



DevCon School

Технологии будущего

Веб приложения в облаке: проектирование и отладка

Alexey Bokov

Senior Program Manager, Microsoft Corp



Q&A

Веб приложения в облаке: проектирование и отладка

Alexey Bokov

Senior Program Manager, Microsoft Corp

abokov@microsoft.com; twitter: @abokov

#msdevcon

Содержание

Веб приложение – основные компоненты

Сервисы для улучшения доступности веб-приложения

Azure DNS

Traffic Manager

Балансировка нагрузки

Application Gateway

Azure Internal Load Balancer

Веб приложения = App Services

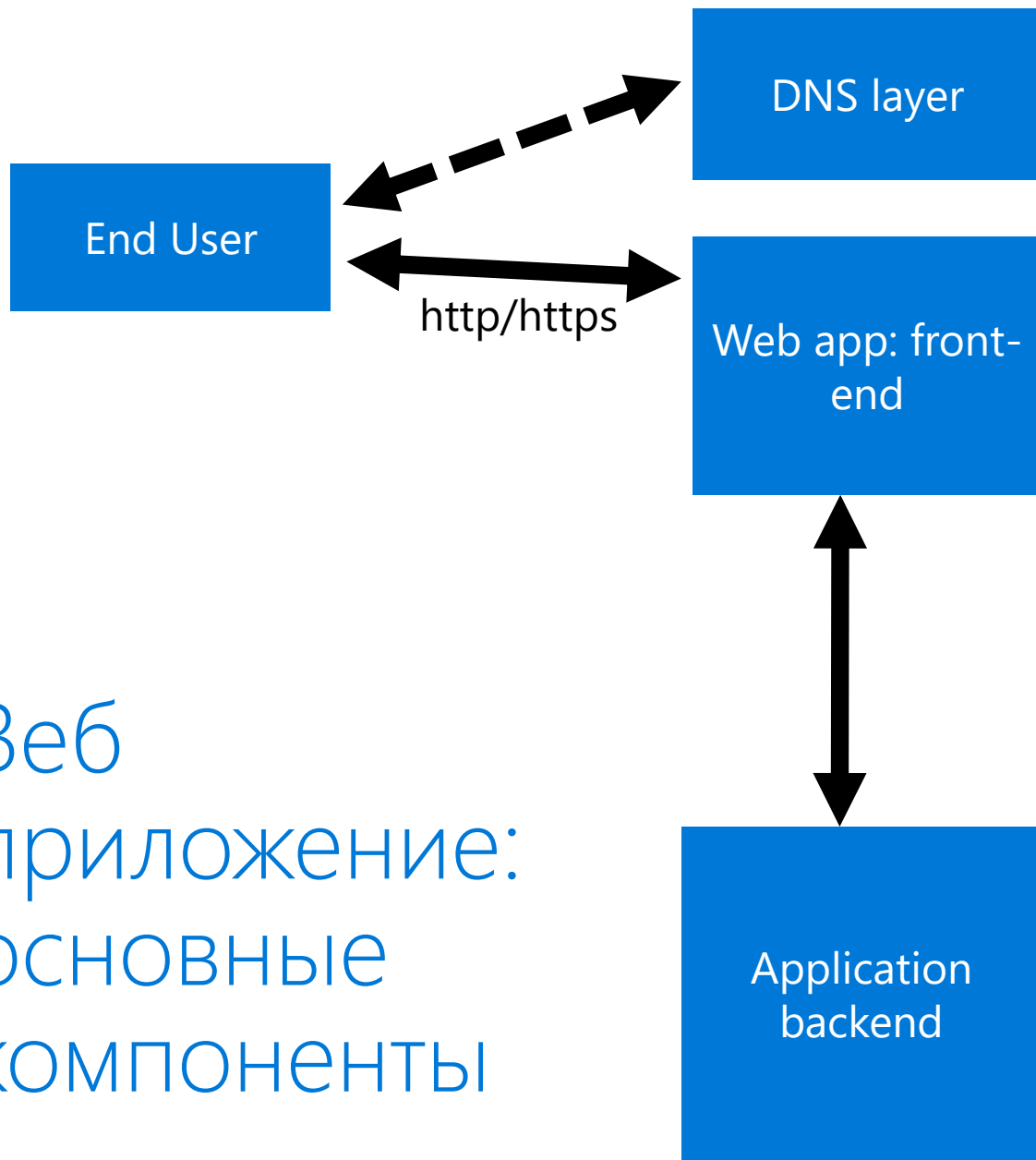
Типы тарифов (plans) и разница между ними

Веб фермы (staging)

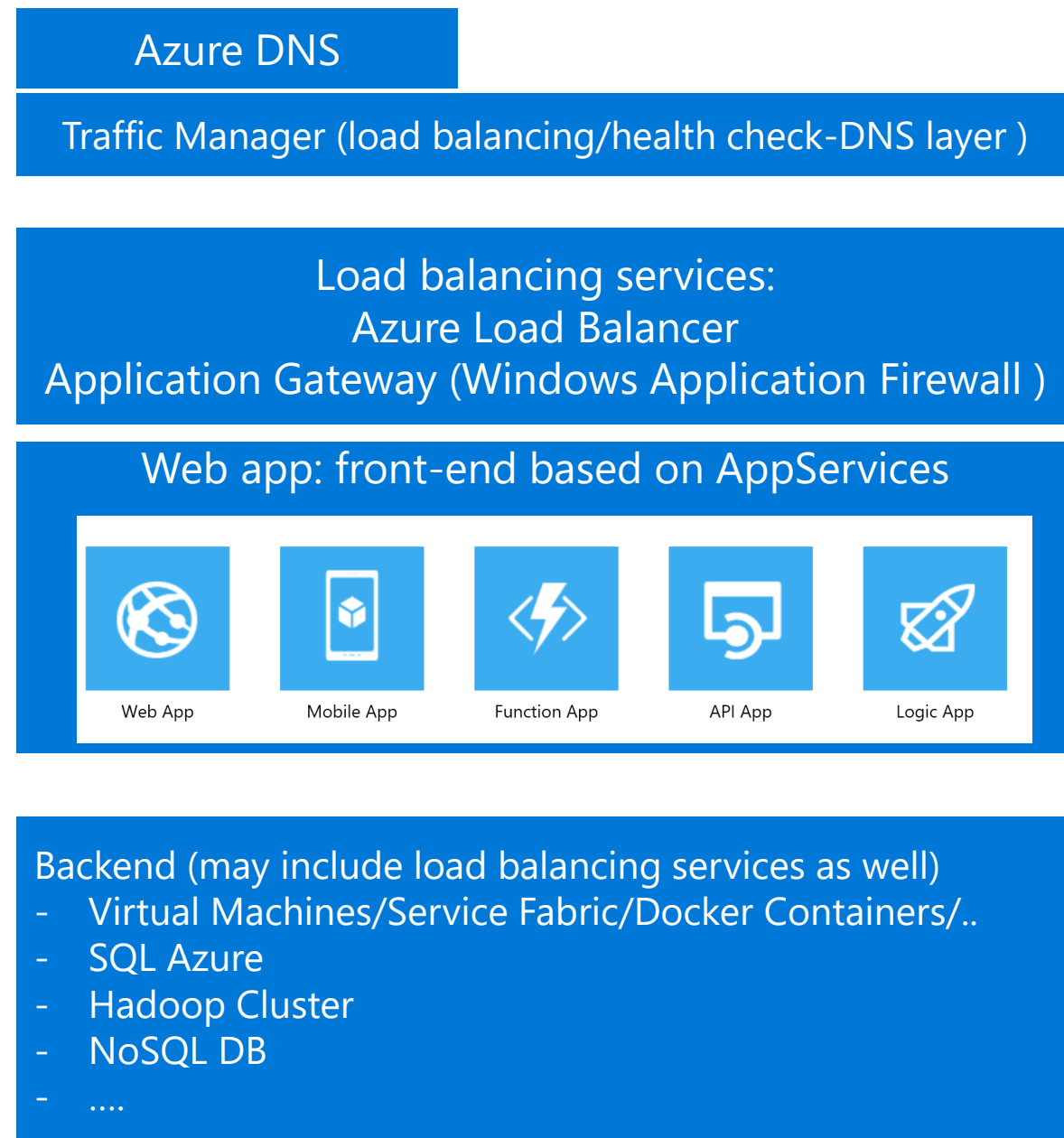
Задачи Web job

Практическая часть

Веб приложение — основные компоненты



Веб
приложение:
ОСНОВНЫЕ
КОМПОНЕНТЫ



Балансировка и отказоустойчивость

Уровень DNS

Azure DNS

Traffic Manager

Уровень приложения

Azure Internal Load Balancer (ALB)

Application Gateway + Web Application Firewall (WAF)

Сравнительная таблица

Сервис	Azure Load Balancer	Applicaion Gateway	Traffic Manager
Стек	Транспортный уровень Level 4	Уровень приложения Level 7	DNS
Поддерживаемые протоколы	Весь стек TCP	HTTP/S	Все, к которым применим DNS
Конечные точки (endpoints)	Виртуальные машины, облачные сервисы в Azure	Любой IP – внутренний (внутри Azure) или внешний	Любой сервис у которого есть FQDN или публичный IP
Виртуальные сети	Публичные (в интернет) и внутренние	Публичные(в интернет) и внутренние	Только публичные
Мониторинг	Пробы	Пробы	Пробы http/s GET

Сервисы для улучшения доступности веб-приложения

#msdevcon

Azure DNS

Сервис хостинга записей DNS

Для хостинга DNS записей в Azure DNS используется глобальная сеть DNS name серверов

Для ответов на запросы DNS используется anycast протокол, ответ будет от ближайшего DNS сервера

Поддерживается только делегирование доменов (т.е. покупать надо у регистратора)

Azure DNS поддерживает общеупотребимые DNS записи типов, включая A, AAAA, CNAME, MX, NS, SOA, SRV, TX, а также wildcards (*)

Azure Traffic Manager

Работает на уровне DNS

- Повышение надежности и доступности для критических приложений
- Улучшение времени ответа для веб приложений с высокой нагрузкой – позволяет использовать веб приложения в обалке Azure или _любом_другом_ веб хостинге
- Позволяет делать обновления и вести технические работы без прерывания работы веб сайта
- Использовать для веб сайта свой хостинг и облако Azure – Traffic Manager поддерживает внешние, не-Azure конечные точки и позволяет реализовывать сценарии использования облака как backup хостинга для основной реплики веб сайта
- Распределение трафика для больших веб проектов – возможно использование вложенных правил для профилей Traffic Manager профилей

