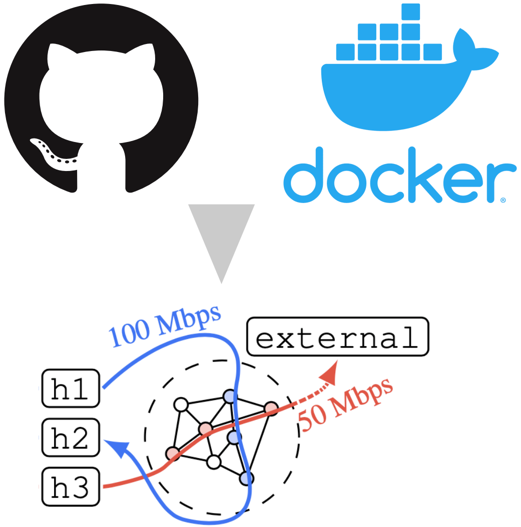
# Generating representative, live network traffic out of millions of code repositories



Tobias Bühler, Roland Schmid,
Sandro Lutz, Laurent Vanbever

ETH Zürich nsg.ee.ethz.ch

ACM HotNets

Nov 14 2022

# Today, we only have a few gold nuggets of network data available

CAIDA

Intrusion Detection Evaluation Dataset (CIC-IDS2017)

RIPE Atlas

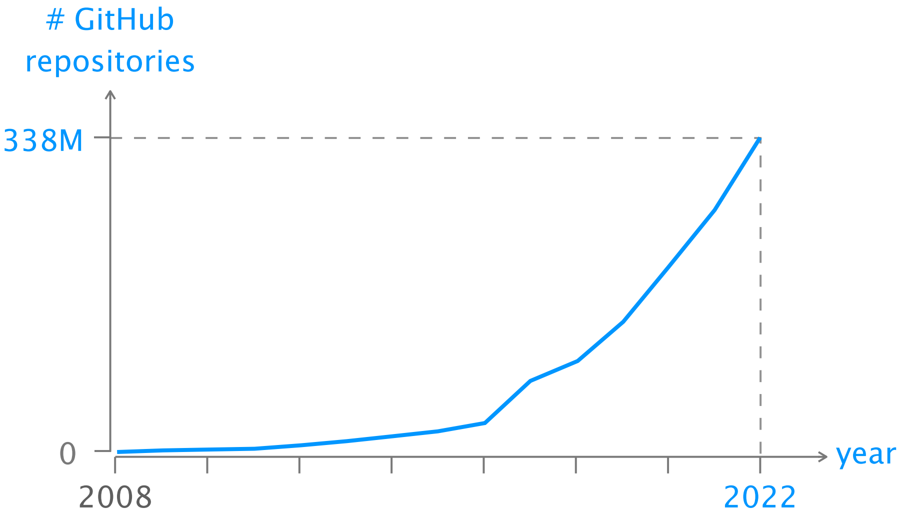
MAWI

Picture: https://www.scienceimage.csiro.au/image/10458

# We believe there exists an entire gold mine/pile of network data

Picture: https://labs.openai.com/s/zD0NTe1h8FPJPCsjSlkZvKMv

# We believe there exists an entire gold mine/pile of network data



Picture: https://labs.openai.com/s/zD0NTe1h8FPJPCsjSlkZvKMv
Numbers: https://octoverse.github.com/
https://en.wikipedia.org/wiki/Timeline_of_GitHub

In order to tap into this gold mine,

we have to bridge the gap from static text/code to actual network data

# In order to tap into this gold mine,
# we have to bridge the gap from static text/code to actual network data

| | |
|---|---|
| Static code analysis | Analyze usage of network functions<br>Extracts high-level traffic insights |
| Running the code | Compile and run each open-source project<br>Generates live traffic which reacts to network events |
| ??? | The next crazy idea |

# In order to tap into this gold mine, we have to bridge the gap from static text/code to actual network data

| | |
|---|---|
| Static code analysis | Analyze usage of network functions<br>Extracts high-level traffic insights |
| Running the code | Compile and run each open-source project<br>Generates live traffic which reacts to network events |
| ??? | The next crazy idea |

# However, executing arbitrary open-source projects is challenging

| | |
|---|---|
| Arbitrary code | How do we build the projects?<br>Arbitrary code, language and APIs |
| Missing documentation | How do we run the projects?<br>Missing commands, dependencies and support |
| Unexpected errors | How do we handle bugs and errors?<br>Unexpected crashes, inputs and runtime |

