End-user Experience Monitoring with ZABBIX
by Jurijs Fedorenko

September 6, 2013
Who am I and
What I do
Senior Systems Administrator at ABLV Bank, AS

• Riga Technical University graduate [ IT Engineering / Programmer ]
• Four years as DevOp at AS Latvenergo Energy Company
• Currently DevOp at ABLV Bank, AS
• ZABBIX Certified Specialist
• Virtualization systems
• Monitoring systems
ABLV Bank, AS

- Currently, ABLV Bank, AS is the largest independent private bank in Latvia.
- Founded on 17 September 1993.
- Representative offices of the Group in many CIS countries.
- ABLV Group includes ABLV Bank, AS and more than 20 subsidiary companies.
- 20 years of valuable experience.
What to monitor in Bank?

- Credit card systems
- Banking systems
- e-Banking systems
- Third party payment gateways
- Inter-branch links
- Clearing systems
- Client line / Call centre
- Databases
- Data warehouse
- Security systems
Our monitoring infrastructure - in numbers and facts

- Number of hosts: 1150
- Number of items: 38973
- Number of triggers: 14014
- Number of users: 106
- Values per second: 578.81

- Test & Development environment
- ORABBIX + custom DB scripts
- Production ZABBIX server
- ZABBIX proxy
- SNMPTT

- Integration with service/project management system
- Automatic incident registering
- Automatic work order registering
- Reports for Enterprise Resource Planning

**ZABBIX** is taking part in resource management!
Our monitoring infrastructure - simplified
Vision
3 levels of monitoring

- Hardware
- Applications
- End-user experience
3 levels of monitoring

1. Infrastructure level monitoring.
   • Server health
   • Network devices
   • Peripheral devices

2. Application level monitoring.
   • Log files
   • App. SNMP traps
   • Running processes

3. End-user monitoring.
   • ?
Problem

- Servers are up & running
- Applications are started
- Network – connected
- Users have access rights
- Service unavailable

- eg. we’ve ran out of licenses.
What is End-user monitoring?

- Broker system
- Credit cards
- Mobile devices
- e-Banking
- SMS messaging
- DBMS
- Security systems
Unique person

• Loves routine work
• Stress-resistant
• IT knowledge
• Low salary
• Insomnia
• Punctual
• Loyal
End-user monitoring must be a robot
System must be able to

- Run all the applications, that our users and clients are using
- Run applications from user’s perspective
- Run on the same OS, platform, browser etc.
- Time every step, while running apps.
- Handle errors and non-standard situations
- Collect useful information about occurred error
- Send collected data to ZABBIX
- Must be easy to implement
- Must be easy to maintain
- Must be quick to learn
- Notify administrators
Failed to find

- Expensive
- Complicated (learning & implementing)
- Overlaps functionality with ZABBIX
- Lacks integration capability
- Not customisable – just a built-in functionality
- Third party unable to deliver
AutoIt key features

- Easy to learn BASIC-like syntax
- Simulate keystrokes and mouse movements
- Manipulate windows and processes
- Interact with all standard windows controls
- Create Graphical User Interfaces
- ~400 built-in functions
- Scripts can be compiled into standalone executables
- Obfuscation & Encryption
- Free of charge

- «Polished» and dependable product – first release in 1999
- COM support
- Regular expressions
- Directly call external DLL and Windows API functions
- Scriptable RunAs functions
- Well documented
- Large community-based support forums
- Digitally signed for peace of mind
AutoIt example

AutoItSetOption("PixelCoordMode", 2)
AutoItSetOption("MouseCoordMode", 0)

; Start timer
$startTime = TimerInit()

Run("mspaint.exe")

WinWaitActive("untitled - Paint")
AutoItSetOption("SendKeyDown", 100)

Sleep(4000)
MouseClickDrag ( "lcf", 350, 400, 400, 300 )
MouseClickDrag ( "left", 400, 300, 450, 400 )
MouseClickDrag ( "left", 375, 350, 425, 350 )

Sleep(4000)

