Ke clean

Sciifii

Credits

Sciifii has been done by TeamWiigen:

- Coders: Arasium, Teton and Fanta
- Testers: JeanMi, Jicay, Thetataz, Vlad, Oranda (and many others)

Greetings

Waninkoko, for all job he's done on the Wii and because he has always answered/helped us when we need some infos about his cios.
Da_letter_a and Damysteryman for their cioscorp/Darkcorp knowledge.
Everybody who has helped us and we have forget J

Contacts:

teamWiigen@gmail.com

Table of Contents

Introduction

Birth of Sciifii

In 2009, Arasium has done a guide to helps everybody who wants to hack their Wii. This guide was quite good and like every good job, a lot of people has copy/paste it on warez boards or other forums without including links to the original guide (for French people, you can find on Wiigen forum, under the member tutorial section).
But, the worst isn't these thieves. It is the noobs who aren't able/don't want to read the guide. As everybody knows, always repeating the same thing is very boring. So Arasium has the idea of a homebrew that will do the entire job. No more stupid questions (after Sciifii v1 or v2 release, we realize that we can't avoid stupid questionsJ), no more reading and overall no need to always update the guide.
So the Sciifii v1 project was born. This first version wasn't very powerfull due to a lack of configuration, some bugs etc…. But now, Sciifii has evolved!
Enjoy our job

The goal of Sciifii

A lot of people see a hack tool when they take a look to Sciifii. But Sciifii isn't only this. Our goal was to create a very flexible application that can be used for a lot of things.
So yes, Sciifii is provided with a configuration file wich will hack your Wii. But it is very easy to create more apps (for example of firmware downgrader can be done very quickly). Actually, we can describe Sciifii has a tiny task sequencer. All the application can be modified in the xml. And **this** is the power of Sciifii.

Changelog

- Added a log system
- The cios installer was rewritten
- Cmios included (done with the new ciosInstaller)
- Cios r19 base 37 added in slot 250
- The libs were rewritten
- Now, fully customizable in the xml. No more embedded datas.
- Improved the config.xml
    - Removed the flags (it was a stupid idea done for the v2)
    - CorpInstaller and SystemUpdater have their one configuration (so you can use them more than once)
    - Added a file system step
    - Added a file manager to export some patches/datas from the executable
- And some bug fixed J

How to use Sciifii

Prerequisites

First, you need an internet connection. Sciifi use some files we can't give you because there are copyrighted. But Sciifii can download them from Nintendo update servers. You can use your Wii internet connection or a computer.
Then you need a brain (I say that, because some people will always try to not use it), an sd card and a Wii J

Installation methods

Sciifii needs to download items from the network. So you can launch Sciifii and he will automatically download all the needed files for you.
But keep in mind that you can also use our pc application to download these files. In fact, using the pc tool is a lot quicker. So perhaps you will prefer our pc app (or perhaps your Wii isn't connected to the networkJ)

Sciifii modes

When you start Sciifii you will see a menu. This menu will show all the available modes. In Sciifii, a mode is a default installation option set. In fact, each mode will automatically activate/deactivate some options.
We provide these modes for users who don't care about how Sciifii does his job. But we suggest you to take a look to the advanced mode (at least, just to have an idea on what Sciifii really does on your Wii)

- Hack your Wii ! (Light Mode): this will only install the waninkoko cios rev17b and the priiloader.
- Hack your Wii ! (Full Mode with LoaderGX): This will do the same that lite mode but it will also install the GX USB Loader.
- Hack your Wii ! (Full Mode with Wiiflow): This will do the same that lite mode but it will also install Wiiflow.
- Unhack your Wii: This mode will remove all cios from your Wii and will update it to firmware 4.2. No channel will be removed. So I suggest users to remove their channels before using this (with any title delete for example). Once the uninstallation done, you can remove the HBC and/or bootMii with Hackmii installer.

Sciifii options

Like explained just above, Sciifii modes will only active some options. You can see this options in the advanced menu. I will describe here every option you can find in the config file provided with sciifii. You can do a custom installation selecting the required options in the advanced menu.

- Restore Trucha Bug : This will restore the trucha bug in the ios36. The ios 36 used as base to create the trucha ios is the latest version.
- Install cios 38 rev17b: This will install the cios38 rev17b from waninkoko. The installation will be done under the ios36. So you need an ios36 with the trucha bug (or use the first option).
- Update: This will update you Wii to the firmware 4.2. The installation is done under the cios249 (so be sure to have it or activate the cios option). The cioses (ie 249, 250 etc..) won't be modified.