Azure Traffic Manager

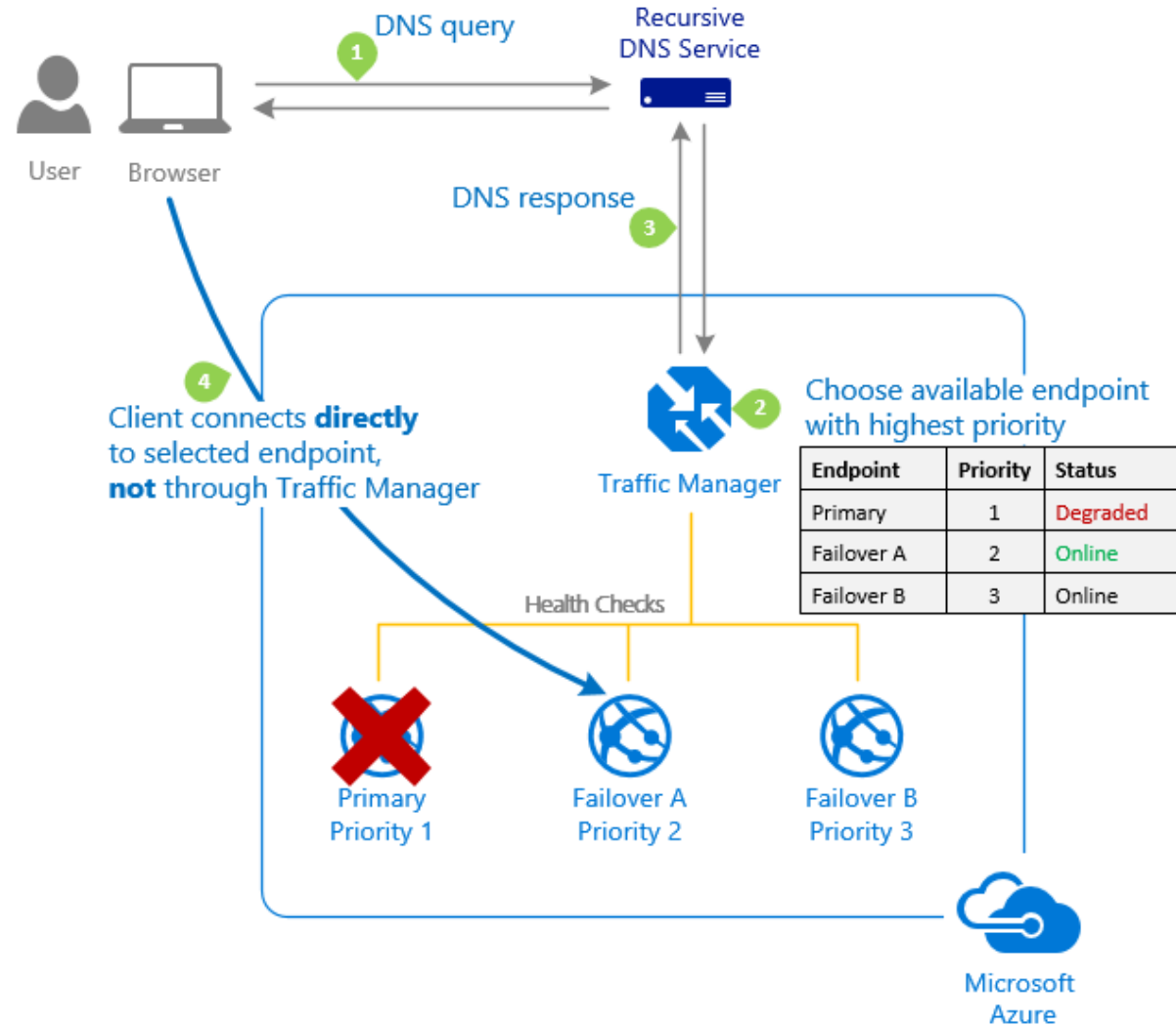
Методы распределения трафика

Priority: Весь трафик идет на основную реплику, при проблемах с основной репликой, подключаются backup реплики согласно их приоритету

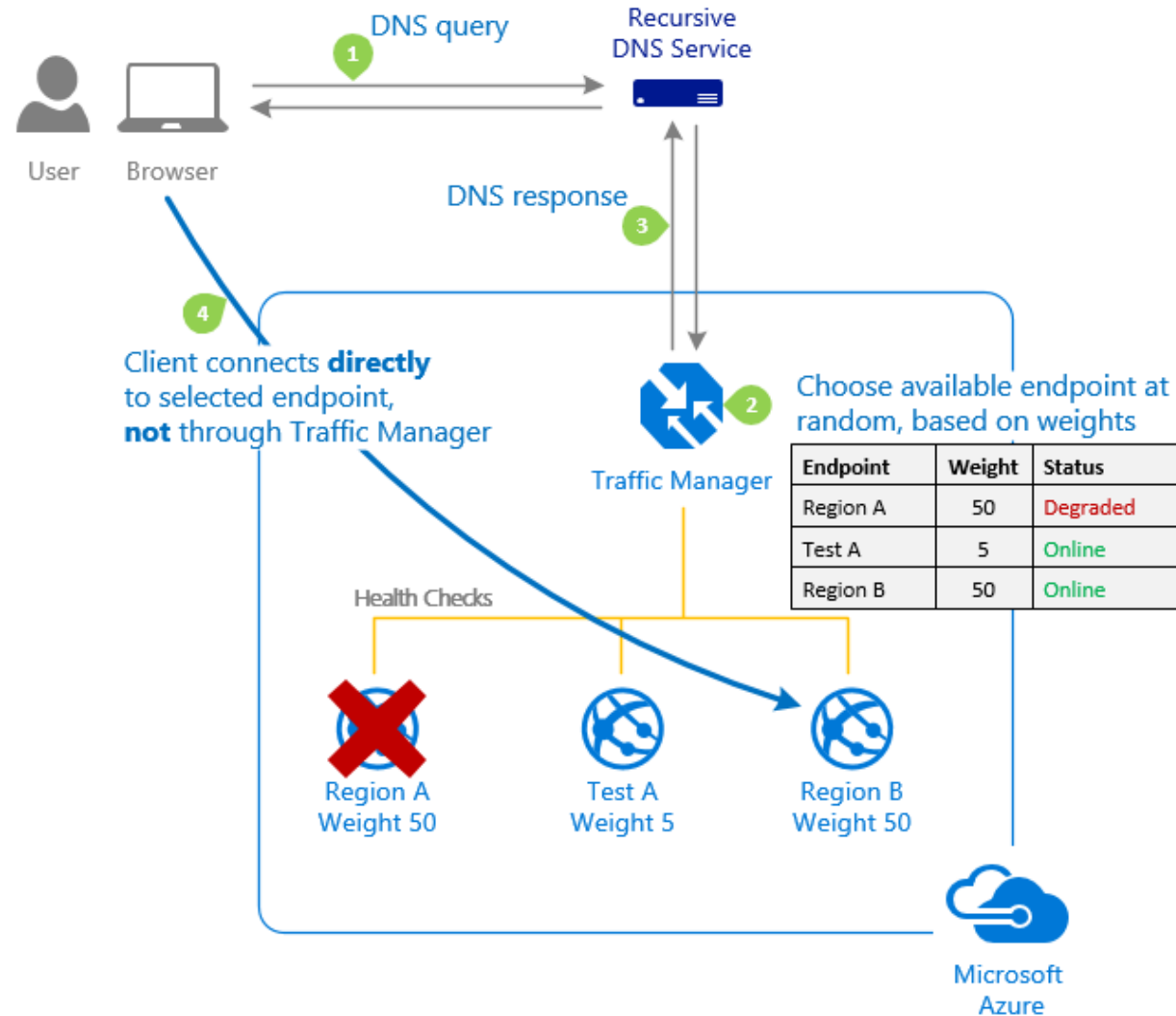
Weighted: Трафик распределяется между репликами (конечными точками) согласно их весу

Performance: Распределение трафика с учетом производительности и георасположения по отношению к пользователю

