Stories from the Trenches at GoDaddy: How Big Data Insights Equal Big Money

Felix Gorodishter, Principal Architect

GoDaddy

2018.03.01

@fgorodishter

Who am I?

- Felix Gorodishter
- Speak fluent Russian ☺
- Deving since ‘96
- With GoDaddy since ‘09
- Currently Principal Architect
- Started using Elastic v0.9

- Contact:
    - felix@godaddy.com
    - @fgorodishter

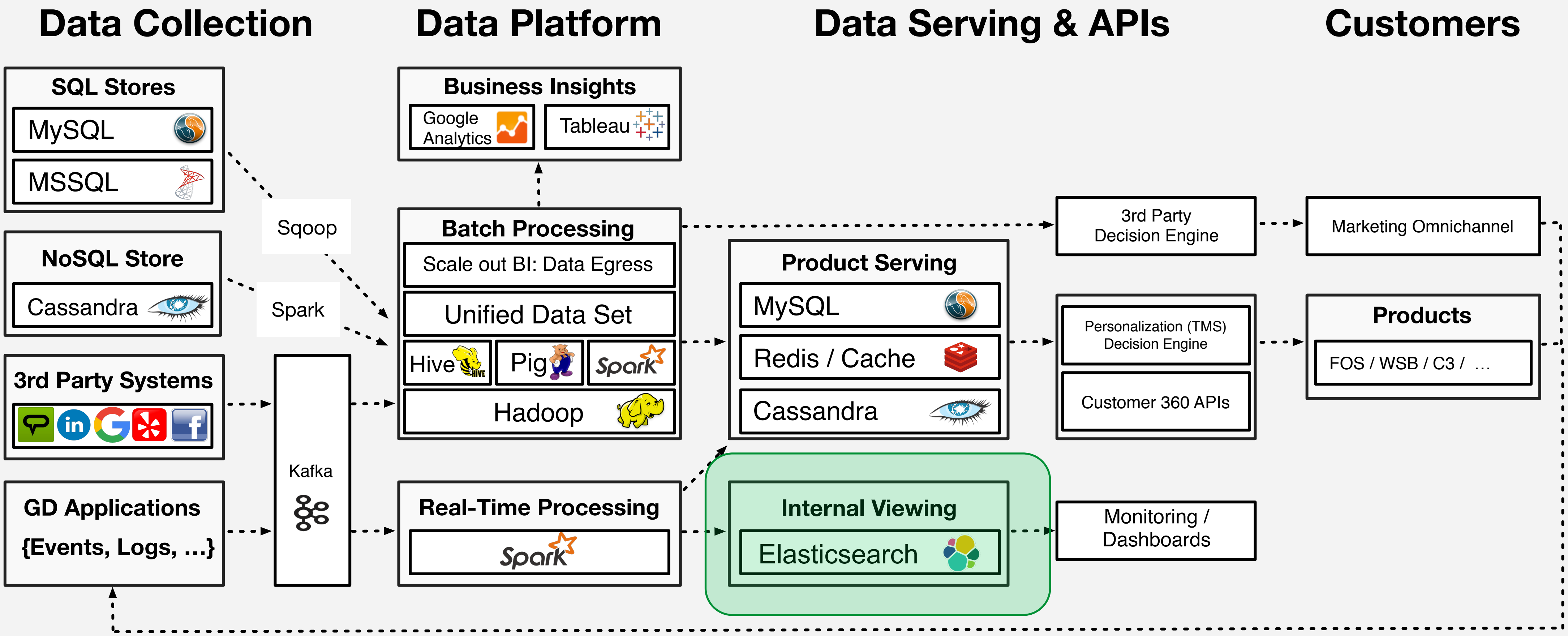
GoDaddy …

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

- 17.3 M Customers worldwide (56 markets)
- 75 M Domains under management
- 10 M Websites hosted / 24 Datacenters
- 18 B DNS queries daily
- 2 B Attacks blocked monthly
- 85 K Servers
- 7000 Employees