We leverage the rise of automation frameworks
which allow to compile and run arbitrary code

# We leverage the rise of automation frameworks which allow to compile and run arbitrary code

Docker containers

Are a standalone, executable package

Contain all the code and its dependencies

# We leverage the rise of automation frameworks which allow to compile and run arbitrary code

Docker containers

Are a standalone, executable package

Contain all the code and its dependencies

Orchestration files

Define how multiple containers are configured

A single command builds and starts all of them

Our vision is to combine big data and container solutions
to generate representative, live network traffic

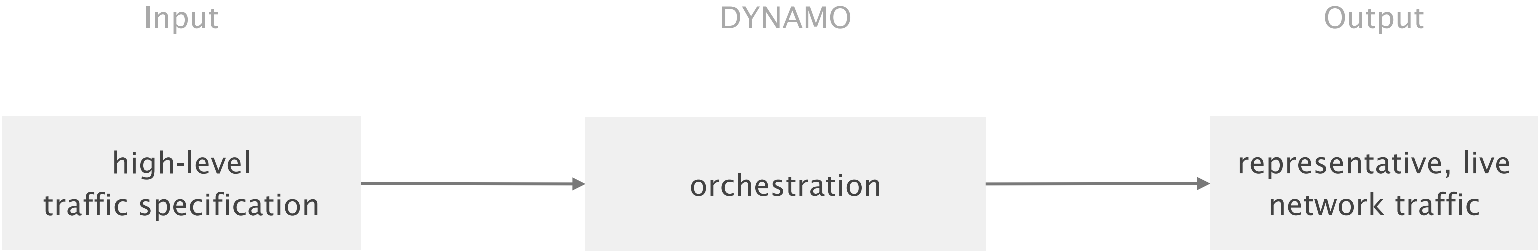
Our vision is to combine big data and container solutions to generate representative, live network traffic

with respect to a
given user specification

Our vision is to combine big data and container solutions to generate representative, live network traffic

