Measurement laboratory 4. (vimia315) 2. measurement

# Measurement guide and report template

**Investigating the application layer**

# V06a (2016. February 17.)

1. Modification to tasks and template: HUSZERL Gábor, BME MIT, 2012-2014.
2. English translation: NASZÁLY Gábor, BME MIT, 2014.
3. Modification to tasks and template: NASZÁLY Gábor, BME MIT, 2016.

|  |  |
| --- | --- |
| Location: | **Room: I.B141, place number: <xy>** |
| Date: | **2016. ???. ??.** |
| Students: | **<name> <course> <group>**  **<name> <course> <group>** |
| Name of this report: | **ml4\_ml4a<group>\_2.doc** |
| Lecturer: | **???** |

**Fill in only the parts marked with bright yellow!**

# Rating of this measurement

Your preparation will be tested before the measurement by asking five simple questions. It is necessary to give correct answers to at least two questions to pass the test. Without at least two right answers the measurement cannot be rated and needed to be repeated.

The measurement report is either accepted or asked to be extended based upon the formal and theoretical quality of the document (and also by asking questions if needed). If the report is acceptable a mark for the measurement is given based upon the result of the test.

There are **beginner level** and **advanced level** (marked by +) tasks. For the exam at the end of the semester a mark good (4) can be granted if the student is able to do the basic tasks. For the mark excellent (5) the advanced level tasks are also required to complete!

# 1. Dynamic TCP/IP configuration of network nodes

**Aim of this task:** getting familiar with the DHCP protocol. Take care to observe the **very first** boot of the operating systems in! If you missed the first system start ask help from the lecturer! (It is worth to observe not only the first but the succeeding boot processes too but it is not required during these tasks.)

**Preparation:** copy the two linked virtual machine folders (\*\_linked\_clone) located as subfolders in V:\ML4\_M2\_application\_level\_networking to drive D:! In the case drive D: already contains these folders delete them before copying unless the lecturer asks you to leave the folders in peace (in case the folders were pre-copied to drive D: by the lecturer).

## 1.1 Dynamic configuration of Ubuntu 12.10

|  |
| --- |
| 1. Start capturing network traffic in Wireshark! 2. Assure that the network type of the virtual machine is set to “bridged”! 3. Start the virtual machine (answer with “copied it” if asked)! 4. After the login prompt appears log in to the system (*user/user*)! Figure out the MAC address of the virtual machine by using the command line tool *ifconfig* inside a terminal! |
| <<MAC address>> |
| 1. Shut down the system (either by issuing sudo poweroff in a terminal or using the GUI)! After quitting from VMware Player stop capturing network traffic! 2. Construct a *display filter* to show only network traffic belonging to the virtual machine! |
| <<display filter>> |
| 1. Analyze the traffic! |
| <<screen shot>> |
| 1. What protocols are used by Ubuntu Linux (how and why)? |
| * protocol: description |

## 1.2 Dynamic configuration of Windows 7

|  |
| --- |
| 1. Start capturing network traffic in Wireshark! 2. Assure that the network type of the virtual machine is set to “bridged”! 3. Start the virtual machine (answer with “copied it” if asked)! Be patient as the operating system is booting (it can consume 3-4 minutes). After the machine started log in (*meres/MeresLabor4*)! 4. After the desktop appears figure out the MAC address of the virtual machine using the command line tool *ipconfig*! |
| <<MAC address>> |
| 1. Shut down the operating system (Turn Off Computer)! After exiting VMware Player stop capturing network traffic! 2. Construct a *display filter* to show only network traffic belonging to the virtual machine! |
| <<display filter>> |
| 1. Analyze the traffic! |
| <<screen shot>> |
| 1. What protocols are used by Windows 7 (how and why)? |
| * protocol: description |