GoDaddy

Data Flows



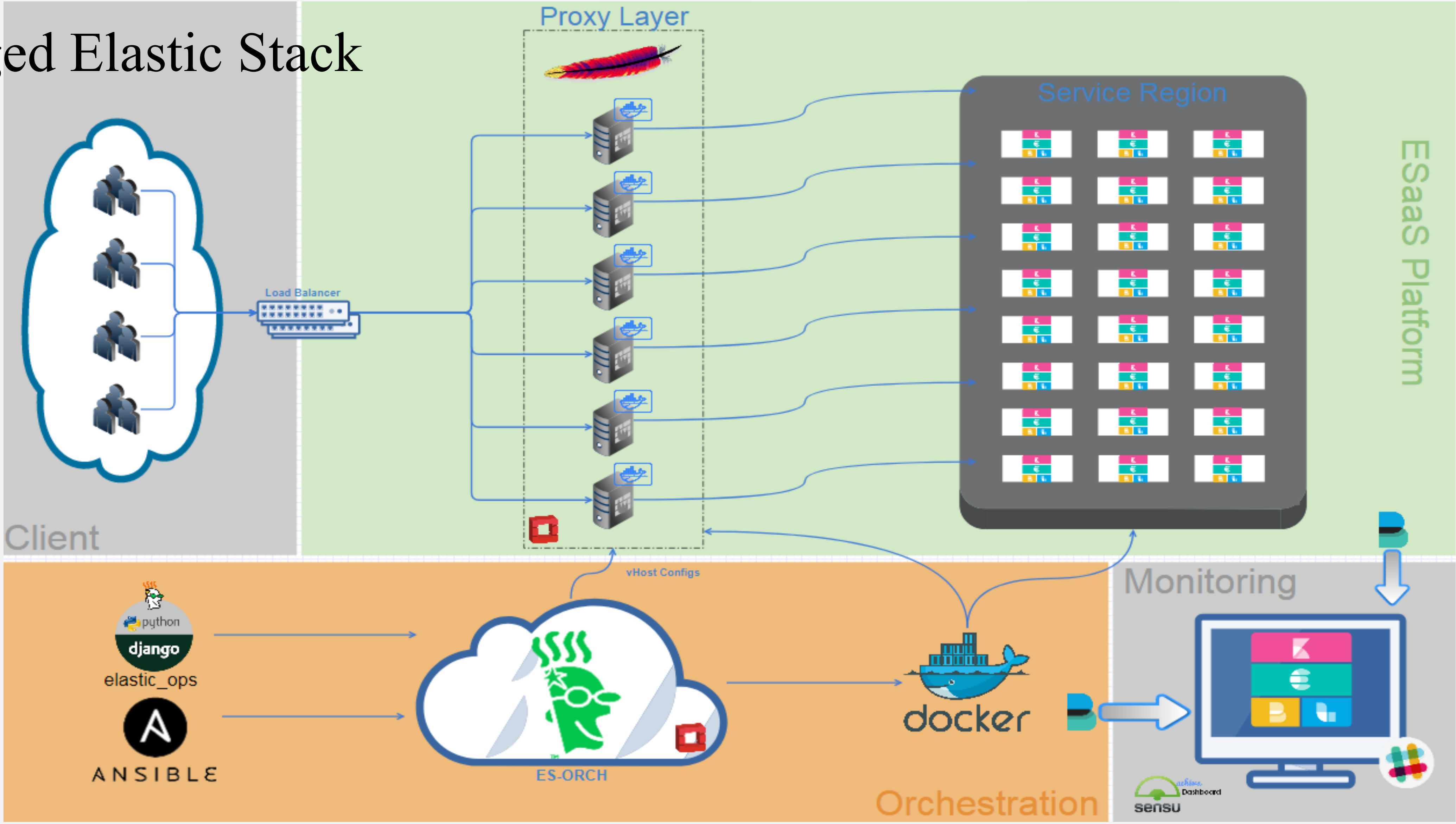
11 PB Managed HDFS

13 TB New data per day in HDFS

200K Messages per second

GoDaddy

Managed Elastic Stack



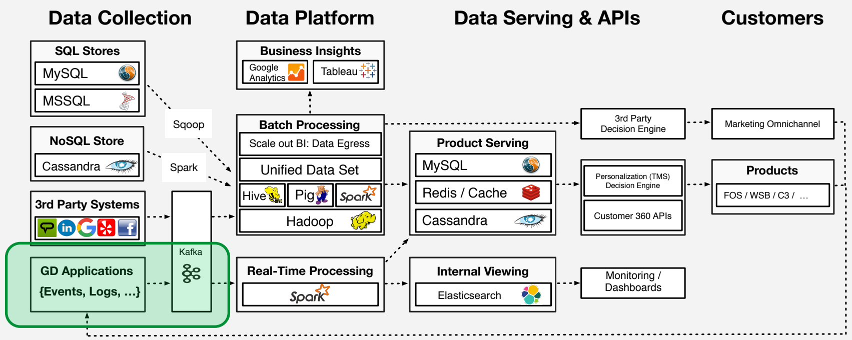
61 Managed Clusters

766 Containers

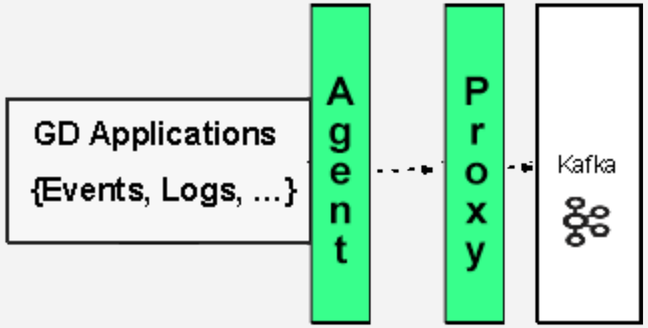
271 TB Indexed Data

GoDaddy

Data Collection



Current State



What we did

- Wrote agents for Linux and Windows *(in 2014)*
- Agent exposed local port on every server so teams can natively ship data over HTTP (and UDP too)

```
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 Collection – 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

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
```

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

GoDaddy

GoDaddy™

Winning Patching

Q: Are you patching?

A: Isn't that just magic?

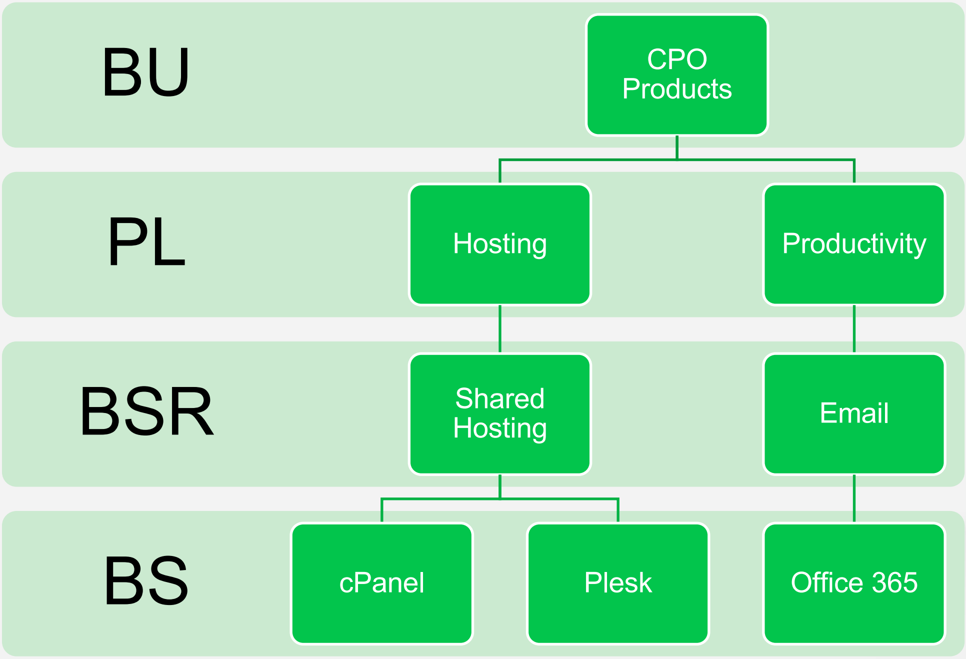
Patching – What is it?

- 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:

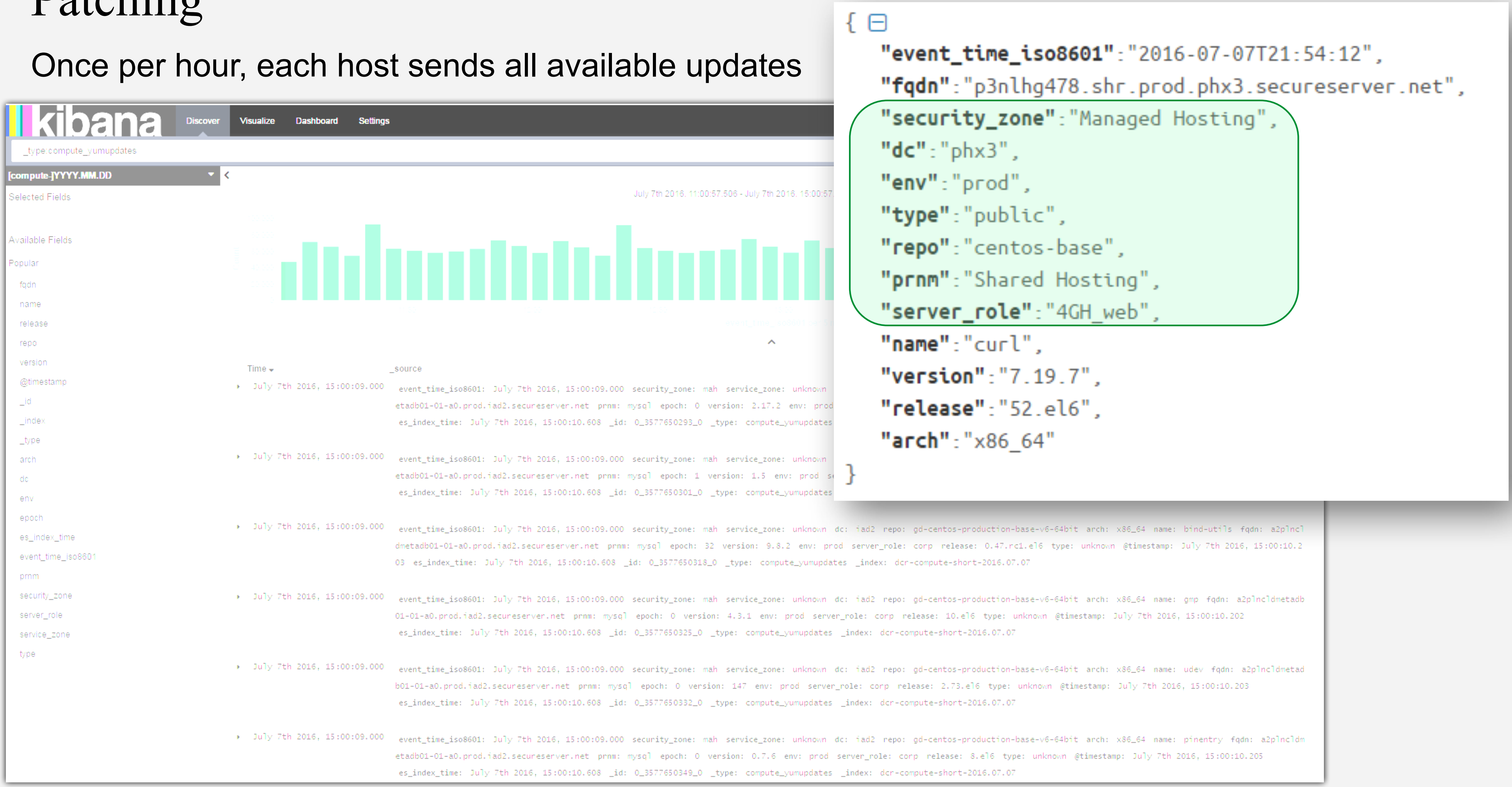
- Business Unit (BU)
- Product Line (PL)
- Business Service Rollup (BSR)
- Business Service (BS)