- Remove the cioses: This will removed the cioses with the original ioses from Nintendo.
- Priiloader: this will install the priiloader and the hack.ini file on your sd card.
- Corp: It will install the cioscorp/darkcorp on your Wii. This includes the cmios, but, if you want the cios56 and 58, you need to extract them from GH5/RB and a game wich used the cam.
- GXLoader: It will install the GX Loader on your sd and the GX Loader channel on your Wii.
- Wiiflow: The same as GX Loader but for Wiiflow.

Here is a summary of the options activated in each mode:

Hack your Wii ! (Light Mode)	TBR + cIOS + Preloader
Hack your Wii ! (Full Mode with LoaderGX)	TBR + cIOS + Preloader + GXLoader
Hack your Wii ! (Full Mode with Wiiflow)	TBR + cIOS + Preloader + Wiiflow
Unhack your Wii	TBR + cIOS + Update + Uninstall

Error resolutions

Sciifii can work on a lot of Wii. But, it can encounter some problems with some configurations. Generally, these problems are due to an old cioscorp.
The cioscorp increase the ioses revision number to their max value. So we can't manipulate them easily.
In case of problems during the first installation phase of Sciifii (ie the TBR), we suggest you to use advanced mode and select this items: cIOS, update (to remove the old corp), Preloader, corp (if you want to reinstall it), and an usb loader.
If the error persists, you can send us a mail with your ip. We will take a look at our logs in order to identify what happens.

Sciifii customization

As already explained, Sciifii is just a task sequencer. We will see here how to modify the configuration file and we will describe each option of the configuration.
Actually, the configuration file need to be stored on sd:/sciifii/ folder. Perhaps we will update our app to use another device (but you will always need your sd to use bannerbomb or any other hack).
Be carefull, every item in the configuration file is case sensitive.

Global structure

First, the root tag must be called "sciifii". Sciifii is composed of logs, files, modes, options, steps, a disclaimer and some options. Here is a table of the "sciifii" tag structure:

Name	Element type	Cardinality	Type	Format	Remarks
Version	Attribute	1	Int	Decimal	This must match the Sciifii version. If not, Sciifii will refuse it.
MenuMessage	Attribute	0-1	String	-	This message will be prompt on the menu.
AllowAdvancedMode	Attribute	0-1	Bool	true/false	True is the default value. If this is set to

					false, the advanced won't be available.
workingDirectory	Attribute	0-1	String	usb:/ or sd:/	This element will be used as a temp directory. All downloaded files will be putted here by default. sd:/sciifii/temp is the default value.
logs	complex node	0-1	log	-	See the log xml element
files	complex node	0-1	file	-	See the file xml element.
modes	complex node	1	mode	-	See the mode xml element.
options	complex node	1	option	-	See the option xml element.
steps	complex node	1	-	-	See the description of available steps.
Disclaimer	node	0-1	String	-	This text will be displayed in the disclaimer

Log section

Now, it is possible to log what Sciifii does. There are actually three different versions of the loggers: a file logger, a gecko logger and a web logger.
Three types of events are logged: errors, warnings and information. For the file and gecko loggers, the logs are composed of strings (send to file or gecko).
But the web logger is quite different. It consists of an http call with some parameters. You need to provide the page to call, but the parameters can't be defined. If you want to create your own web page you need to catch these parameters.

Name	Element type	Cardinality	Type	Format	Remarks
type	Attribute	1	String	gecko file web	This defines the logger type to use.
category	Attribute	0-1	String	error warning info all	This indicates what kind of message will be logged by the logger. "all" is the default value.
path	Attribute	0-1	String	usb:/ or sd:/	This will be used by the file logger in order to create the log file.
url	Attribute	0-1	String	A valid internet url (http get)	Used by the web logger. The parameters send to the web page are: line, message, file, application and version.

Files section

In the v3 of Sciifii, we have created a file manager. The goal of this manager is to download and store some required files. In the previous version of Sciifii, a lot of files were embedded in the homebrew. This was a bad idea, because the dol became very heavy and we can't update Sciifii without recompiling it. The file manager is configured with the files section of the configuration file. Each managed file is described with a "file" element in the files section.
If the file needs to be downloaded, the FileManager can also validate it with the sha algorithm.
If Sciifii asks for a file that isn't in the FileManager, the file manager will try to find it using the working directory and the key.
Here is a description of the attributes of the file element:

Attribute	Type	Cardinality	Format	Remarks
url	String	1	a valid internet url	This url will be used to download the file
sha1	String	0-1	a valid internet url	If provided, this url needs to be a file containing the hash of the file to be download.
key	String	1	Don't use / in this string	This is a key used to get the file from the file manager.
path	String	0-1	sd:/ or usb:/	This is the full path were the file need to be stored. If the file is missing, the file manager will download it. If the path is not provided, Sciifii will create the path like this: workingDirectory/key

Modes section

This section will describe all the modes available on Sciifii. These modes will be displayed in Sciifii menu. Each mode is represented by a mode element in the elements section.