## 1.3 + Advanced: operation of NAT

**Aim of this task**: to understand the operation of NAT (Network Address Translation) by tracking the route of a specific packet and by observing the modifications in addresses made by the NAT component.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1. Change the network type for the Linux virtual machine to “NAT” **before** starting the machine! 2. Just like with task 1.1 start the virtual machine and log in after the login prompt appears! 3. Figure out the IP address belonging to the VMnet8 virtual adapter of the host computer (*ipconfig*). 4. Launch **two** instances of Wireshark! In one of them start capturing the network traffic of the physical interface while in the other start capturing network traffic for the VMnet8 adapter! 5. As both Wireshark instances are running (and as such capturing packets happens on both adapters simultaneously) ping the host alaplab.mit.bme.hu from the **guest** operating system! 6. Filter the traffic to show only packets belonging to ICMP messages! Compare the two captures! What does NAT? | | | | |
| <<Wireshark1 screen shot>>  <<Wireshark2 screen shot>> | | | | |
| 1. Pick an *ICMP request* message and select the corresponding packets from both captures! 2. Fill the table bellow with the MAC and IP addresses of these selected packets! Also record the computer (host or guest) and its adapter to which the address (MAC or IP) in question belongs to! | | | | |
|  | Source MAC | Target MAC | Source IP | Target IP |
| VMnet8 |  |  |  |  |
| Physical |  |  |  |  |

# 2. Basic settings of an Ethernet switch

**Aim of this task:** make modifications to the basic configuration of an Ethernet switch via the serial interface.

Before a manageable network device – like the D-Link DES-3010G entry level manageable switch used in the laboratory – can be used we need to modify their basic settings because of the following:

* **The default settings expose a security risk!**
  + Default passwords are well known!
  + Or sometimes the default behavior is not use passwords at all!
* In most of the cases the default settings do not satisfy our needs. Usually even the most basic functions don’t work properly (or don’t work at all).
* The device is usually unreachable through the network because its network configuration (like its IP address, the address of the default gateway) is missing or contains invalid settings.

Manageable Ethernet switches (which are basically multiport Ethernet bridges doing frame filtering and relaying) operate with default settings unless we reconfigure them. As we have spent money for a relatively expensive manageable switch (instead of a relatively cheap unmanageable switch) we want to access its full range of features and not just the basic ones. For this reason we need to reconfigure them.

## 2.1 Initial settings

The possible default IP configuration of networked equipments is either not having any settings at all or to have some settings assigned either statically or dynamically by a server (e.g. using DHCP). If the default IP configuration is not appropriate or the equipment is unreachable on the network for some other reason, it is still possible to communicate with the device via the serial line (as nearly all manageable equipments have an RS-232 port). It is worth to have a look at the figure showing the network topology of the laboratory (can be found in a different document).

|  |
| --- |
| 1. During the following tasks we will use the command line interface (CLI) of the switch. First launch a PuTTY then establish a connection to the switch via the serial port COM2 using 9600 bit/s data rate with 8 data bits, no parity bit and 1 Stop bit (no flow control)! (If you see a pre-defined session named “Switch serial” you are free to use that (after you assured that its settings are appropriate). 2. Check if the switch is powered (if not then turn it on and wait until the firmware is loaded)! Log in to the equipment! According to the default settings the *UserName* and *PassWord* fields are blank. If you are unable to log on with the default settings you may try labadmin/labadmin (as the user name and the password). If you are still unable to log in ask for help by the teacher (who knows how to enter to the switch without knowing a valid user name and password pair).    1. After the firmware loaded the following prompt appears: DES-3010G:4#    2. If the switch was already turned on before you made the connection via the serial line you can make the switch to display the prompt by pressing ENTER. 3. Revert all configuration options to the factory defaults (reset system) to start with a well know configuration! Issuing this command also causes the device to reboot. Make **two** screenshots: one is about the *bootloader* and the other is about the *firmware* and record also the version numbers of the hardware and the firmware! |
| <<screenshot of bootloader>>  <<screenshot of firmware>> |
| 1. What are the default settings of the switch (select a command which show approximately one page length summary information about the most important settings)? |
| <<Screenshot>> |
| 1. What are the IP settings? (Make a query which displays specifically the IP settings in a more detailed view.) |
| <<Screenshot>> |
| 1. Configure the switch to acquire its IP address using DHCP! |
| <<Screenshot>> |
| 1. Record the new (DHCP assigned) IP address of the switch! |
| <<Screenshot>> |
| 1. List the current accounts (if any)! |
| <<Screenshot>> |
| 1. Create an account having administrative privileges using labadmin / labadmin as user name and password! (We ask you to set this authentication information to make life a bit easier during the measurement. **Never** use such a trivial user name and password pair in real life!) |
| <<Screenshot>> |
| 1. Restart the device! Check the settings you modified in previous steps (IP configuration and accounts). What have happened? Why (hint: what is the NVRAM of the switch)? Solve the problem! |
| <<Screenshots>>  <<Explanation>> |
| 1. Synchronize the clock of the device to a public SNTP server (for example: 0.hu.pool.ntp.org)! |
| <<Screenshot>> |
| 1. Check the time! |
| <<Screenshot>> |
| 1. Is the time displayed correct? If not solve the problem (hint: check time zone settings)! |
| <<Screenshot>> |