GoDaddy

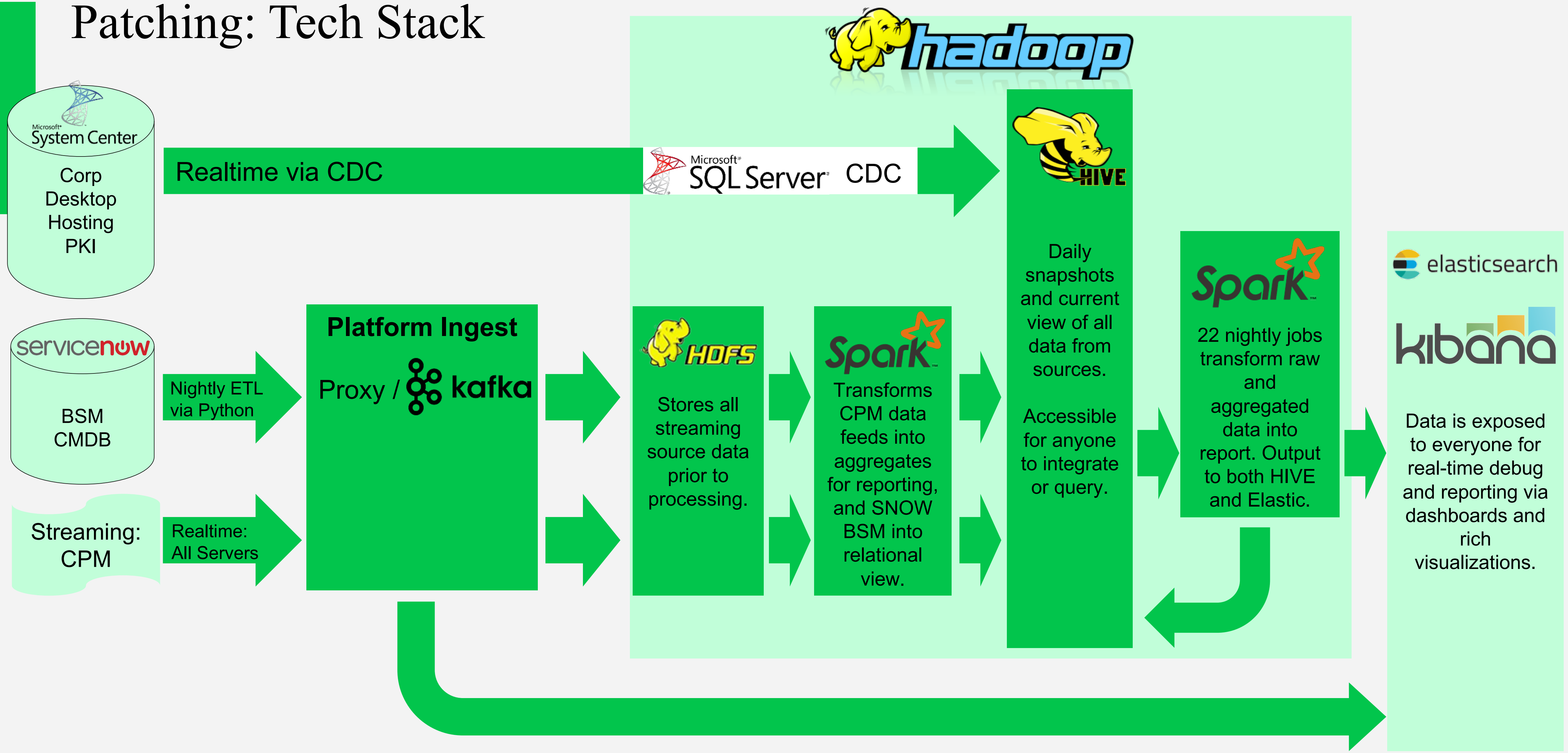
Patching

Once per hour, each host sends all available updates



GoDaddy

Patching: Tech Stack

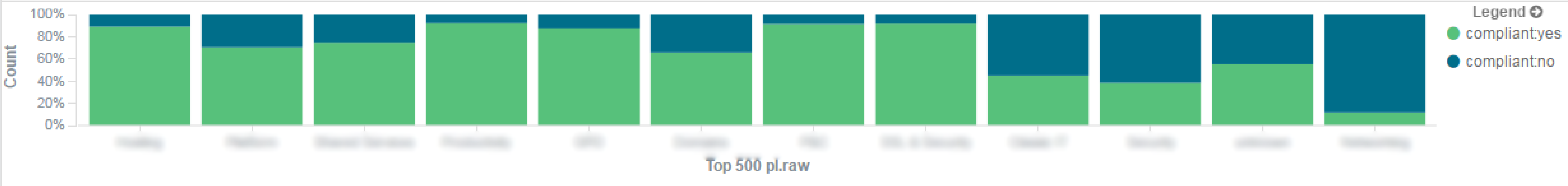


GoDaddy

Patching: PL Listing - Since Patch Tuesday

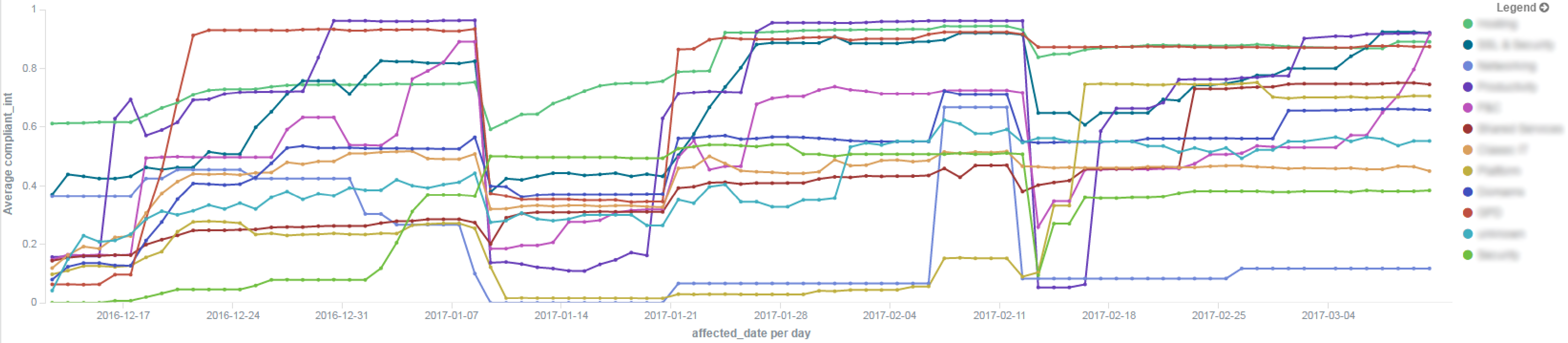
Top 500 bu.raw	Top 500 pl.raw	Count	Sum of compliant_int	Average complian
[illegible]	[illegible]	7,701	6,856	0.89
[illegible]	[illegible]	1,151	1,061	0.922
[illegible]	[illegible]	955	628	0.658
[illegible]	[illegible]	481	440	0.915
[illegible]	[illegible]	2,652	1,869	0.705
[illegible]	[illegible]	2,304	1,718	0.746
[illegible]	[illegible]	269	121	0.45
[illegible]	[illegible]	171	157	0.918
[illegible]	[illegible]	138	53	0.384
[illegible]	[illegible]	17	2	0.118
[illegible]	[illegible]	794	694	0.874
unknown	unknown	67	37	0.552

Patching: Product Lines Rollup - Since Patch Tuesday



Patching: Product Lines - Since Patch Tuesday

Patching: Product Lines Compliant Percentages



Patching Look-back Heatmap

pl.keyword: Descending

0.958	0.958	0.944	0.95	0.956	0.958	0.918	0.967	0.919	0.974	0.969	0.938	0.973	0.975	0.909	0.895	0.74	0.945
0.907	0.86	0.883	0.897	0.869	0.852	0.812	0.985	0.982	0.965	0.982	0.832	0.992	0.987				
0.893	0.887	0.768	0.802	0.857	0.907	0.905	0.936	0.845	0.927	0.748	0.891	0.927	0.732	0.866	0.852	0.4	1
0.891	0.898	0.776	0.913	0.742	0.934	0.878	0.876	0.438	0.911	0.916	0.944	0.956	0.813	0.961	0.399	0.375	0.398
0.745	0.711	0.775	0.727	0.764	0.666	0.845	0.917	0.869	0.754	0.716	0.747	0.946	0.745	0.519	0.592	0.83	0.945
0.649	0.361	0.479	0.449	0.494	0.713	0.626	0.659	0.682	0.933	0.979	0.518	0.953	0.96	0.886	0.835	0.317	0.905
0.632	0.211	0.361	0.676	0.258	0.645	0.111	0.214	0.357	0.929	1	0.929	1	1	0.929	0.889	0.778	1
0.592	0.614	0.588	0.599	0.478	0.56	0.511	0.821	0.823	0.89	0.908	0.681	0.949	0.959	0.88	0.905	0.726	0.977
0.592	0.685	0.485	0.333	0.348	0.521	0.407	0.782	0.721	0.877	0.743	0.502	0.918	0.815	0.851	0.858	0.425	0.758
0.486	0.158	0.175	0.275	0.222	0.159	0.681	0.069	0.143	0.268	0.203	0.139	0.226	0.189	0.179	0.074	0.654	0.89
0.39	0.324	0.386	0.622	0.305	0.48	0.743	0.791	0.8	0.858	0.865	0.879	0.849	0.788	0.762	0.285		
0.119	0.66	0.363	0.415	0.39	0.534	0.407	0.194	0.148	0.908	0.883	0.051	0.389	0.348	0.502	0.685	0.36	0.91
							0.935	1	0.9	1	0.233	1	0.958				
								1	1	1	1	1	1	0.72	1	1	1
													1	0.917	0.972	0.718	0.995
														1	1	1	1
														0.986	0.986		
														0.958	1	0.972	1
														0.711	0.933	0.148	0.961
															0.882	0.803	0.893
																0.953	1
																0.109	0.127

2016-09-30 2016-12-31 2017-03-31 2017-06-30 2017-09-30 2017-12-31

Patch Month

GoDaddy

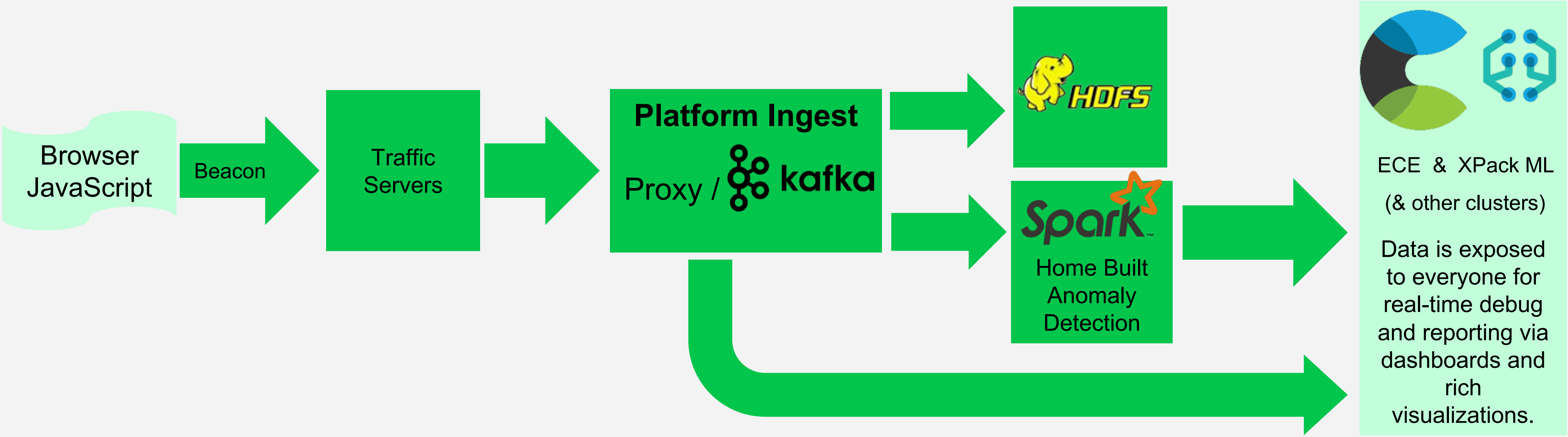
RUM / User Events

Q: Isn't that just GA?

