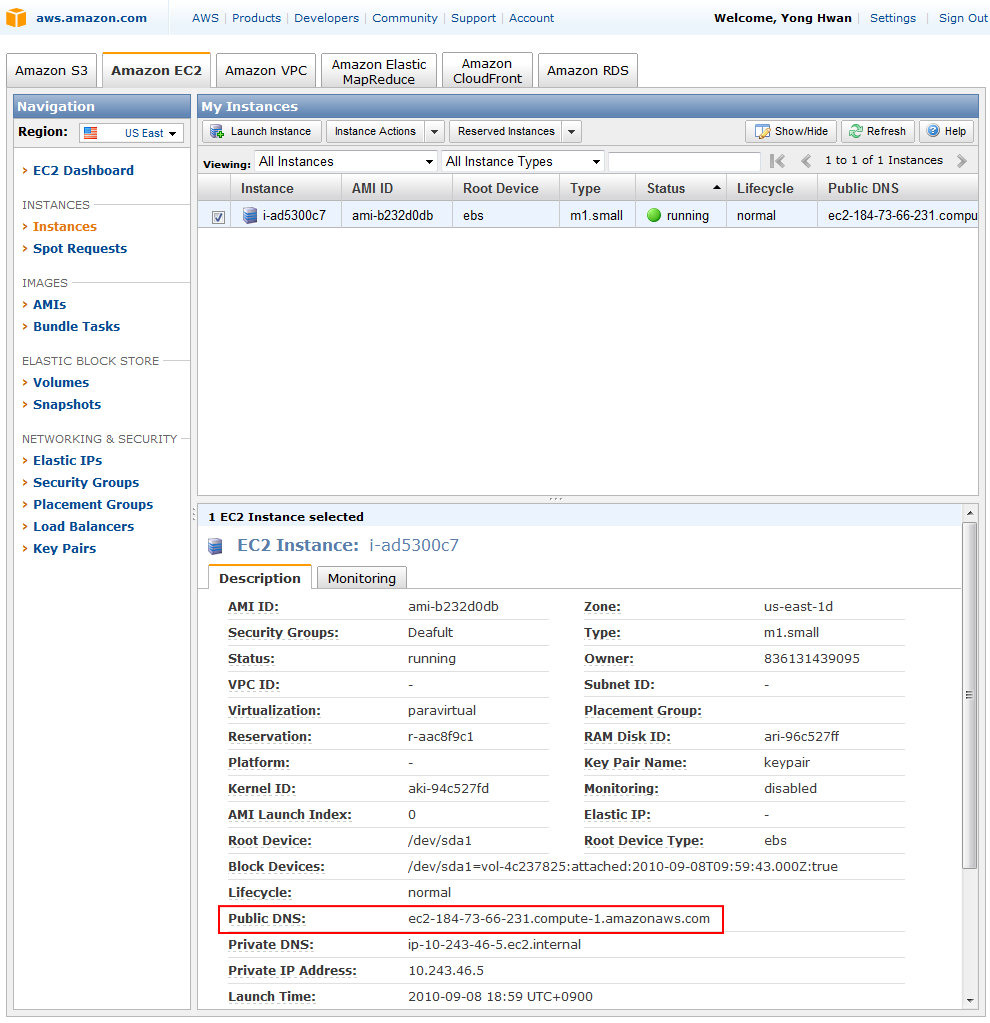


그림 21 EC2 화면

* 1. **EBS (Elastic Block Storage)**

EBS는 EC2 인스턴스에 첨부(Attach)될 수 있는 가상의 하드디스크(적확히 말하면 S3보다는 작은 분산 스토리지)이다. 하나의 EC2 인스턴스에는 여러 개의 EBS 볼륨이 마운트(mount)될 수 있으며, 하나의 볼륨 크기는 1GB~1TB이다. 실제로 저장될 때는 S3 서비스를 이용해서 저장되는데, 흥미로운 점은 분산 파일 구조를 채택하기 때문에 IO 퍼포먼스(Performance)가 상당히 높은 편이며, EBS는 부팅 파티션(Booting Partition)으로도 마운트가 가능하다.

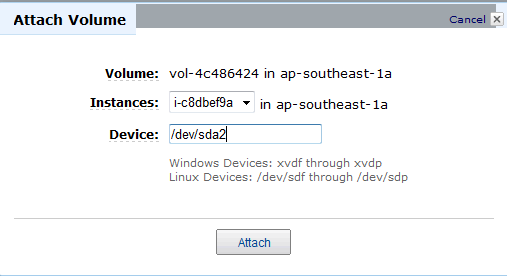


그림 22 EBS Mount 화면

또한 특정 시점에 EBS의 이미지를 S3에 저장하여 백업용도로 사용 가능하다

EBS의 라이프 사이클(life cycle, 수명주기)과 실제 사용에 대한 정책은 다음과 같다. 종료(Terminate)하지 않으면 계속 과금되는 구조를 가지고 있으며, 사용량과 사용시간에 대해서 비용을 지불하는 정책이 있다.

