СибГУТИ

СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ

Инструменты Эксплуатации

План занятия

Теория

- Операционные системы
- Мониторинг
- Средства Диагностики
- Bash и другие скриптовые языки программирования

Практика

1) Установить СУБД (postgres или mysql(mariadb))
2) Установить веб сервер (apache или nginx, php-fpm,php)
3) Сделать Инвентаризацию в доме и погрузить данные в Субд + парсим результат в веб

Что это вам даст

- Понимание инструментов и средств Эксплуатации
- Узнать о средствах диагностики
- Как использовать скриптовые языки программирования

Операционные системы

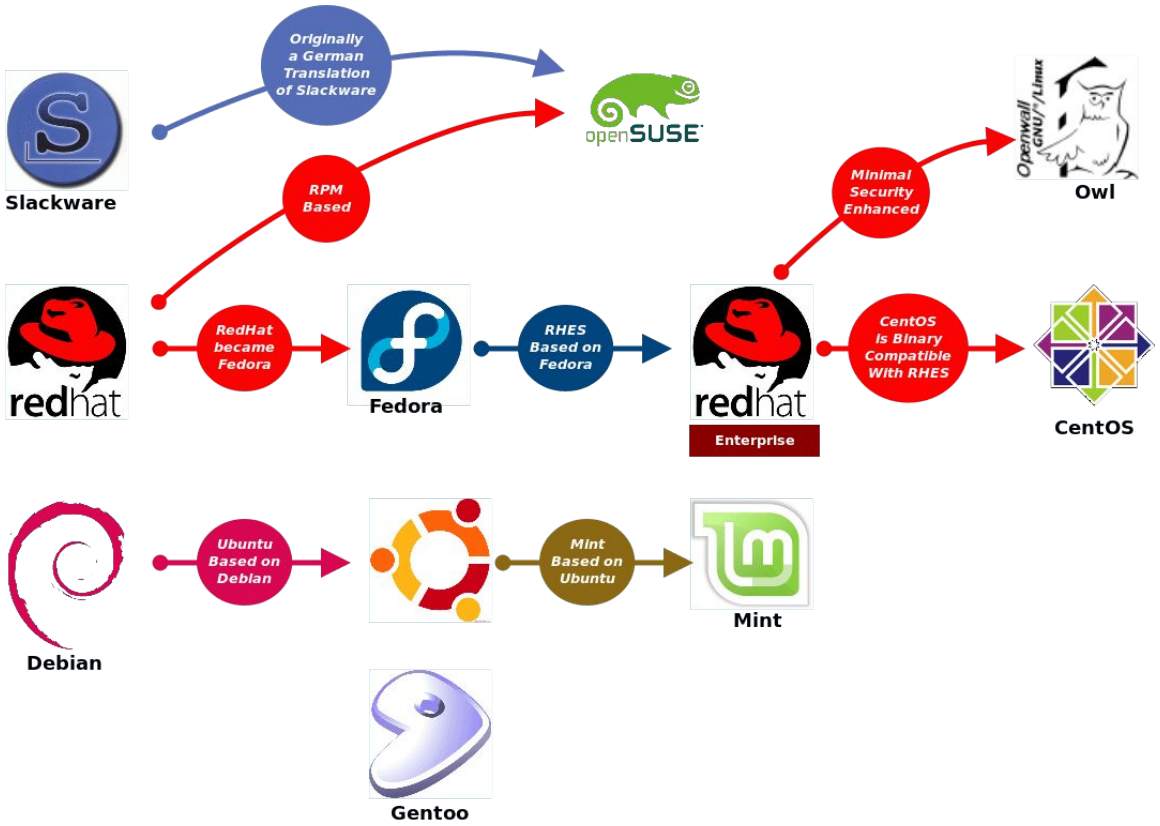
Microsoft Windows Server System

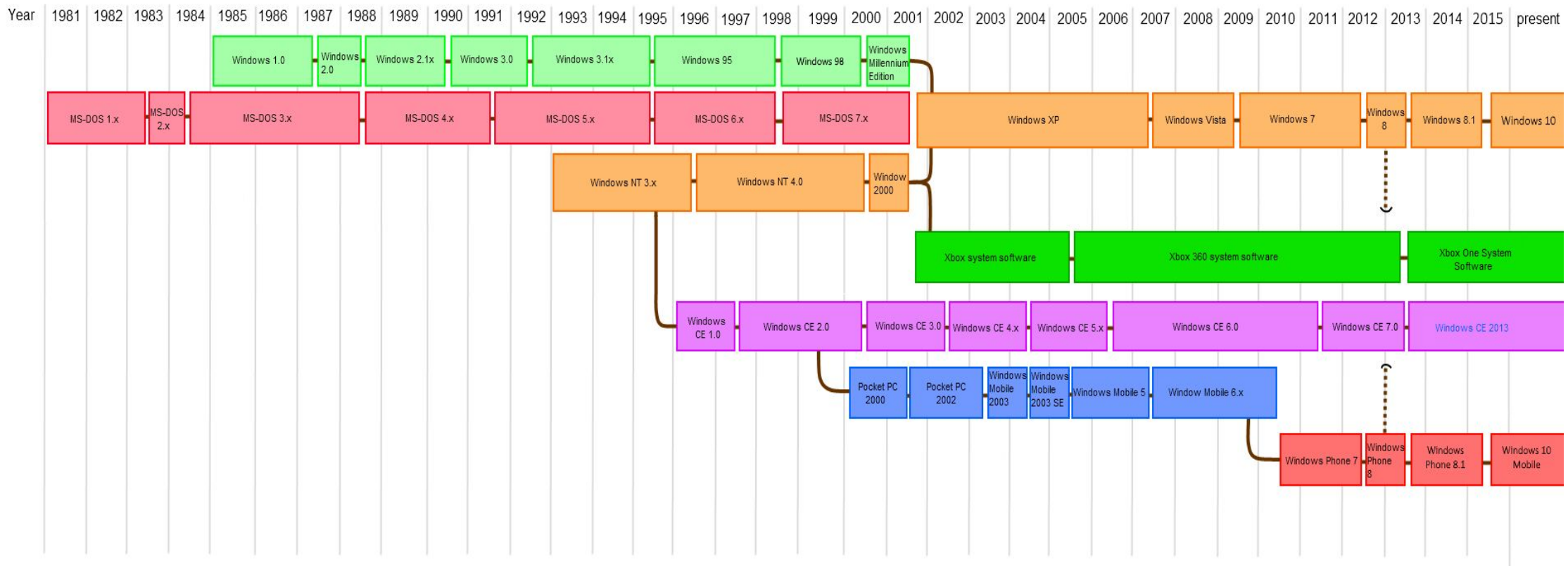
Mac OS X

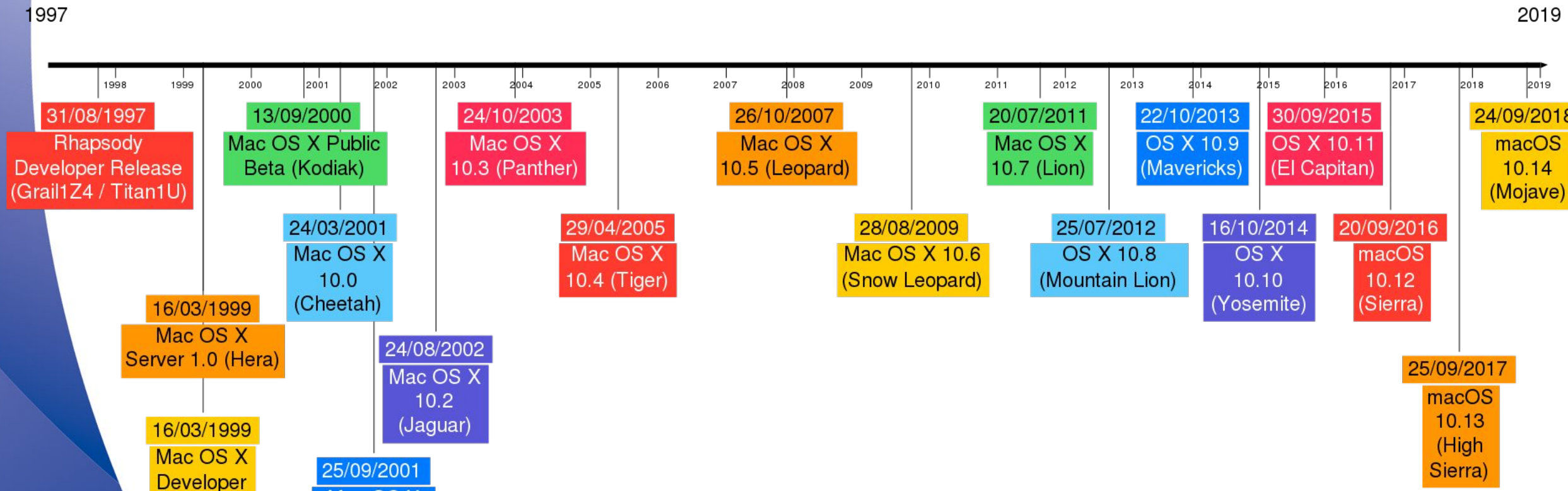
android

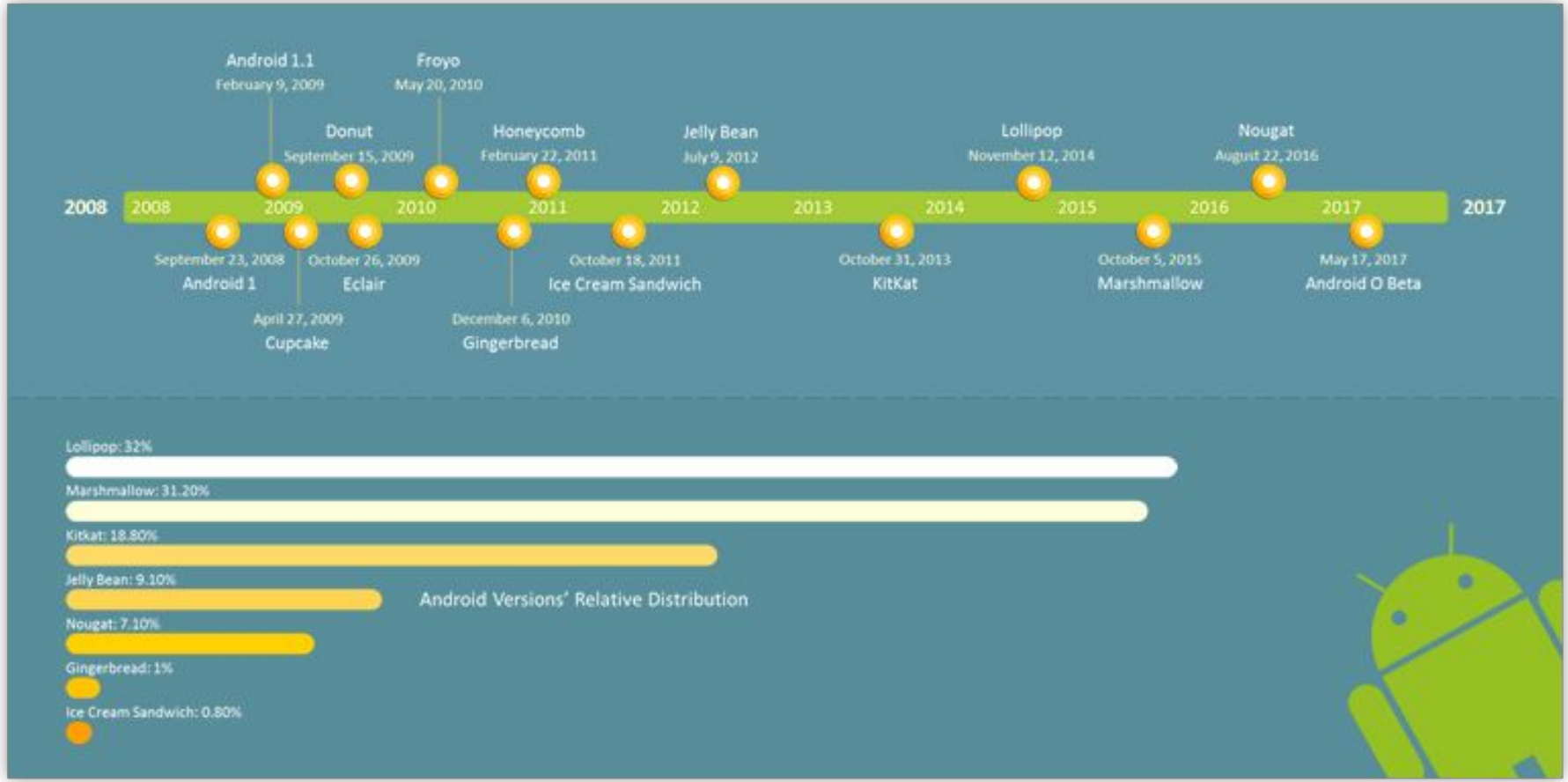
Дистрибутивы Linux

Как развивалось и множилось




Year
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
present
Windows 1.0
Windows 2.0
Windows 2.1x
Windows 3.0
Windows 3.1x
Windows 95
Windows 98
Windows Millennium Edition
MS-DOS 1.x
MS-DOS 2.x
MS-DOS 3.x
MS-DOS 4.x
MS-DOS 5.x
MS-DOS 6.x
MS-DOS 7.x
Windows XP
Windows Vista
Windows 7
Windows 8
Windows 8.1
Windows 10
Windows NT 3.x
Windows NT 4.0
Window 2000
Xbox system software
Xbox 360 system software
Xbox One System Software
Windows CE 1.0
Windows CE 2.0
Windows CE 3.0
Windows CE 4.x
Windows CE 5.x
Windows CE 6.0
Windows CE 7.0
Windows CE 2013
Pocket PC 2000
Pocket PC 2002
Windows Mobile 2003
Windows Mobile 2003 SE
Windows Mobile 5
Window Mobile 6.x
Windows Phone 7
Windows Phone 8
Windows Phone 8.1
Windows 10 Mobile