RUM / User Events

Our JS – Traffic2

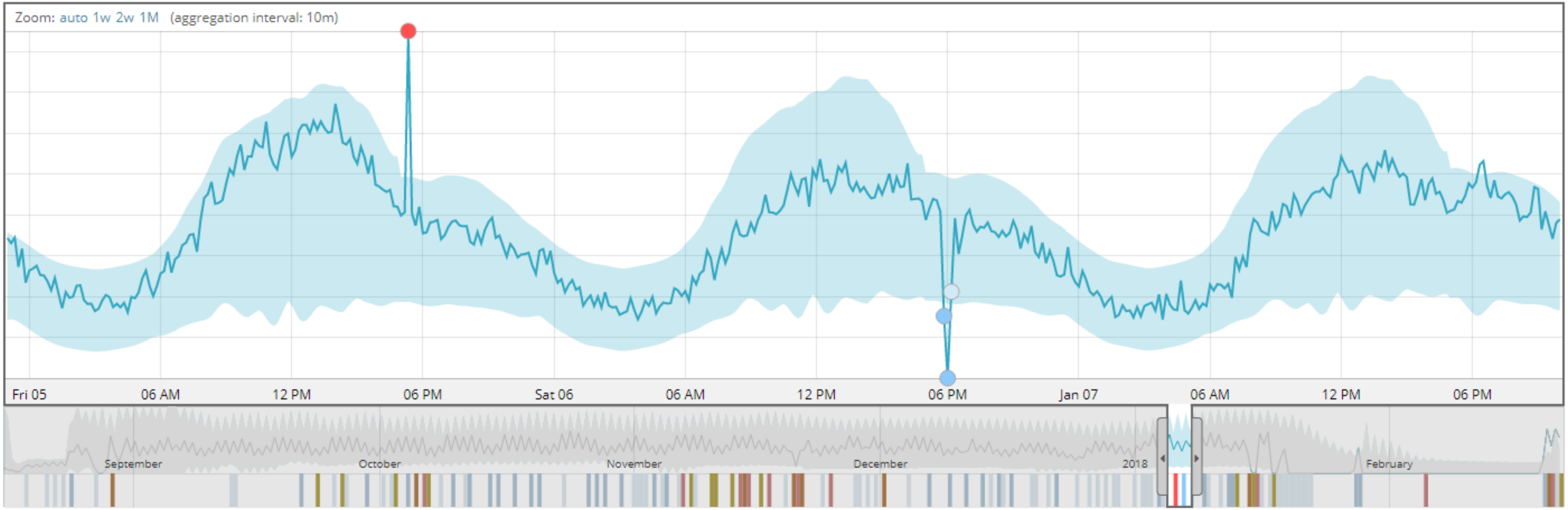
- 100% fidelity clickstream / event data
- Ability for teams to act quickly on streamed data
- Ability to join data to other datasets – ie. network monitoring / flow
- Support for our split testing & personalization frameworks



User Events

GoCentral Product

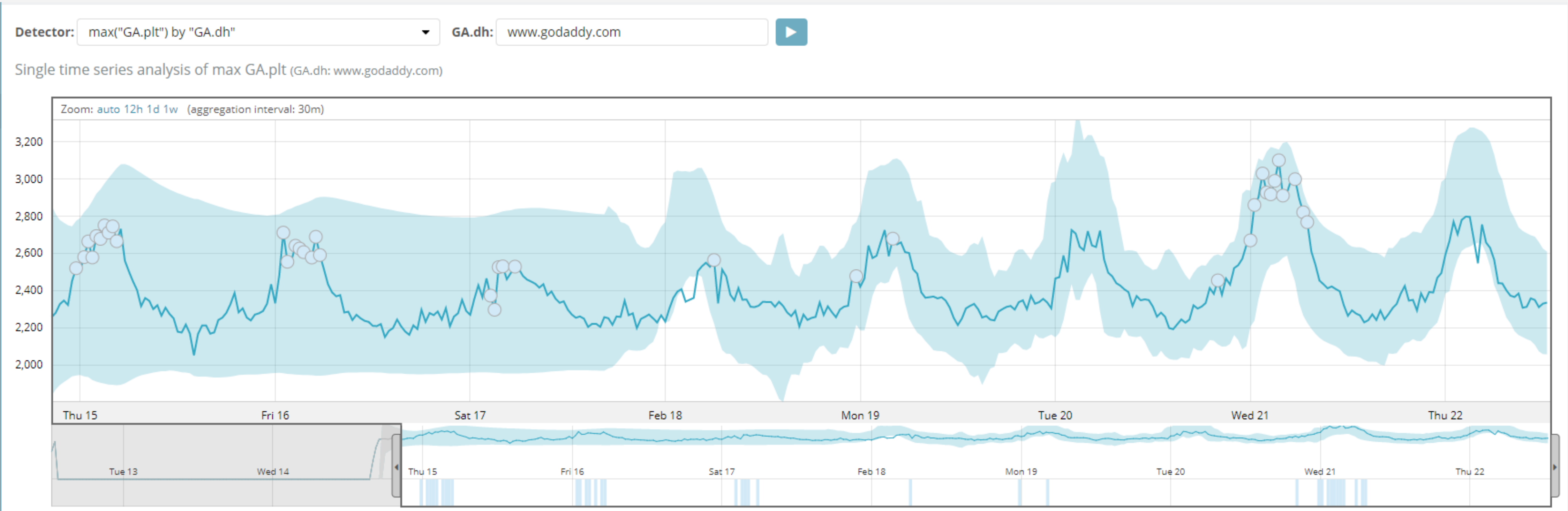
- We track every aspect of customer interaction / lifecycle via Traffic2
- Recently started to analyze this via ECE + XPack ML



