

Создание масштабируемых приложений

*на примере сервиса сканирования веб сайтов в
Лаборатории Касперского*

Обо мне

Лаборатория Касперского

Лаборатория контентной фильтрации

Группа развития инфраструктуры

Наша работа

- Сервис получения спама
- Генерация автоправил (Даниил Зайцев)
- Стенды сравнения продуктов
- Сервис сканирования сайтов

Общий инструментарий

- XP (extreme programming)
- DevOps

Системы контроля версий

1. Git



git

Системы контроля версий

1. Git
2. Mercurial



git



Системы контроля версий

1. Git
2. Mercurial
3. Fossil



git

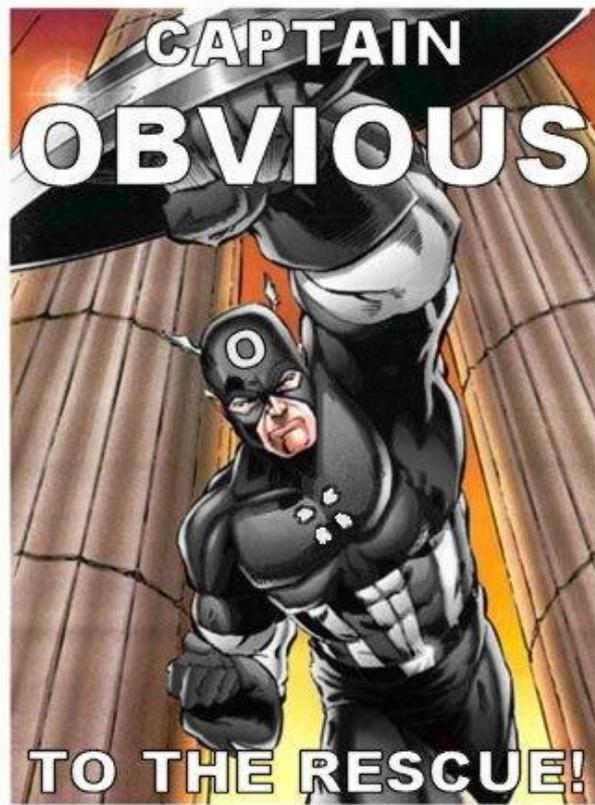


FOSSIL



Что хранить?

1. Код



Что хранить?

1. Код
2. Вспомогательные скрипты

Что хранить?

1. Код
2. Вспомогательные скрипты
3. Файлы конфигурации

Что хранить?

1. Код
2. Вспомогательные скрипты
3. Файлы конфигурации
4. Тестовые данные

Что хранить?

1. Код
2. Вспомогательные скрипты
3. Файлы конфигурации
4. Тестовые данные
5. Входные данные необходимые для работы вашего ПО

Что хранить?

1. Код
2. Вспомогательные скрипты
3. Файлы конфигурации
4. Тестовые данные
5. Входные данные необходимые для работы вашего ПО
6. Специфические зависимости (`libVasya.so`)

Какая польза?

```
git clone <project> <directory>
```

Поздравляю! Вы великолепны.

Модульное тестирование

- Регрессии не миф!

Модульное тестирование

- Регрессии не миф!
- Если разработчиков, более одного - суровая реальность

Модульное тестирование

- Регрессии не миф!
- Если разработчиков, более одного - суровая реальность
- Замечательное ощущение, когда видишь такое

```
C:\askyb\GTestSample\Release\GTestSample.exe
[=====] Running 3 tests from 3 test cases.
[-----] Global test environment set-up.
[-----] 1 test from MultiplyThreeTest
[ RUN     ] MultiplyThreeTest.InputNumber
[      OK ] MultiplyThreeTest.InputNumber (0 ms)
[-----] 1 test from MultiplyThreeTest (0 ms total)

[-----] 1 test from CompareCharTest
[ RUN     ] CompareCharTest.InputChar
[      OK ] CompareCharTest.InputChar (0 ms)
[-----] 1 test from CompareCharTest (0 ms total)

[-----] 1 test from SampleClassTest
[ RUN     ] SampleClassTest.InputNumber
[      OK ] SampleClassTest.InputNumber (0 ms)
[-----] 1 test from SampleClassTest (0 ms total)

[-----] Global test environment tear-down
[=====] 3 tests from 3 test cases ran. (0 ms total)
[ PASSED ] 3 tests.
```

