Логирование

План

- Что такое логи и зачем нужны логи?
- Централизованная система логирования: основные компоненты, требования, примеры
- Elastic Stack
- Логирование приложения

Логи

Зачем нужны логи?

- ...
- ...
- ...

Зачем нужны логи?

- Видимость и понимание того, как работают наши системы
- Поиск ошибок и их причин
- В отличие от метрик, содержат полный контекст работы системы или процесса

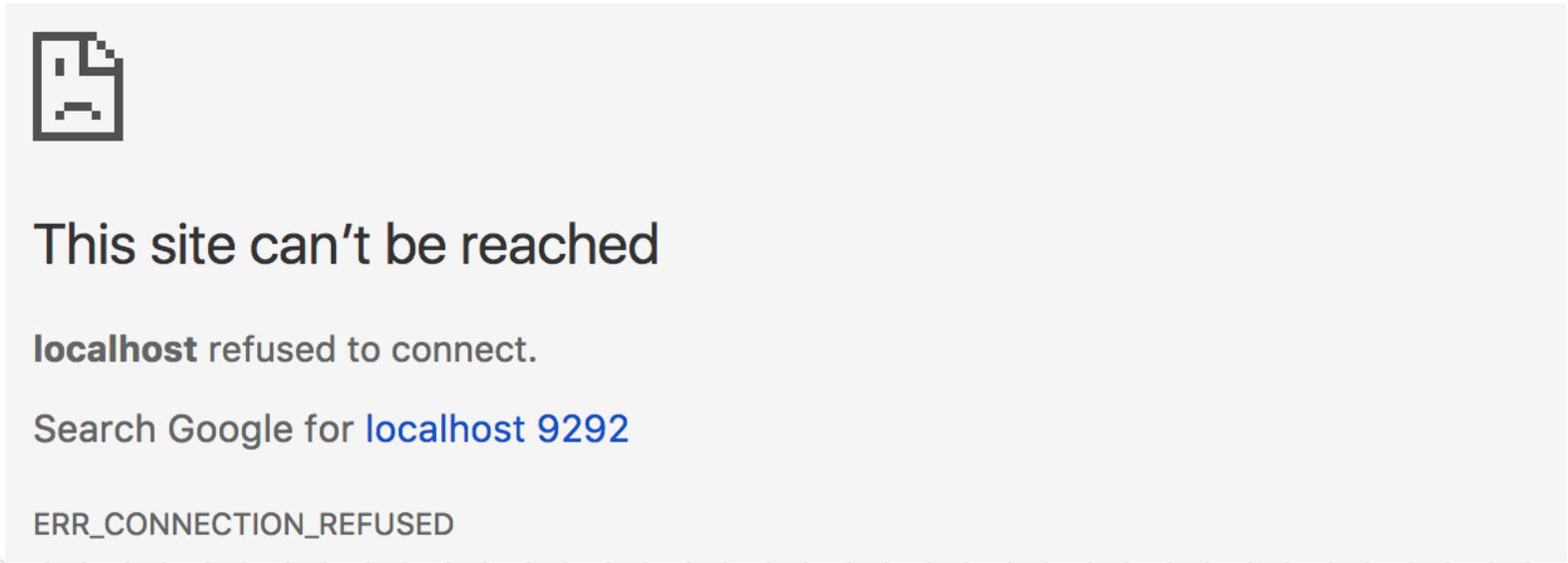
Пример применимости логов

Запустили ui сервис в фоновом режиме

```
$ docker-compose up -d ui

reddit_post_db_1 is up-to-date
Starting reddit_post_1 ...
Starting reddit_post_1 ... done
Starting reddit_ui_1 ...
Starting reddit_ui_1 ... done
```

Но приложение не работает :(Что делать?



Ищем ошибку

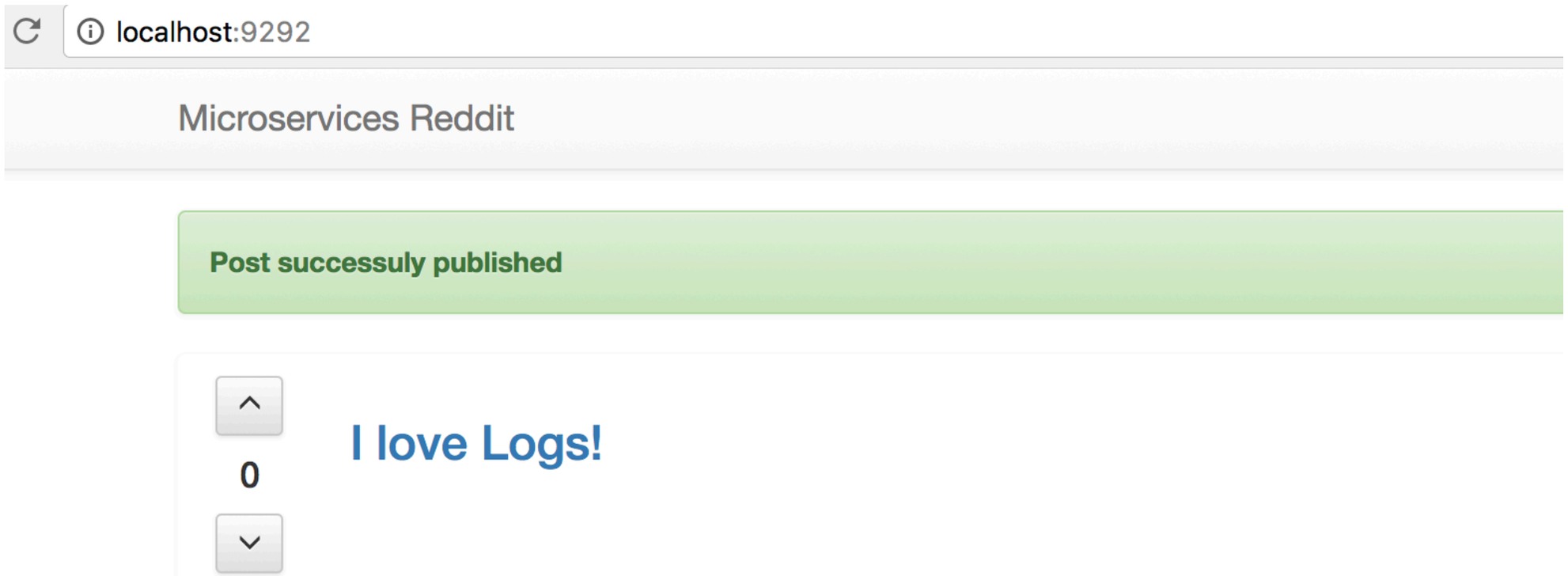
Смотрим логи и находим ошибку

```
$ docker-compose logs ui

Attaching to reddit_ui_1
ui_1              | Puma starting in single mode...
ui_1              | * Version 3.10.0 (ruby 2.3.3-p222), codename: Russell's Teapot
ui_1              | * Min threads: 0, max threads: 16
ui_1              | * Environment: development
ui_1              | ! Unable to load application: SyntaxError: /app/ui_app.rb:60: syntax error,
unexpected keyword_end, expecting end-of-input
ui_1              | config.ru:1:in `require': /app/ui_app.rb:60: syntax error, unexpected
keyword_end, expecting end-of-input (SyntaxError)
ui_1              | from config.ru:1:in `block in <main>'
ui_1              | from /usr/local/bundle/gems/rack-2.0.3/lib/rack/builder.rb:55:in
`instance_eval'
ui_1              | from /usr/local/bundle/gems/rack-2.0.3/lib/rack/builder.rb:55:in
`initialize'
ui_1              | from config.ru:in `new'
ui_1              | from config.ru:in `<main>'
ui_1              | from /usr/local/bundle/gems/rack-2.0.3/lib/rack/builder.rb:49:in `eval'
ui_1              | from /usr/local/bundle/gems/rack-2.0.3/lib/rack/builder.rb:49:in
`new_from_string'
```