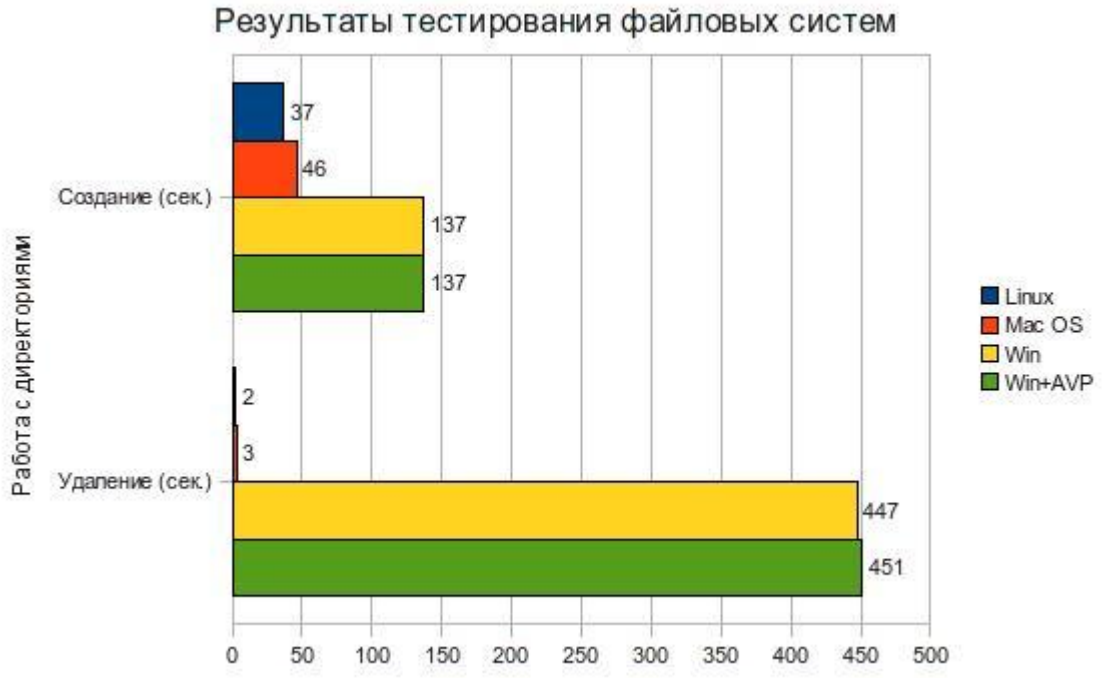

1997
2019
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
31/08/1997
Rhapsody Developer Release (Grail1Z4 / Titan1U)
16/03/1999
Mac OS X Server 1.0 (Hera)
16/03/1999
Mac OS X Developer Preview
13/09/2000
Mac OS X Public Beta (Kodiak)
24/03/2001
Mac OS X 10.0 (Cheetah)
25/09/2001
Mac OS X 10.1 (Puma)
24/08/2002
Mac OS X 10.2 (Jaguar)
24/10/2003
Mac OS X 10.3 (Panther)
29/04/2005
Mac OS X 10.4 (Tiger)
26/10/2007
Mac OS X 10.5 (Leopard)
28/08/2009
Mac OS X 10.6 (Snow Leopard)
20/07/2011
Mac OS X 10.7 (Lion)
25/07/2012
OS X 10.8 (Mountain Lion)
22/10/2013
OS X 10.9 (Mavericks)
16/10/2014
OS X 10.10 (Yosemite)
30/09/2015
OS X 10.11 (El Capitan)
20/09/2016
macOS 10.12 (Sierra)
25/09/2017
macOS 10.13 (High Sierra)
24/09/2018
macOS 10.14 (Mojave)


Android 1.1
February 9, 2009
Froyo
May 20, 2010
Donut
September 15, 2009
Honeycomb
February 22, 2011
Jelly Bean
July 9, 2012
Lollipop
November 12, 2014
Nougat
August 22, 2016
2008
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2017
September 23, 2008
Android 1
October 26, 2009
Eclair
April 27, 2009
Cupcake
December 6, 2010
Gingerbread
October 18, 2011
Ice Cream Sandwich
October 31, 2013
KitKat
October 5, 2015
Marshmallow
May 17, 2017
Android O Beta
Lollipop: 32%
Marshmallow: 31.20%
Kitkat: 18.80%
Jelly Bean: 9.10%
Nougat: 7.10%
Gingerbread: 1%
Ice Cream Sandwich: 0.80%
Android Versions' Relative Distribution

Сравнение производительности ОС

1) Создание и удаление директории
 - Во время теста создётся большое количество вложенных директорий, а затем удаляется.
Операции проводятся встроенным в ОС средствами: mkdir/rmdir/rm

	Linux	Win	Win+AVP	MacOS
Создание (сек.)	37	137	137	46
Удаление (сек.)	2	447	451	3



2) Синтетический тест, с использованием MYSQL

Тесты проводились по 3 раза без заметной разницы, количество созданных в mysql записей равно 237201, на компьютерах

