Ch01. 프로메테우스란 무엇인가?

1.1. Intro

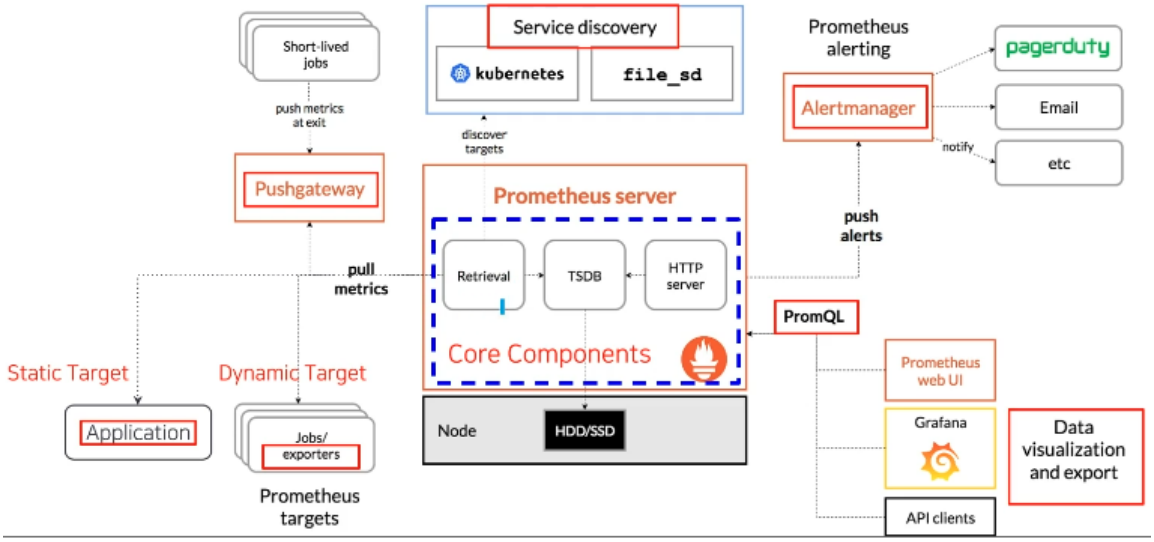
- make by Go, Apache 2.0 License
- 2nd Graduate CNCF Project, at 2016
- Monitored Code: Go, Java, C#/.Net, Python, Ruby, node.js, Haskell, Erlang, Rust
- Monitored Framework: k8s, docker, EC2, GCE, Openstack
- Existed Exporter: HAProxy, MySQL, PostgreSQL, Redis, JMX, SNMP, Consul, Kafka
- Base Data Model: Lable = Key-Value & Time-line > Used by PromQL, Integrated Grafana

1.2. What is Monitoring

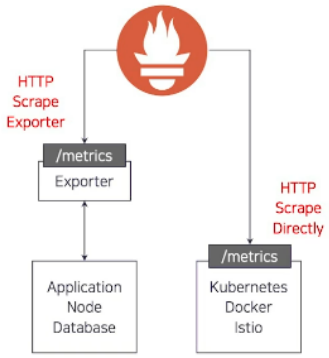
- Generalized Monitoring Usage
  - Alert
  - Debugging
  - Trending
  - Plumbing: made by Automic Monitoring Data Pipeline, and renewal by Pipeline
- **General Monitoring Scope: All is Event, Even Status is Event. AND Event has contexts**
  - (Event) Profiling: tcpdump? code debug build?
  - Tracing: Sparsed & Focused > Stack trace, Distributed trace(OpenZipkin, Jaeger)
  - Logging: Transaction log, Request log, Application log, Debug log
  - Metric: Simplify Specified Event Aggregation (e.g. Specified HTTP URL Request Metric)
- Prometheus is a metric-based monitoring system with cardinality characteristics. It’s NOT Logging. In other words, examplly, Prometheus focuses on the fact that there were 40 DB calls, 2 purchases, and 4 failures in the last minute. Details of all requests and information on required resources are of interest to Logging and Profiling.

1.3. Prometheus Architecture

- Architecture Diagram



- Metric: Pull Metrics from Pushgateway-Proxy and Exporters
  - Client libraries for instrumenting application code, a Pushgateway for supporting short-lived jobs
  - Special-purpose Expoters for services like HAProxy, StatsD, Graphite, etc
- Prometheus Server: scrapes and stores time series data(Events)
- Alertmanager: handle alert
- **Pull Metrics & Pull Scrap & Scrap Metrics**



1.4. Prometheus does fit or not?

- FIT:
  - Prometheus는 순전히 숫자로 된 시계열을 기록하는 데 적합합니다. 기계 중심 모니터링과 고도로 동적인 서비스 지향 아키텍처 모니터링 모두에 적합합니다. 마이크로서비스의 세계에서 다차원 데이터 수집 및 쿼리 지원은 특별한 강점입니다.
  - Prometheus는 중단 중에 신속하게 문제를 진단할 수 있도록 하는 시스템으로서 안정성을 위해 설계되었습니다. 각 Prometheus 서버는 독립형이며 네트워크 스토리지 또는 기타 원격 서비스에 의존하지 않습니다. 인프라의 다른 부분이 손상되었을 때 이를 사용할 수 있으며 이를 사용하기 위해 광범위한 인프라를 설정할 필요가 없습니다.
  - 하나의 장소에 Metrics정보가 저장되고, 필요시 자유롭게 다시, 확인 및 활용 가능하고, 시각화에 유리
  - 수 많은 Metrics를 직접 매번 취급하지 않아 가벼우며, 빠른 파악과 알림에 유리
  - Pull기반이 아닌 경우, Pushgateway에 의한 수집 구성도 가능
- NOT FIT:
  - 프로메테우스는 신뢰성을 중시합니다. 오류 상태에서도 시스템에 대해 사용 가능한 통계를 항상 볼 수 있습니다. 요청당 청구와 같이 100% 정확도가 필요한 경우 수집된 데이터가 충분히 자세하고 완전하지 않을 가능성이 있으므로 Prometheus는 좋은 선택이 아닙니다. 이러한 경우 청구를 위해 데이터를 수집하고 분석하는 데 다른 시스템을 사용하고 나머지 모니터링을 위해 Prometheus를 사용하는 것이 가장 좋습니다.

    - 100%보장이 없으므로, 빌링데이터나, 어떤 사실의 근거로 사용하는 경우, 시계열 메트릭이 아닌, 인덱싱 성향의 로깅성격인 경우

1.x. Added

- 최신의 공식 소개 내용은 다음을 참고: https://prometheus.io/docs/introduction/overview/

+==