with respect to a
given user specification

traffic/applications that
react to network events



Input

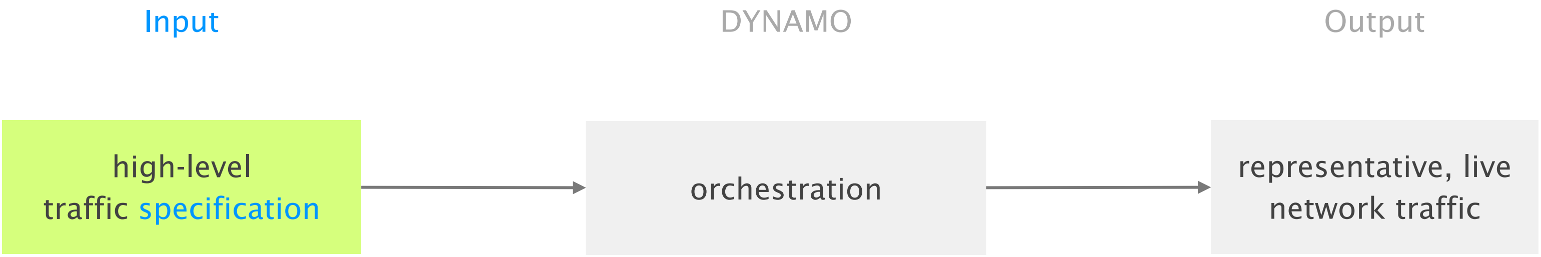
DYNAMO

Output

high-level
traffic specification

orchestration

representative, live
network traffic



Input

DYNAMO

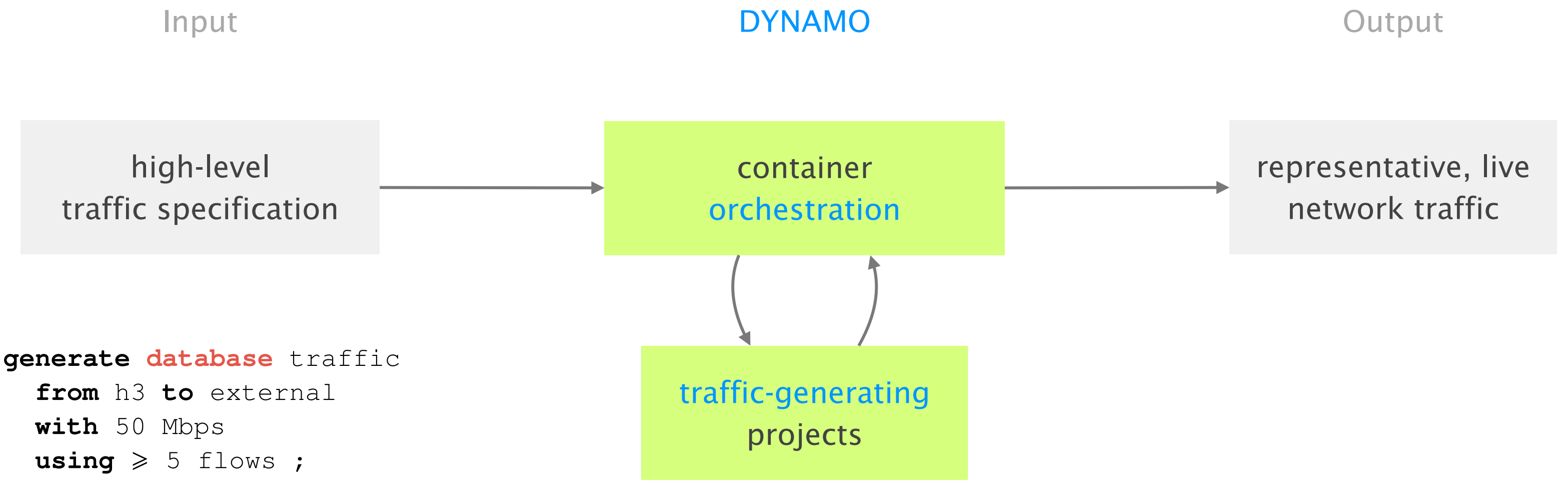
Output

high-level traffic specification

orchestration

representative, live network traffic

```
generate database traffic
  from h3 to external
  with 50 Mbps
  using ⩾ 5 flows ;
```