Правим код

Исправляем синтаксическую ошибку и приложение работает.
Спасибо логам!



Что такое лог?

- Журнал событий происходящих во время работы системы или процесса

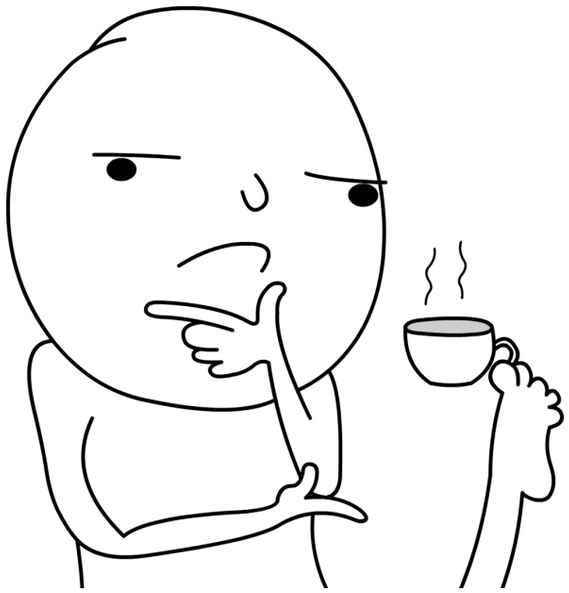
Где должны храниться логи?

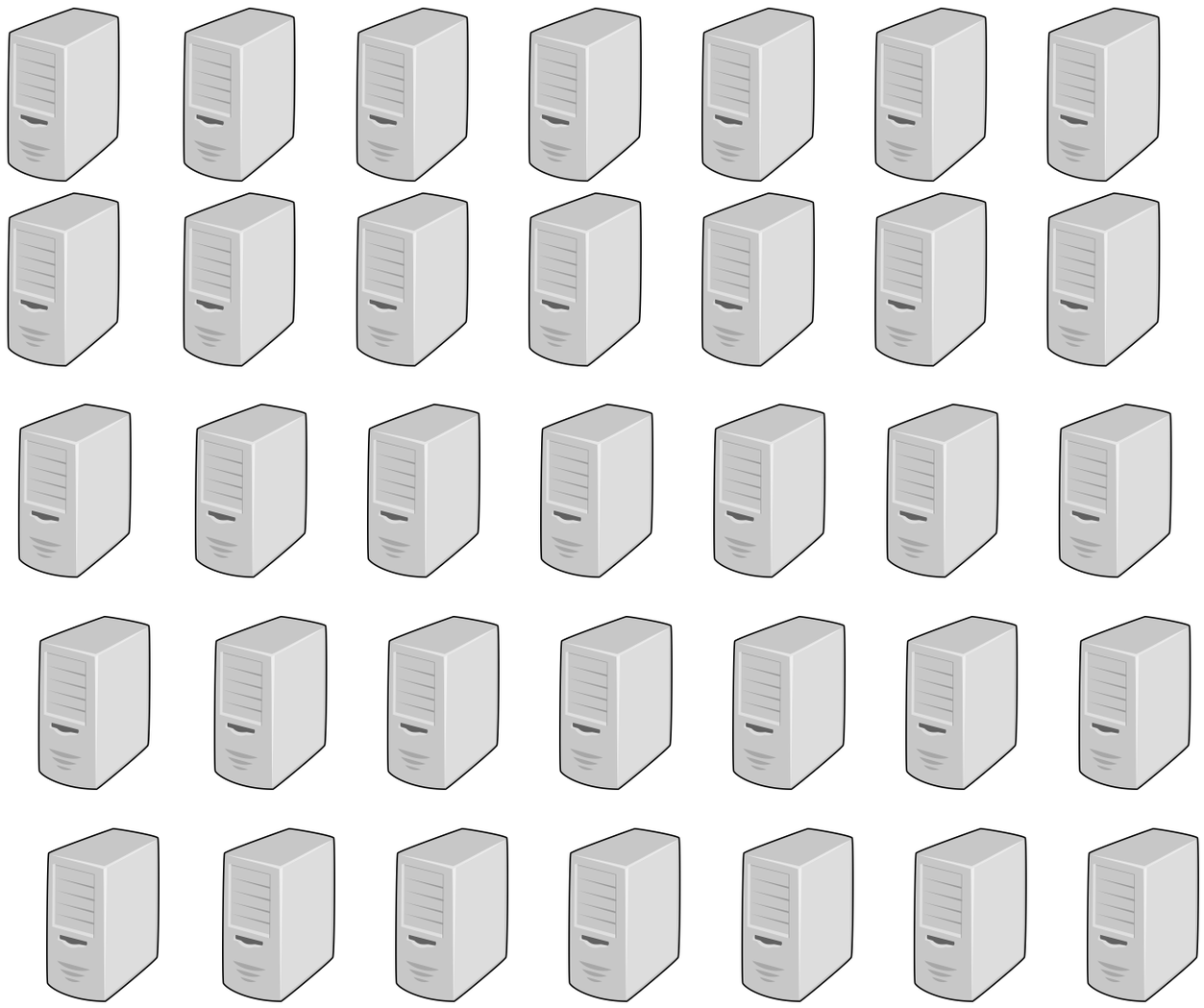
- ...
- ...
- ...