RUM - **Facets / Findings**

- Analyze by Source Geo → Datacenter → Page/Site
- 75th Percentile is most useful for ML on this dataset
- Top 1000 sites are interesting – but unique every hour/day
- We leverage Advanced ML job with aggregations:
  - date_histogram by 5m
  - terms agg top N
  - percentiles 75 for page load time (and other timings)


```
{
    "cts": "1519286437470",
    "dh": "woo.godaddysites.com",
    "dl": "woo.godaddysites.com/foo",
    "dp": "/foo",
    "ua": {
      "orig": "Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 ...",
      "browser": {
        "..."
      },
      "os": {
        "name": "Windows 7",
        "..."
      }
    },
    "ds":"2000",
    "ClientIp": "172.17.249.139",
    "ClientIp_Geo": "...",
    "plt": 2718,
    "dns": 5,
    "tcp": 195,
    "srt": 178,
    "pdt": 445,
    "rrt": 730,
    "dit": 1628,
    "dct": 1629,
    "@timestamp": "2018-02-22T08:00:25.844Z"
  }
```

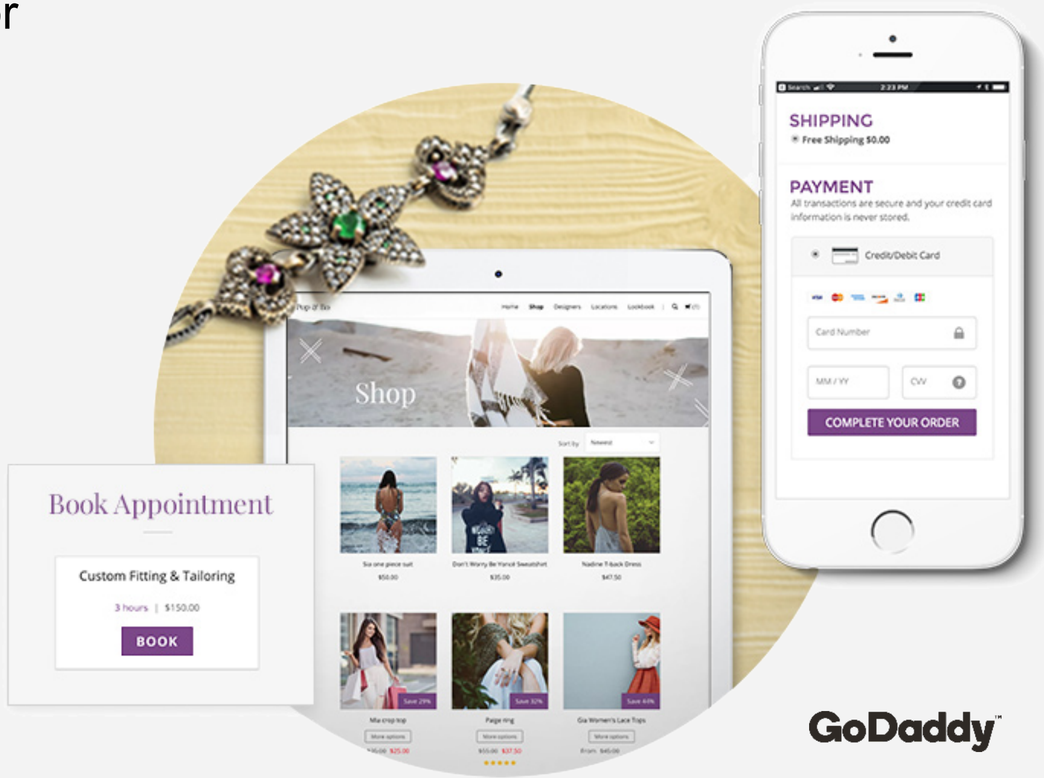



Blog Post:
http://x.co/MLAgg
by Rich Collier

Business KPIs

GoCentral KPIs

- GoCentral team moves extremely fast – 13,500 code/config deployments in 2017
  - “If you don’t stop and look around once in awhile, you could miss it.” – Ferris Bueller
- Free trial product so we analyze by cohorts
  - If I bought January 1st, on January 14th I’ll be in the 14 day cohort
- Business level KPIs are trailing indicators
  - Activate – when customer setup the product they signed up for
  - Publish – when customer launches their initial website
  - Conversion – when customer switches from Free Trial to Paid
  - Auto Renew – whether account has auto-conversion enabled



