WHAX

WHAX - MODULARISED 2

What's new ? 2

Distribution Change 2

Modularity 2

Customizing WHAX 4

Creating modules: 4

Integrating modules into the ISO file: 6

Conclusion 6

WHAX - MODULARISED

Developers info – alpha 0.0.0.1

What's new ?

There are some major new features in WHAX which add huge functionality to Whoppix, and may change the way we use live distributions. But before we get all fired up, a bit of general information:

Distribution Change

WHAX is so longer based on Knoppix, but on SLAX, a Slackware live cd. One of the main reasons I decided to try this is due to the wonderful world of modularity which SLAX uses. For us, this modularity means that we can easily customize our version of WHAX, and include whichever modules we like. All the tools have been compiled to "WHAX Modules which can be easily added or removed, depending on our needs.

Modularity

Modularity also makes software updates much easier – for example, if a new version of <great app> (such as Metasploit Framework) has been released, we can now download the updated module, and update our livecd / usb stick instantly. This new setup allows for the greatest flexibility, as modules are easy to create (even from windows) – which means that if we want to add that tool (or wallpaper) we always wanted, but never appeared in Whoppix, we can now do it ourselves! A wonderful new software interface makes this whole process s breeze.

WHAX will come in two flavours – The "regular edition" which is not modular, but bundled up into a small number of modules. This would be the best choice for someone who needs as a live cd. The "dev-edition" is broken down to more than 200 modules, each which can be replaced or updated. This would be the best choice for someone who wants to modify for his own needs, or for an HD install.

Customizing WHAX

New modules will be released on a regular basis, and will be announced and put online for download, once they are tested and ready. A list of the current modules can be found at: http://www.whoppix.net/whoppix-3.0-modules.html

Creating modules:

NOTE: It's probably easiest working from within a HD install to make modules, but it is not necessary.

Modules (.mo) files can be created several ways:

- Creating a module from source code (amap example):

  Pre-requisites:

  - WHAX-iso - www.iWHAX.net
  - amap - http://thc.org/releases/amap-5.1.tar.gz
  - checkinstall -http://asic-linux.com.mx/~izto/checkinstall

1. Boot WHAX and open a shell.
2. Change directory to /tmp, get/wget the current amap-source at thc.org and extract it (*tar xzf amap-5.1.tar.gz*).
3. Change into the amap-directory and configure the tool using '*./configure --prefix=/usr*'. Compile the sources using 'make'.
4. If the make-process is done, run '*checkinstall*' (instead of 'make install'). Checkinstall will create a Slackware, RPM or Debian compatible package. For more details about checkinstall visit the checkinstall-homepage[1]
5. After checkinstall finishes building the package you should have a new slackware package in the same directory called 'amap-5.1-i386-1.tgz'.

6. Use tgz2mo to convert the tgz slackware package to a module. Run *'tgz2mo amap-5.1-i386-1.tgz amap-5.1-i386-1.mo*' and checkinstall will create the module named 'amap-5.1-i386-1.mo'.

- Copying directory structures (framework 2.4 example):

  Pre-requisites:

  - WHAX-iso - www.iWHAX.net
  - framework-2.4 - http://www.metasploit.com/tools/framework-2.4.tar.gz

1. Boot WHAX and open a shell.
2. Change directory to /tmp, get/wget framework-2.4 and extract it (*tar xzf framework-2.4.tar.gz*).
3. Create a directory-structure using *'mkdir -p build/pentest/exploits*' and move the extracted framework-2.4 folder to build/pentest/exploits/.
4. Run *'dir2mo build/ framework-2.4.mo*' – this will create the module.

Other interesting tools like rpm2tgz also exist, and make it possible to convert rpm packages to slackware packages.

Integrating modules into the ISO file:

1. On your windows box install and run MySLAX-creator as administrator.
2. Select the ISO where you would like to add the new module and mount it using 'Mount'.
3. After changing to the next window you're able to add new modules. Do so and pick the module you just created and copied to your windows box.
4. Create the new ISO using 'Create ISO', burn it and boot it. Log in as root, change to /PENTEST/exploits/framework-2.4 and you will magically see framework 2.4.

Conclusion

With this new modularized system, WHAX will always be up-to-date with all the new tools and security fixes. Downloading the update and updating the ISO or USB pen drive takes minutes, and new modules can be custom made and added easily.

Enjoy a great release!

The WHAX Team – {muts, ports}

Scripts: makepentest.sh

Copies the updated /pentest folder into a temp folder, and generates .mo's out of them.

```
#!/bin/bash
echo "Deleting /tmp/BUILD/TMP/pentest/  /tmp/BUILD/TMP/pentest/"
rm -rf  /tmp/BUILD/PENTEST/*
rm -rf  /tmp/BUILD/TMP/pentest/
mkdir -p /tmp/BUILD/TMP/pentest/
ls -l /pentest/ |awk '{print $9}'|cut -d"/" -f1> maindirs
for maindir in $(cat maindirs);do
        rm -rf  /tmp/BUILD/TMP/pentest/
        echo "Processing $maindir"
        ls -l /pentest/$maindir |awk '{print $9}'|cut -d"/" -f1>subdirs
        echo "Looping Subdirs"
        for subdir in $(cat subdirs);do
                rm -rf  /tmp/BUILD/TMP/pentest/
                echo "          making /tmp/BUILD/TMP/pentest/$maindir"
                mkdir -p /tmp/BUILD/TMP/pentest/$maindir
                echo "          copying /pentest/$maindir/$subdir /tmp/BUILD/TMP/pentest/$maindir"
                cp -rf /pentest/$maindir/$subdir /tmp/BUILD/TMP/pentest/$maindir
                mkdir -p /tmp/BUILD/PENTEST/$maindir
                echo "          MO-ing $subdir$(date +%Y%m%d).mo"
                dir2mo /tmp/BUILD/TMP/ /tmp/BUILD/PENTEST/$maindir/$subdir-$(date +%Y%m%d).mo
        done
done
exit 0
```