Input

DYNAMO

Output

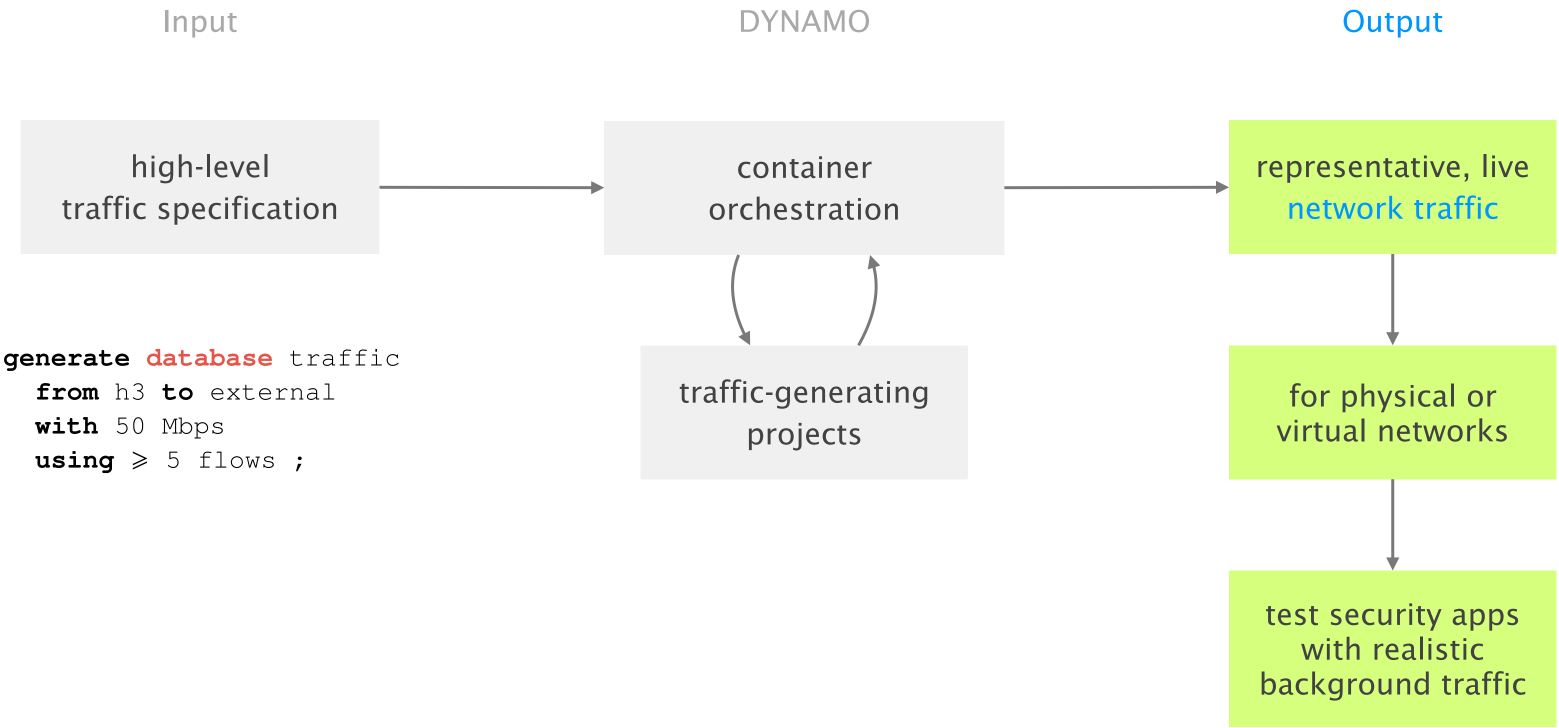
high-level traffic specification

container orchestration

traffic-generating projects

representative, live network traffic

```
generate database traffic
  from h3 to external
  with 50 Mbps
  using ⩾ 5 flows ;
```



Input

DYNAMO

Output

high-level traffic specification

container orchestration

traffic-generating projects

representative, live network traffic

for physical or virtual networks

test security apps with realistic background traffic

```
generate database traffic
  from h3 to external
  with 50 Mbps
  using ⩾ 5 flows ;
```

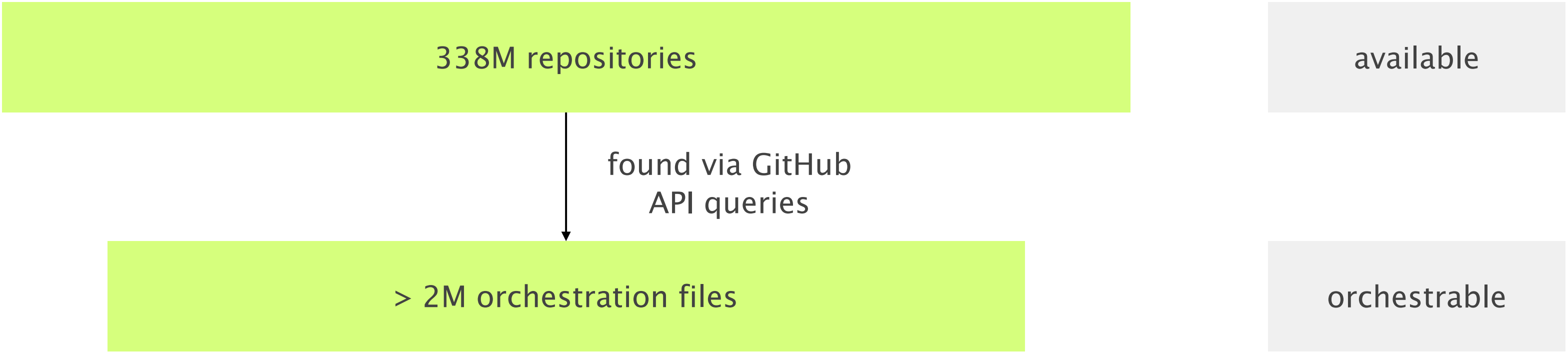
# DYNAMO first searches for traffic-generating projects