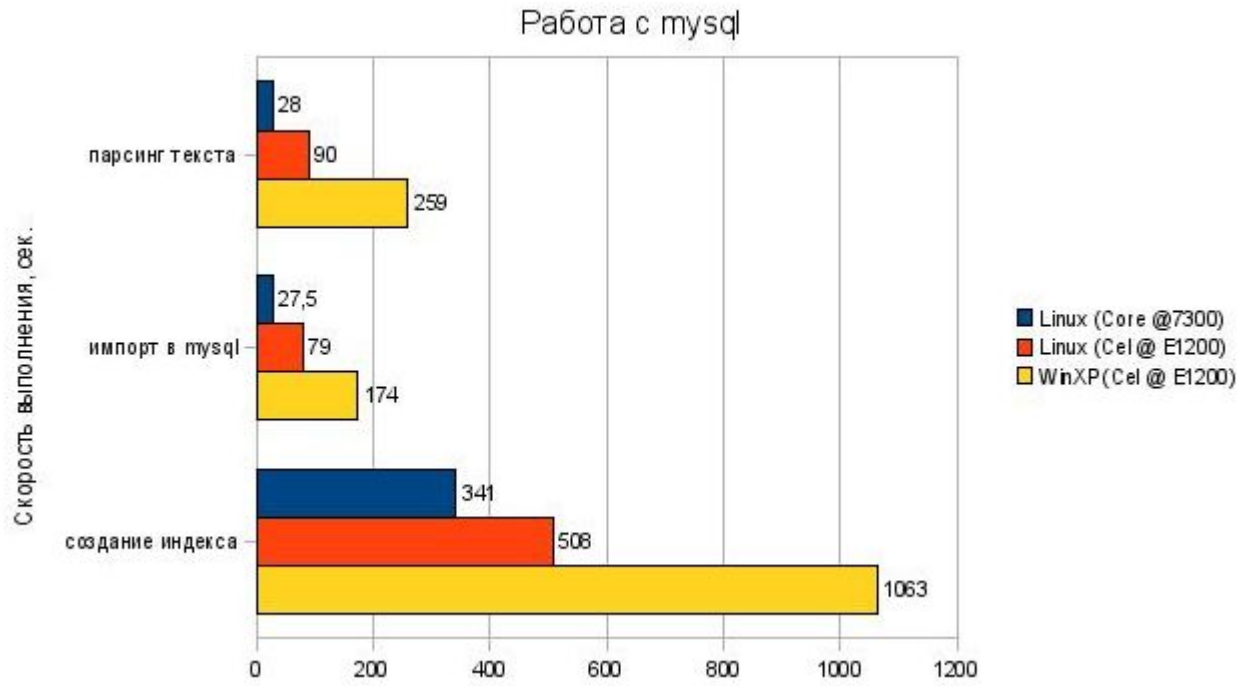
Debian Intel(R) Core(TM)2 Duo CPU E7300 @ 2.66GHz
Ubuntu Intel(R) Celeron(R) CPU E1200 @ 1.60GHz
WinXP Home Intel(R) Celeron(R) CPU E1200 @ 1.60GHz

Для проведения теста необходимы mysql сервер и клиент, интерпретатор python

Суть теста:

распарсить текстовый файл с данными о файловой системе на внешнем ftp сервере, создать 3 таблицы: файлов, путей и расширений, сгенерировать sql файл для таблицы файлов
вставить все данные в mysql
создать индекс

	Linux (Duo CPU E7300)	Linux (Cel @ E1200)	Windows (Cel @ E1200)
парсинг txt (сек)	28	90	259
import mysql (сек)	27,5	79	174,5
создание индекса (сек)	341	509	1063


Работа с mysql
Скорость выполнения, сек.
парсинг текста
импорт в mysql
создание индекса
28
90
259
27,5
79
174
341
508
1063
0
200
400
600
800
1000
1200
Linux (Core @7300)
Linux (Cel @ E1200)
WinXP (Cel @ E1200)

Мониторинг

ZABBIX

Nagios®

New Relic.

MUNIN

RRDtool
logging & graphing

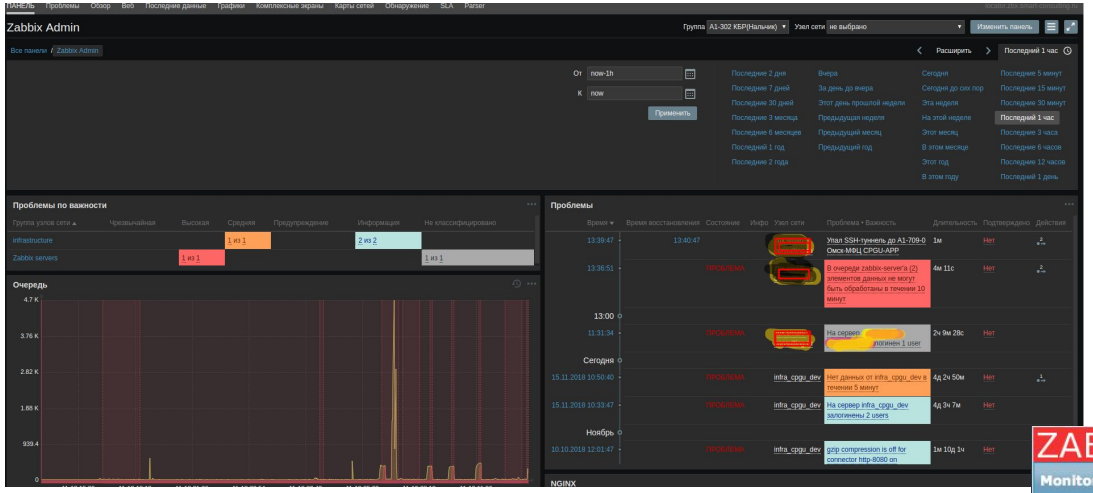
TOOLS
pingdom

elasticsearch + logstash + kibana

ЗАЧЕМ ВСЁ ЭТО НУЖНО???