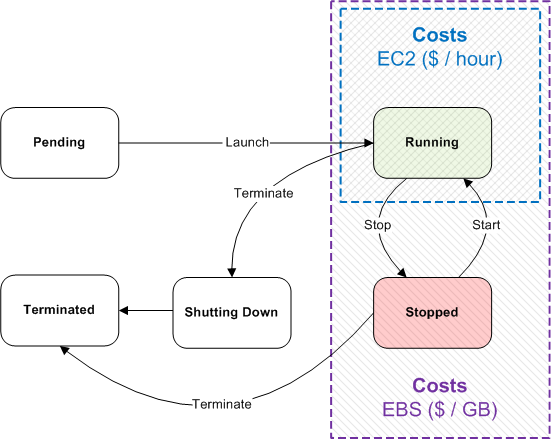


그림 23 EBS 비용구조

* 1. **S3 (Simple Storage Service)**

S3는 대용량 Blob 데이터에 대한 저장을 위해서 디자인 되었다. 파일, 이미지, 동영상과 같은 큰 사이즈의 데이터를 저장한다. 저장될 수 있는 데이터의 수와 크기는 제한이 없으며, 저장되는 데이터의 크기는 레코드당 1byte에서 최대 5GB를 지원한다.

S3는 Bucket(도메인 이름)을 생성하고, 파일을 올리는 구조로 되어 있다. http, https, Torrent에서 접근할 수 있는 API 인터페이스를 제공하고 있다.

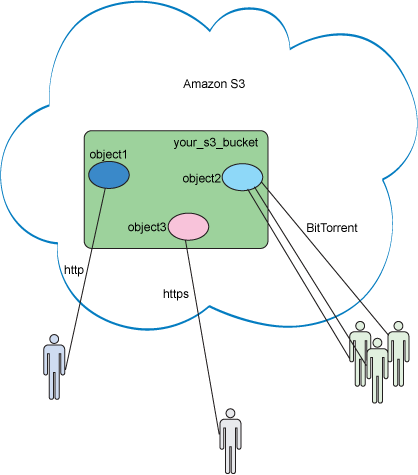


그림 24 S3 인터페이스 및 S3 구조

S3의 라이프 사이클(수명주기)과 사용에 대한 정책은 사용 중일 때에 시간대별로 과금 한다.

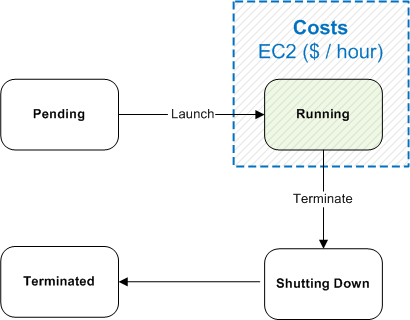


그림 25 S3 비용 구조

Smugmug에서는 아마존의 S3를 사진 데이터(Data)로 활용하여 사진 공유 서비스를 하고 있다. 파이어폭스(Firefox) 플러그인처럼 S3 연동을 지원하는 애플리케이션과 3rd party 플러그인과 여러 언어로 만들어진 라이브러리(library)들이 많이 공유되어 있다.

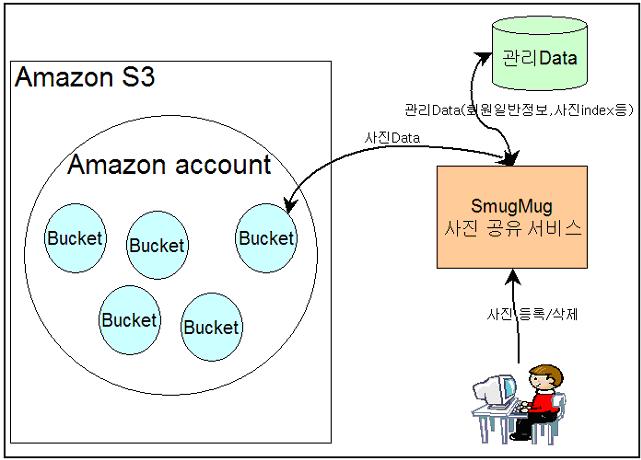


그림 26 Smugmug의 S3 사용예

S3에 파일을 올리는 과정은 그리 어렵지 않다. Bucket 하나를 만들고, 파일을 Upload하면 그림 27과 같이 웹에서 바로 확인할 수 있다.