Статический анализ кода

“Потерянный Бозон Хиггса”

```
if ( A != 2 || B != 3 || A != 3 || B != 2 )
```

Статический анализ кода

“Потерянный Бозон Хиггса”

```
void G4QEnvironment::DecayBaryon(G4QHadron* qH)
{
    ....
    else if(qM<mSzPi) // Only Lambda+PiM is possible
    {
        fQPDG=lQPDG;    // Baryon is Lambda
        fMass=mLamb;
        sQPDG=pimQPDG; // Meson is Pi-
        sMass=mPi;
    }
    else if(qM<mSzPi) // Both Lambda+PiM & Sigma0+PiM are possible
    {
        if(G4UniformRand()<.6)
        {
            ....
        }
    }
}
```

Статический анализ кода

“Потерянный Бозон Хиггса”

```
void G4QEnvironment::DecayBaryon(G4QHadron* qH)
{
    ....
    else if(qM<mSzPi) // Only Lambda+PiM is possible
    {
        fQPDG=lQPDG; // Baryon is Lambda
        fMass=mLamb;
        sQPDG=pimQPDG; // Meson is Pi-
        sMass=mPi;
    }
    else if(qM<mSzPi) // Both Lambda+PiM & Sigma0+PiM are possible
    {
        if(G4UniformRand()<.6)
        {
            ....
        }
    }
}
```

Динамический анализ кода

- Самый страшный враг сервисов 24/7...

Динамический анализ кода

- Самый страшный враг сервисов 24/7...



Динамический анализ кода

- Самый страшный враг сервисов 24/7...



Динамический анализ кода

- Профилировка
- Анализ покрытия кода тестами
- Отладка памяти

Динамический анализ кода

- Профилировка
- Анализ покрытия кода тестами
- Отладка памяти

```
ArrayType receiving( 3, 5 );

SomeAnotherArrayType init = { 1.0, 1.0, 1.0,
                             1.0, 1.0, 1.0,
                             1.0, 1.0, 1.0,
                             1.0, 1.0, 1.0,
                             1.0, 1.0, 1.0,
                             };

std::copy( init.cbegin(), init.cend(), receiving.begin() );
```

Динамический анализ кода

- Профилировка
- Анализ покрытия кода тестами
- Отладка памяти

```
ArrayType receiving( 3, 5 );

SomeAnotherArrayType init = { 1.0, 1.0, 1.0,
                             1.0, 1.0, 1.0,
                             1.0, 1.0, 1.0,
                             1.0, 1.0, 1.0,
                             1.0, 1.0, 1.0,
                             };

std::copy( init.cbegin(), init.cend(), receiving.begin() );
```

Интеграционное тестирование

- Вынесение длительных тестов

Интеграционное тестирование

- Вынесение длительных тестов
- Изоляция от окружения разработки

Интеграционное тестирование

- Вынесение длительных тестов
- Изоляция от окружения разработки
- Тестирование совместной работы
компонентов