Раньше:
Мониторились, в основном, системные показатели: CPU, память, диски, сеть. Этого вполне хватало, потому что там крутилось одно приложение на php, и ничего больше не использовалось. Проблема в том, что по таким показателям обычно мало что можно сказать. Либо работает, либо нет. Что именно происходит с самим приложением, выше уровня системных показателей понять сложно.
Если проблема была на уровне приложения (не просто “сайт не работает”, а “сайт работает, но что-то не так”), то клиент сам писал или звонил, сообщал, что есть такая-то проблема, мы шли и разбирались, потому что сами мы такие проблемы заметить не могли.
Сейчас:
Надо мониторить не только дискретное “работает/не работает”, а гораздо больше градаций. Что, в свою очередь, позволяет ловить проблему до того, как все рухнет.
Кроме того, теперь надо следить и за бизнес-показателями.

Почему?
Усложнение самих систем, конечно, повлекло за собой большее количество возможных проблем. Появились метрики приложений, количество запущенных тредов у Java application, частота garbage collector pauses, количество событий в очереди. Очень важно, чтобы мониторинг также следил за масштабированием систем. Допустим, у вас Kubernetes HPA. Надо понимать, сколько запущено подов, и с каждого запущенного пода должны идти метрики в систему мониторинга приложения, в apm.



ZABBIX

Help | Get support | Print | Profile | Logout

Monitoring | Inventory | Reports | Configuration | Administration

Dashboard | Overview | Web | Latest data | Triggers | Events | Graphs | Screens | Maps | Discovery | IT services

Search

History: トリガーの設定 » ダッシュボード » ユーザープロファイル » Dashboard » Overview

PERSONAL DASHBOARD

Favorite graphs

- vSphere 001: CPU utilization
- vSphere 002: CPU utilization
- vSphere 003: CPU utilization

Graphs »

Favorite screens

- Zabbix server performance
- JBoss performance
- Oracle RAC
- Network map

Screens »

Favorite maps

- Network devices
- VMWare production

Maps »

Status of Zabbix

Parameter	Value	Details
Zabbix server is running	Yes	localhost:10051
Number of hosts (monitored/not monitored/templates)	85	47 / 0 / 38
Number of items (monitored/disabled/not supported)	502	493 / 0 / 9
Number of triggers (enabled/disabled) [problem/ok]	291	291 / 0 [10 / 281]
Number of users (online)	2	1
Required server performance, new values per second	7.7	-