# DYNAMO first searches for traffic-generating projects

338M repositories

available

# DYNAMO first searches for traffic-generating projects



338M repositories

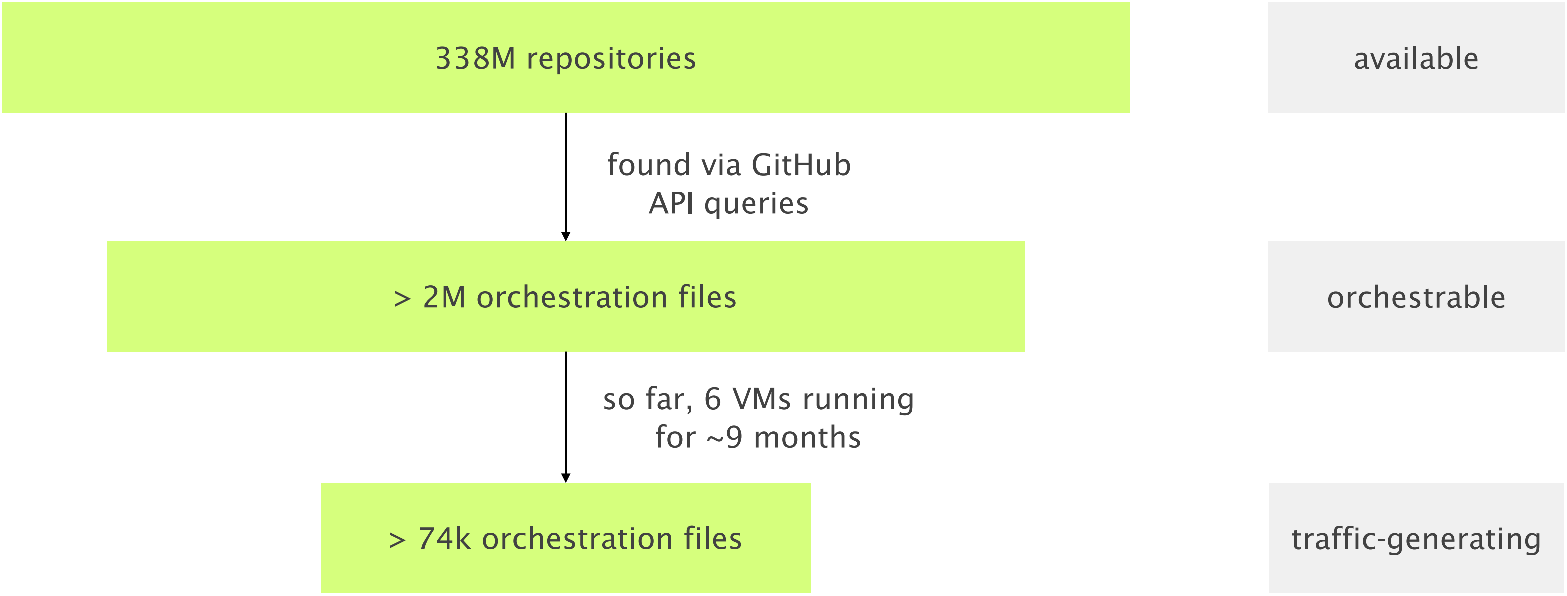
found via GitHub API queries

> 2M orchestration files

available

orchestrable

# DYNAMO first searches for traffic-generating projects



338M repositories

found via GitHub API queries

> 2M orchestration files

so far, 6 VMs running for ~9 months

> 74k orchestration files

available

orchestrable

traffic-generating

# Users then specify traffic requirements in a Declarative Traffic Specification Language

What kind of traffic?

Between which hosts?

How much traffic?

Example specification

```
generate web traffic
  from h1 to h2
  with 100 Mbps ;
generate database traffic
  from h3 to external
  with 50 Mbps
  using ⩾ 5 flows ;
```

# Users then specify traffic requirements in a Declarative Traffic Specification Language

What kind of traffic?

Between which hosts?

How much traffic?

Example specification

```
generate web traffic
  from h1 to h2
  with 100 Mbps ;
generate database traffic
  from h3 to external
  with 50 Mbps
  using ⩾ 5 flows ;
```

# Users then specify traffic requirements in a Declarative Traffic Specification Language

What kind of traffic?

Between which hosts?

How much traffic?

## Example specification

```
generate web traffic
  from h1 to h2
  with 100 Mbps ;
generate database traffic
  from h3 to external
  with 50 Mbps
  using ⩾ 5 flows ;
```

# Users then specify traffic requirements in a Declarative Traffic Specification Language

What kind of traffic?

Between which hosts?

How much traffic?

## Example specification

```
generate web traffic
  from h1 to h2
  with 100 Mbps ;
generate database traffic
  from h3 to external
  with 50 Mbps
  using ⩾ 5 flows ;
```

# Users then specify traffic requirements in a Declarative Traffic Specification Language

What kind of traffic?

