1) The difference between lastlog and wtmp is: though they both have to do with recording the logins of users, wtmp is just a binary log file that appends new entries to it. The lastlog on the other hand just stores the time of last login from every user. It is a binary file that is indexed by the UID of the users.

   A reasonable rotation policy for each would be: wtmp rotation policy would depend entirely on your systems needs and context, if you have a lot of users logging in then you might be using a lot of disk space, so more frequent rotations might be necessary, I would use logrotate to handle this monthly as I think that's a good amount of time for this file. Lastlog does not need to be rotated unless you have a lot of new users coming to the system, if so then logrotate can rotate the /var/log/lastlog file whenever it gets to a certain size.

2) Listening to ICMP redirects could allow unauthorized users to compromise a network by: an Attacker forging ICMP redirect packets, they are very easy to fake and once the attacker was in, they could alter routing tables and diver traffic to hosts of their choice.

3) The MTU (maximum transmission unit) of a network is the largest packet size that can be sent in a network. If the MTU is set too high then it will fragment the packet, if the router has a do not fragment flag, it'll just send 2 smaller packets. If the packet is too small then it will avert fragmentation and increase the performance.

4) There are 4 /18 subnets, they are:

   143.123.0.0/18

   143.123.64.0/18

   143.123.128.0/18

   143.123.192.0/18

   The subnet mask is: 255.255.192.0 = 18

b) 2^(18-2) = 2^16 = a total of 65536-2=65534 hosts. With 16382 on each subnet (65528).

c) 143.123.67.124 belongs to the 143.123.64.0/18 subnet

d) the 4 broadcast addresses are:

143.123.63.255

143.123.127.255

143.123.191.255

143.123.255.255

5) a. There are 3 ethernet frames on the /24 network, first would be an ARP request from the host with the ip 128.138.2.255. Then it would be ARP reply from the router of the connected host. Lastly it would be the data from the source to the destination within an ip packet that goes to the connected routers/hosts.The source and destination ip address/ethernet frames for these 3 things are:

ARP request: 128.138.2.4 (EA1) to 128.138.2.255 (router mac address:FF:FF:FF:FF:FF:FF)

ARP reply from router: 128.138.2.1(EA2) to 128.138.2.4 (EA1)

Data sent: 128.138.2.4(EA1) to 128.138.129.12 (EA2)

B. with a /16 network the answer would change. The ARP request would be different, wouldn't have to go through a router, data would be sent straight to ethernet address

c. with a /26 network it would change, ARP request ip would change from 128.138.2.255 to 128.138.2.63

6) For this one I tried going into CSVM and finding the .rpm files but that didn't work. I realized that there was a URI link on ivans example, I then used the commands:

zypper ar -f nfs://lbc/csvm1_2/opensuse/distribution/15.0/repo/oss leap15full

and

zypper ar -f nfs://lbc/csvm1_2/opensuse/updates/leap/15.0/oss updateOSS

And then removed the opnSUSE repo I had.

For b. I first used the command zypper list-updates to see the updates (quite a few) and zypper update to update all the repos (that took quite a bit). I then rebooted to restart all those processes.

7) To check if it was running, I did the command ps -A | grep rsyslog and if it returned a process then I knew it was running. I also just used $ systemctl status rsyslog and it was in fact active

8) Man pages

   1) dmesg ~ this command prints the kernel ring buffer. The output typically looks like: [8.851351] EXT4-fs (sda4): mounted filesystem with ordered data mode Opts: (null), it is essentially a data structure that records data messages related to kernel operation. With dmesg you can also control the buffer as well.

      OPTIONS: -C, this clears the ring buffer, a nice option if you need to see how the kernel is gonna operate after doing a specific task related to operation of the kernel.

   2) lastlog ~ this command is for reporting the most recent login time for users. You can specify users and times as well. The data is taken from the /var/log/lastlog file which logs all the login times for users(it is a binary file).

OPTIONS: -u, this option specifies the user for their last login. Important if you have a lot of users and don't want to list through every user.

3) logger ~ this command inserts user specified messages into the system log. The syntax for this command is just $ logger [options] [message]. The exact file that gets edited with this command is /var/log/syslog.

   OPTIONS: -i, this logs the process id of the logger process into each line. I think this is a helpful option as the pid is a very important part of the process if you were logging something with a process.

4) logrotate ~ this admin command is for rotating system logs. You can compress the log or even mail the log as well. It is primarily a tool for cleaning up some log files that simply take up too much space on the system.

   OPTIONS: -v, showing the messages every time rotation occurs may be handy just for knowing your system and when and how it cleans up itself.