Continuous integration

~~LEEROY~~



Jenkins

Jenkins

Benutzer

Build-Verlauf

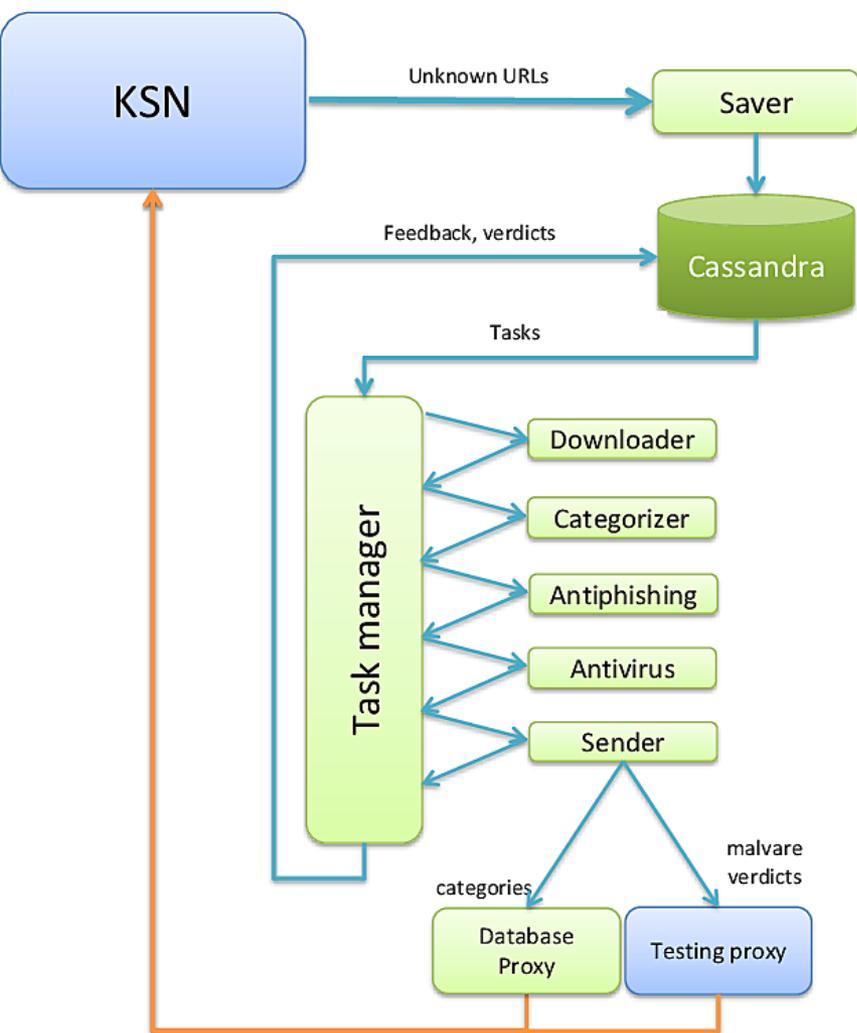
Build Warteschlange

S	W	Name ↓	Letzter Erfolg	Letzter Fehlschlag	Letzte Dauer
		android	44 Minuten (cm_manta-userdebug)	1 Stunde 14 Minuten (cm_iewel-userdebug)	30 Minuten
		android-build-all-lunches	2 Monate 24 Tage (#146)	4 Monate 16 Tage (#109)	17 Sekunden
		cm-build-all	8 Stunden 5 Minuten (#122)	Unbekannt	2 Minuten 1 Sekunde
		cm_daily_build_cycle	8 Stunden 5 Minuten (#235)	6 Monate 22 Tage (#25)	0.48 Sekunden
		recovery	31 Minuten (4f1cac2a9bac78321ed06c7b00360f34)	6 Minuten 32 Sekunden (9c5440be96d4d1eee566a4862337b4e0)	3 Minuten 20 Sekunden
		submission-test	3 Stunden 31 Minuten (gerrit-test-34890)	15 Stunden (gerrit-test-34681)	16 Minuten

Symbol: [S](#) [M](#) [L](#)