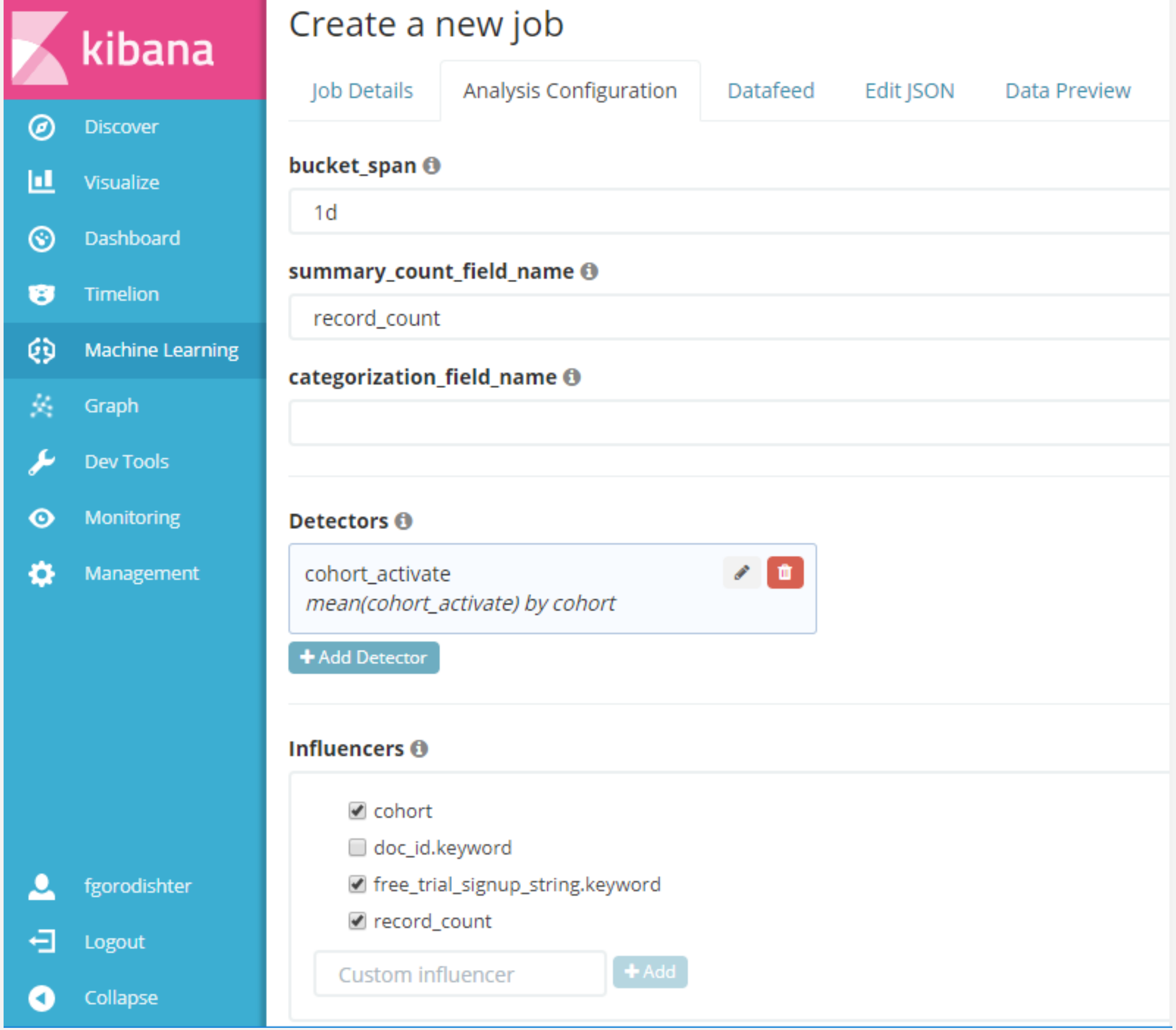
GoDaddy

GoCentral KPIs

PySpark Approach:

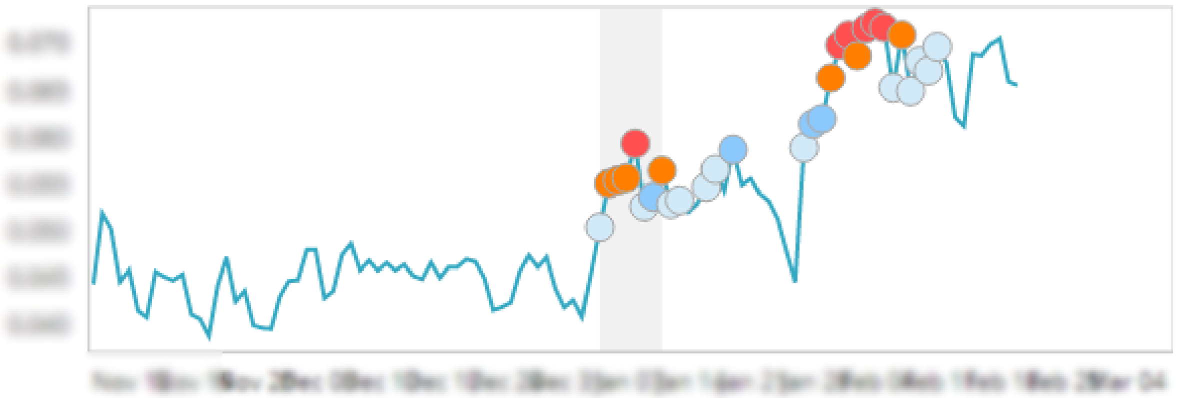
```
## Build Dataset
df = df \
     .withColumn('cohort_activate’, \
         F.when((F.datediff(df.activate_date, df.signup_date) <= cohort), 1).otherwise(0)) \
     .withColumn('cohort_end_date', F.date_add(df.signup_date, cohort))
df2 = df.groupBy(df.cohort).agg({“cohort_activate” : “avg”})

## Ingest into ES
df2.write.format("org.elasticsearch.spark.sql").mode("append").save(“index/type")

## Ask ML to process
start_payload = {"end": latest_available_date}
requests.post(es + '/_xpack/ml/anomaly_detectors/' + job + '/_open')
requests.post(es + '/_xpack/ml/datafeeds/datafeed-' + job + '/_start', json=start_payload)
```

* This is a code fragment

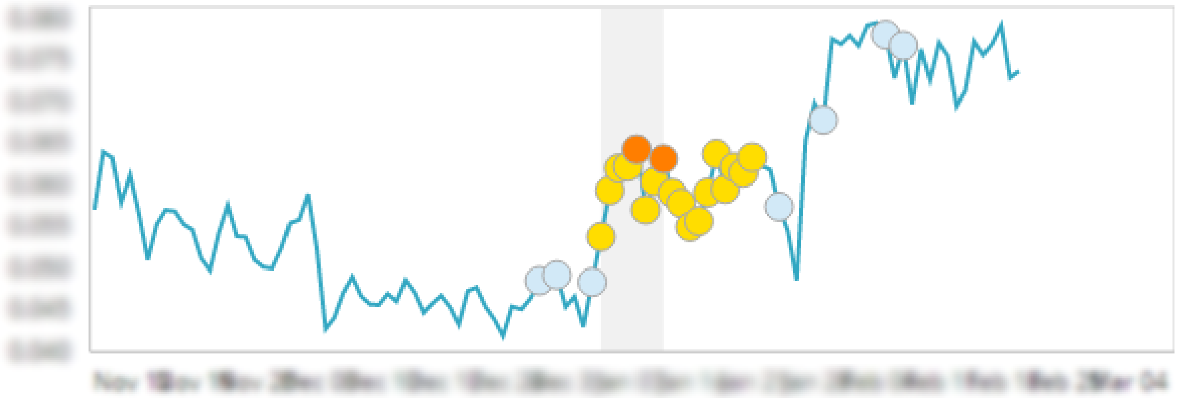

kibana
Discover
Visualize
Dashboard
Timelion
Machine Learning
Graph
Dev Tools
Monitoring
Management
fgorodishter
Logout
Collapse
Create a new job
Job Details
Analysis Configuration
Datafeed
Edit JSON
Data Preview
bucket_span
1d
summary_count_field_name
record_count
categorization_field_name
Detectors
cohort_activate
mean(cohort_activate) by cohort
+ Add Detector
Influencers
cohort
doc_id.keyword
free_trial_signup_string.keyword
record_count
Custom influencer
+ Add

Anomalies

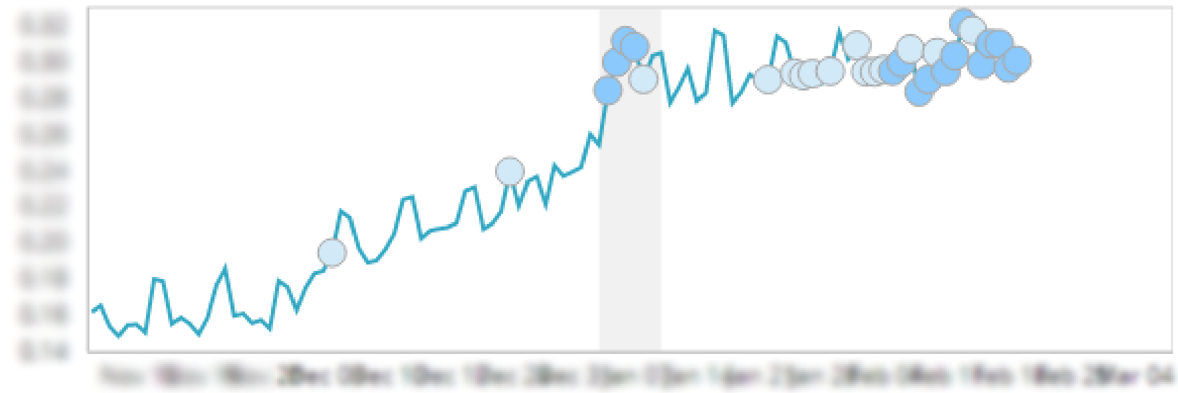
mean(cohort_conversion) by cohort - cohort 32 ℹ View