## 2.2. Additional settings

|  |
| --- |
| 1. Ask the teacher a short UTP cable! After you start capturing network traffic in Wireshark connect two of the switch’s port together using this cable (**only for a few second**)**!** |
| <<Screenshot>> |
| 1. Enable Spanning Tree Protocol (STP) and repeat the previous step! |
| <<Screenshot>> |
| 1. What can be observed and what is the cause? |
| <<Description>> |

## 2.3 +Advanced: remote access to the CLI

|  |
| --- |
| 1. Start capturing network traffic in Wireshark! 2. Log in to the switch over the network via **telnet** and not directly via the serial interface (you can use the pre-defined settings in PuTTY named “Switch telnet”)! 3. Query the time then log out! 4. Stop capturing network traffic and evaluate the results! |
| <<Screenshot>> |
| 1. Try out the “Follow TCP stream” functionality of Wireshark! |
| <<Screenshot>> |
| 1. What protocols are used (and how) by telnet? |
|  |
| 1. How much is the overhead of telnet (primarily during the time when you type commands). |
|  |
| 1. Why is some information duplicated? |
|  |
| 1. Can be the telnet protocol considered as a secure way of communication? |
|  |

# 3. The HTTP protocol[[1]](#footnote-1)

During this task we observe simple HTTP requests and responses to discover the fundamental aspects of the protocol’s operation.

To accomplish this we will use an IIS Express web server running inside a Windows 7 virtual machine and a web browser running on the host computer.

Preparation:

1. Before starting the virtual machine set the network type to “NAT”.
2. Start the virtual machine! After the boot process completes log in and start the web server using the start\_iis\_express.cmd shortcut on the Desktop (**as administrator**)! (This will launch two separate web server instances running in two separate consoles. The web servers serve two different web locations.)
3. Disable TCP port number to service name resolution in Wireshark (uncheck View / Name resolution / Enable for transport layer).
4. Web browsers aggressively make use of various caching techniques. This way network traffic can be reduced significantly as web servers are allowed to retransmit the whole content of the page only if it was changed after the last query. If not they can answer with the 304 status code meaning the page was not modified. To avoid this situation and to make the web server to always transmit the whole page use Ctrl+F5 if you want to refresh a page!

## 3.1 Investigating a simple request

**Aim of this task:** is to observe the protocols used during the download of a simple web page.

|  |
| --- |
| Figure out the IP address of the virtual machine then start capturing network traffic by Wireshark on one of the virtual interfaces in the host machine (there are two of them, you have to use the one which is on the same subnet as the virtual machine). Then visit the web page on http://<IP\_address\_of\_the\_virtual\_machine> in a browser started on the host machine.  Filter out all the network traffic which does not belong to the downloading of the web page (like the traffic of other machines, the NetBIOS protocol, Windows file sharing and so on.) |
| <Display filter> |
| Record the packets remained after applying the filter! |
| <<Screenshot>> |
| By evaluating the result describe the protocols used during the download process (what are those protocols, how are they used, ...)! |
|  |
| Investigate the process of the communication by the help of Statistics / Flow Graph with General Flow option selected! |
| <<Screenshot>>  <<Description>> |
| How many TCP connections have been established? Why? |
|  |