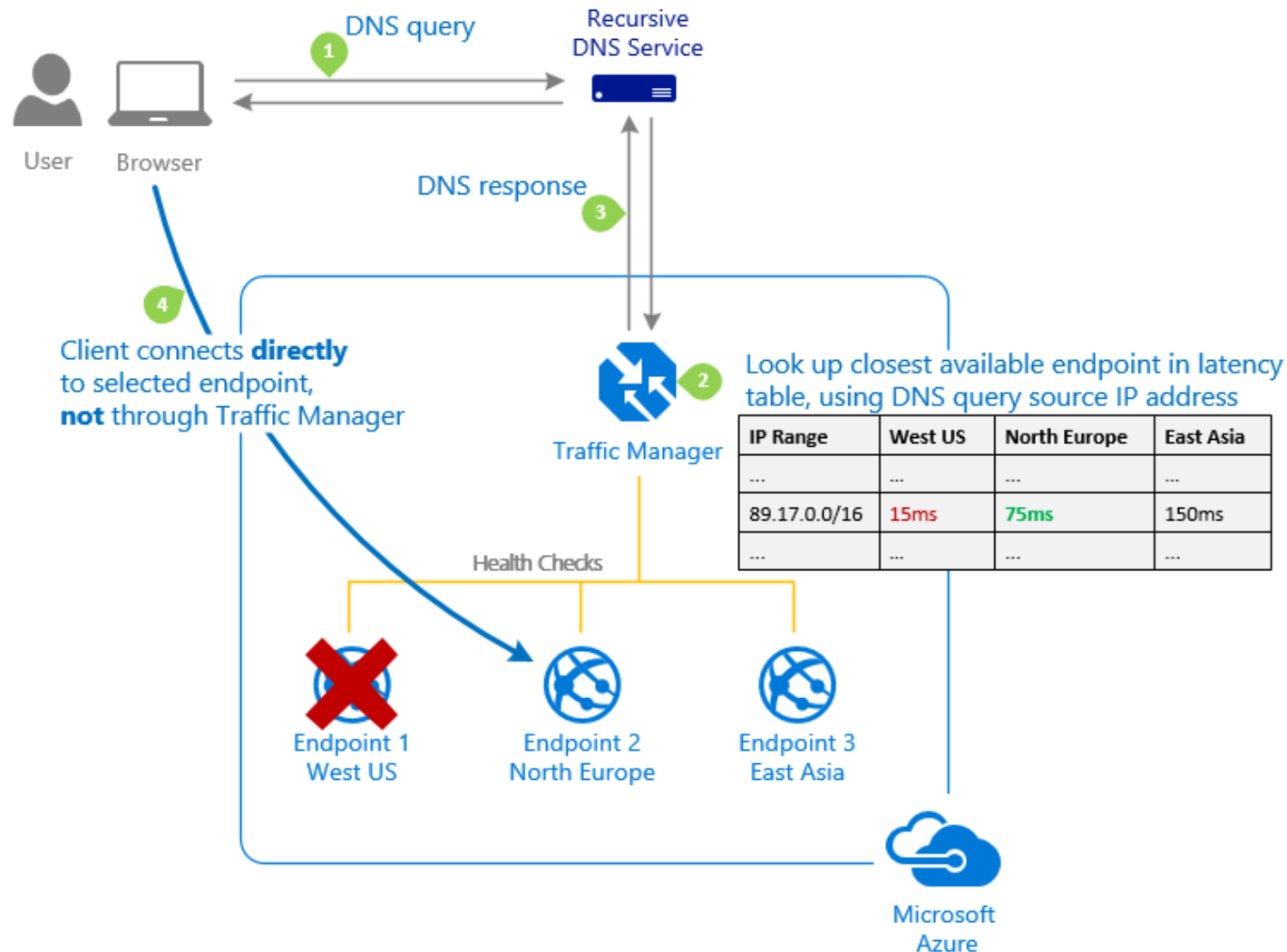
Azure Traffic Manager: priority



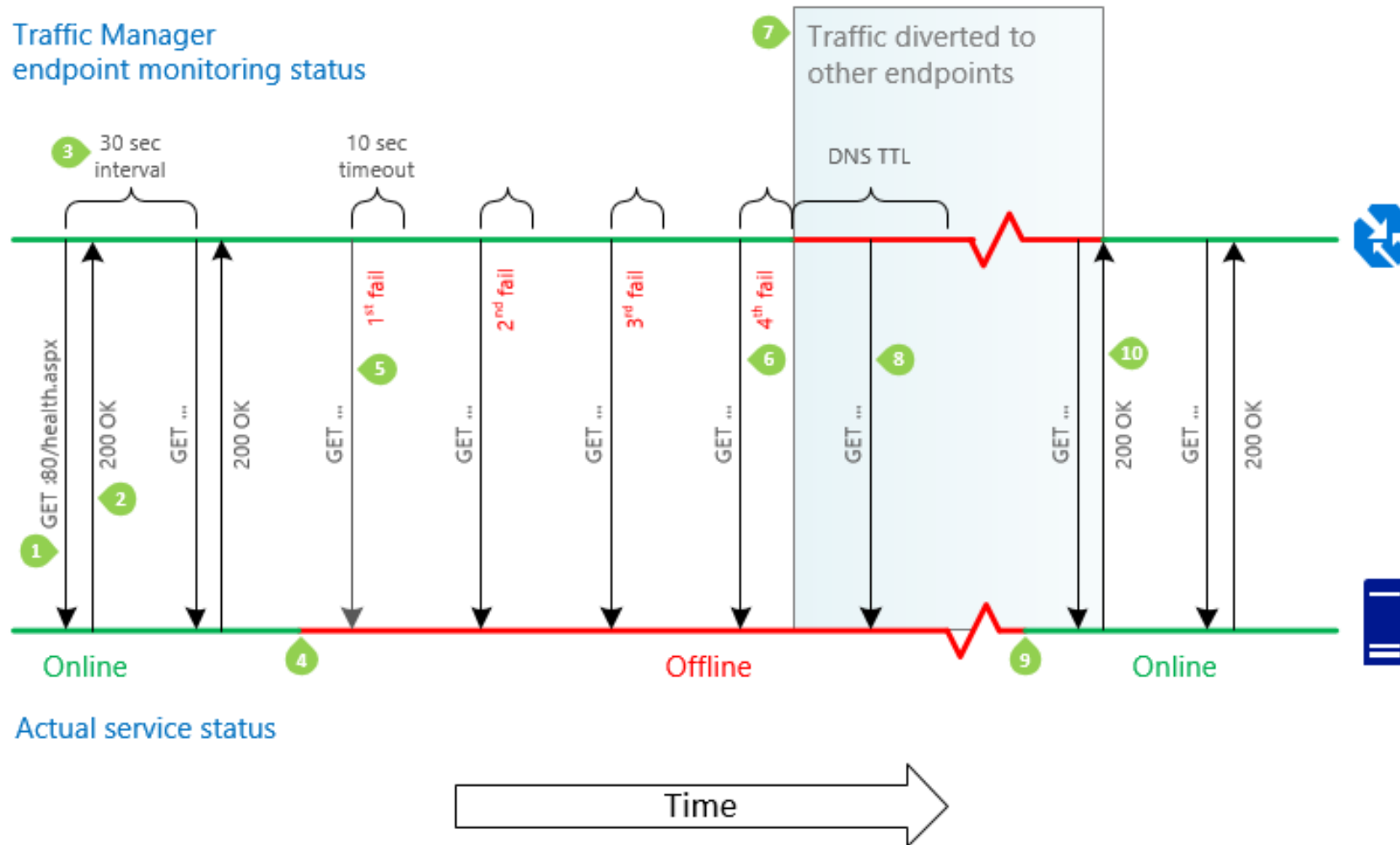
Azure Traffic Manager: weighted



Azure Traffic Manager: performance



Azure Traffic Manager: пример



Балансировка нагрузки

#msdevcon

Azure Internal Load Balancer

Azure Load Balancer (ALB)

Azure Internal Load Balanced

Работает на Layer 4 (TCP, UDP)

Основные сценарии

Балансировка интернет трафика на виртуальные машины

Балансировки трафика внутри кластеров:

Между виртуальными машинами в VNET

Между виртуальными машинами внутри одног облачного сервиса

Между ресурсами на своем хостинге и облачными виртуальными машинами внутри VNET

Перенаправление внешнего трафика на виртуальную машину

Azure Load Balancer – функционал [1/2]

Распределение трафика на основе хэша:

- По умолчанию : 5-tuple (исходный IP, исходный порт, конечный IP, конечный порт, тип протокола) хэш для назначения трафика доступным серверам
- Привязка к конкретному серверу (Stickiness) _только_ внутри транспортной сессии
- Пакеты из одной TCP или UDP сессий всегда будут направлены в один и тот же конечный сервер
- Когда клиент закрывает и переоткрывает соединение или запускает новую сессию (при сохранении исходного IP неизменным) исходный порт может измениться – и это может привести к использованию другой конечного сервера при балансировке

Перенаправление портов

Azure Load Balancer – функционал [2/2]

Автоматическое переконфигурирование

В случае изменений в кластере (scale up/down)

Мониторинг сервисов с помощью запросов *probe*

Guest agent probe (on PaaS VMs aka Web/Worker roles): проверяем статус агента внутри виртуальных машин

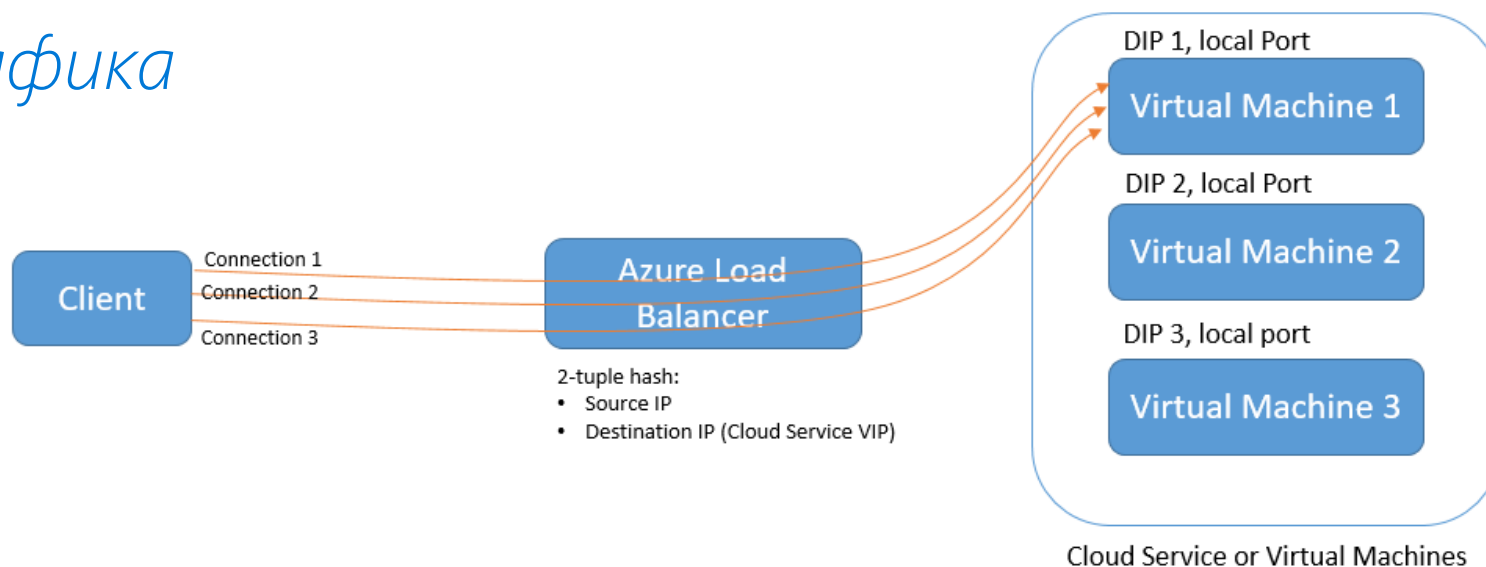
HTTP custom probe: проверяем приложение каждый 15 секунд через TCP ACK или HTTP 200 в ответе