Attribute	Type	Cardinality	Format	Remarks
text	String	1	-	This is the text displayed on screen
options	String	1	A list of options name.	Each option name must be separated by |. Every option in this attribute must have a corresponding option in the options section.
flag	String	0-1	-	This is a flag than can modify Sciifii execution. Actually, the only flag used is Uninstall.

Options section

The options are items we can switch on/off. These options are here to define the tasks need to execute. The modes are custom sets of options. We can see the options to be switch on in the options attribute of a mode element.

If the advanced mode is available, we will see in the advanced menu these options.

Attribute	Type	Cardinality	Format	Remarks
name	String	1	-	The name is the identifier of an option.
text	String	1	-	This text will be displayed in the advanced menu.

Steps section

This section is very important. It is in this section we will define all the tasks of the sequencer. The number and the types of tasks have no limits.
So this chapter will detail every available step.
But, before starting the descriptions, you have to know that every step has an “option” attribute. This attribute indicate to Sciifii what are the options that can switch on this task. If one (or more) of this options are on, the task will be executed. If no option is defined, the task will be always executed. This attribute is optional and the default value is an empty string.

TitleDowngrader

This task will downgrade an ios. In order to do that, it will start to install the latest ios version and modify the tmd in the Wii temp folder to modify the ios revision to 0. After this operation, we can install an ios with a higher version.
The tag name is “TitleDowngrader” and here is its content:

name	element type	cardinality	type	format	Remarks
id	attribute	1	u64	hexa, without 0x	This is the title full id.
revision	attribute	1	u16	decimal	This is the wanted revision

IOS reloader

The IOS reloader indicates that Sciifii need to reload under another ios. There isn’t any verification on the ios. So be sure that the ios isn’t a stub.
The tag is “IOSReloader” and here is the content description:

name	element type	cardinality	type	format	Remarks
id	attribute	1	u32	decimal	This is the ios number (not the full id)
user	attribute	0-1	u16	-1 or 0	This corresponds to an enum value. -1 wich is the default value does nothing. 0 will identify Sciifii as SU after the reloading

Cios installer

This is the more complicated task. This task will allow you to patch an ios and install it on your Wii. Near every cios can be done with this task. If you don’t really know what you are doing, we suggest you to don’t use this.
A cios is composed of many items. First, the original ios is patched with some simple patches. These patches will just replace a binary pattern by another one.

But a cios is also composed of plugins and additional modules. It is possible to insert them as well. We don't write this document to explain you how the hack is done, but how Sciifii works. So we won't deep furthermore into the ios hack.
Before installing the cios, this task will remove the existing ios (to avoid revision conflict). So you need to be under an ios with enough privileges.
The cios installer tag is "CiosInstaller":

name	element type	cardinality	type	format	remarks
source	attribute	1	u32	decimal	This is the ios base number
revision	attribute	1	u16	decimal	The ios base revision
slot	attribute	1	u32	decimal	The slot where to install the cios
ciosRevision	attribute	1	u16	decimal	The revision of the cios
modules	complex node	0-1	module	-	This is the place where you need to describe all the additional modules
plugins	complex node	0-1	plugins	-	You need to put the plugins here
patches	complex node	0-1	various patches	-	patch list

Cios modules

A module is an elf file compiled with arm compiler. The specified modules will be inserted to the ios and the task will do all manipulations job for you. This xml part is quite simple.
A modules tag is "module":

name	element type	cardinality	type	format	remarks
file	attribute	1	string	-	It refer to a file manager item
position	attribute	0-1	u16	decimal	If the module needs a specific position in the ios, you can specify it.

Cios plugins

A plugin is a piece of code we insert in an existing module/elf. With Sciifii you can use what we name additive plugins (the code will be added into the module as a new section or in an existing section) or replacement plugins (the plugin will replace an entire module section).
We can make the distinction between the two different plugins type using the header tag. The header tag will describe how we insert a new plugin. If the header tag isn't provided, the plugin will be inserted in an existing section.
We will describe the "plugin" tag, then we will try to explain how Sciifii detects the different plugins type:

name	element type	cardinality	type	format	remarks
dest	attribute	1	string	-	The plugin will be applied on the ios module specified in the dest attribute (FFS ES, DI etc…)

file	attribute	1	string	-	Reference a file manager item wich contains the plugin.
offset	attribute	0-1	u64	hexa without 0x	This indicates the plugin position in its section. The good section will be automatically found.
bss	attribute	0-1	u64	hexa without 0x	This defines the new bss size.
segment	attribute	0-1	u32	decimal	It indicates wich section to replace. 0 to create a new section.
header	node	0-1	-	-	This describe the program header for the new section
handle	node	0-n	-	-	This is patches required to enable the plugin.