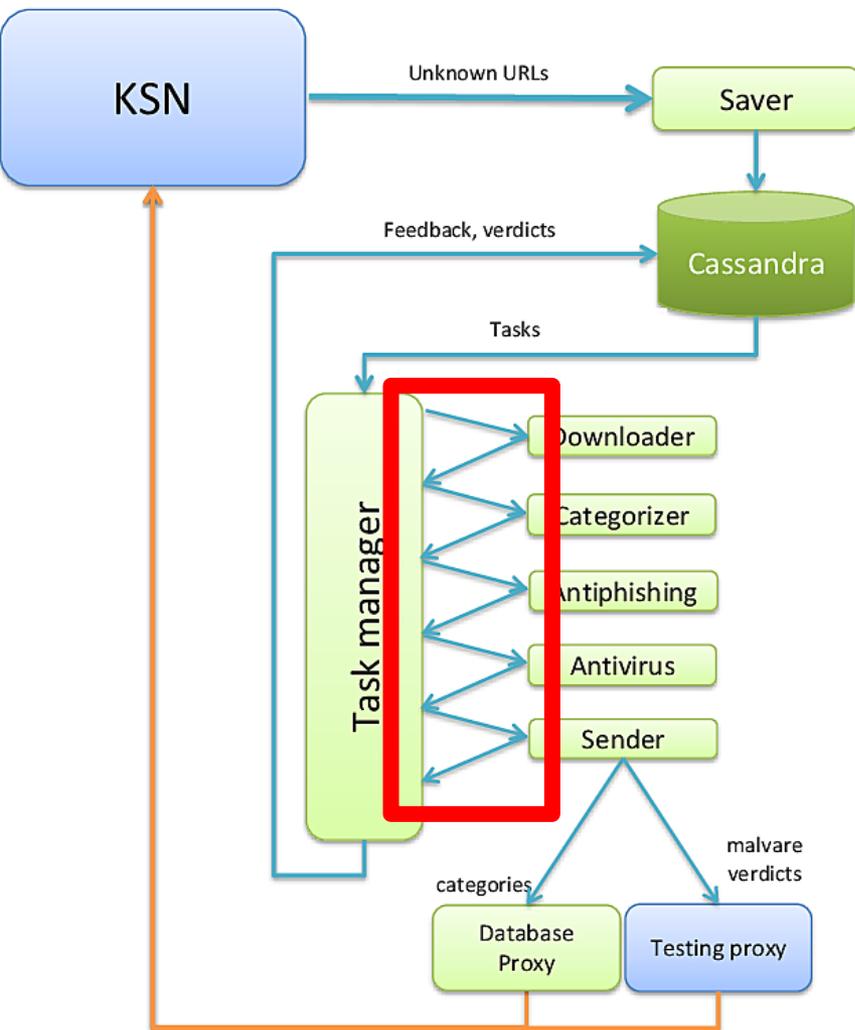
[Legende](#) [RSS Alle Builds](#) [RSS Nur Fehlschläge](#) [RSS Nur jeweils letzter Build](#)

Схема сервиса



1. Получение запроса (из “облака”).
2. Сохранение в БД
3. Помещение в очередь задач
4. Процессинг на конвейере сканеров
5. Ответ в “облако”
6. Сохранение новых данных в БД