5) rpm ~ this command is a package manager. With rpm you can install and update packages when necessary. A package is essentially a zip of files and meta-files that all builds to make a program.

   OPTIONS: -i, probably one of the most used option, this is for installing the package. -f is also a good for if you're installing from a url or uri.

6) rsyslog.conf ~ This is the configuration file for rsyslogd. It is located in /etc/rsyslog.conf. The file essentially contains the rules for how the system may log.

OPTIONS: N/A

7) rsyslogd ~ or reliable syslogd, is a utility that handles message logging. The relevant files for this utility is in /etc/rsyslog.conf, /dev/log, /var/run/rsuslogd.pid and prefix/lib/rsyslog. Rsyslogd is susceptible to DOS attacks by flooding the daemon with syslog messages, this cloud clog the disk space on the system.

   OPTIONS: -i, this option seems useful if you wanted multiple logging instances. Perhaps you need multiple logging rules and types for 1 system.

8) syslog (3) ~ This c library routine is for sending messages to the systems logger. The main function in this library is openlog(), this function connects the systems logger to the c program. It is automatically called by syslog(), which itself, generates the log message with the parameters priority and format.

   OPTIONS: N/A

9) syslog (8) ~ this utility is the system logging service, you can have different implementation as the syslog service, such as syslogd, rsyslogd, and syslog-ng. To find the options for start parameters or other params for the daemon is located in /etc/sysconfig/syslog.

   OPTIONS: N/A

10) systemd.journal-fields ~ This word processing package deals with special journal fields. There are trusted (journal fields that are only added by the journal and can’t be changed from client code) fields that have an underscore and an = sign to denote the value.

OPTIONS: N/A

11) wtmp ~ this is similar to lastlog in that it takes the last login and logout of users. It is a file /var/log/wtmp, however it is only readable by who and lastb, since it has binary data within it.

    OPTIONS: N/A

9) At first glance, they both look like lines of gibberish to me, like lines you would see in an install log, just a bunch of lines only relevant to the kernel or hardcore system admins, I did noticed the messages file had actual time stamps for each log as well as the pid for the logger, much more readable for a human, I grepped through dmesg to try to find similar keywords in the ring buffer to the messages file but I didn't find much. With my username I found [3.619780] systemd[1]: Set hostname to <christian2>. What I basically noticed is that although the log syntax is a bit different, dmesg's entries are recorded in messages, I noticed that messages also has a lot of log messages from other things like application-related entries that weren't in dmesg
10) rsyslog is mainly configured by the rsyslog.conf file. There is also the rsyslog.d directory where it seems a bunch of smaller files are used to config rsyslog, but ours only has remote.conf which configures for remote logging. The way the logging works in our rsyslog service is rsyslog first takes a series of inputs with the assistance of modules. It is then passed to a serious of rulesets, default or user specified. Depending on the rules, a action is committed, this is where the logging gets sent to the outgoing logging files and services.

The first thing I would change is limiting the amount of log messages the service will accept. I was reading about some of the security issues that rsyslog has and I think this will reduce the chances of getting DOS'd. To do this I added the lines

$SystemLogRateLimitInterval 1

$SystemLogRateLimitBurst 25

To the rsyslog.conf file. This essentially limits the amount of messages I get, specifically if for every 1 second, if the system receives more then 25 logs, then limiting will occur. The second thing I would change is logging cron files, I feel like knowing whether those jobs have worked would be a good thing to log. I just added the line $ cron.* /var/log/cron to the file.

11) Logrotate is what is being used to rotate my log files, or me there was no mention of rotating log files within any of the cron.[daily, monthly, etc] files.The configuration file for this is located in /etc/logrotate.conf and there is a service for it called logrotate.service, which runs one time at bootup. The execStart is equal to /usr/sbin/logrotate and /etc/logrotate/conf. According to the conf file it rotates log files weekly, it also compresses the files. I noticed that /etc/logrotate.d had a directory full of custom rotations. Messages was not rotated, to start rotating messages if it gets to a size of 512 kb I first created a messages file in logrotate.d, I then created a function for it, but it failed to restart. I then found in the custom rotation file syslog and found a reference to messages already so I deleted that and the service worked. Before messages had over 8000 lines of entries and now it is empty.

Report: Overall this lab was on more theory based side then the practical side, I enjoyed the last few question because it actually had me changing the system. I liked the last one in particular because it was interesting learning about the way these log files are managed from getting too big. Overall this lab took me about 12 hours. I think 5 was the hardest question though because I had a lot of problems was a) was actually trying to say. The most boringest part was of course the man pages.