Храним локально?

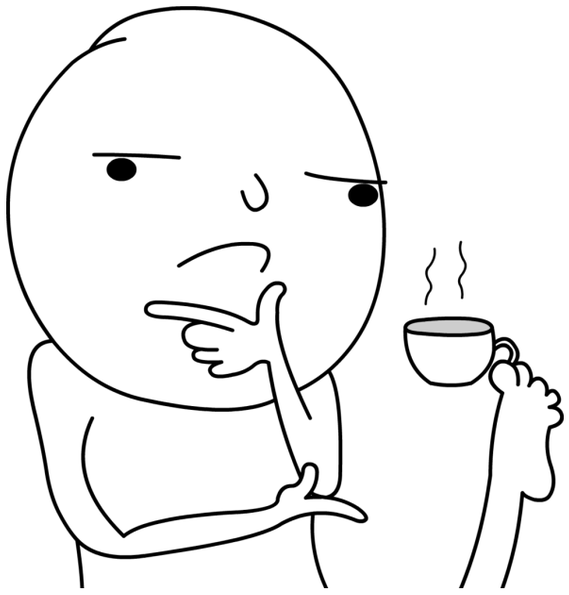
- Не понимаем, что где происходит, пока сами не зайдем в систему и не посмотрим
- При большой ферме серверов не хватит рабочего времени на обход всех машин
- Нет возможности быстро локализовать проблему - следовательно, и ее решить

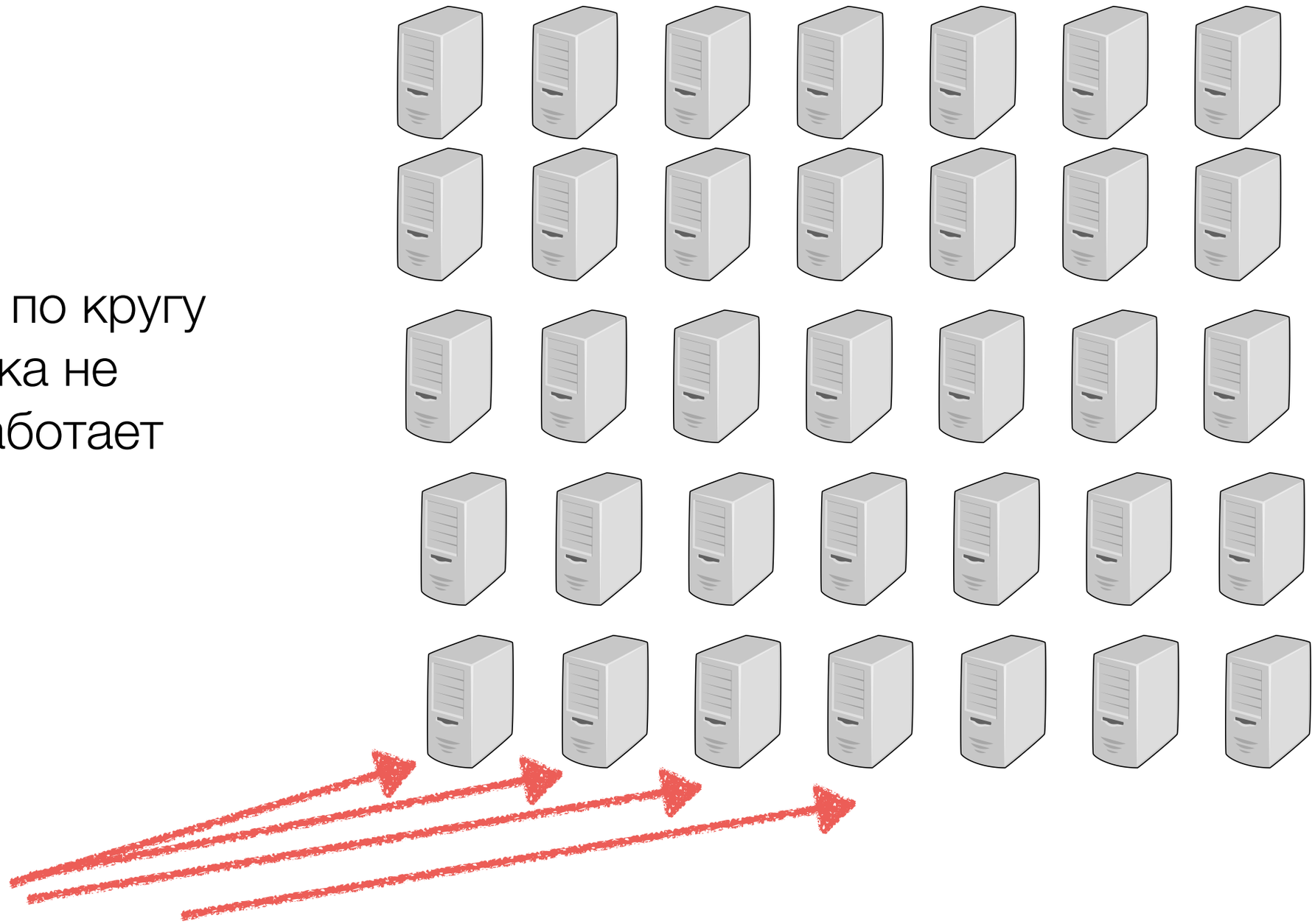
и что из них
работает, а что нет?