TCP custom probe: проверяем установление TCP сессии на указанном порту

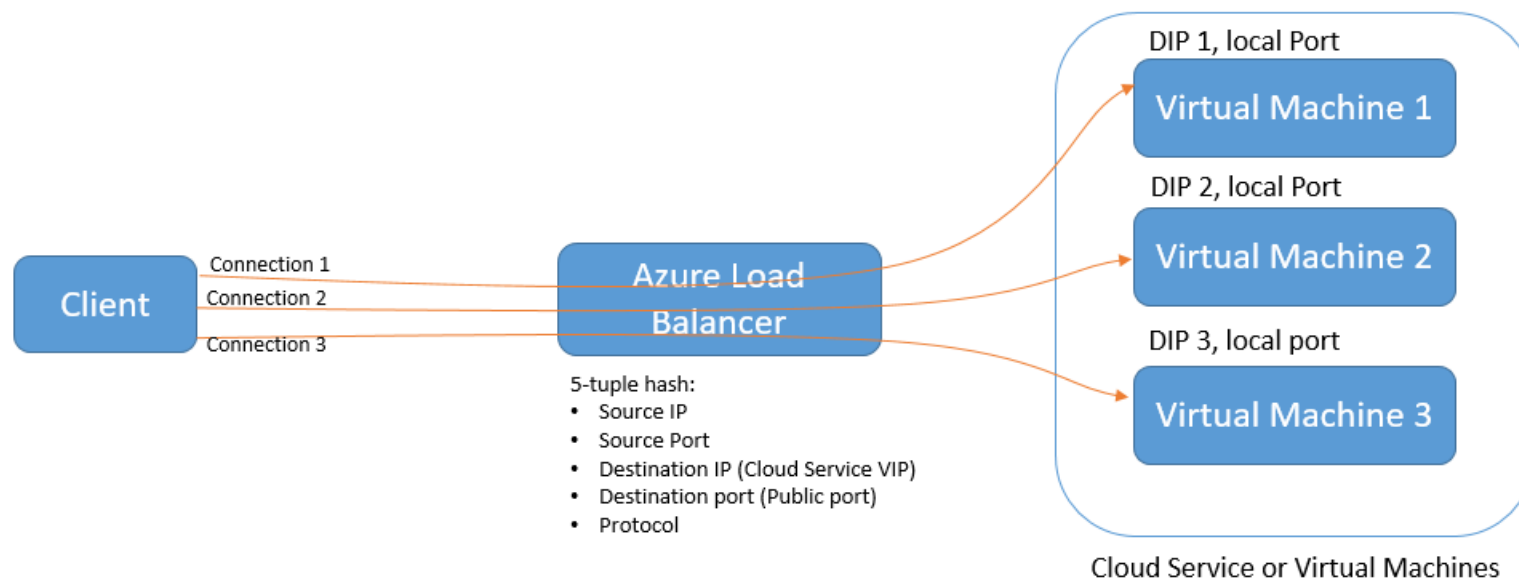
Azure Load Balancer

Виды распределения трафика

IP Affinity mode: use (Source IP, Destination IP, Protocol) to map traffic

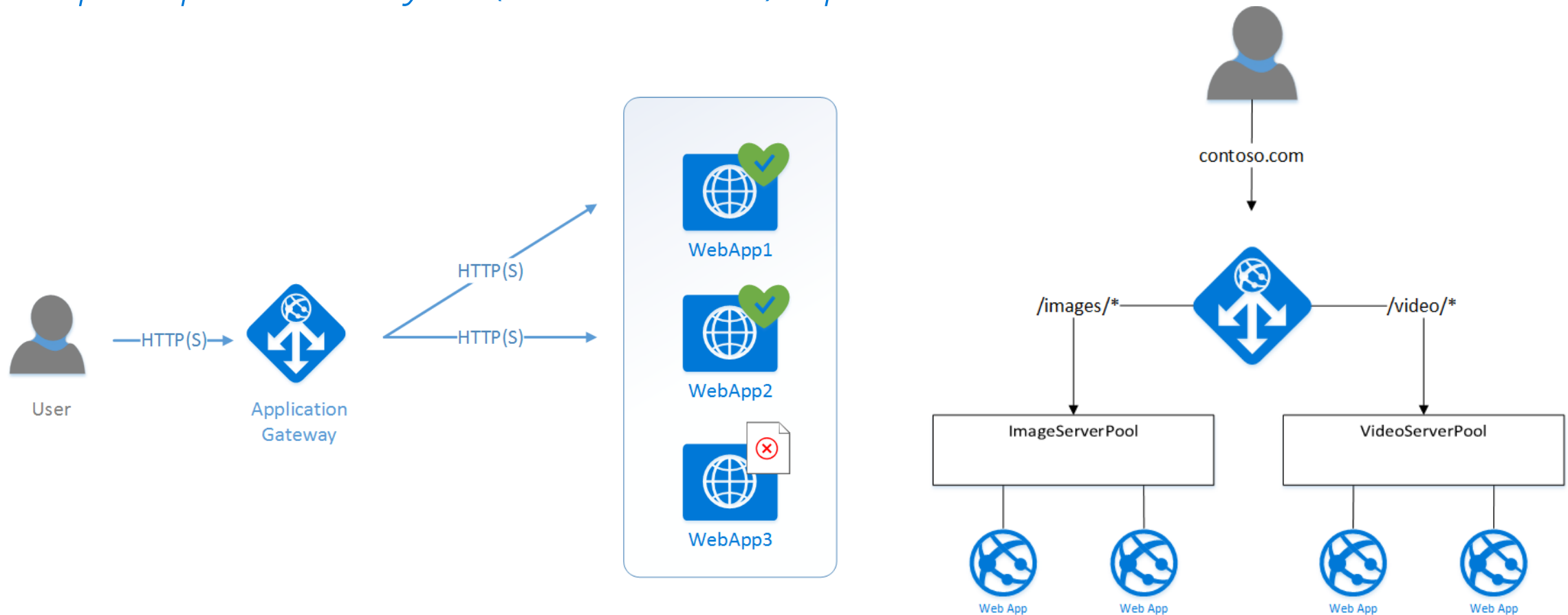


Hash based mode: use (source IP, source port, destination IP, destination port, protocol type) hash to map traffic, stickiness only within a transport session



Azure Application Gateway

Балансировка и перенаправление запросов, сетевой экран и проверка статуса (health check) приложения



Azure Application Gateway

Application Gateway работает на layer-7

Ключевые сценарии:

Балансировка HTTP

Привязка сессии на базе Cookie при балансировке

Снятие нагрузки Secure Sockets Layer (SSL) с конечного веб-сервера

Перенаправление трафика по маске из URL

Перенаправление трафика между разными сайтам

Azure Application Gateway

Типичные сценарии для балансировки HTTP L7

Приложения, которым требуется обработка запросов от пользователя всегда на одном и том же сервере в кластере внутри одной сессии

Снять SSL нагрузку с веб фронтенда

Перенаправление или балансировка http запросов внутри одного TCP соединения на разные бэкенд сервисы

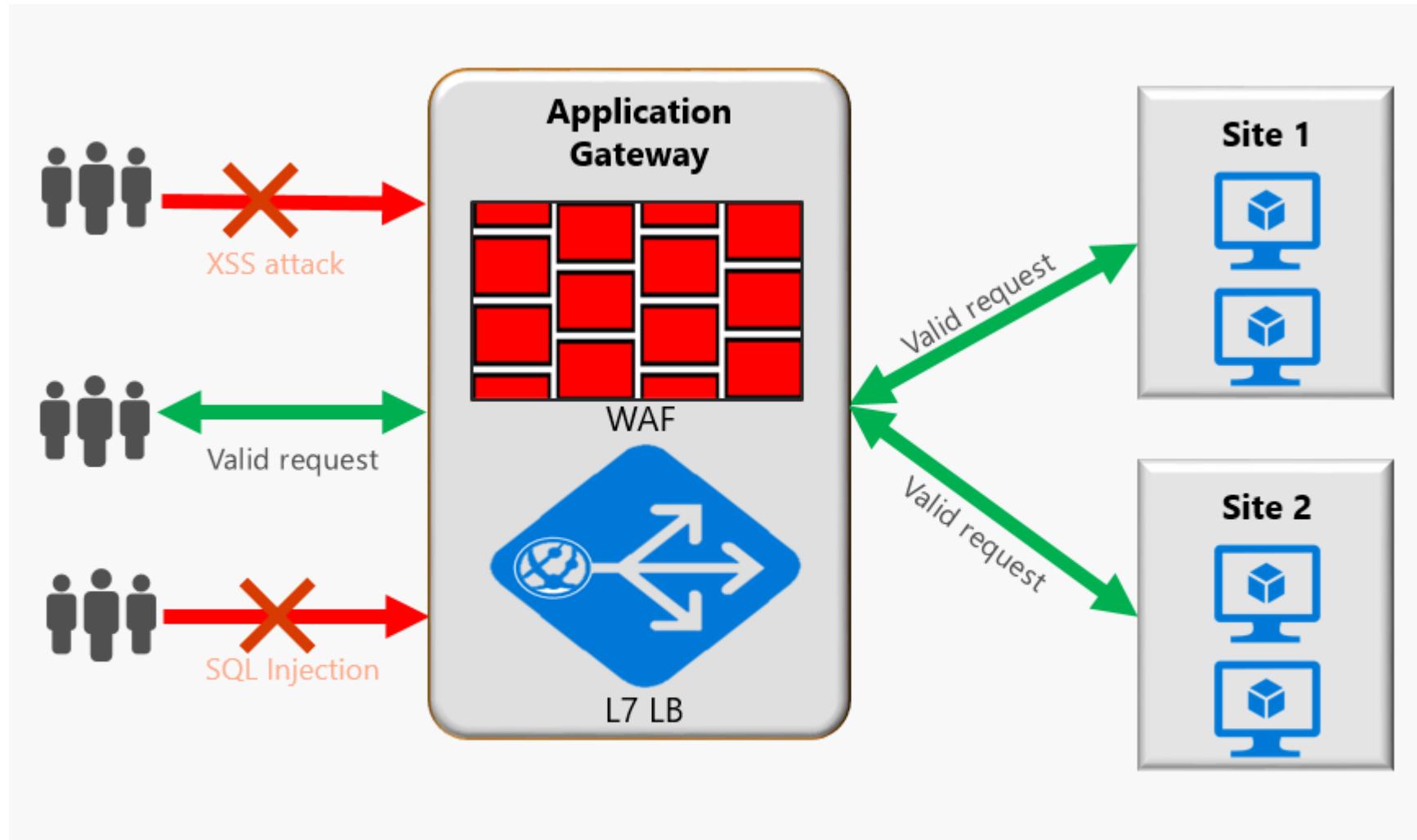
Azure Application Gateway

В зависимости от типа Application Gateway имеет разные характеристики, small – рекомендуется для dev/qa

Back-end page response	Small	Medium	Large
6 K	7.5 Mbps	13 Mbps	50 Mbps
100 K	35 Mbps	100 Mbps	200 Mbps

Azure Application Gateway

Web Application Firewall (WAF)



Web Application Firewall

Базовый уровень защиты против топ 10 угроз согласно OWASP

SQL injection : GET `http://testphp.vulnweb.com/artists.php?artist=-1 UNION SELECT 1,pass,cc FROM users WHERE uname='test'` HTTP/1.1

Cross site scripting (XSS) protection

Распространенные атаки в интернет: command injection, HTTP request smuggling, HTTP response splitting, and remote file inclusion attack

Нарушения протокола HTTP, в том числе такие как отсутствующий host user-agent или accept headers

Защита HTTP DoS включая HTTP flooding и защита от slow HTTP DoS

Защита от ботов, интернет сканнеров и тп

Защита и определение неверное конфигурации веб-сервисов (i.e. Apache, IIS, etc)

OWASP = Open Web Application Security Project - owasp.org

Web Application Firewall

Application Gateway WAF может использоваться в двух режимах

Detection mode – WAF мониторит и логирует все события:

- *Logging diagnostics внутри Application Gateway должна быть включена ON в секции Diagnostics*
- *WAF логи должны быть выбраны и включены N*

Prevention mode – блокирует атаки и предотвращает доступ к атакуемому ресурсу:

- *Злоумышленник получает 403, соединение закрывается*
- *Атака логирует в логах*

Альтернативным вариантом для защитного экрана является использование кластера с Barracuda application firewall из Azure Marketplace настроенного в качестве фронт-енда перед веб сайтом.