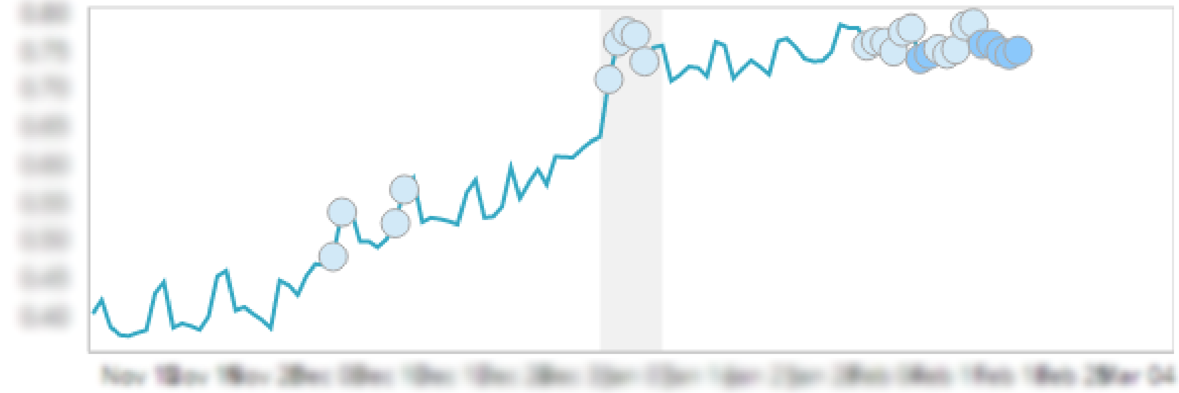
mean(cohort_auto_renew) by cohort - cohort 32 ℹ View



mean(cohort_publish) by cohort - cohort 32 ℹ View



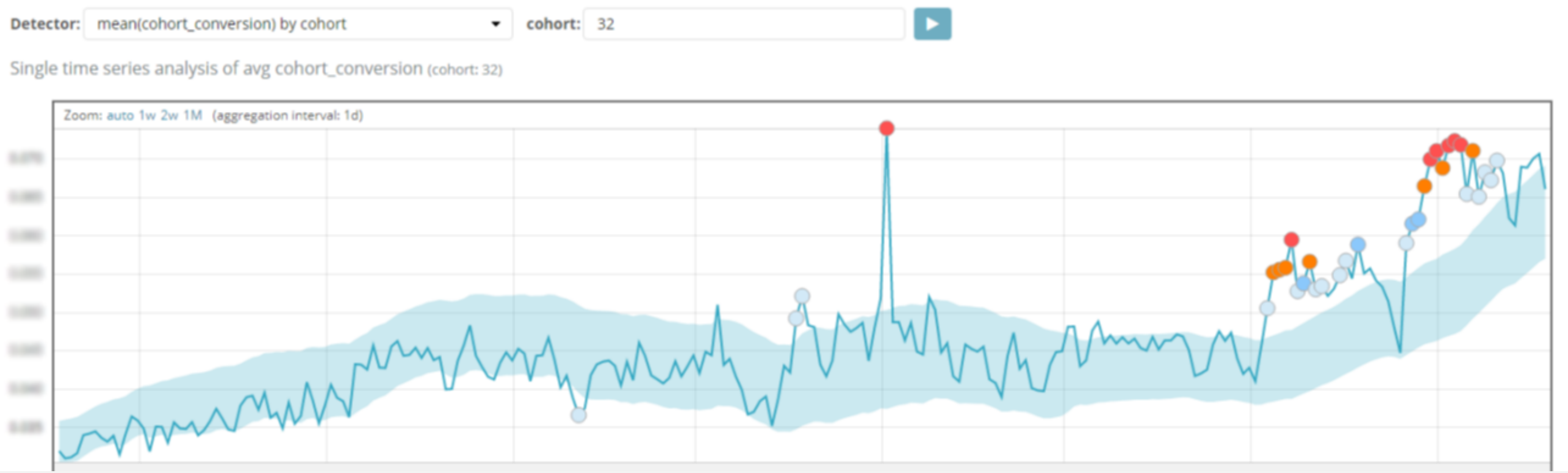
mean(cohort_activate) by cohort - cohort 32 ℹ View



Severity threshold: ⚠ minor ▾ **Interval:** Auto ▾

time	max severity	detector	found for	influenced by	actual	typical	description	job ID
January 7th 2018	⚠ 89	mean(cohort_conversion) by cohort	32	cohort: 32 free_trial_signup_string.keyword: 2017-12-07 record_count: [illegible]	[illegible]	[illegible]	↑ 1.4x higher	gocentral-kpi-bycohort-agg-cor
January 4th 2018	⚠ 66	mean(cohort_conversion) by cohort	32	cohort: 32 free_trial_signup_string.keyword: 2017-12-04 record_count: [illegible]	[illegible]	[illegible]	↑ 1.3x higher	gocentral-kpi-bycohort-agg-cor

GoCentral KPIs



- Model Plot FTW!

ECE / ML Key Learnings

Hardest part is your data

Bulk of project was spent figuring out what data was actionable versus vanity and formatting to take best advantage of ML.

Make data ingest idempotent

Leverage custom document _id field so you can reload same data easily.

You will try & retry jobs

Tooling is powerful, but figuring out the right mix of detectors, influencers, etc is dataset specific. Set aside a sprint or two for this.

Advanced jobs are your friend

We found ourselves running most ML workloads on Advanced jobs due to their power in configuring and enabling model plot (see below).

Alerting / Watcher is Hard

Plan to spend time on watcher configuration, especially for advanced notification. Its getting better, but still more to do.

Be mindful of updates

Updates to Elastic may require stopping all ML jobs at a minimum and restarting. Alternatively may require recreating job if model changed to take advantage of new features.

Business likes model plots

The visualization with a model-plot is extremely convincing / powerful. Use it where you can afford.

* Advanced jobs require JSON config!

Wait for updates to bake

It’s a pretty good practice for any production workload, but ECE is new and has more moving pieces.
Build a dev cluster and upgrade at-will for new features – especially in ML!

Leverage the Elastic team

We’ve had an incredible relationship with the Elastic team.

But wait, there’s more!

Replaced our SIEM with Elastic + Hadoop

Tracking our CICD Pipelines / Code Health

Monitoring & Alert Correlation / Analysis

System Availability / Impact Analysis

We can’t wait for…

Elastic APM

Guess what …. We’re hiring!

x.co/jobplz | godaddy.com/jobs

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

Questions at the AMA or …

Felix Gorodishter

@fgorodishter

felix@godaddy.com