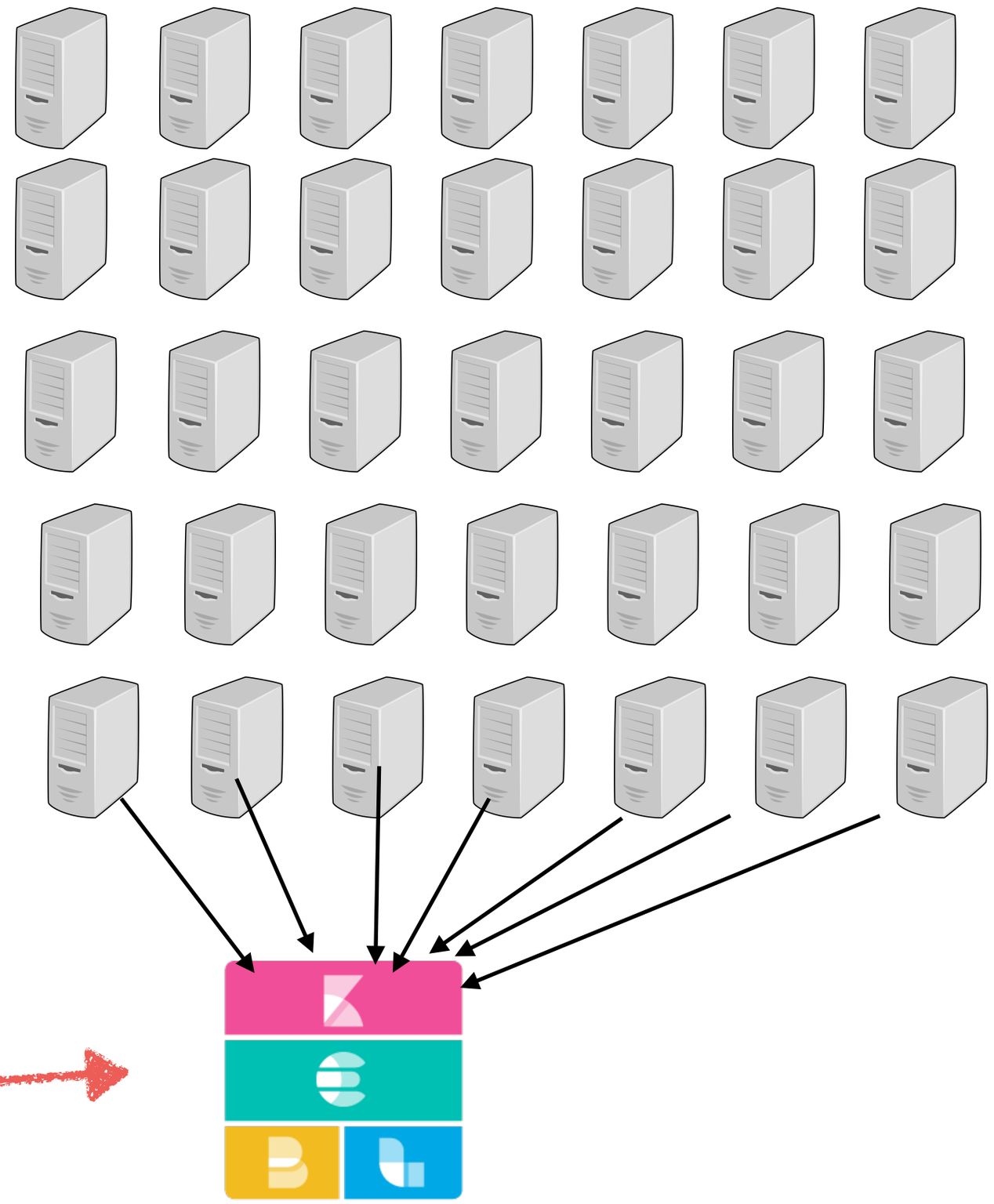


Начну обходить по кругу
все машины, пока не
найду, где не работает





Пускай хосты отдают всю информацию центральному серверу, буду обращаться только к нему



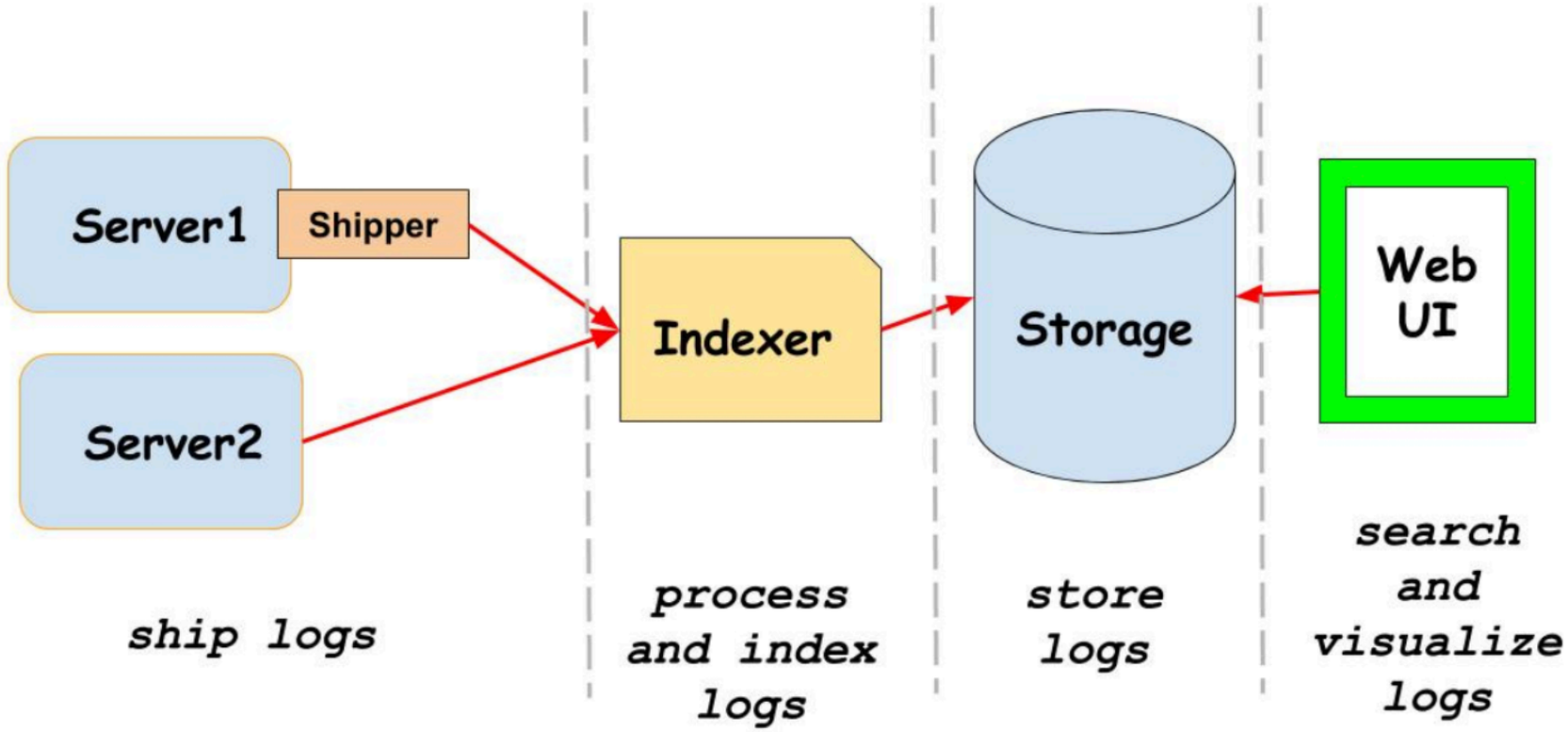
Централизованная система логирования (ЦСЛ)