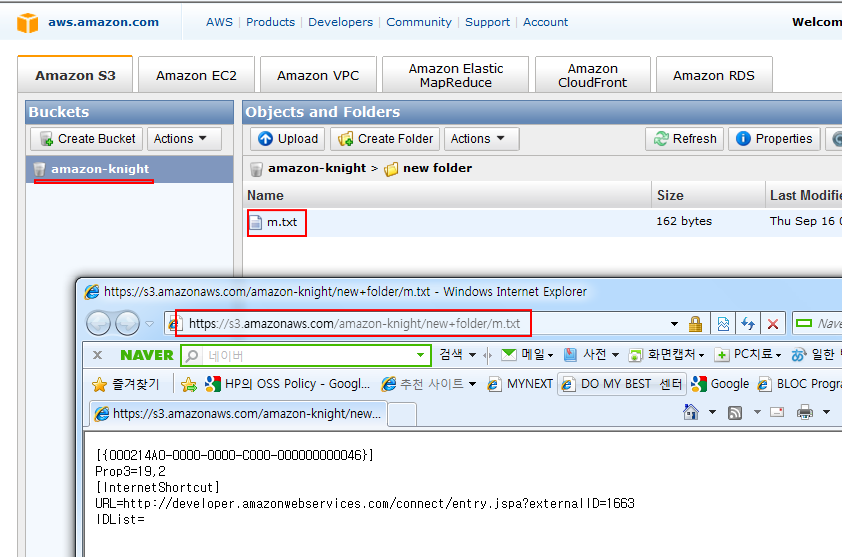


그림 27 S3 사용

참고로, S3에 대한 구현이 궁금할텐데, 위키피디아를 보면 공개가 되어 있지 않다고 되어 있지만, Dynamo:Amazon’s Highly Available Key-Value Store(<http://www.allthingsdistributed.com/files/amazon-dynamo-sosp2007.pdf>)을 근거로 프로토콜과 어떤 식으로 구현되었는지를 추측할 수 있다.

EBS와 S3를 비교할 수 있겠는데, S3가 훨씬 비싸고, 확장성이 좋다. 그러나, 가격과 속도측면에서는 EBS가 훨씬 좋다. EC2 인스턴스와 EBS와 S3간의 비교 및 API 내용은 다음과 같다.

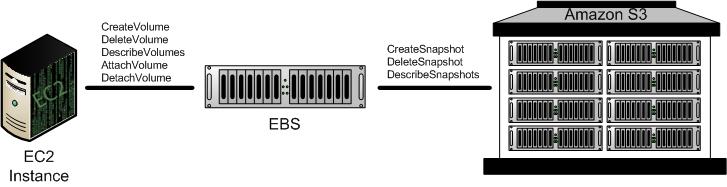


그림 28 EC2, EBS, S3 연관

* 1. **Simple DB**

Simple DB 서비스는 Key-Value 타입의 데이터를 저장하기 위한 데이터 저장소 서비스이다. 키 밸류 타입의 데이터를 저장하고, 대용량의 데이터 저장 및 빠른 검색을 지원하며, 밸류(Value)에 들어가는 데이터의 형에는 제약이 없다. 이런 특성을 Schemeless라고 한다. 관계형 데이터 저장이 필요 없는 데이터 구조에서 데이터 저장의 유연성을 부여해준다.

좀 더 구체적으로 설명하면 최초 인증하면서 받은 AWSAccessKeyId 안에 도메인(domain)을 지정하고, 그 밑에 있는 아이템(item)을 지정하고 속성(attribute)을 저장하는 방식을 사용한다. 도메인(Domain)/아이템(Item)/속성(Attribute)의 개념을 사용하여 마치 윈도우나 리눅스의 디렉토리처럼 쓸 수 있도록 하고 있다.

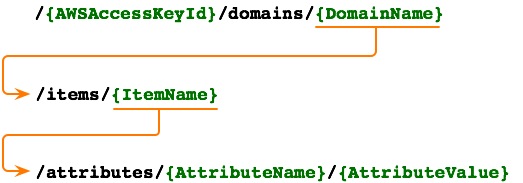


그림 29 도메인(Domain)/아이템(Item)/속성(Attribute) 연관도

RDB의 특성을 가지지는 않지만, 데이터 모델(Data Model)을 RDB처럼 쓸 수 있도록 지원이 가능하다. 다음의 예제는 RDB 처럼 사용할 수 있도록 Simple DB를 이용한 예제이다.

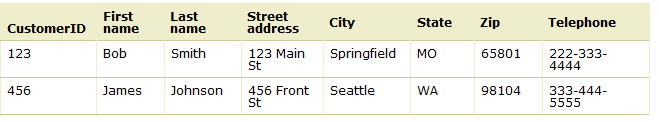


그림 30 Simple DB 예시

테이블 정보를 추가하려면, 다음의 정보를 API를 통해서 전달한다.

|  |
| --- |
| **PUT** (item, 123), (First name, Bob), (Last name, Smith), (Street address, 123 Main St.), (City, Springfield), (State, MO), (Zip, 65801), (Telephone, 222-333-4444)  **PUT** (item, 456), First name, James), (Last name, Johnson), (Street address, 456 Front St.), (City, Seattle), (State, WA), (Zip, 98104), (Telephone, 333-444-5555) |

만약 여기서 컬럼이 추가되는 경우에는 다음과 같이 쉽게 적용이 가능하다.

|  |
| --- |
| PUT (item, 789), First name, Deborah), (Last name, Thomas), (Street address, 789 Garfield), (City, New York), (State, NY), (Zip, 10001), (Telephone, 444-555-6666), (Email, dthomas@xyz.com) |

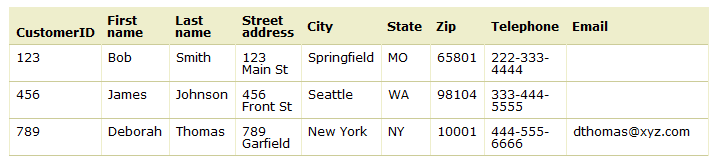


그림 31 Simple DB 예시

그러나, 단순(Simple) DB는 사이즈(size)나 길이(length)의 제한을 걸어 작은 크기의 데이터만을 수용하도록 설계되어 있다. 대용량 데이터는 서비스하지 않는다.