Application Gateway vs Load Balancer

Тип	Azure Load Balancer	Application Gateway
Протокол	UDP/TCP	HTTP/HTTPS
Пробы	Интервал 15 секунд, индикация ошибки через два цикла по таймауту, поддерживают кастомизацию	30 секунд, индикация ошибки через 5 последовательных ошибок при наличии трафика и одна проба при ждущем режим, поддерживают кастомизацию
SSL	Нет	да

Веб приложения = App Services

Azure App Services



Web App



Mobile App



Function App



API App



Logic App

App Service планы

Рекомендуется для try/dev/test				Рекомендуются для коммерческого продакшна	
	Free	Basic	Shared	Standard	Premium
Число приложений	10	100	Unlimited	Unlimited	Unlimited
Disk space	1 GB	1 GB	10 GB	50 GB	500 GB
Max instances	--	--	Up to 3	Up to 10	Up to 50
SLA	--	--	99.95%	99.95%	99.95%
Auto scale	--	--	--	Supported	Supported
Geo-distributed environment	--	--	--	Supported	Supported
VPN connectivity	--	--	--	Supported	Supported

App Service планы: продолжение

	Free	Basic	Shared	Standard	Premium
Custom domain	10	100	Unlimited	Unlimited	Unlimited
SSL certificates	--	--	Unlimited SNI SSL certs	Unlimited SNI SLL + 1 IP SSL included	Unlimited SNI SLL + 1 IP SSL included
Auto backups/day	--	--	Up to 3	Up to 10	Up to 50
Active mobile devices	500 / day	500 / day	Unlimited	Unlimited	Unlimited
Offline sync/day	500 calls	1k calls	1k calls	Unlimited	Unlimited
Staging environment	--	--	--	5	20

App Service планы

Autoscale : по расписанию или уровню загрузки CPU (который вычисляется как _среднее_ значение по кластеру)

* Scale by

Description

Settings

Add Profile

Geo-distributed environment: веб-сайты внутри плана могут размещаться в разных регионах

Auto backups: бэкапы для App configuration, файлов и Azure SQL Database/Azure MySQL подключенных к приложению, бэкапы хранятся в сторадж аккаунте, делаются по расписанию

Active mobile devices: related to push/call notifications for mobile devices via Notification Hub.

Offline sync: sync for mobile apps under Azure Mobile Apps, for offline mode.

Staging environments: поддержка размещения в стейджингах, включая разделение трафика в пропорциях между разными окружениями

Возможности Azure Websites

Надежность

Спроектированы для критически важных приложений

Hybrid Connections / VPN Support
Scheduled Backup
Azure Active Directory Integration
Site Resiliency, HA, and DR
Web Jobs
Role Base Access Control
Audit / Compliance
Enterprise Migration
Client Certs
Redis Caching
IP Restrictions/ SSL
Web Sockets
SQL, MySQL, DocDB, & Mongo

Глобальность

Оптимизированы для высокого уровня доступности и автоматического масштабирования

Automated Deployment
AutoScale
Built-in Load Balancing
WW Datacenter Coverage
End Point Monitoring & Alerts
App Gallery
DR Site Support
Wildcard Support
Dedicated IP address
HTTP Compression
WebJobs
Sticky Sessions

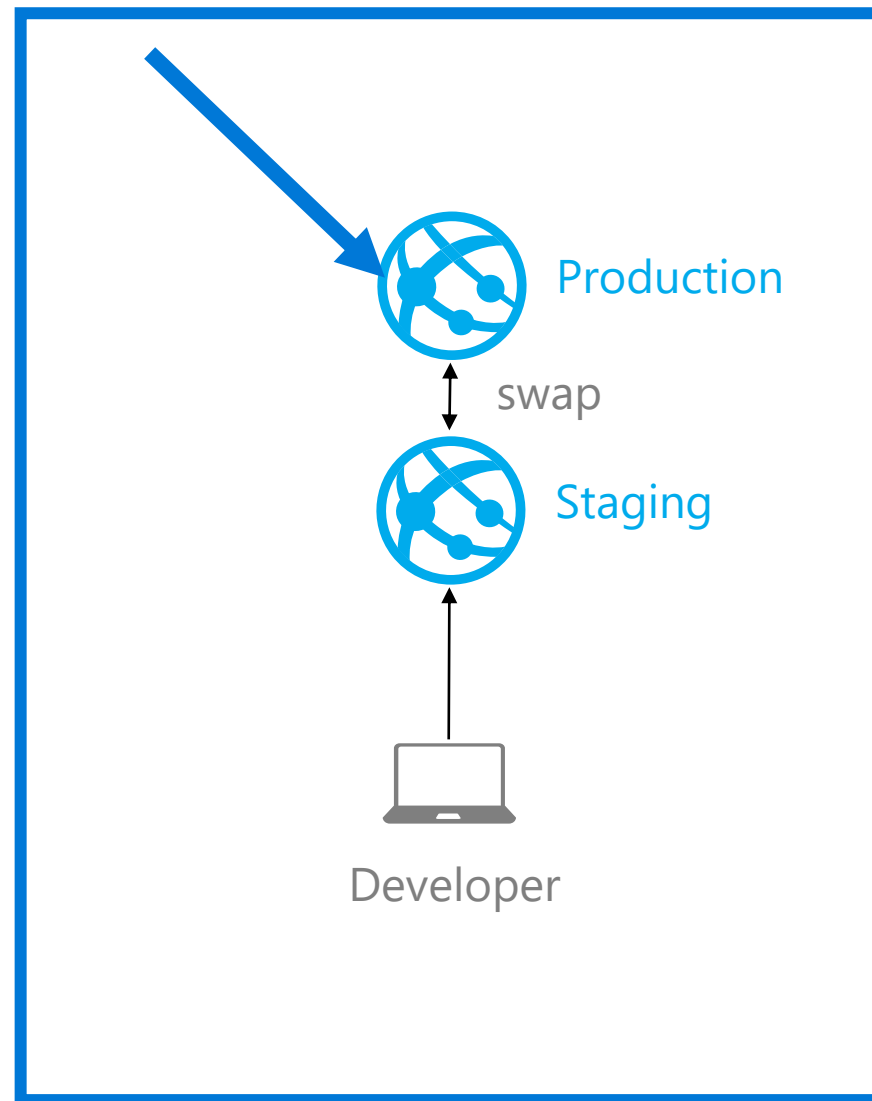
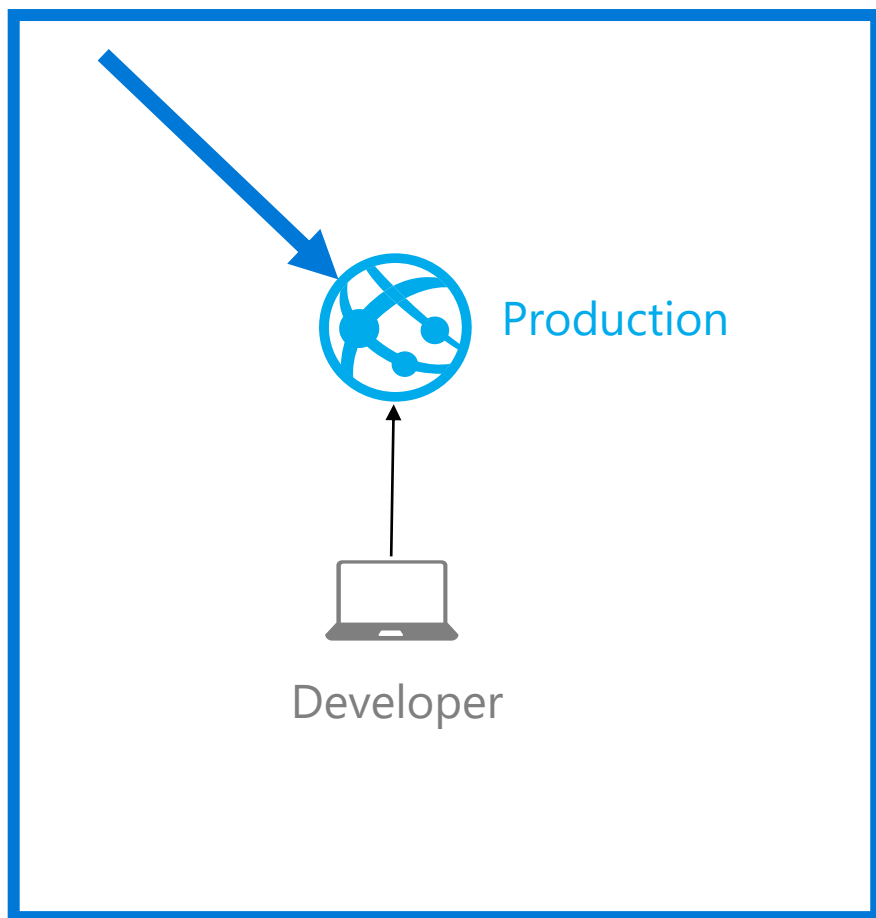
Сделаны для DevOps

Удобные инструменты платформы для Continuous Deployment

Remote Debugging w/ Visual Studio
Site Staging Slots
Testing in Production
Continuous Integration/Deployment
Git, Visual Studio Online and GitHub
App & Site Diagnostics
OS & Framework Patching
Site Extensions Gallery
NET, PHP, Python, Node, Java
Framework Installer
Browser-based editing
Auto-Healing
Logging and Auditing