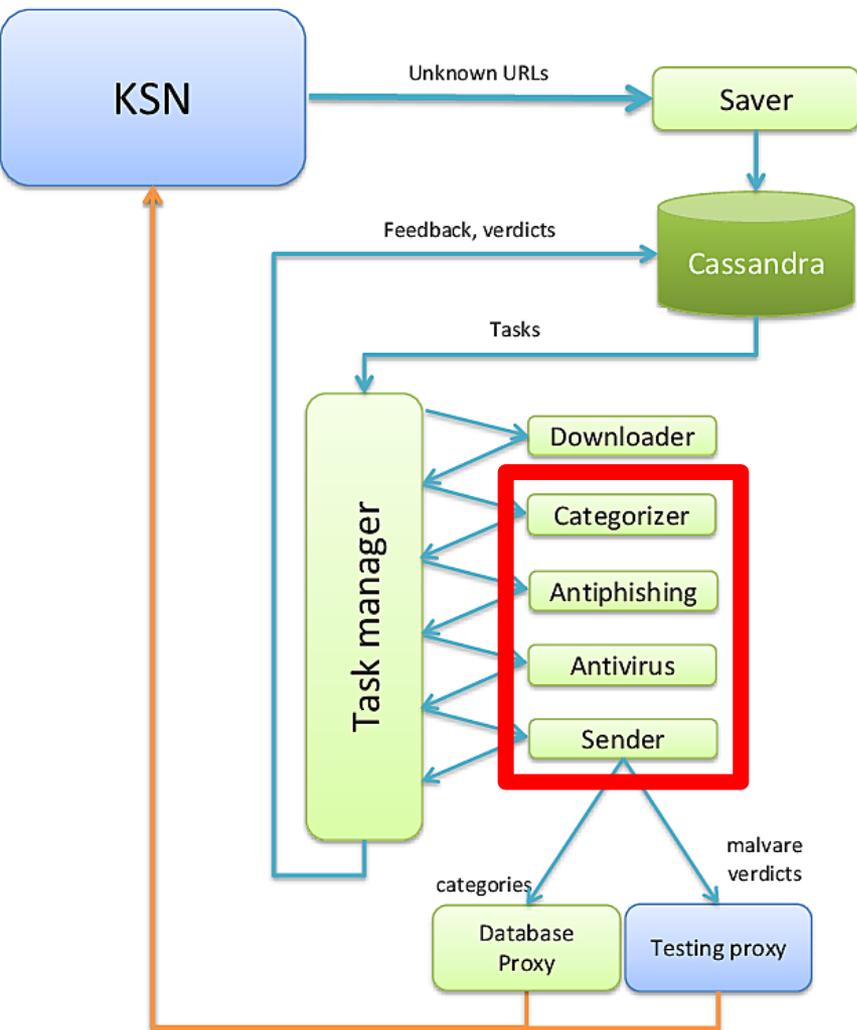
Горизонтальное масштабирование



На уровне машин

- распределенные очереди

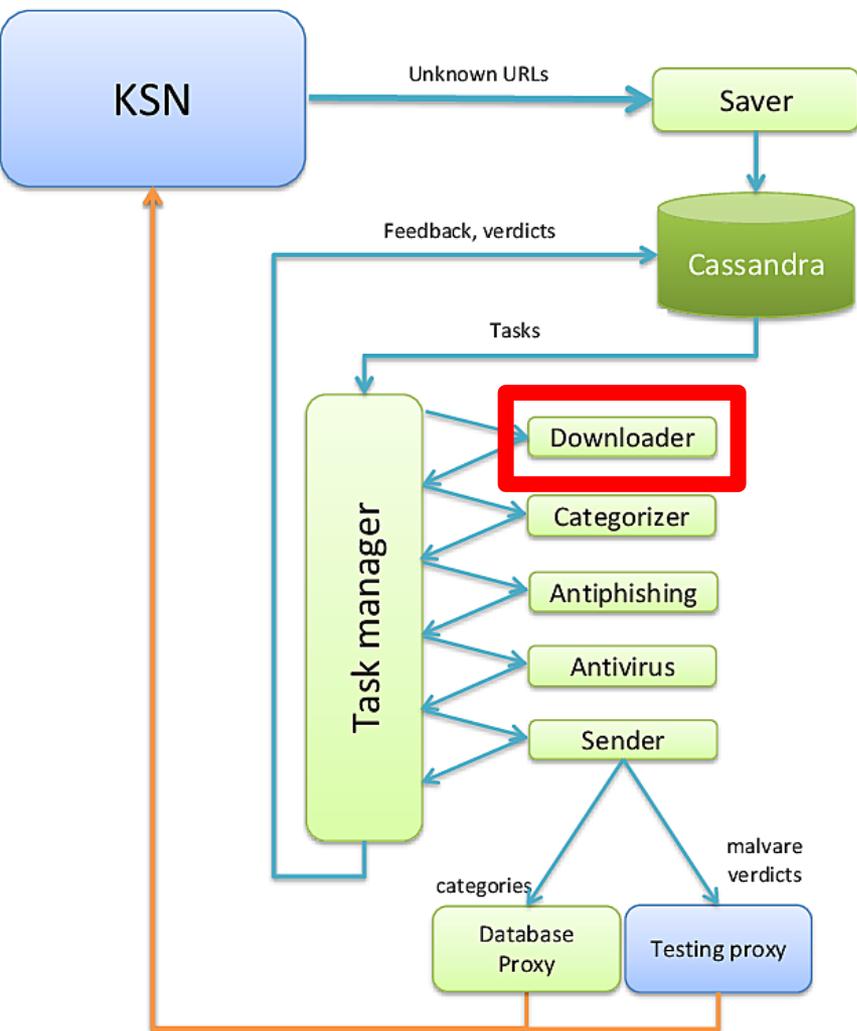
Горизонтальное масштабирование



На уровне процессоров/ядер

- МНОГОПОТОЧНОСТЬ

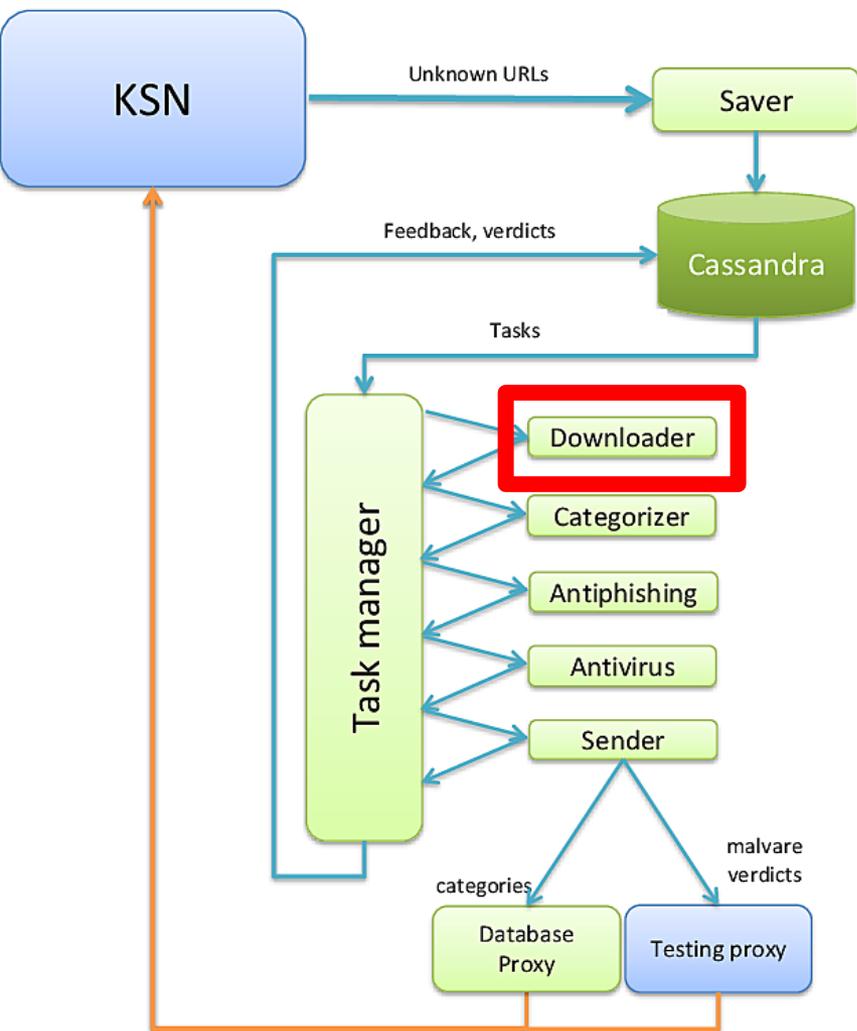
Горизонтальное масштабирование



На уровне процессоров/ядер

- МНОГОПОТОЧНОСТЬ
- МНОГОПРОЦЕССНОСТЬ

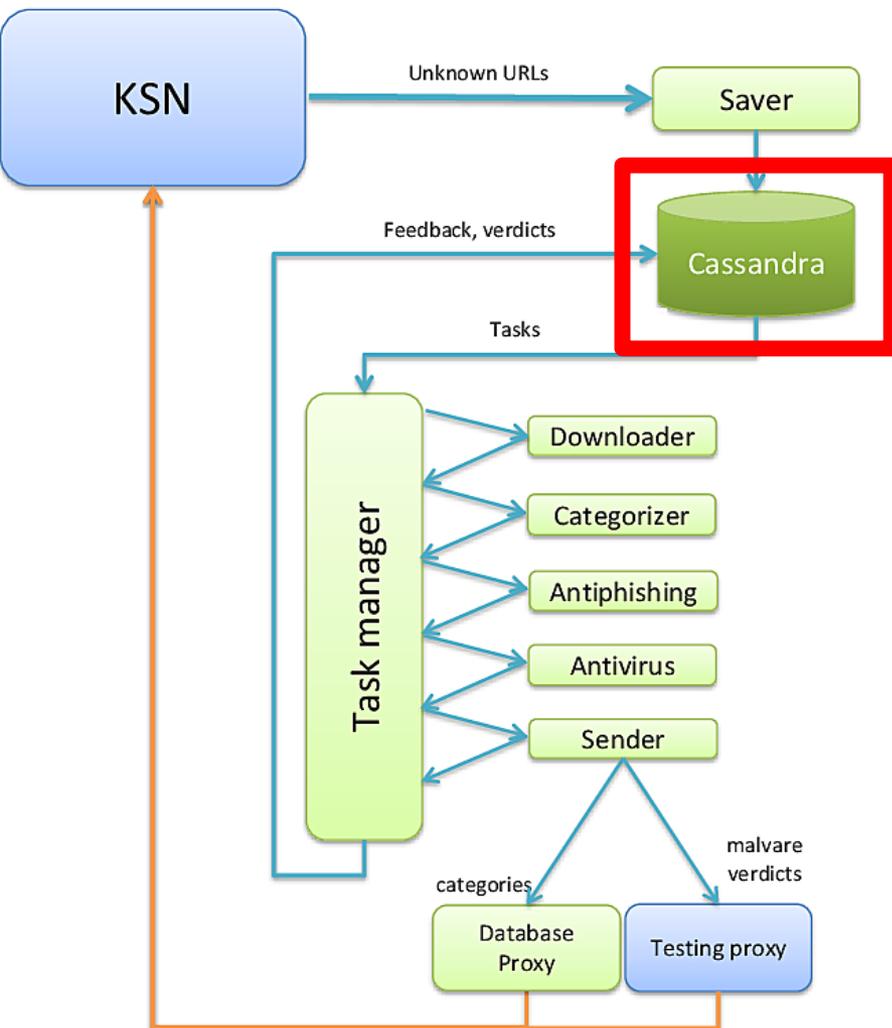
Горизонтальное масштабирование



На уровне процессоров/ядер

- **МНОГОПОТОЧНОСТЬ**
- **МНОГОПРОЦЕССНОСТЬ**
- **утечки памяти в третьесторонних компонентах**