Updated: 02:41:40 AM

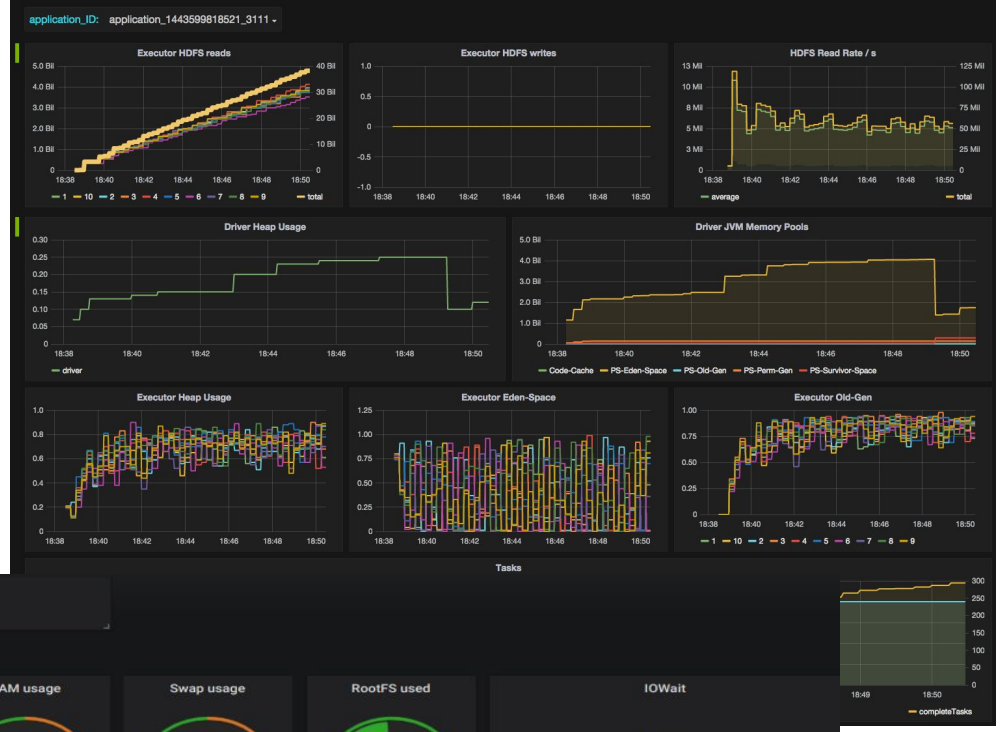
System status

Host group	Disaster	High	Average	Warning	Information	Not classified
Business System	0	0	0	0	0	0
Clouds	0	0	0	0	0	0
Database servers	0	0	0	0	0	0
JBoss instances	0	0	0	3	0	0
Network Devices	0	0	0	0	0	0
Private Cloud	0	0	0	5	0	0
Web servers	0	0	0	0	0	0
Zabbix servers	0	0	0	2	0	0