- Центральный сервер(ы) агрегирует всю информацию по логам
- Единая точка доступа ко все информации
- Возможность проведения анализа по всем системам

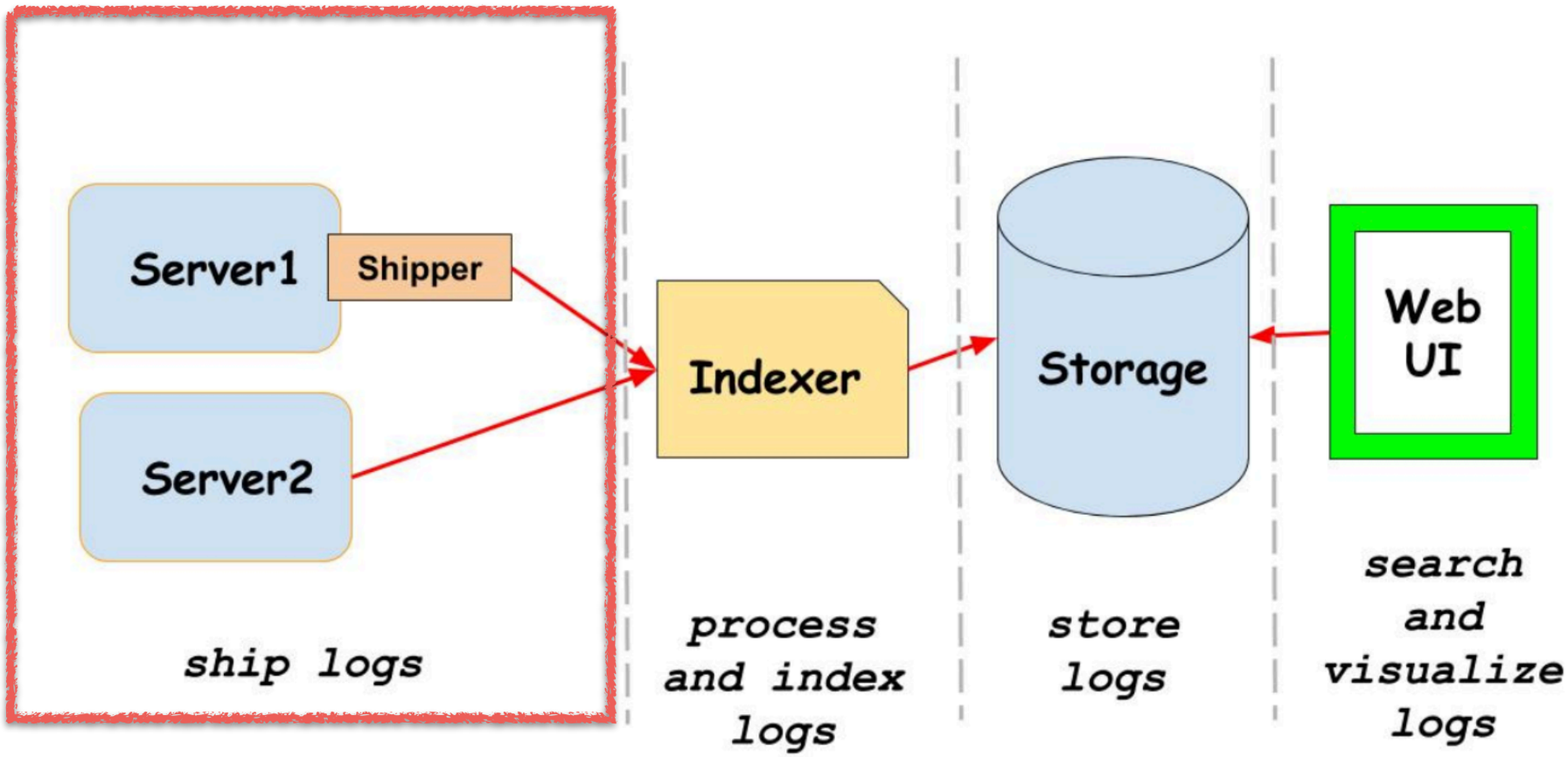
Но ...