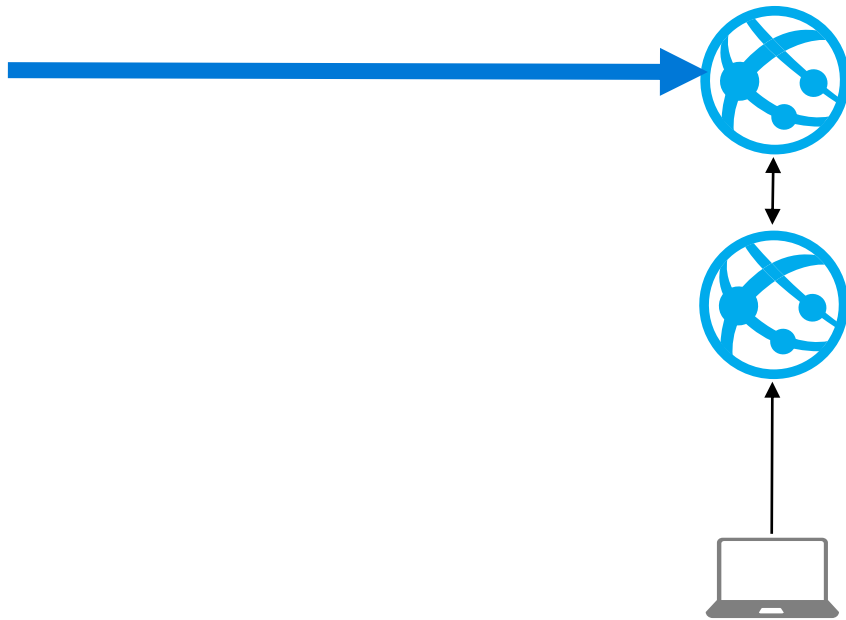
Продакшн и разработка

Используем стейджинги



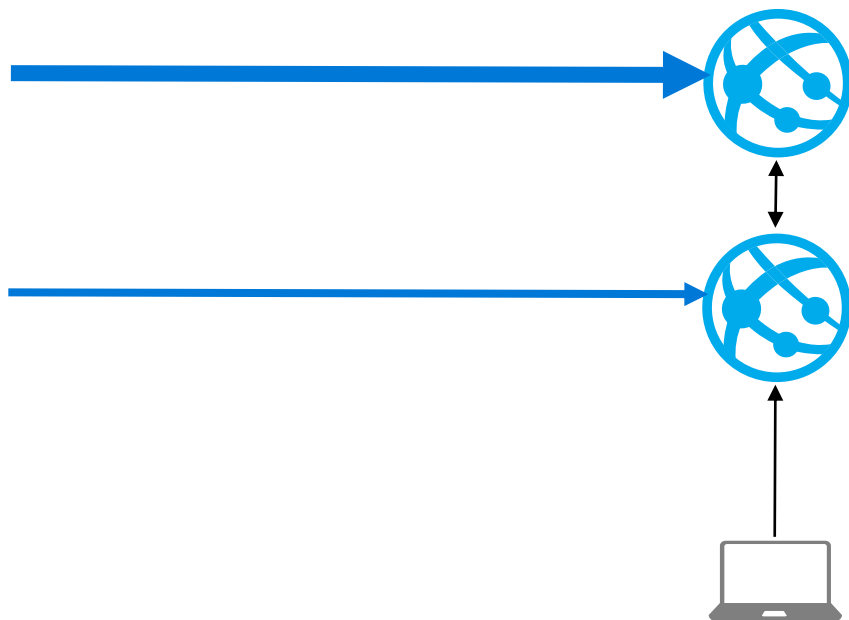
Разработка и выкладывание кода

Классический подход – переключаемся между
новым и старым кодом



Разработка и выкладывание кода

Делим реальный трафик между новым и старым кодом



Отладка кода

Kudu – панель управления (*System Control Management*)

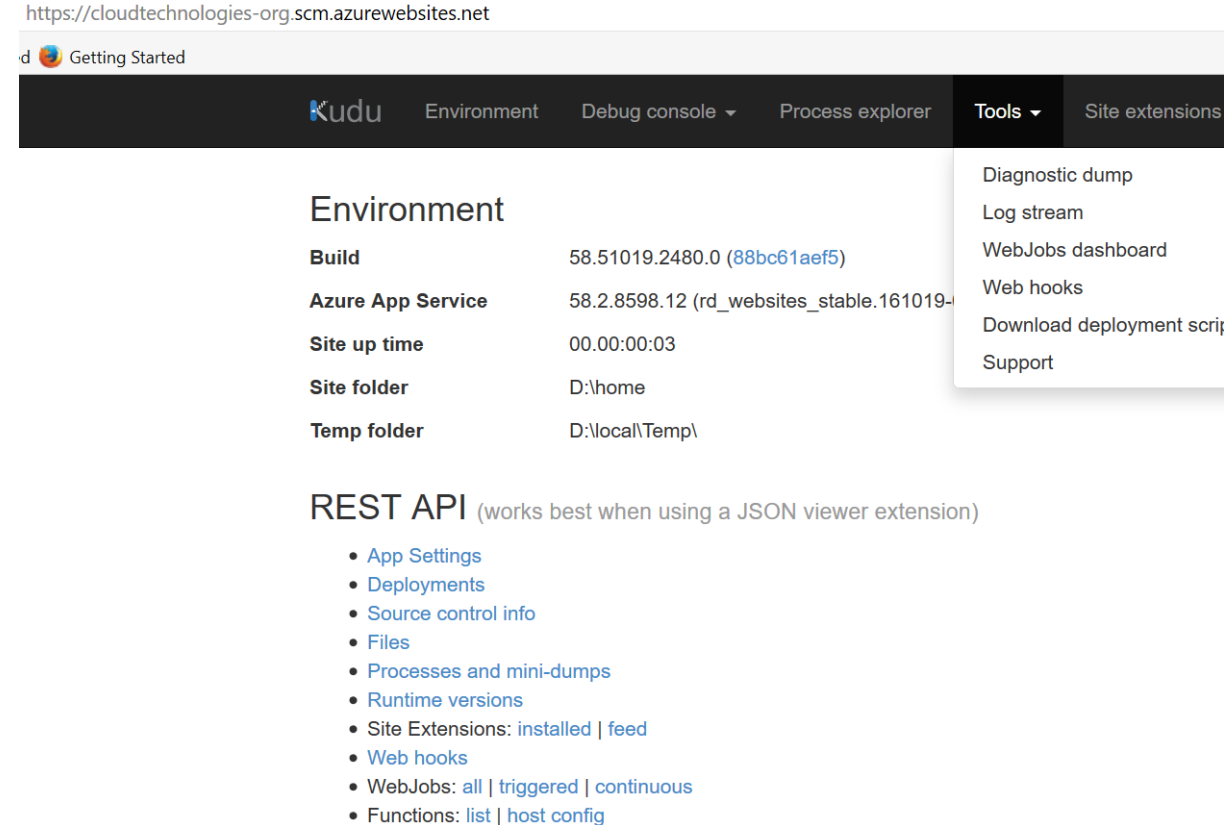
<https://<mySite>.scm.azurewebsites.net/>

Автоматическая авторизация

Выполняется в том же контексте
что и веб сайт

Доступ к файлам, логам и
переменным окружения из
консоли

Отличный инструмент для отладки
и администрирования



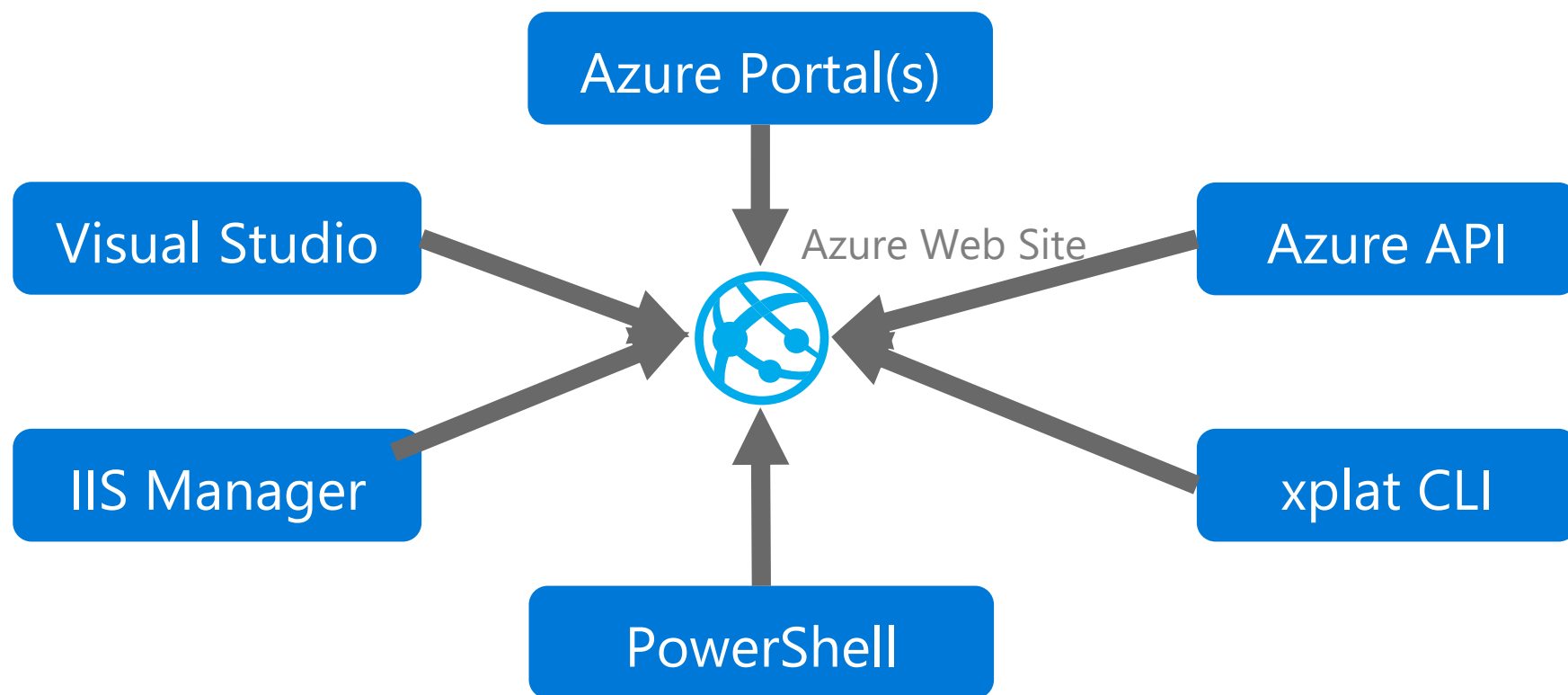
The screenshot shows the Kudu management interface for an Azure Web Site. The URL in the browser is <https://cloudtechnologies-org.scm.azurewebsites.net>. The page has a dark navigation bar with the Kudu logo and links to Environment, Debug console, Process explorer, Tools, and Site extensions. The Tools menu is open, showing options like Diagnostic dump, Log stream, WebJobs dashboard, Web hooks, Download deployment scrip, and Support. The main content area is divided into two sections: Environment and REST API. The Environment section displays a table with site details, and the REST API section provides a list of available endpoints.

