Managed Virtual Machines

- **Elasticsearch**
  - elasticsearch.cs.vt.edu
  - Elasticsearch in Docker
    - Kibana

- **KGI**
  - kgi.cs.vt.edu
  - Gitlab Runner

```
<table>
<thead>
<tr>
<th>CONTAINER ID</th>
<th>IMAGE</th>
<th>COMMAND</th>
<th>CREATED</th>
<th>STATUS</th>
<th>PORTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>615682167def</td>
<td>gitlab/gitlab-runner</td>
<td>*/usr/bin/dumb-init ...</td>
<td>23 hours ago</td>
<td>Up 23 hours</td>
<td></td>
</tr>
<tr>
<td></td>
<td>gitlab_gitlab-runner_1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CONTAINER ID</th>
<th>IMAGE</th>
<th>COMMAND</th>
<th>CREATED</th>
<th>PORTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>34c6596b6ead</td>
<td>docker.elastic.co/elasticsearch/elasticsearch:7.10.0</td>
<td>*/tini -- /usr/local...</td>
<td>2 weeks ago</td>
<td>es03</td>
</tr>
<tr>
<td></td>
<td>9200/tcp, 9300/tcp</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0cbcaaa3dd3e</td>
<td>docker.elastic.co/elasticsearch/elasticsearch:7.10.0</td>
<td>*/tini -- /usr/local...</td>
<td>2 weeks ago</td>
<td>es01</td>
</tr>
<tr>
<td></td>
<td>0.0.0.0:9200--&gt;9200/tcp, 9300/tcp</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1436c0c931bde</td>
<td>docker.elastic.co/elasticsearch/elasticsearch:7.10.0</td>
<td>*/tini -- /usr/local...</td>
<td>2 weeks ago</td>
<td>es02</td>
</tr>
<tr>
<td></td>
<td>9200/tcp, 9300/tcp</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Elasticsearch

- Index data from content teams
- Store ingested documents and metadata
- Provide access to Front End for display and interaction
Gitlab

- VT CS hosted Gitlab instance
  - git.cs.vt.edu
- Provides almost complete DevOps system
  - Version Control
  - Runner
  - Container Registry
Kubernetes

- Created projects for each team
- Volume mounts
- Airflow integration
- Load Balancing for Front End interface
Postgres

- Postgres instance running on Kubernetes
  - Service and Goals tables
  - Reasoner
  - Airflow
- Database was also used by other teams
  - Front End - users and permissions
Using the API
Service API

- CRUD API to pass to Front End for registering services

```python
@api.route('/services/', methods=['GET'])
def get_services():
    ...  
    Returns the list of available services within
         the service metadata table in the db
    ...  
    services = services_model.select()
    return jsonify(services)
```
Reasoner

- Mines workflows from the reasoner table
- API built to make it accessible from the frontend
- API call returns relevant information to the Front End
Mining and Running a Workflow
## Mining a workflow

<table>
<thead>
<tr>
<th>service_id</th>
<th>service_name</th>
<th>service_description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>warc-to-json</td>
<td>Converts incoming WARC file to Json</td>
</tr>
<tr>
<td>5</td>
<td>add-id</td>
<td>Add ID field</td>
</tr>
<tr>
<td>6</td>
<td>add-username-field</td>
<td>Extracts Username field</td>
</tr>
<tr>
<td>7</td>
<td>add-timestamp-field</td>
<td>Extracts Timestamp field</td>
</tr>
<tr>
<td>8</td>
<td>add-hashtags-field</td>
<td>Extracts Hashtag field</td>
</tr>
<tr>
<td>9</td>
<td>add-mentions-field</td>
<td>Extracts Mentions field</td>
</tr>
<tr>
<td>10</td>
<td>add-geolocation-field</td>
<td>Extracts Geolocation field</td>
</tr>
<tr>
<td>11</td>
<td>add-keywords-field</td>
<td>Extracts Keyword field</td>
</tr>
<tr>
<td>12</td>
<td>add-twirole-classification-field</td>
<td>Adds field with Twirole classification (string value is either &quot;male&quot;/&quot;female&quot;/&quot;brand&quot;)</td>
</tr>
<tr>
<td>13</td>
<td>merge-fields</td>
<td>Merge all extracted fields</td>
</tr>
<tr>
<td>14</td>
<td>generalized-indexing-using-els</td>
<td>Index for elasticsearch</td>
</tr>
</tbody>
</table>
Apache Airflow

- Used to run workflows on Kubernetes
- API created for the frontend to trigger workflows
- Workflow status and service logs made available to the frontend through the API
CI/CD

- Gitlab Runner
  - KGI VM
- Virginia Tech Computer Science Container Registry
  - container.cs.vt.edu
- Virginia Tech Computer Science Gitlab
  - git.cs.vt.edu
Unit Testing

- Created unit testing framework
- Worked with content teams through testing manager
- Goal: All services have unit tests and integration tests
  - Not quite met
Challenges

- Docker rate limits
- Elasticsearch security
- Ceph and NFS storage mounting
- Gitlab CI/CD interactions
- Communication
- Airflow currently only handles I/O through files
Future Work

- Service API
  - Fix update operation
  - Connect to Front End

- Add tests
  - Specifically for Service API and others
Questions?