- Централизованная система логирования не отменяет локального хранение логов
- Локальное хранение логов по-прежнему является самым надежным способом хранения
- Возможна потеря логов, если центральный сервер загружен или не доступен

Основные компоненты ЦСЛ

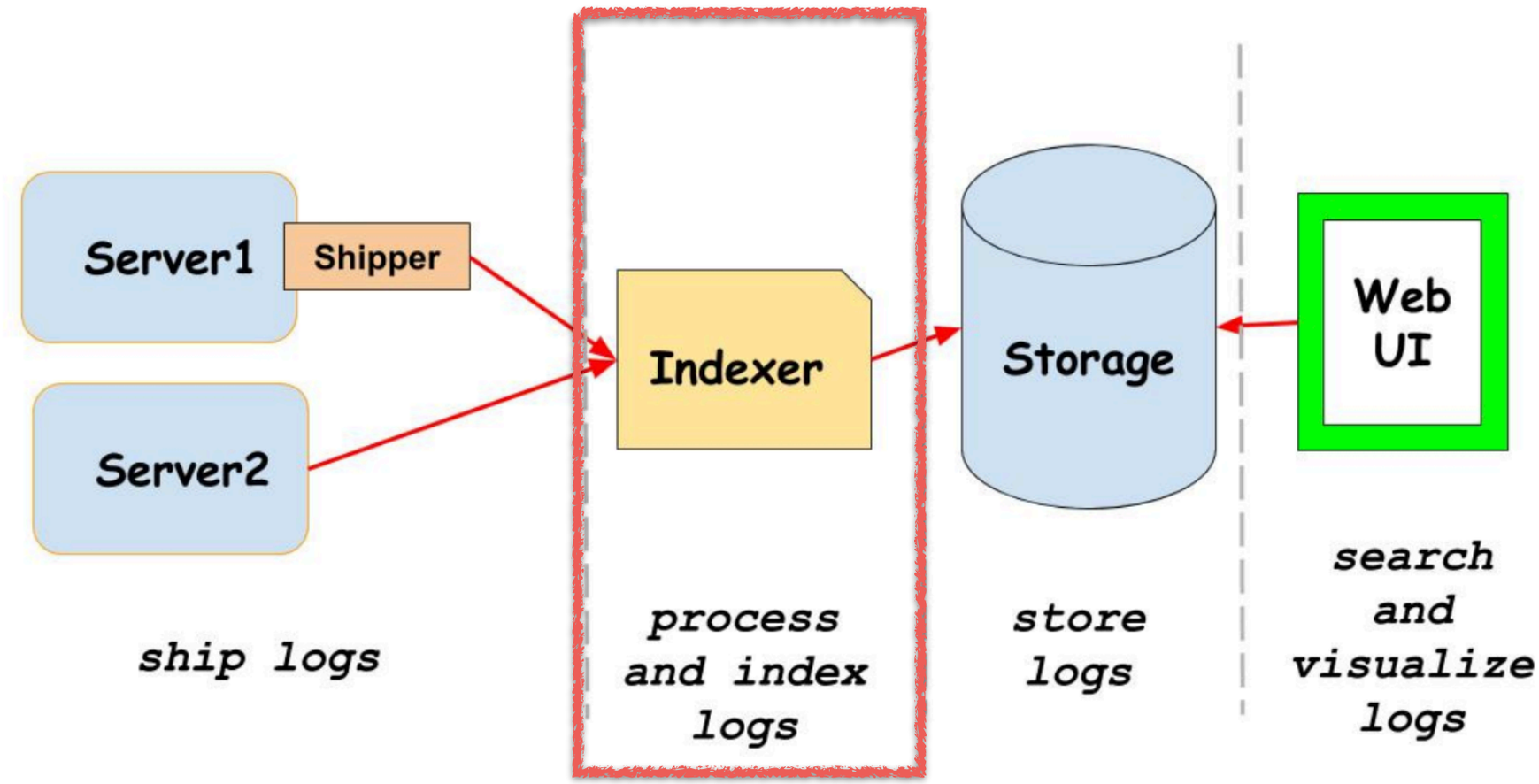


Отправка логов (shipping)



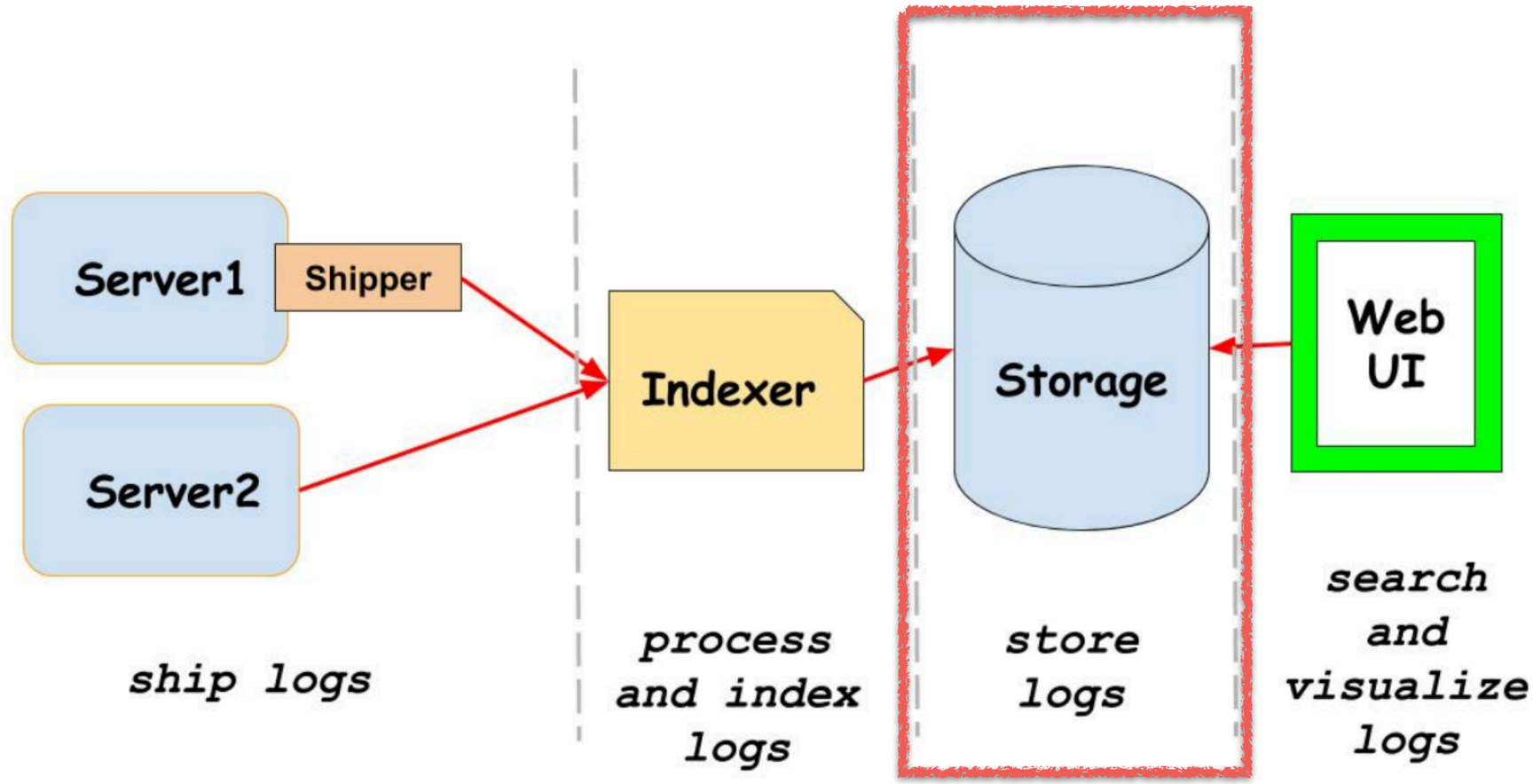
- Клиентские библиотеки
- Shippers: beats, nxlog, rsyslog, syslog-ng, fluentd, etc.