Environment	
Build	58.51019.2480.0 (88bc61aef5)
Azure App Service	58.2.8598.12 (rd_websites_stable.161019-)
Site up time	00:00:00:03
Site folder	D:\home
Temp folder	D:\local\Temp\

REST API (works best when using a JSON viewer extension)

- [App Settings](#)
- [Deployments](#)
- [Source control info](#)
- [Files](#)
- [Processes and mini-dumps](#)
- [Runtime versions](#)
- Site Extensions: [installed](#) | [feed](#)
- [Web hooks](#)
- WebJobs: [all](#) | [triggered](#) | [continuous](#)
- Functions: [list](#) | [host config](#)

Управление и администрирование



Создание webjobs

Через ftp

Для triggered job копируем все в путь (внутри вебсайта):

site/wwwroot/app_data/jobs/triggered/{job name}

Для continuous job копируем все в путь (внутри вебсайта):

site/wwwroot/app_data/jobs/continuous/{job name}

Используя SDK REST API

(например, PUT

/api/zip/site/wwwroot/App_Data/jobs/continuous/{job name}/)

Через портал azure.com – загружаем zip файл

NEW JOB

Basic WebJob settings

NAME

mytestjob



CONTENT (ZIP FILES - 100MB MAX) ?



BROWSE FOR FILE...

HOW TO RUN ?

Run continuously

Run on a schedule

Run on demand

Процесс выполнения webjobs

- Архив распаковывается во временную папку (путь к ней в переменной %WEBJOBS_PATH%)
- Сначала выполняется поиск файла по маске run.{file type extension} (например run.cmd или run.exe)
- Если ничего не найдено для всех типов файлов, то тогда выполняется поиск первого файла с поддерживаемым расширением в указанном порядке: .cmd, .bat, .exe, .ps1, .sh, .php, .py, .js
- Поиск выполняется только в корневой папке
- Рекомендованное имя файла для запуска задач = run.cmd

.cmd, .bat, .exe (using windows cmd)

.ps1 (using powershell)

.sh (using bash)

.php (using php)

.py (using python)

.js (using node)

.jar (using java)

Размещение внутри сайта

Файловая система: site\wwwroot\App_Data\jobs\{job type}\{job name}

Снаружи (через POST)

<https://{sitename}.scm.azurewebsites.net/api/webjobs/triggered/{webjobname}/run>

Веб интерфейс: <https://{sitename}.scm.azurewebsites.net/azurejobs/#/jobs>

Данные веб-сайта доступны: d:\home (%HOME%)

При запуске WebJobs копируется во временную папку (см %WEBJOBS_PATH%) и там запускается

Для хранения и обработки данных рекомендуется использовать папку:

%WEBROOT_PATH%\App_Data\jobs\%WEBJOBS_TYPE%\%WEBJOBS_NAME%

или %WEBJOBS_DATA_PATH%

Настройки конфигурирования (GET) : <https://{sitename}.scm.azurewebsites.net/api/webjobs>

Настройки

WEBJOBS_RESTART_TIME – время рестарта в секундах, после завершения предыдущего запуска (независимо от статуса завершения) – только для continuous jobs

WEBJOBS_IDLE_TIMEOUT – время бездействия в секундах, при превышении этого времени задача будет принудительно остановлена. Бездействие определяется по отсутствию нагрузки на CPU и операциям ввода-вывода на консоль/логи. Только для triggered jobs

WEBJOBS_HISTORY_SIZE – размер истории запусков, только для triggered jobs

WEBJOBS_STOPPED – если установлено в 1, то запуск задач запрещен и все задачи (включая уже запущенные) принудительно останавливаются.

Управление остановкой задач

В ряде случаев требуется принудительная остановка задачи и в этом случае существует механизм сообщения задаче о такой остановке перед тем как она будет удалена.

Это может происходить в случаях:

Рестарта/остановки/изменение настроек веб-сайта/web.config

Остановки задачи (через API)

Обновления файлов webjob

Как это работает:

Continuous : если требуется принудительная остановка то создается файл %WEBJOBS_SHUTDOWN_FILE% и задаче дается 5 секунд на остановку. Соответственно внутри задачи должен быть мониторинг наличия этого файла как флага принудительной остановки через 5 секунд

Triggered: по умолчанию 30 секунд ожидания

Управление запуском через settings.job

Этот файл должен находиться в корневой папке там же где и выполняемый файл.

По умолчанию для всех задач (кроме .JS) при каждом запуске выполняется копирование всех файлов в `%TEMP%\jobs\{job type}\{job name}\{random name}` где и происходит запуск задачи.

Вторая опция называется in place – при этом запуск производится в том же месте где выложен код webjobs (`wwwroot/app_data/job,....`) Это устанавливается в settings.job при помощи `{ "is_in_place": true }`

Дополнительно в этом файле можно менять время ожидания принудительно остановки задач

```
{ "stopping_wait_time": 60 }
```

Переменные окружения

Variable name	Description	Example
HOME	Web site root directory	D:\home
WEBJOBS_DATA_PATH	Job data directory	D:\home\data\jobs\type\name.
WEBJOBS_NAME	Job name	
WEBJOBS_PATH	Temporary directory, where job is running	C:\DWASFiles\Sites\~1sitename\Temp\jobs\type\name\lpej4hrk.fks
WEBJOBS_RUN_ID	An unique ID of the job run (an UTC timestamp of the run). There's a data folder created for every run in %WEBJOBS_DATA_PATH%\%WEBJOBS_RUN_ID%	201409090707416329
WEBJOBS_TYPE	Job type	triggered or continuous
WEBROOT_PATH	Web site root directory	D:\home\site\wwwroot
WEBSITE_SITE_NAME	Web site name	

Пример

```
@echo off
```

```
set LOG=%WEBJOBS_DATA_PATH%\%WEBJOBS_RUN_ID%\session.log
```

```
echo WEBJOBS_PATH is %WEBJOBS_PATH%
```

```
echo Running script... >> %LOG%
```

```
date >>%LOG%
```

```
set RESULT=%ERRORLEVEL%
```

```
echo Result code is %RESULT%
```

```
rem Dumping session log to Azure log (standard output) when it exists
```

```
if exist %LOG% (
```

```
    echo Session log:
```

```
    type %LOG%
```

```
)
```

```
rem Propagating exit code to Azure
```

```
exit %RESULT%
```


Azure Web Site: балансировка нагрузки ARR

Балансировка между нодами осуществляется при помощи ARR – Application Request Routing

По умолчанию (однако) включен режим Session Affinity в котором по куки клиент попадает на одну и ту же ноду – ARRAffinity

SETTINGS

- Application settings
- Authentication / Authorization
- Backups
- Custom domains
- SSL certificates
- Networking
- Scale up (App Service plan)
- Scale out (App Service plan)
- Security Scanning



Python version ⓘ Off


Platform ⓘ 32-bit 64-bit

Web sockets ⓘ Off On

Always On ⓘ Off On

Managed Pipeline Version Integrated Classic

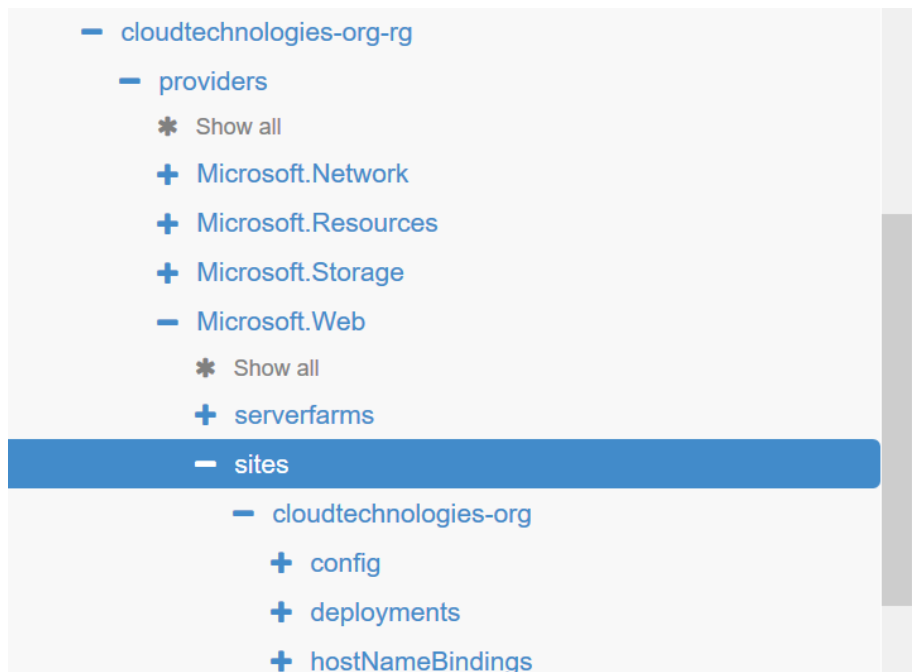
 You can improve the performance of your state-less applications by turning off the Affinity Cookie, state-full applications should keep the Affinity Cookie turned on for increased compatibility. Click to learn more. 