다른 특징 중의 하나는 지리적 복제(Geo Replication)이 가능하다는 것이다. Simple DB에 저장된 데이터는 물리적으로 떨어진 아마존의 데이터 센터에 복제되기 때문에 데이터의 접근성을 향상 시키고 장애 시 데이터에 대한 안정성을 보장한다.

Simple DB는 랜잭션(Transactions)기능을 제공하고 있는데, 특정 값이 맞으면 데이터를 추가 또는 삭제할 수 있다. (Conditional Puts/deletes) 그리고, 읽기 성능을 높이기 위해서 바로 추가/삭제/변경된 정보를 바로 반영하지 않는 정책(Eventually Consistent Read)을 디폴트로 사용하고 있다.

변경된 데이터에 대한 Consistent Read를 원한다면, 항상 일관성(consistency)을 보장할 수 있도록 API를 사용할 때 파라미터로 consistentRead key의 value를 true(consistentread=true)로 해서 전달하면, 추가/삭제/변경된 정보를 바로 반영해서 볼 수 있다.

Simple DB에 대해서 성능을 측정 내용(<http://huanliu.wordpress.com/2010/03/03/%EF%BB%BFthe-cost-of-eventual-consistency/>)에 따르면, 굉장히 빠른 속도를 보장하고 있다.

단일 쓰레드 기반 테스트 결과는 다음과 같다.

|  |
| --- |
| GetAttribute : 22ms  GetAttribute with ConsistentRead : 23ms  PutAttribute : 110ms  PutAttribute Conditional : 118ms |

병렬 쓰레드를 이용한 결과는 다음과 같다. Conditional/ConsistentRead 사용 차이에 대한 변화는 상당히 적은 편이며, 쓰레드가 많아지면서 속도가 저하되지는 않는다.

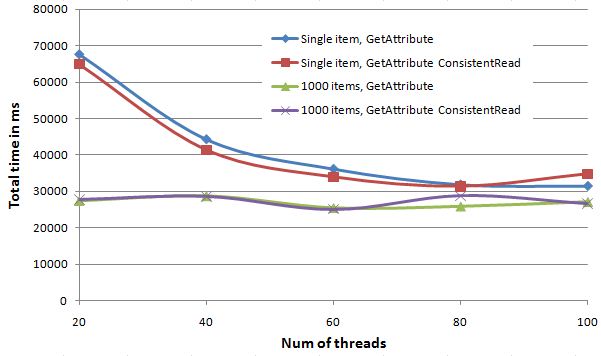


그림 32 병렬쓰레드 환경에서 GetAttribute 테스트 결과

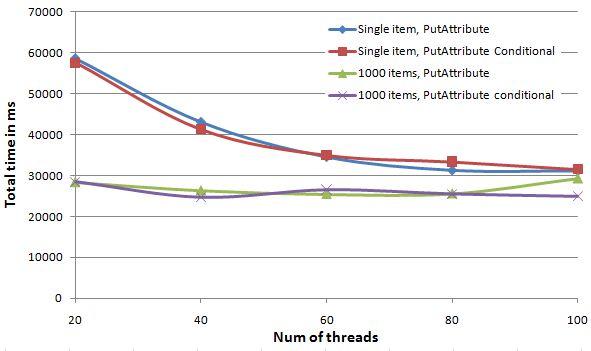


그림 33 병렬쓰레드 환경에서 PutAttribute 테스트 결과

그리고, API를 이용할 때는 Https 를 통한 SOAP과 REST 방식을 사용하여 안정성을 확보하려고 하고 있다.

다른 서비스와 달리 Simple DB는 과금 정책이 독특하게 처리되어 있다. S3보다 비싸면서, 비용이 소수점까지 나오기 때문에, 비용 청구구조를 파헤친 블로거가 있으니, 내용은 참조하면 좋겠다. (<http://www.daemonology.net/blog/2008-06-25-dissecting-simpledb-boxusage.html>)

* 1. **RDS (Relational DataBase Service)**

RDB 서비스는 MySQL 기반의 관계형 데이터 베이스 서비스를 제공하며, MySQL 기능(Feature)을 충분히(fully) 지원한다. AWS GUI 콘솔 또는 API 서버를 통해서 쉽게 다룰 수 있다.

데이터베이스 아키텍쳐 중의 하나인 Query-off loading 아키텍쳐를 지원한다는 것이다. 이 아키텍쳐는 Read Transaction이 많은 경우, 하나의 마스터(Master) DB에 Create/Update/Delete를 일으키고 여러 개의 종속된(Slave) DB에 데이터를 복사하여 여러 개의 Slave DB에서 Read 관련 트랜잭션(Transaction)을 수행함으로써 Read Transaction을 분산 시켜서 대규모 처리를 가능하게 한다.