## 3.2 The structure of HTTP headers

**Aim of this task:** is to observe an HTTP request and the corresponding response to it.

|  |
| --- |
| Investigate the content of the HTTP traffic (right click on one of the HTTP packets the select Follow TCP Stream)! |
| <<HTTP traffic content>> |
| What type of HTTP request was generated by the client? Describe it in more detail (like what is the purpose of the give request, how does it work, what resource was requested and so on)! |
| <<Request>>  <<Description>> |
| What HTTP headers were sent by the client? |
|  |
| What kind of information can be figured out of these headers (like regarding some properties of the operating system or anything else)? |
|  |
| Identify the building blocks of the server’s answer! |
|  |
| What status code has been returned, what does it mean? |
| <<status code>>  <<meaning of it>> |
| Record another network traffic capture during the request of the following URL: http://<IP\_address\_of\_the\_virtual\_machine>/notexists.html  What status code has been returned in this case? What does it mean? |
| <<status code>>  <<meaning of it>> |

## 3.3 + Advanced: investigation of other HTTP functions

**Aim of this task:** is to get familiar with the *Keep-Alive* function of HTTP and to observe how it is possible to serve multiple web sites by a single web server.

|  |
| --- |
| Let’s investigate a little bit more complicated web page! Capture the network traffic as you visit the page on http://<IP\_address\_of\_the\_virtual\_machine>/bme/vik.html  How many TCP connections were established by the browser in this case? And for what purposes are they used by the browser? |
|  |
| You can make a list of the resources requested by the help of Statistics / HTTP / Requests… |
| <<screenshot>> |
| To this point every time we requested a resource a TCP connection has been established and then closed by the client after the arrival of the response. This is not too effective. To resolve this issue *persistent connections* have been introduced in HTTP 1.1. If the value of Connection field is *Keep-Alive* then the client may keep the TCP connection open and request other resources afterwards.  Capture the network traffic as you visit the page on:  http://<IP\_address\_of\_the\_virtual\_machine>/keep-alive/vik.html How many TCP connections were established by the browser in this case? And for what purposes are they used by the browser? |
|  |
| It is possible to serve multiple web sites by a single web server. To distinguish among the possible sites web server usually use the information bellow:   * target IP address, * target port number, * the so called *host* header (the Host field in the header of the request which contains the DNS name or the IP address used during the request).   Let’s observe some request to different web sites! Capture the network traffic as you visit the web page on http://<IP\_address\_of\_the\_virtual\_machine>:8000*/* (the usage of port numbers 80, 8000 or 8080 is just convention, it is perfectly possible to configure a web server to listen on any other port number). (Warning: if you use any other port but the standard 80 then Wireshark won’t recognize the packets as HTTP just as TCP.) What is the value of the *Host* field now? |
|  |

## 3.4 Authentication methods (basic)

**Aim of this task:** is to investigate the basic method of HTTP authentication.

|  |
| --- |
| There can be contents on the Web what are not public. In this case we need to turn off allowing *anonymous* accesses to that given resource and use some authentication method.  Capture the network traffic as you download the web page on:  http://<IP\_address\_of\_the\_virtual\_machine>/authentication/basic  (Use the same login information as with the virtual machine itself!) Evaluate the traffic captured and try to understand the way HTTP’s basic authentication operates!  How is it signaled by the server that the client is about to request a protected resource? |
|  |
| In what form (encoding) is the password sent by the client? |
|  |
| Can this form be considered as a safe way of sending passwords? |
|  |
| Click on the Next link on the page! Were you presented a login dialog again by the browser in this case? If yes, why? If not how do you think the web server gets the authentication information? |
|  |
| Let’s try out the following! Open a new tab and request the same page:  http://<IP\_address\_of\_the\_virtual\_machine>/authentication/basic  Close the browser completely then reopen it and request the page once again! In which cases were authentication information requested from the user? Why? |
|  |