ARR Affinity Off On 



Azure Web Site: балансировка нагрузки ARR

Либо в Azure Resources Explorer (resources.azure.com) :



```
107 "contentAvailabilityState": 0,  
108 "runtimeAvailabilityState": 0,  
109 "siteConfig": null,  
110 "deploymentId": "cloudtechnologies-org",  
111 "trafficManagerHostNames": [  
112   "cloudtechnologies-org.trafficmanager.net"  
113 ],  
114 "sku": "Standard",  
115 "premiumAppDeployed": null,  
116 "scmSiteAlsoStopped": false,  
117 "targetSwapSlot": null,  
118 "hostingEnvironment": null,  
119 "hostingEnvironmentProfile": null,  
120 "microService": "WebSites",  
121 "gatewaySiteName": null,  
122 "clientAffinityEnabled": true,  
123 "clientCertEnabled": false,  
124 "hostNamesDisabled": false,  
125 "domainVerificationIdentifiers": null,  
126 "kind": null
```



Azure Web Site: балансировка нагрузки ARR

Либо в web.config

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <system.webServer>
    <httpProtocol>
      <customHeaders>
        <add name="Arr-Disable-Session-Affinity" value="true" />
      </customHeaders>
    </httpProtocol>
  </system.webServer>
</configuration>
```



Failover – main hosting on-premise

Azure- backup

Azure DNS:

*.cloudtechnologies.org CNAME cloudtechnologies-org.trafficmanager.net



Azure Traffic Manager profile:

DNS layer: cloudtechnologies-org.trafficmanager.net Priority Routing (failover)



Endpoint list:

default (priority 1) - External (on-premise) hosting: infobox.cloudtechnologies.org (IP = 92.243.94.73)

1st backup (priority 2) – Azure Web App: cloudtechnologies-org.azurewebsites.net

2nd backup (priority 3) – Azure Linux Web App: cloudtech-org-linux-php.azurewebsites.net

Application gateway load balancing

Azure DNS:

gateway.cloudtechnologies.org A record = 52.164.244.33



Azure Application Gateway (Web Application Firewall is ON)

URL= 52.164.244.33 (68a951a5-e787-4cd5-a2a3-4f8e1ce2bfc1.cloudapp.net)



Backend pool (appGatewayHttpListener with rule1 (no Cookie affinity, 80)):

InfoBox hosting: infobox.cloudtechnologies.org (DNS A record IP = 92.243.94.73)

Azure Web Site: инструменты разработчика

Fiddler – для отладки HTTP трафика на клиентской Windows машине

Tcpdump – популярный инструмент для linux, умеет дампит UDP трафик

Httpry – тул для логгирования трафика на linux

Digwebinterface.com - правильный способ для проверки DNS записей, умеет использовать разные name-сервера

Все веб-сайты System Control Management Kudu -

<https://<mySite>.scm.azurewebsites.net/>

Для эмуляции простой нагрузки на веб сайт можно использовать **Apache Benchmark** – ab

Важный тул для работы с Azure : **resources.azure.com**

Логи можно загрузить через **Kudu** -

<https://<mySite>.scm.azurewebsites.net/api/zip/LogFiles>

Azure Web Site: полезные тонкости

- Доступ к KUDU можно получить вот так
`https://{userName}:{password}@{siteName}.scm.azurewebsites.net/deploy {userName} и {password}` - параметры из Publisher profile
- Для определения хоста (в случае кластера) полезно смотреть на hostname
- Для того чтобы смотреть логи из Azure Storage можно пользоваться Azure Storage Explorer - storageexplorer.com/
- Всё что есть в Azure – доступно через REST API (то есть можно через cUrl использовать API)
- _Почти_всё_ что есть в REST API есть в SDK – powershell (windows), xplat cli (node.js based) – linux/unix + есть SDK для многих языков программирования
- _Многое_ что есть в SDK доступно в портале.
- Названия параметров в SDK и портале могут отличаться 😊
- В SDK есть 4 environment – Public cloud, Azure China, Azure Government, Azure Stack.
- Через SDK можно посмотреть на _все_ preview/beta через "Get-AzureProviderFeature -ListAvailable"

Практическая часть

Hands-on experience

aka Лабораторная работа

Лабораторная работа 1 - Облачный сайт как бэкап реплика

Лабораторная работа 2 – Облачный сайт как часть гибридного кластера с моим ХОСТИНГОМ

Лабораторная работа 3 – Application Gateway как firewall и балансировщик для моего хостинга

Лабораторная работа 4 – разработка и тестирование (ARR, github и stagings) в вебе

Хаклаба - Для тех кто всё сделал или кому неинтересны другие лабы

Лабораторная работа 1

Облачный сайт как бэкап реплика

- 1) Создаем два разных Azure Web App - %web1% и %web2%
- 2) На каждом из них делаем кастомизацию дефолтной страницы – для того чтобы их отличать друг от друга – через `github` или `ftp`
- 3) Настраиваем Azure Traffic Manager в режиме Priority infobox.cloudtechnologies.org (IP = 92.243.94.73)
- 4) Добавляем %web1% и %web2% как backup реплики
- 5) Проверяем что веб трафик перенаправляется на хостинг в infobox – см логи в infobox. Для доступа = последний слайд в этой деке с доступом для Linux
- 6) Проверяем что посылаются пробы от TrafficManager – их надо найти в логах
- 7) (опционально) Добавляем домен %мой_домен%.cloudtechnologies.org – ask Alexey Bokov и CNAME к нам
- 8) Эмулируем (или делаем на самом деле kill для процесса apache2) отказ веба на основной реплике на infobox. Эмулировать можно путем Disable в профиле End-point в Traffic Manager – запоминаем время t1
- 9) В _портале_ проверяем когда и через какое время подключается backup реплика - запоминаем время t2
- 10) Как только отобразилось в портале – проверяем записи в dig-e, запоминаем время t3
- 11) Ждем когда изменения дойдут до клиентского устройства – проверяем на ноутбуке, мобильном – время t4
- 12) Делаем красивый слайд с t1, t2, t3, t4

Лабораторная работа 2

Облачный сайт как часть гибридного кластера с моим хостингом

- 1) Создаем два разных Azure Web App - %web1% и %web2%
- 2) На каждом из них делаем кастомизацию дефолтной страницы – для того чтобы их отличать друг от друга – через github или ftp
- 3) Настраиваем Azure Traffic Manager в режиме **wieghted** infobox.cloudtechnologies.org (IP = 92.243.94.73)
- 4) Добавляем %web1% и %web2% как backup реплики - _веса_ всех реплик сделать одинаковыми
- 5) Проверяем что веб трафик распределяется по round robin – на клиенте делаем запрос, обнуляем DNS кэш, опять делаем запрос, обнуляем кэш и тп – должны увидеть все три хостинга, запоминаем кол-во попыток N
- 6) Проверяем что посылаются пробы от TrafficManager – их надо найти в логах на InfoBox (см последний слайд) и на web sites
- 7) (опционально) Добавляем домен %мой_домен%.cloudtechnologies.org – ask Alexey Bokov и CNAME к нам
- 8) Дожидаемся когда на клиенте сайт будет ресовится на любой из 3 хостингов, запомнили хостинг – отключаем его (физически или эмулируем). Эмулировать можно путем Disable в профиле End-point в Traffic Manager – запоминаем время t1
- 9) В _портале_ проверяем когда и через какое время отключается плохая реплика - запоминаем время t2
- 10) Как только отобразилось в портале – проверяем записи в dig-e, запоминаем время t3
- 11) Ждем когда изменения дойдут до клиентского устройства – проверяем на лэптопе, мобильном – время t4
- 12) Делаем красивый слайд с N,t1, t2, t3, t4

Лабораторная работа 3

Application Gateway как firewall и балансировщик для моего хостинга

- 1) Создаем **Application Gateway**, включаем **firewall** и диагностику для него, режим = **detection**
- 2) Создаем профиль в **TrafficManager** который будет вести на **Gateway** (тип любой, так конечная точка одна).
- 3) В качестве **backend pool** добавляем infobox.cloudtechnologies.org (IP = 92.243.94.73) либо любой свой хостинг если есть.
- 4) Проверяем что веб трафик доходит до infobox (см последний слайд для доступа) – ищем в логах свои запросы, пробы от Application Gateway и TrafficManager
- 5) (опционально) Добавляем домен %мой_домен%.cloudtechnologies.org – ask Alexey Bokov и CNAME к нам
- 6) Ищем ARR Affinity куки в логах веб сервера - смотрим разницу при запросе через все доступные DNS имена (в теории разницы не должно быть)
- 7) Смотрим ARR на клиенте – Fiddler или httprry
- 8) Выключаем куки, смотри в логах что они пропали.
- 9) Смотрим логи firewall, там пока ничего нет – теперь делаем небольшой DoS при помощи Apache Benchmark, добиваемся того чтобы она отмечал запросы как DoS
- 10) Включаем режим Prevention и добиваемся 403, запоминаем примерно кол-во запросов
- 11) Подключаем SSL offload (опционально)

Лабораторная работа 4

Разработка и тестирование (ARR, github и stagings) в вебе

- 1) Создаем Azure Web Site (linux | windows – попробовать на оба варианта)
- 2) Делаем профиль на traffic manager ссылающийся на вебсайт (и/или) (опционально) Добавляем домен %мой_домен%.cloudtechnologies.org – ask Alexey Bokov и CNAME на профиль
- 3) Создаем слот для тестирования и разработки
- 4) Делаем репозиторий на github и подключаем его к этому слоту и проверяем что слот обновляется
- 5) Переключаем слот в продакшн и смотрим что продакшн обновился – должно быть мгновенное обновление
- 6) Делаем правку для слота и комитим в него
- 7) Распределяем трафик между двумя стейджингами, смотрим в логах через Kudu куда приходят запросы
- 8) Возвращаем в режим когда весь трафик идет на один стейдж, делаем scale up на два инстанса
- 9) Открываем в Kudu консоль на обоих инстансах – смотрим hostname (что разные)
- 10) Делаем бранк в github для продакшна, делаем там изменение, смотрим через Kudu что апдейт произошел на обоих нодах.
- 11) Добавляем новый web app в другом регионе и делаем там web job для мониторинга нашего первого web app – если сайт не откликается либо get запрос выдает пустой html – делаем нотификацию

Хаклаба

- 1) Прикрутить Application Gateway с backend pool настроенным на 2 ноды – одна нода это infobox, другая нода это Azure Web Sites (можно Linux Web Site)
- 2) Настраиваем DNS (ask Alexey Bokov for domain name) -> Traffic Manager -> Application Gateway
- 3) Смотрим на стороне сервера как работает получение запросов – на apache2/infobox и IIS на web sites, смотрим логи и самое главное заголовки http запросов – обращаем внимание на host и на ответ от веб сервера. Интересно найти пробы от Traffic Manager и Application Gateway
- 4) Делаем еще хак - _перед_ Application Gateway включаем Azure Load Balancer (и перенастраиваем соотв. Traffic manager на ALB)
- 5) Опять смотрим логи – там много интересного 😊
- 6) Эмулируем DoS – apache benchmark на самое верхнее DNS имя, ожидаем 403 – смотри что отразилось в логах Firewall (а не ALB ли там был забанен по IP ? 😊
- 7) Делаем красивый слайд с выводами по лабе

Linux

*Доступ к тестовому серверу
expired right after lab is finished (31 Oct 2016)*

Username=testuser_1

Password=OOO#VS15D9

IP = 92.243.94.73

DNS= infobox.cloudtechnologies.org

Логи apache2 находятся в папке
/var/log/apache2/



Q&A

Веб приложения в облаке: проектирование и отладка

Alexey Bokov

Senior Program Manager, Microsoft Corp

abokov@microsoft.com; twitter: @abokov

#msdevcon

Помогите нам стать лучше!

На вашу почту отправлена индивидуальная ссылка на электронную анкету. 2 ноября в 23:30 незаполненная анкета превратится в тыкву.

Заполните анкету и подходите к стойке регистрации за приятным сюрпризом!

#msdevcon

Оставляйте отзывы в социальных сетях. Мы все читаем. Спасибо вам! 😊

