Big Data for Operational Insights

Felix Gorodishter | @fgorodishter | felix@godaddy.com
Who am I?

• Felix Gorodisher
• Speak fluent Russian 😊
• Deving since ‘96
• With GoDaddy since ‘09
• Currently Principal Architect

Contact:
• felix@godaddy.com
• @fgorodisher
GoDaddy ...

Our vision is to radically shift the global economy toward life-fulfilling independent ventures.

- 14.5 M Customers worldwide (56 markets)
- 63 M Domains under management
- 10 M Websites hosted / 9 Datacenters
- 15 B DNS requests daily
- 2 B Attacks blocks monthly
- 37 K Servers
- 5000+ Employees
There are two approaches ...

1. Plan what you ingest based on what you’ll actually use
2. Ingest everything and figure it out later

* This project started 2013
Hadoop

Data Collection
- SQL Stores
  - MySQL
  - MSSQL
- NoSQL Store
  - Cassandra
- 3rd Party Systems
  - GD Applications {Events, Logs, …}
- GD Applications {Events, Logs, …}

Data Platform
- Business Insights
  - Google Analytics
  - Tableau
- Batch Processing
  - Scale out BI: Data Egress
  - Unified Data Set
    - Hive
    - Pig
    - Spark
- Data Serving & APIs
  - Product Serving
    - MySQL
  - Redis / Cache
  - Cassandra
  - Hadoop
- Real-Time Processing
  - Elasticeirch

Customers
- 3rd Party Decision Engine
- Marketing Omnichannel
- Products
  - FOS / WSB / C3 / …

6PB Managed → 21 PB by EOY
13 TB New data per day in HDFS
200K Messages per second
Data ingest

Requirements
• Collect application & system logs from Linux & Windows servers
• Local port where users could HTTP POST and UDP send messages
• AUTH around data access
• Delivery guarantees

First Attempt

Challenges
• Old Kafka versions (< 0.9) lacked AUTH / Security
• Teams had to work hard to support protobuf + Kafka

Apps write direct to Kafka in protobuf
**Data ingest**

**Current State**

- GD Applications (Events, Logs, …)
- Agent
- Proxy
- Kafka

**What we did**

- Wrote agents for Linux and Windows
- Agent exposed local port on every server so teams can natively ship data over HTTP (and UDP too)
  ```sh
curl
  -H "Content-Type: application/json"
  -X POST
  -d '{"fqdn":"hostname", "data":"felixtest"}'
  http://localhost:<PORT>/v1/foo/bar
  ```

- Data from hosts is WRITE-ONLY into pipeline
Data ingest – for Operations

Our Agent(s) – CPM (Collector Process Manager)
- Operations/SRE needed more primitives
- Built it to be pluggable – Python on Linux & C# .NET on Windows
- Always ship base meta-data about sender
- Allow for tail or scheduled workloads

Linux
- `/var/log/*` (known useful stuff)
- `/etc/passwd`
- `/etc/group & /etc/login.groups`
- `/etc/yum.conf & /etc/yum.repos.d/*`
- `rpm -qa`
- `yum check-update`

Windows
- Application – event log
- System – event log
- Security – event log

Per Message Meta-data

```
[gd-linux-system-collector]
type = private
dc = phx3
env = staging
server_role = hypervisor
service_zone = phx-private-gen-zone-1
security_zone = mgmt
product_name = compute
```
Other feeds

- Syslog
- sFlow / Netflow / Packet
- Network Route
- SNMP Traps
- IP SLA
- Metrics – Server & Appliances
- APM
- External Alarms
- And many more!
Winning Patching

Q: Are you patching?
A: Isn’t that just magic?
Patching – What is it?

- Measure and report on the compliance and risk of our server fleet
- Support static and ephemeral infrastructure
- Support Windows & Linux
- Provide transparency in the data and collection
- Give the raw data to the teams
- Leverage the same data for ops to exec reporting
Business Service Mapping (BSM)

We leverage 4 layers:

- **Business Unit (BU)**
- **Product Line (PL)**
- **Business Service Rollup (BSR)**
- **Business Service (BS)**

```
<table>
<thead>
<tr>
<th>BU</th>
<th>CPO Products</th>
</tr>
</thead>
<tbody>
<tr>
<td>PL</td>
<td>Hosting</td>
</tr>
<tr>
<td></td>
<td>Productivity</td>
</tr>
<tr>
<td>BSR</td>
<td>cPanel Hosting</td>
</tr>
<tr>
<td></td>
<td>Email</td>
</tr>
<tr>
<td>BS</td>
<td>cPanel HTTP</td>
</tr>
<tr>
<td></td>
<td>cPanel APP</td>
</tr>
<tr>
<td></td>
<td>Office 365</td>
</tr>
</tbody>
</table>
```
Patching

Once per hour, each host sends all available updates.
Every 4 hours ETL via Sqoop

Realtime via CDC

Platform Ingest

Nightly ETL via Python

Realtime: All Servers

Stores all streaming source data prior to processing.

Transforms CPM data feeds into aggregates for reporting, and SNOW BSM into relational view.

Daily snapshots and current view of all data from sources.

Accessible for anyone to integrate or query.

22 nightly jobs transform raw and aggregated data into report. Output to both HIVE and Elastic.

Data is exposed to everyone for real-time debug and reporting via dashboards and rich visualizations.
Patching: Approach & Learnings

- Servers are the source of truth
- Prioritize some data sources above others

```sql
coalesce(cpm.compliant, sccm.compliant, spacewalk.compliant, spacewalk_pki.compliant, desktop.compliant, 'unknown') compliant
```

- Store intermediate tables
- Make process / scripts idempotent
- Partition or store separate tables for output (especially keep forever*)
  - patching_2017031400 (%Y %m %d %H)
- Create views

```sql
CREATE VIEW patching AS SELECT * FROM patching_2017031400;
```

- Track performance / success
- Build example / canned reports at different levels and allow people to fork them

* For some value of forever that’s cost effective
But wait, there’s more!

- Replaced our SIEM with Elastic + Hadoop
- Page load / RUM Analysis & Anomaly Detection
- Monitoring & Alert Correlation / Analysis
- OpenStack Allocation vs. Utilization Reporting
- Infrastructure Capacity Planning
If I were to do it today ...

Evaluate open source tooling ... it has improved tremendously!

logstash  fluentd  Flume
If I were to do it today ...

Enforce schema + Schema Registry
Guess what .... We’re hiring!

x.co/jobplz | godaddy.com/jobs

Arizona, California (SD, LA, SF, Sunnyvale), Iowa, Massachusetts, Washington, and more!

Questions?

Felix Gorodishter
@fgorodishter
felix@godaddy.com
Rate me: x.co/rateplz