; Now quit by sending a "close" request to the paint
WinClose("untitled - Paint")

Sleep(500)
Send("n")

WinWaitClose("untitled - Paint")

; Stop timer
$deltaTime = TimerDiff( $startTime )
$deltaTime = $deltaTime - 2500

; Send data to Zabbix
Run("C:\zabbix\zabbix_sender.exe -z zbx_srv -p 10051 -s autoit -k autoit_test -o " & $deltaTime")
Sleep(500)

; Finished!
End-user monitoring «pilot»

Diagram:
- Test application
  - Autotest script
  - ZABBIX sender
    - ZABBIX server
- Windows scheduler
- Log file
Pros

• Easy to script
• Has useful tools
• Good at «timing»
• Easy to integrate with ZABBIX
• Autonomous – no DB, no conf.
• Can detect non-standard situations

Cons

• Not able to notify
• No error handling mechanism
• Not enough debug information
• Unable to run multiple scenarios concurrently
Wrapper. Main duties

- Handle the list of executables and AutoIt scripts
- Set executing frequency
- Run scripts exclusively
- Run as Win service
- Debug information
- Scheduling
- Log file
End-user monitoring with wrapper

Applications to run

Wrapper parent

Wrapper config.

Error handling scripts

Shared space

Log file

ZABBIX sender

AutoIt script

Administrators and Help-desk

ZABBIX server
Wrapper. Key features

- Executing scripts exclusively
- Scheduling (1-7, 00:00-23:59)
- Multiple scheduled periods for each script
- Set frequency for each period (e.g., in weekdays run tests more frequently, than in weekends)
- Execute any command – not only AutoIt scripts (e.g., sending 0 while idling)
- Set number of retries, before item becoming «unsupported»
- Set separate frequency for «unsupported» items

```plaintext
timeout=600
frequency=180
retryPenalty=60
unreachableTimeout=120
retryPenalty=60
unreachableTimeout=3600
#
# Comment lines must start with "#" - no leading spaces allowed!
# UserParameter=hostname;metric:C:\Path\to\executable;metricFrequency;timeout;retries;schedule([1-7]-[1-7],hh:mm-hh:mm)
# UserParameter=hostname;metric:C:\Program Files\AutoIt3\AutoIt3.exe C:\Scripts\test.au3;600;30;2;1-7,10:31-12:00
```
Wrapper flowchart

Begin

Load configuration file

Get current time

Is wrapper paused

Sleep 5 sec.

Y

Other script already running

N

Go to next script in queue

N

Script scheduled to run now

Y

“Frequency” amount of time passed since previous run

N

Set next time to run

Create lock file

Fork

child

Run script

“unreachable”

Y

Reset “unreachable” counter

Reset “frequency”

N

End

Sleep 1 sec.

N

Timeout reached

Y

Save screenshot

Kill “child” process

Close application

Send error message to ZABBIX

Remove lock file

Increase “unreachable” counter

Reached “unreachable” threshold

Y

Set “frequency” to “unreachable freq.”
Tux has dogtail

- Dogtail is a GUI test tool
- Object oriented & procedural APIs
- Uses AT-SPI framework (supports GNOME & GTK+ applications)
- Written in Python (if you can do it in Python, you can do it in dogtail)
- Sniff – graphical representation of GUI elements in hierarchy
- Dogtail recorder – script recording and playback mechanism
- Alternative – LDTP (WinLDTP)
Gain
Example 1 – VM migration

One morning we discovered, that there is «random» performance improvement in almost every system. It looked like this:

![Graph showing performance improvement](image)

Few seconds – not worth mentioning?
5sec * 20app * 5x/d * 500emp * 250d/y * 5y =~10 years
Example 2 – from spy movies

Log in the surveillance system.
Move through the list of cameras.
Check, if date/time has changed.

* Example created with private IP camera, with no relation to surveillance system of bank.
**Camera pic. used from site www.liepaja.lv.
Thank you!

Question session!*

*aka Q&A

P.S. Work hard, be kind, and amazing things will happen! /Conan O'Brien/