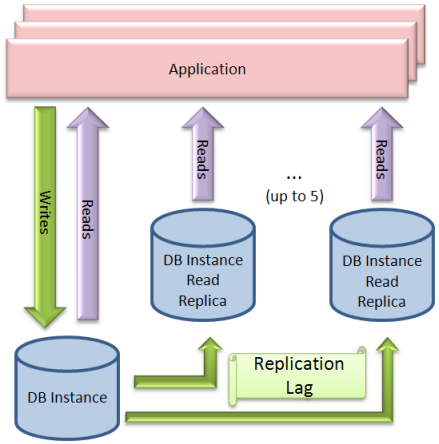


그림 34 RDS 특징

자동 회복(recovery)은 5분, 백업은 최대 8일 이내로 이루어지도록 되어 있다. 또한, 여러 지역으로 나누어진 AWS Zone(일종의 IDC개념)에서 자동 장애조치(failover) 기능을 제공하고 있다. 소프트웨어 패치를 하거나, 스토리지에 이상이 있거나, DB를 확장하기 위해서 DB 인스턴스를 종료(shutdown) 또는 up을 하는 상황, 네트워크 이상, 가용성 이상, 컴퓨팅(computing)에 이상이 있는 상황에 자동으로 장애조치가 되어 신뢰성을 높이는 작업을 진행하고 있다.

EC2와 마찬가지로 DB 인스턴트 클래스(Instance class) 를 지정할 수 있다.

|  |  |
| --- | --- |
| class | 정보 |
| Small DB Instance | 1.7 GB memory, 1 ECU (1 virtual core with 1 ECU), 64-bit platform, Moderate I/O Capacity |
| Large DB Instance: | 7.5 GB memory, 4 ECUs (2 virtual cores with 2 ECUs each), 64-bit platform, High I/O Capacity |
| Extra Large DB Instance: | 15 GB of memory, 8 ECUs (4 virtual cores with 2 ECUs each), 64-bit platform, High I/O Capacity |
| High-Memory Extra Large Instance | 17.1 GB memory, 6.5 ECU (2 virtual cores with 3.25 ECUs each), 64-bit platform, High I/O Capacity |
| High-Memory Double Extra Large DB Instance | 34 GB of memory, 13 ECUs (4 virtual cores with 3,25 ECUs each), 64-bit platform, High I/O Capacity |
| High-Memory Quadruple Extra Large DB Instance | 68 GB of memory, 26 ECUs (8 virtual cores with 3.25 ECUs each), 64-bit platform, High I/O Capacity |

표 4 RDS Type

* 1. **SQS (Simple Queue Service)**

SQS는 IBM MQ나 JAVA의 JMS와 같은 전통적인 Queue 서비스이다. Queue를 통해서 EC2 간의 신뢰할 수 있는메시징(Messaging)이나 비동기(Asynchronous) 아키텍쳐 구성을 지원한다.

무제한의 Queue와 Message를 생성할 수 있다. Queue에 저장되는 메시지는 개당 최대 64Kb까지 지원하며, 최대 14일까지 Queue에 저장될 수 있다. 동시에 읽기 및 전송이 가능하다. 읽기 단위의 락을 제공하다. 특히 무제한의 서버에서 무제한의 메시지를 일고 쓸 수 있도록 확장성(scalability)을 제공한다.

또한 가격이 저렴해서 만개의 요청(Request) 당 1센트 부과하는 정책을 사용하고 있다.



그림 35 SQS 특징

* 1. **SNS(Social Networking Service)**

일반적인 웹 서비스 모델은 클라이언트 요청에 대해서 웹 서버가 응답하는 구조로 되어 있다. 이에 비해서 알림(Notification)서비스는 서버가 클라이언트에 요청을 보내는 모델이다. 대표적으로 핸드폰의 SMS나 이메일 푸쉬 서비스 등이 이에 해당하는데, 아마존 AWS에서는 이러한 형태의 알람 서비스(Notification Service)를 제공한다.

아마존 AWS의 알람 서비스는 HTTP(HTTPS)와 SMTP(JSON) 프로토콜만을 지원한다.

기본적인 모델은 클라이언트가 주제(Topic)에 구독(Subscription)을 신청하면 등록된 클라이언트들에서 이벤트가 있을 경우 알람(Notification)을 보내주는 모델이다.

게시-구독(Publish-Subscription) 모델에서 게시자(Publisher)는 가장 먼저 주제(Topic)를 만들고, 이에 대한 정책을 지정한다. 그 이후에, Notification을 받고자 하는 구독자(Subscriber)가 구독(subscription)을 신청하면 된다. 게시자가 메시지를 주제에 게시(Publish)하면, SNS 는 구독자에 알람(notification)을 전달한다.