## 3.5 + Advanced: other authentication methods

**Aim of this task:** is to get familiar with NTLM authentication connection on Windows and the SSL/TLS protocol.

|  |
| --- |
| We have many opportunities to raise the level of security during the authentication phase. We can use more secure authentication methods or we can encrypt the whole application level part of the message.  Let’s have a look at a more secure authentication method at first. Request the URL bellow:  http://<IP\_address\_of\_the\_virtual\_machine>/authentication/windows  What kind of authentication method is used on this page? What information is sent by the server in this case? |
|  |
| Can this authentication method be considered as safe? |
|  |
| How does the server inform the client the authentication methods it supports? |
|  |
| What can be the disadvantage of this authentication method? |
|  |
| Another option is to use the basic authentication method but to encrypt the whole traffic above the transport layer using SSL/TLS.  Capture the network traffic as you request the page bellow: https://<IP\_address\_of\_the\_virtual\_machine>:44300/authentication/basic  Record the content of the certificate too!  What port is used with HTTPS? |
|  |
| What can be observed now (about the HTTP messages)? |
|  |
| For how long is the certificate of the web page valid? |
|  |
| Not all web site use HTTPS for all of the web pages? What can be the cause? |
|  |

## 3.6 Sending data back to the server by HTTP

**Aim of this task:** is to investigate the possibilities for the clients to send additional data back with the request to the server.

|  |
| --- |
| If the client wants to send data to the server there are more possibilities.  The simplest case is to pass the data as *parameters* within the URL (query string).  Capture the network traffic as you click on the *query* link on the page bellow: http://<IP\_address\_of\_the\_virtual\_machine>/data/querystring/  In what form is the data passed to the server? |
|  |
| There can be arbitrary number of URL parameters with arbitrary names. Let’s try out some handmade requests and observe the result sent back by the web server! |
|  |
| Another frequently used option is to pass the data using input fields of a *form*.  Start capturing network traffic then request the page bellow. Fill in the form and submit it. Wait for the page generated as the response from the server. After the response page loaded you are free to stop capturing traffic. http://<IP\_address\_of\_the\_virtual\_machine>/data/form/  What HTTP method is used typically in this case? |
|  |
| How the parameters are passed using this method? What are these parameters now? |
|  |
| How many parameters are passed? Where do all these values come from? |
|  |

## 3.7 Optional: additional ways to send data

**The aim of this task:** is to observe the operation of a simple *Web Service*, and to observe the standard way to pass parameters in this case.

(This task is not needed to be solved but it is worth to go through it if you have time.)

|  |
| --- |
| The methods used in previous examples are just the most basic ones. If more complex data structures are needed to pass or more complex actions are about to take there are many methods and technologies to accomplish these tasks. Here we focus on a SOAP web service over HTTP. Web services are self describing entities making machine-to-machine cooperation possible over the network[[2]](#footnote-2).  Launch a browser inside the **virtual machine** and open the following page: http://localhost/data/ws/hello.svc  Our web service has been implemented in .NET. We can see the basic page generated by .NET framework. Look at the WSDL description of the service! What kind of operations is provided by this service? |
|  |
| Copy *HelloWorldConsoleClient.exe* and *HelloWorldConsoleClient.config* from C:\inetpub\wwwroot\data\ws\client\HelloWorldConsoleClient to the desktop of the host machine! Update the IP address stored in the *address* value of the *endpoint* element in the configuration file to the current address of the virtual machine!  Start capturing network traffic as you launch the program from command line and then access the remote services (you need an arbitrary parameter)! Explain the communication observed! |
| <<Screenshot>>  <<Description>> |
| How is the parameter passed? |
| <<Screenshot>>  <<Explanation>> |

1. The protocol is defined under IETF’s RFC 2616 (<http://www.ietf.org/rfc/rfc2616.txt>). This document can be consulted if something is uncertain about HTTP. [↑](#footnote-ref-1)
2. It is hard to give an exact definition. Further information on web services can be found here:  
   Web Services Glossary, <http://www.w3.org/TR/ws-gloss/> [↑](#footnote-ref-2)