Between which hosts?

How much traffic?

Example specification

```
generate web traffic
  from h1 to h2
  with 100 Mbps ;
generate database traffic
  from h3 to external
  with 50 Mbps
  using ⩾ 5 flows ;
```

# Given a specification,
# DYNAMO generates matching live traffic

Example specification

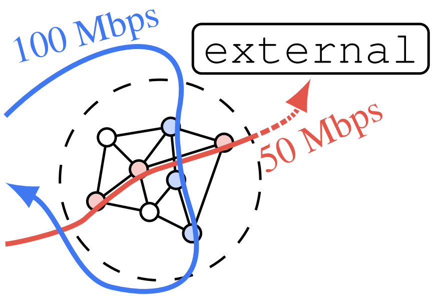
```
generate web traffic
  from h1 to h2
  with 100 Mbps ;
generate database traffic
  from h3 to external
  with 50 Mbps
  using ⩾ 5 flows ;
```

# Given a specification,
# DYNAMO generates matching live traffic

Example specification

```
generate web traffic
  from h1 to h2
  with 100 Mbps ;
generate database traffic
  from h3 to external
  with 50 Mbps
  using ⩾ 5 flows ;
```

Traffic generation



Send live traffic through
a given user network

# To achieve that,

# DYNAMO needs to orchestrate matching containers

Example specification

```
generate web traffic
  from h1 to h2
  with 100 Mbps ;
generate database traffic
  from h3 to external
  with 50 Mbps
  using ⩾ 5 flows ;
```

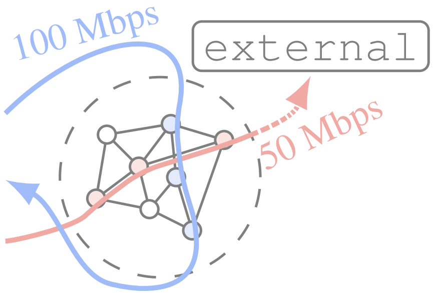
↓

## Identified projects

2× Project #5:
100 Mbps **web** traffic

Projects #7 and #18:
50 Mbps **database**
traffic using 7 flows

Traffic generation



Send live traffic through
a given user network

# To achieve that,

# DYNAMO needs to orchestrate matching containers

Example specification

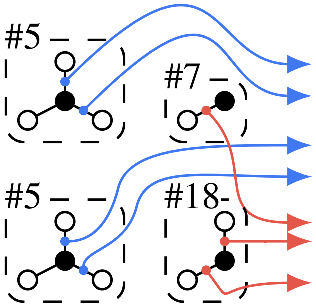
```
generate web traffic
  from h1 to h2
  with 100 Mbps ;
generate database traffic
  from h3 to external
  with 50 Mbps
  using ⩾ 5 flows ;
```

Identified projects

2× Project #5:
100 Mbps **web** traffic

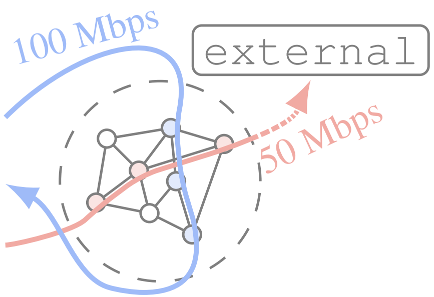
Projects #7 and #18:
50 Mbps **database**
traffic using 7 flows

Setup



Run the correct containers

Traffic generation



Send live traffic through a given user network

# To achieve that,
# DYNAMO needs to orchestrate matching containers

Example specification

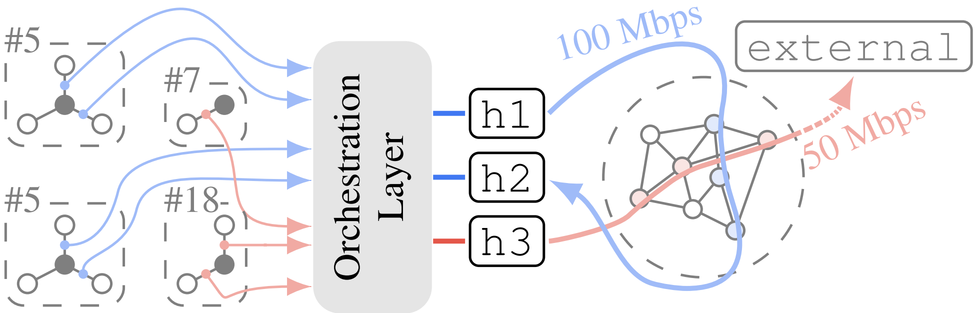
```
generate web traffic
  from h1 to h2
  with 100 Mbps ;
generate database traffic
  from h3 to external
  with 50 Mbps
  using ⩾ 5 flows ;
```