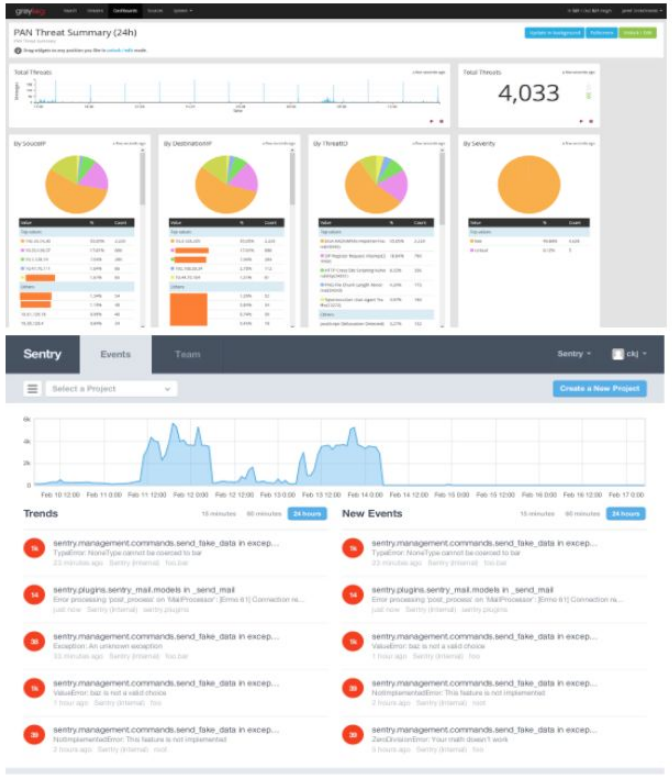
Updated: 02:41:41 AM

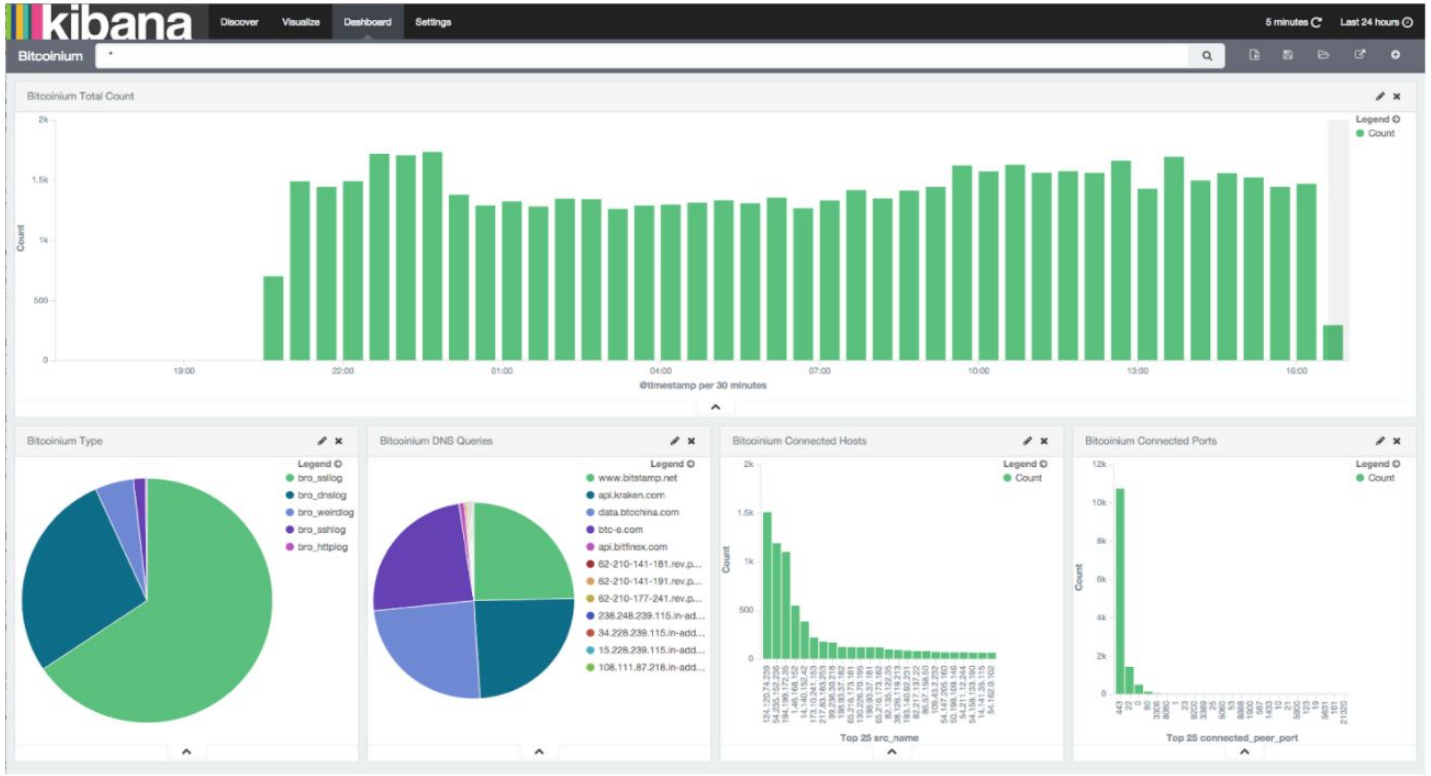
Host status

Host group	Without problems	With problems	Total
Business System	17	0	17
Clouds	2	0	2
Database servers	2	0	2
JBoss instances	0	3	3


application_ID: application_1443599818521_3111
Executor HDFS reads
Executor HDFS writes
HDFS Read Rate / s
Driver Heap Usage
Driver JVM Memory Pools
Executor Heap Usage
Executor Eden-Space
Executor Old-Gen
Tasks


21.0 GB
12
2.147 GB
16.9 GB
211 GB
Quick overview
Uptime
3.877 week
LA medium
0.9
Zombies
0
Processes
282
Threads
325
CPU usage
8%
RAM usage
6%
Swap usage
7%
RootFS used
48%
IOWait
0.2%
CPU
CPU usage
MFC_Nso_DB: user
MFC_Nso_DB: system
MFC_Nso_DB: iowait
MFC_Nso_DB: softirq
Load averages
MFC_Nso_DB: short
MFC_Nso_DB: medium
MFC_Nso_DB: long
Memory usage
Processes


PAN Threat Summary (24h)
4,033
Sentry
Events
Team
Select a Project
Create a New Project
Trends
New Events


kibana
Discover
Visualize
Dashboard
Settings
Bitcoinium
Bitcoinium Total Count
Count
@timestamp per 30 minutes
Bitcoinium Type
bro_sslog
bro_dnslog
bro_weirdlog
bro_sshlog
bro_httplog
Bitcoinium DNS Queries
www.bitstamp.net
api.kraken.com
data.btcchina.com
btc-e.com
api.bitfinex.com
Bitcoinium Connected Hosts
Top 25 src_name
Bitcoinium Connected Ports
Top 25 connected_peer_port

Средства Диагностики

Материалы:
https://habr.com/company/ua-hosting/blog/281519/

#!Bash

https://linuxconfig.org/bash-scripting-tutorial

Пример

```
#!/bin/bash

echo -e "Hi, please type the word: \c "
read  word
echo "The word you entered is: $word"
echo -e "Can you please enter two words? "
read word1 word2
echo "Here is your input: \"$word1\" \"$word2\""
echo -e "How do you feel about bash scripting? "
# read command now stores a reply into the default build-in variable $REPLY
read
echo "You said $REPLY, I'm glad to hear that! "
echo -e "What are your favorite colours ? "
# -a makes read command to read into an array
read -a colours
echo "My favorite colours are also ${colours[0]}, ${colours[1]} and ${colours[2]}:-)"
```

#!/usr/bin/python

```
#!/usr/bin/python3

import os
import subprocess
import paramiko
import sys
import re
import time
#Подключение по ssh
host = "point"
port = 22
user='root'
secret= 'neskazhupassword'
#Команда для выполнения
monitoring = "curl   localhost:8080/rest/monitoring/ -s | sed -e 's/<br>//g' | sed -e 's/<\/b>//g' | sed -e 's/<br\/>/#/g' | sed -e 's/<b>//g'| sed -e 's/<html><body>//g' |  sed -e 's/<\/body><\/html>//g' | sed 's/ /_/g
#Подключение
client = paramiko.SSHClient()
client.set_missing_host_key_policy(paramiko.AutoAddPolicy())
client.connect(hostname=host, username=user, password=secret, port=port)


#Выполнение команды на удалённом сервере
stdin, stdout, stderr= client.exec_command(monitoring)
#Вывод команды
data = stdout.read() + stderr.read()
# Преобразование русского текста
data2 = data.decode('utf8')
#print (data2)
# Закрытие соединения
data3=re.findall(r'[^# ]+', data2)
client.close()

#Список из проверок встречающихся в выводе мониторинга
check_code = ['FAIL','WARN']
#Проверка совпадений в строке
result = [s for s in data3 if  any(xs in s for xs in check_code ) ]
print (result)
```