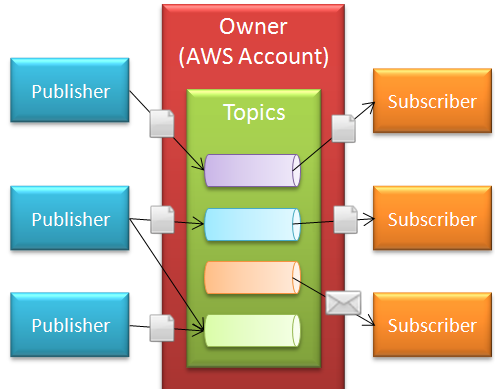


그림 36 SNS 특징

특별히 SNS는 SQS와 비슷하게 저렴한 가격에 판매를 하고 있다. 매달 100,000개 API 요청(Request)에 대해서 무료이고, 그 이후, 100,000개당 6 센트를 부과하는 정책을 사용하고 있다. 알람(Notification) 비용 청구는 Http와 SMTP 프로토콜에 따라 다른데. Http의 경우는 API 요청과 동일한 정책을 사용하지만, 이메일(Email)의 경우는 30~100배 정도 비싼 정책을 사용하고 있다.

* 1. **과금**

아마존 AWS 서비스를 사용한 결과는 한 달 후에 알 수 있다. 일반적으로 사용하고 나서 다음 달에 비용이 청구되는(지출되는) 종량제의 형태이다.



그림 37 과금 요청서

EC2의 On Demand Instance를 썼을 때는 종량제로 되어 한 달에 한 번씩 결제를 요청하는 메일이 날아온다. 그러나, 예약된 인스턴스(Reserved Instance)를 선택하면 1년 또는 3년 동안의 긴 시간 동안 사용할 수 있는데, 정액제로 지불할 수 있다.

* 1. **CloudWatch와 ELB (Elastic Load Balancer)**

CloudWatch는 모니터링 서비스로서, EC2, EBS, S3, ELB 을 대상으로 자원 사용량(resource ulitization), 성능을 모니터링한다. 만약 EBS나 S3의 디스크(Disk)에 결함(Fault)이 발생했을 때, 같은 존(zone, 영역) 또는 다른 존(IDC 개념)으로 데이터 백업(Data backup)이 이루어진다. EBS->EBS, EBS->S3, EBS->다른 zone의 EBS, S3->다른 영역의 S3로 이동할 수 있도록 지원한다.

EC2 인스턴스에 대한 시스템 모니터링 화면은 다음과 같다.

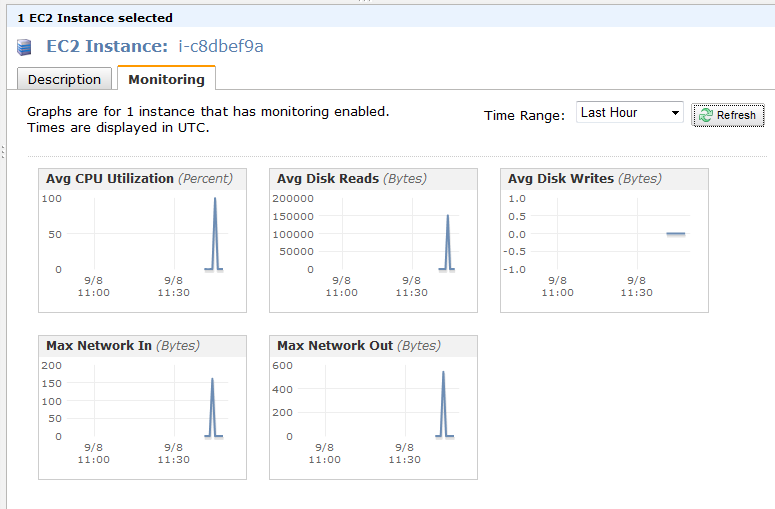


그림 38 EC2 모니터링

EC2 인스턴스에 마운트(Mount)된 EBS에 대해서도 개별적인 모니터링이 가능하다.