Identified projects

2× Project #5:
100 Mbps **web** traffic

Projects #7 and #18:
50 Mbps **database**
traffic using 7 flows

Setup | Orchestration | Traffic generation



Run the correct containers

Combine to virtual hosts

Send live traffic through a given user network

# DYNAMO enables many use cases

## And we'd love to hear more from you!

Security testing

DYNAMO generates real background traffic

E.g., to combine with attack traffic

Network design

DYNAMO tests applications under different designs

E.g., impact of packet loss on Bitcoin traffic

Trace generation

DYNAMO creates data sets with specific properties

E.g., to complement skewed ML training data

# Our preliminary trace analysis shows the potential of the idea

# Our preliminary trace analysis shows the potential of the idea

We found a wide range of traffic-generating applications

web (HTTP, HTTPS)

crypto (Bitcoin, IPFS)

database (MongoDB, MySQL)

message-broker (RabbitMQ, Apache Kafka)

# Our preliminary trace analysis shows the potential of the idea

We found a wide range of traffic-generating applications

web (HTTP, HTTPS)

crypto (Bitcoin, IPFS)

database (MongoDB, MySQL)

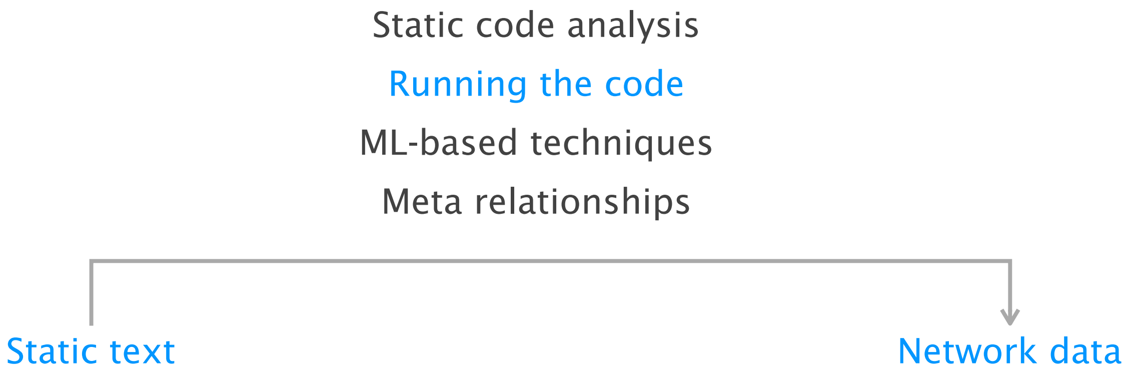
message-broker (RabbitMQ, Apache Kafka)

Some of the applications generate *a lot of* traffic

> 13M pkts (~417 Mbps), a multi-paxos implementation: `thibmeu/imperial-multi-paxos-in-elixir`