Агрегация и трансформация



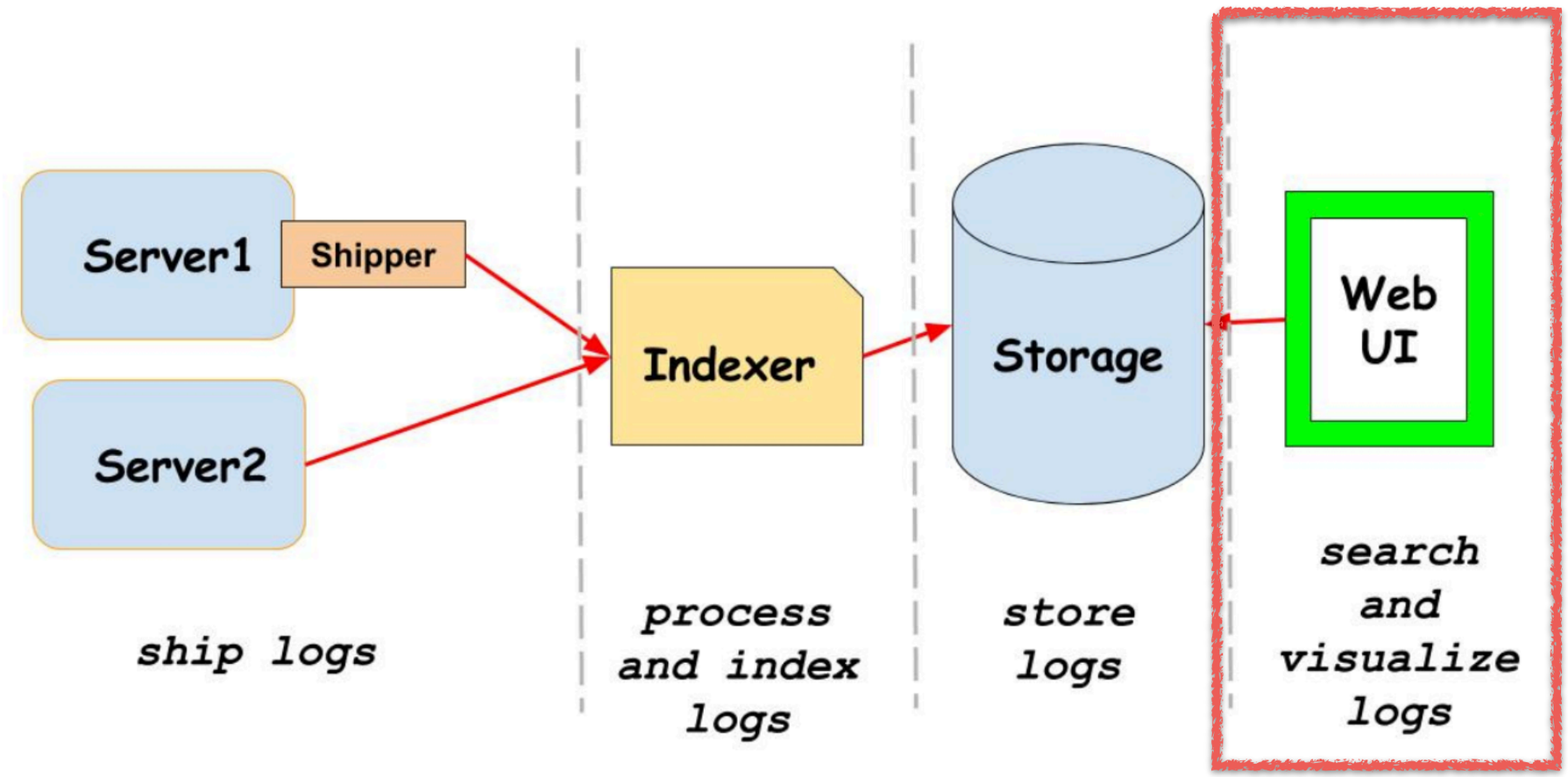
Logstash, Graylog 2, Splunk, etc.

Хранение логов



ElasticSearch, InfluxDB, MongoDB, S3, etc

Визуализация, анализ и алертинг



Kibana, Graylog 2, Grafana, etc.