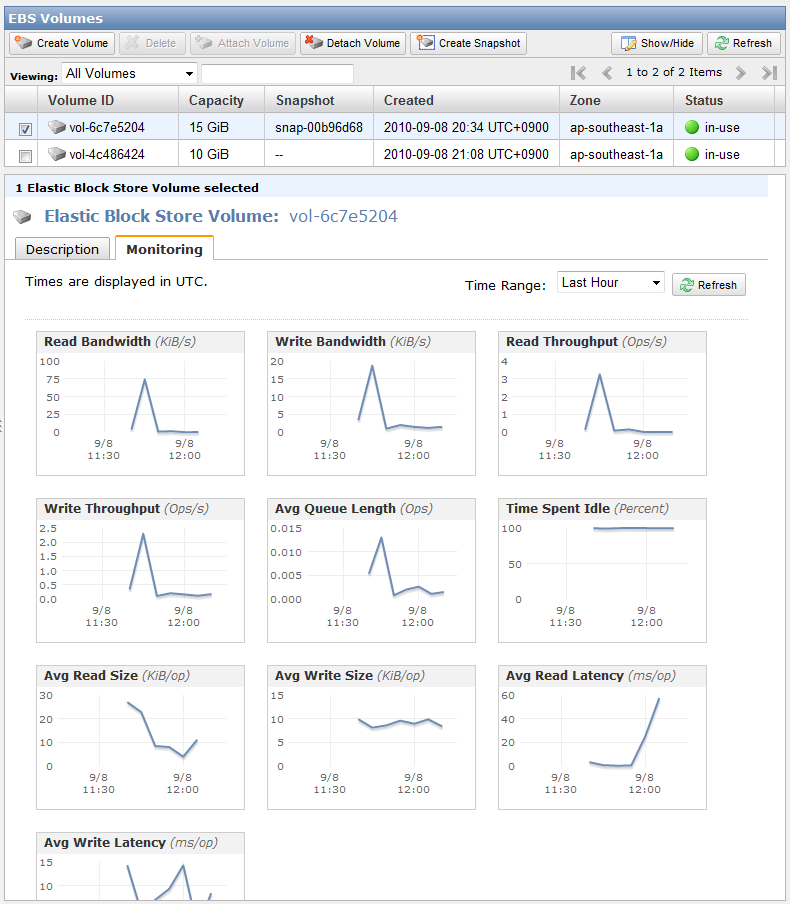


그림 39 EBS 모니터링

ELB는 자동으로 확장성(Scalability)을 지원할 수 있는 API를 제공하고 있으며, 헬스 체크(Health check)를 통한 환경설정이 가능하다. 웹 서버로 활용한다면, L7의 health check 기능과 비슷한 기능을 제공하고 있다. L4 스위칭 장비를 다루듯이 쓰레드홀드(Threadhold)값을 지정하여 시스템 엔지니어 도움 없이도 쉽게 셋팅 할 수 있다.

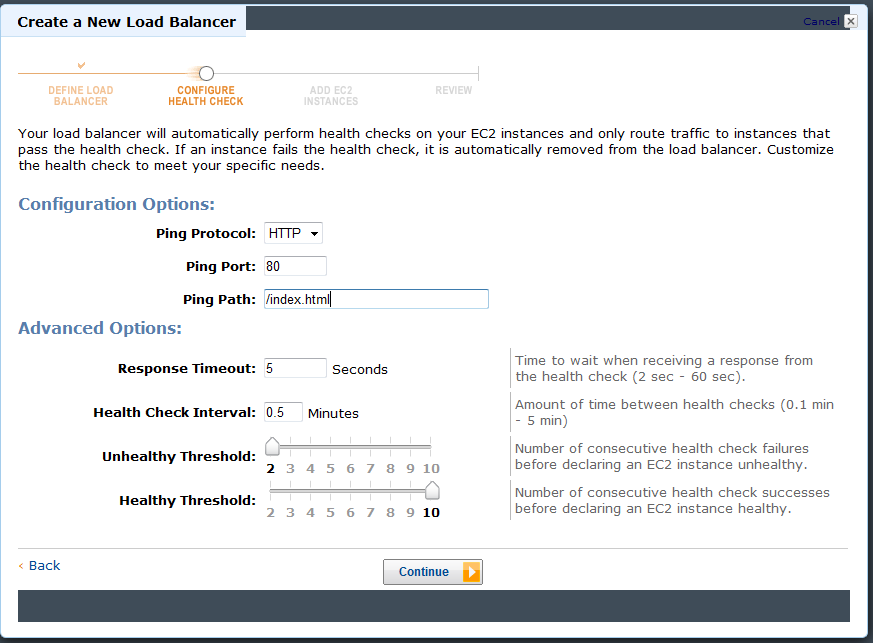


그림 40 ELB 설정 화면

참고로, AWS서비스에서는 일반적인 웹 서비스에 해당하는 L4에 해당되는 서비스는 없다. Health check를 가는 기능을 통해서 서비스의 가용성만 확인할 뿐이다. 실제로는 L4 바인딩이 없어서, AWS 서비스할 때는 3rd party DNS 업체를 이용해야 한다.

AWS는 자동 스케일링(AutoScaling) 기능을 활성화시키면 또 다른 EC2인스턴스가 복제되더라도 DNS 업체에 요청 또는 DNS장비를 셋팅을 해서 요청이 접속되게 해야 한다.

* 1. **기타서비스**

아마존 AWS에서는 기존에 생각하고 있던 IaaS, PaaS 영역을 넘어서서 클라우드 서비스에 대한 다양한 시도를 하고 있다. 운영에 필요한 것은 무엇인지를 많이 고민해온 것들이 서비스로 나와있다.

클라우드 사업을 하고 있는 다른 벤더와 달리 아마존AWS는 다양한 도전을 하고 있다.

VPC (Virtual Private Cloud, VPN 서비스 종류), AWS Import/Export (데이터 backup/recovery/직접 택배로 데이터를 전달), Fulfillment Web Service (Amazon FWS, e-commerce 송장/재고관리), FPS (Flexible Payments Service, 과금시스템), DeyPay 연동, Alexa Web Information Service (Web 분석), Alexa Top Sites (랭킹 서비스), SES (Simple Email Service, 이메일 서비스) 라는 서비스들을 제공하고 있다.