> 367k flows (~4 Mbps), a Telegram proxy: `squizduos/docker-server`

# DYNAMO showcases one approach to bridge the gap from static text/code to actual network data



Static code analysis

Running the code

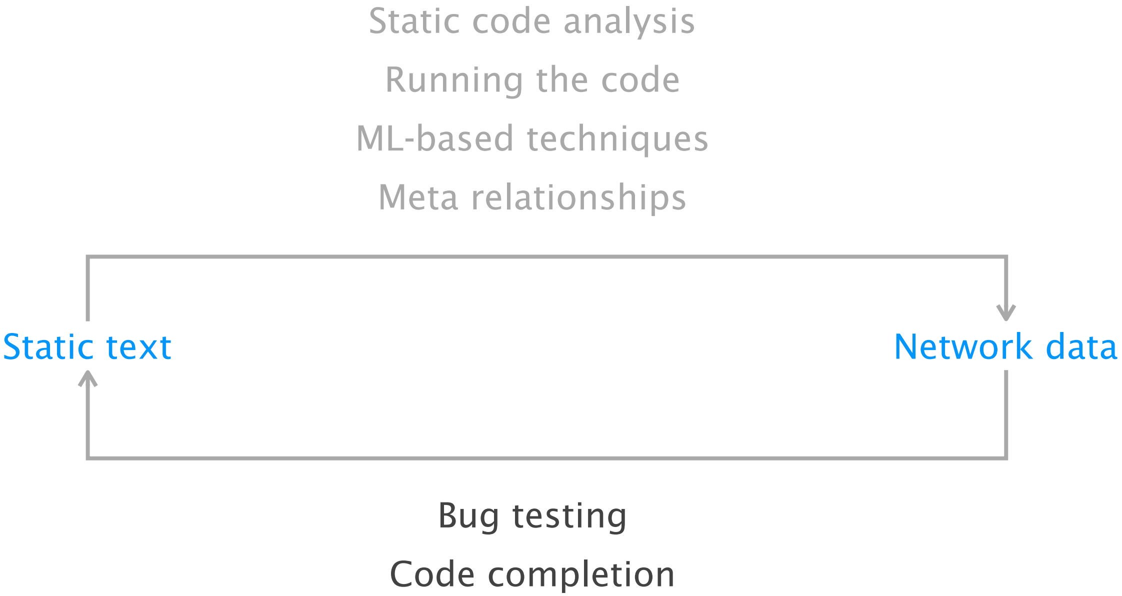
ML-based techniques

Meta relationships

Static text

Network data

# DYNAMO showcases one approach to bridge the gap from static text/code to actual network data



Static code analysis

Running the code

ML-based techniques

Meta relationships

Static text

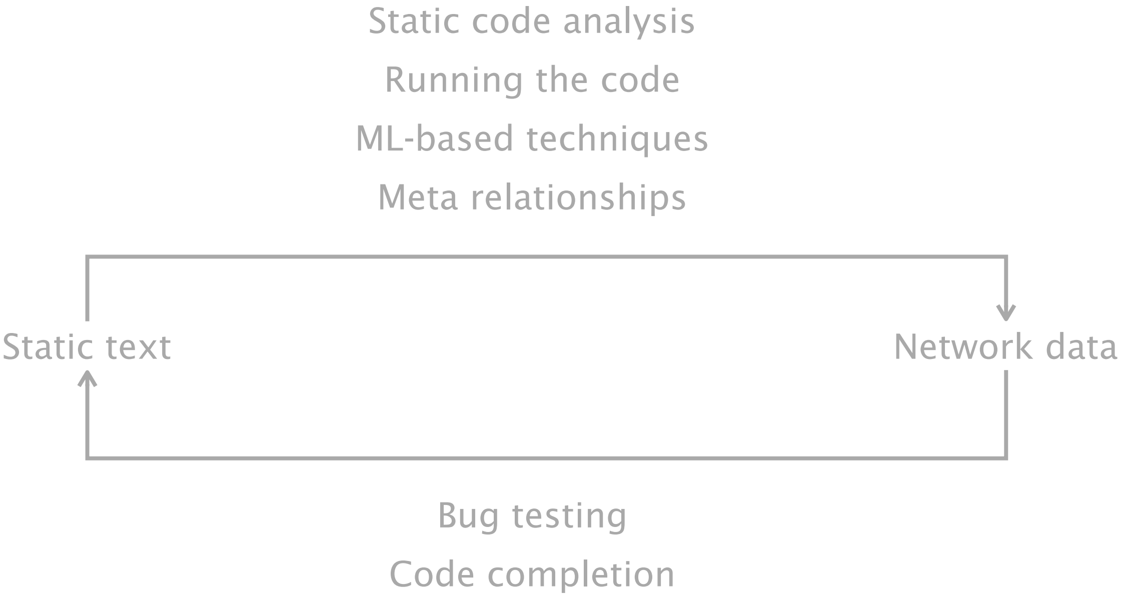
Network data

Bug testing

Code completion

# DYNAMO showcases one approach to bridge the gap from static text/code to actual network data

Traffic-generating projects



Static code analysis

Running the code

ML-based techniques

Meta relationships

Static text

Network data

Bug testing

Code completion