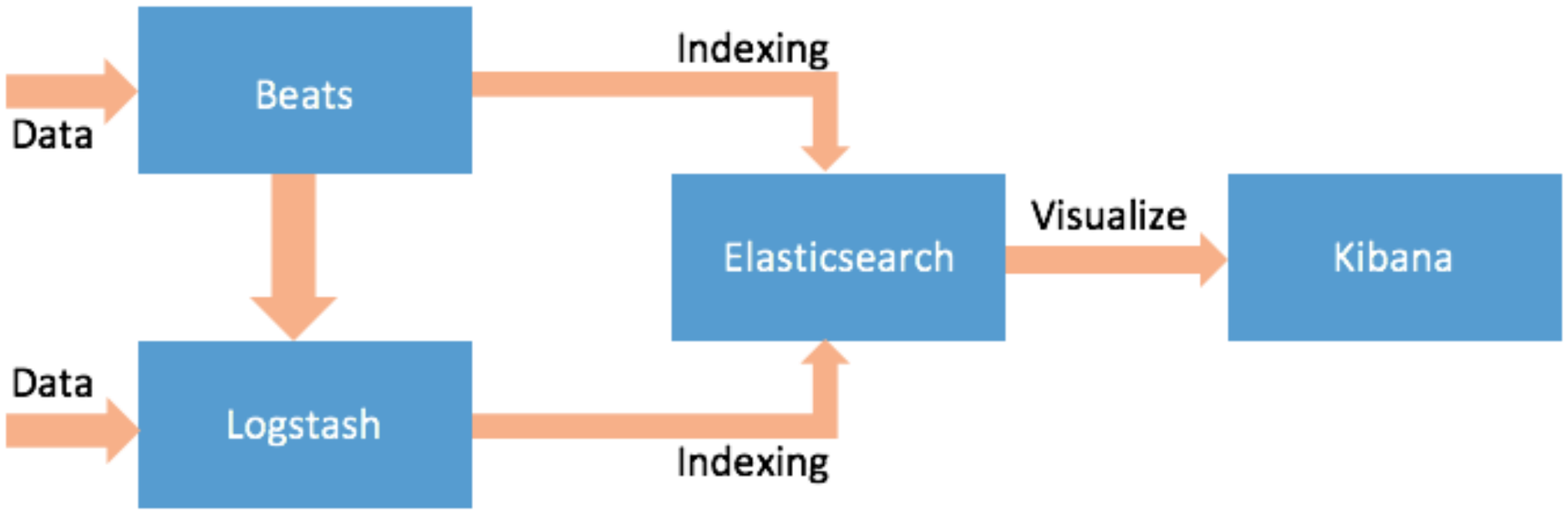
Требования к ЦСЛ

- Горизонтальная масштабируемость
- Надежность (отсутствие потери логов)
- Близость к real-time
- Должна быть недорогой

Примеры ЦСЛ

- Open source: Elastic Stack, Graylog 2
- SaaS: Splunk, Loggly, Papertrail
- Cloud Platform Service: Stackdriver Logging (GCP), CloudWatch (AWS)

Elastic Stack



Логирование приложения

Какую активность приложения логировать?

- Запросы и ответы
- Ошибки
- Вызовы ко всем внешним сервисам и API
- Бизнес события: создание пользователя, платеж
- Время чтения/записи к БД
- etc

Библиотеки для логирования

- Log4j
- Structlog
- Lograge
- etc

Логирование события пример

```
$ docker-compose up post

post_1         |  * Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
post_1         |  * Restarting with stat
post_1         |  * Debugger is active!
post_1         |  * Debugger PIN: 330-452-208
post_1         | 172.21.0.5 - - [02/Nov/2017 16:03:27] "GET /healthcheck HTTP/1.1" 200 -
post_1         | 172.21.0.5 - - [02/Nov/2017 16:03:30] "GET /healthcheck HTTP/1.1" 200 -
post_1         | 172.21.0.5 - - [02/Nov/2017 16:03:33] "GET /healthcheck HTTP/1.1" 200 -
post_1         | 172.21.0.5 - - [02/Nov/2017 16:03:36] "GET /healthcheck HTTP/1.1" 200 -
post_1         | 172.21.0.5 - - [02/Nov/2017 16:04:06] "GET /healthcheck HTTP/1.1" 200 -
post_1         | 172.21.0.5 - - [02/Nov/2017 16:04:08] "POST /add_post HTTP/1.1" 200 -
```

Изменим формат лога в коде приложения

```
@app.route("/add_post", methods=['POST'])
def add_post():
    try:
      title = request.values.get("title")
      link = request.values.get("link")
      created_at = request.values.get("created_at")
    except Exception as e:
      log.warning('bad input data: {}'.format(request.values))
      return 'ERROR'
try:
    mongo_db.insert({"title": title, "link": link, "created_at": created_at,"votes": 0})
except Exception as e:
   log.error("post.created because of {}".format(str(e)), failed=True, title=title, link=link)
   return 'ERROR'
else:
    POST_COUNT.inc()
    log.info("post.created", failed=False, title=title, link=link)
    return 'OK'
```

Логирование события пример

```
$ docker-compose up post
```

```
post_1          |  * Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
post_1          |  * Restarting with stat
post_1          |  * Debugger is active!
post_1          |  * Debugger PIN: 330-452-208
post_1          | 172.21.0.5 - - [02/Nov/2017 16:23:48] "GET /healthcheck HTTP/1.1" 200 -
post_1          | 172.21.0.5 - - [02/Nov/2017 16:23:51] "GET /healthcheck HTTP/1.1" 200 -
post_1          | 172.21.0.5 - - [02/Nov/2017 16:23:54] "GET /healthcheck HTTP/1.1" 200 -
post_1          | 172.21.0.5 - - [02/Nov/2017 16:23:57] "GET /healthcheck HTTP/1.1" 200 -
post_1          | 172.21.0.5 - - [02/Nov/2017 16:24:00] "GET /healthcheck HTTP/1.1" 200 -
post_1          | 172.21.0.5 - - [02/Nov/2017 16:24:03] "GET /healthcheck HTTP/1.1" 200 -
post_1          | 172.21.0.5 - - [02/Nov/2017 16:24:06] "GET /healthcheck HTTP/1.1" 200 -
post_1          | 2017-11-02 16:24.07 post.created failed=False link=https://github.com/
hynek/structlog title=Structlog is awesome!
```

Как получать нужную информацию из логов?

- Использовать структурированный формат логов согласно формату используемой системы логирования
- Парсить существующущий формат логов и извлекать нужную информацию

Примеры формата логов

Одна строка - требует парсинг для извлечения нужной информации:

"Started GET "/" for 127.0.0.1 at 2015-12-10 09:21:45 +0400"

Логи пишутся в формате JSON, который понимает система логирования (парсинг поля не требуется)


```
{
  "method":"GET",
  "path":"/users",
  "format":"html",
  "controller":"users",
  "action":"index",
  "status":200,
  "duration":189.35,
  "view":186.35,
  "db":0.92,
  "@timestamp":"2015-12-11T13:35:47.062+00:00",
  "@version":"1"
}
```