게다가 Amzazon Mechanical Turk이라는 인력 Pool 시장도 서비스하고 있다.

* 1. **운영**
  2. 안정적인 Multi-tenant (멀티 테넌트)

모든 리소스를 공유하는 인프라 기술을 Multi-tenant라고 한다. Gartner는 현존하는 Multi-tenant의 모델들을 그림 41과 같이 발표했는데, AWS의 경우에는 인프라와 플랫폼에 해당된다.

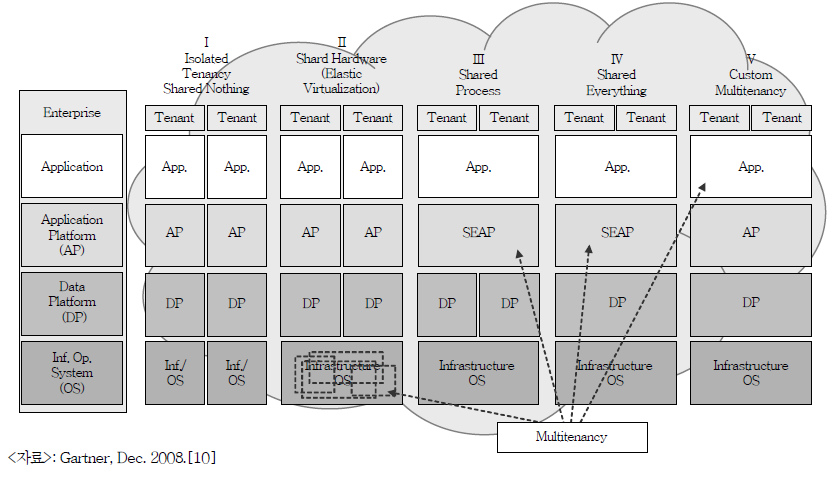


그림 41 Multi-tenant 종류

AWS 서비스는 문제의 한 서비스때문에 다른 서비스에 영향을 주지 않는 보안성(security)과 독립성(isolation)을 제공하고 있다. 컴퓨팅과 네트워크와 스토리지 자원에 대해서 호스팅, 데이터 마이그레이션, 백업, DDOS, 대용량 처리 등에 대해서 항상 일정한 서비스 수준을 유지시킬 수 있는 기술을 가지고 있다.

반면, SalesForce.com, MS Azure와 같은 SaaS 형태의 클라우드에서 이런 부분의 대한 고민이 많다. Database 또는 대용량 스토리지를 Service 단위로 이루어지는 것이 많기 때문이다. 저장공간에 대한 공유라는 점이 신뢰성을 높이기 위해서 Secure Multi-tenant 모델을 세우고, 관련 업체들과 함께 고민을 하고 있다.

반면, Oracle 클라우드 서비스의 경우는 Single-tenant를 제공한다고 발표하였다. 그 이유는 기능/보안 패치나 업그레이드을 임의대로 할 수 있고, 특정 시스템에 문제가 생겼을 때는 전체가 아닌 일부 고객에만 영향을 줄 수 있도록 하는 구조를 가지고 있다.

* 1. **SLA** (Service License Agreement)

Multi-tenancy 기술을 사용하여 비용을 절감하면서 장애나 비정상적인 상황에서도 서비스 품질에 대한 보장(QoS)이 되어야 한다. Qos에 대해서, 고객과 클라우드 서비스 사업자간의 품질의 계약을 SLA라 할 수 있다.

AWS 서비스 중 EC2와 S3에서만 공식적으로 SLA를 작성하였다. S3의 경우는 99.95%의 가용성(uptime/availability)을 보장하고, 그 밑으로 떨어지면 공제를 10~25%정도로 받을 수 있도록 명문화하였다. 한편, 다른 AWS 서비스의 SLA에 대한 공식적인 페이지는 없지만 이메일로 문의하면, 다른 AWS 서비스들도 99.9% 가용성을 보장한다고 알려주고 있다.

참고로 2010년에 S3의 가용성을 측정한 한 회사의 (http://www.quora.com/Amazon-Web-Services/How-stable-is-Amazons-S3-platform) 결과에 따르면, 굉장히 안정성이 높은 것으로 파악되고 있다.

|  |
| --- |
| S3 EU West: 99.979%,  S3 US East: 99.994%  S3 US West: 100%  S3 APAC: 100% |

1. **결론**

지금까지 AWS 서비스를 분석하였다. AWS의 문서들과 API 문서를 보면, 결코 하루 아침에 만들어진 것이 아니라는 것이 느껴진다. 쉽게 쓸 수 있도록 만들어 졌지만, 튼튼하고 강력한 IaaS, PaaS서비스들이 대단하기까지 하다. Amazon.com의 운영 노하우가 자연스럽게 AWS에 이어진 것이라 생각이 된다. 운영을 잘 아는 회사 Amazon AWS가 있어서 많은 도전이 되는 것 같다.