Горизонтальное масштабирование



На уровне подсистемы хранения данных

NoSQL

- Cassandra
- Apache HBase
- MongoDB
- etc

Расширение “узких мест”

Использование жесткого диска:

3 М сайтов; из них **0.1М** с вердиктами

Расширение “узких мест”

Использование жесткого диска:

3 М сайтов; из них **0.1М** с вердиктами

1. Сохраняем всё; удаляем без вердиктов

Расширение “узких мест”

Использование жесткого диска:

3 М сайтов; из них **0.1М** с вердиктами

~~1. Сохраняем всё; удаляем без вердиктов~~

Расширение “узких мест”

Использование жесткого диска:

3 М сайтов; из них **0.1М** с вердиктами

- ~~1. Сохраняем всё; удаляем без вердиктов~~
2. Сохраняем всё; переносим с вердиктами;
удаляем раздел

Расширение “узких мест”

Использование жесткого диска:

3 М сайтов; из них **0.1М** с вердиктами

- ~~1. Сохраняем всё; удаляем без вердиктов~~
- ~~2. Сохраняем всё; переносим с вердиктами;
удаляем раздел~~

Расширение “узких мест”

Использование жесткого диска:

3 М сайтов; из них **0.1М** с вердиктами

- ~~1. Сохраняем всё; удаляем без вердиктов~~
- ~~2. Сохраняем всё; переносим с вердиктами;
удаляем раздел~~
3. Всё держим в памяти; сохраняем только с вердиктами