The header node is very important. Sciifii use this to decide if the plugin is a piece of code to include in an existing section or if the plugin is a full new section (a section is an elf part. You can find it using power-pc-eabi-readelf –a file).

Is header defined?

The plugin will be added to an existing section.

Bss and offset are required.

No

The plugin is a full section.

segment is required.

Yes

Segment = 0 ?

New section

The section will be replaced.

Yes

No

Cios patches

This section is simple. Some usual patches are already defined in Sciifii. We call them prebuild patches. You can insert them in the ios using the “prebuild” tag. You must provide the name attribute to indicate the prebuild patch to use. Here is a list of prebuild patches:

- ES_HashCheck_Old
- ES_HashCheck_New
- ES_Identify
- ES_OpenTitleContent1
- ES_OpenTitleContent2
- ES_ReadContent
- ES_CloseContent
- ES_SetUIDCheck
- ES_TitleVersionCheck
- ES_TitleDeleteCheck

- ES_MEM2Protection
- FFS_PermsCheck
- DIP_UnencryptedLimit
- DIP_EnableDvdVideo
- KoreanKey_EnablePatch

But you can also define your own patches using the “SimplePatch” tag:

name	element type	cardinality	type	format	remarks
module	attribute	0-1	string		This is a module name (ES, FFS etc…). If provided, only this module will be patched.
pattern	attribute	1	data	hexa (with 0x), separated by comas	This pattern will indicate where to apply the patch
patch	attribute	1	data	hexa (with 0x), separated by comas	The patch must have the same size as the pattern.

Composite installer

This task is very simple. This is only a task group. You can add (as child nodes) some tasks. The goal of this item is to apply an option at a group of tasks. The better example is the GXLoader installation.
You can provide a name to this task. The name will be used during the progression report.

Title task

This allows you to manipulate Wii titles. You can install wads, extract titles as wad, decrypt tiles etc…

name	element type	cardinality	type	format	remarks
action	attribute	0-1	enum	-install -uninstall -pack -extract -decrypt	Install (default): will install a wad or a nus title, depends of the source. Uninstall: will uninstall a title based on source information. Pack: download a title from nus and pack it as wad. Extract: extract a title from the Wii and pack it as wad. Decrypt:Extract, decrypt and save a title from the Wii.
wad	attribute	0-1	string	-	Describe a wad source (file manager). Used for Installation or Uninstallation.
id	attribute	0-1	u32	-	This is a title id. It can be used for for every action.
revision	attribute	0-1	u16	-	Used in Pack or Install. It is the title revision (0 for the latest one).
path	attribute	0-1	string	file path (sd or usb)	A wad path if Pack or a folder if Decrypt.

Title installation can be done downloading titles from nus or extracting titles from wad. It depends on the defined attribute. You mustn't define the wad attribute and the id attribute because Sciifii won't be able to choose between nus and wad.
For the title uninstallation, it's the same. The only difference is that Sciifii won't download any title from nus. We only need the titleId to delete it. If you provide a wad file, the title id will be extracted from the wad.
In the other case, the wad attribute will not be used. If you need to define a wad path (for example, with the Pack action), you need to use the path attribute.

File downloader

This task is very simple. The only goal of this task is to download a file managed by the file manager.

Wad Batch installer

This task is very simple too. Sciifii will inspect the folder defined in the "folder" attribute of the "WadBatchInstaller" tag, and it will install all these wads.

File system manipulation

We have created a special task to manipulate the file system (Wii and sd or usb using fat). You can copy, move and delete files and folder.

name	element type	cardinality	type	format	remarks
target	attribute	1	string	path	-
destination	attribute	0-1	string	path	-
type	attribute	1	enum	-file -folder	This indicate if target and destination are folders
recursive	attribute	0-1	bool	-	Default value is false. This is used for folders manipulation (copy/delete)
action	attribute	1	enum	-move -copy -delete	-

The move action will move the file or folder (the target) into the destination folder. If the destination folder doesn't exist, it will be created.
The copy action will do the same, except that the source won't be deleted after the copy. For folder copy, you can specify if the copy is recursive or not.
The delete action will delete the target directory or file. For folder deletion, you can use the recursive attribute. If recursive is set to false and the folder isn't empty, you will have an error.
For the directory move or copy with recursive attribute, you can create the target folder in the target directory.
Example:
Copy sd:/foo sd:/bar => will copy all the content of foo into bar
Copy sd:/foo/ sd:/bar => will copy foo into bar. So you will have sd:/bar/foo/content