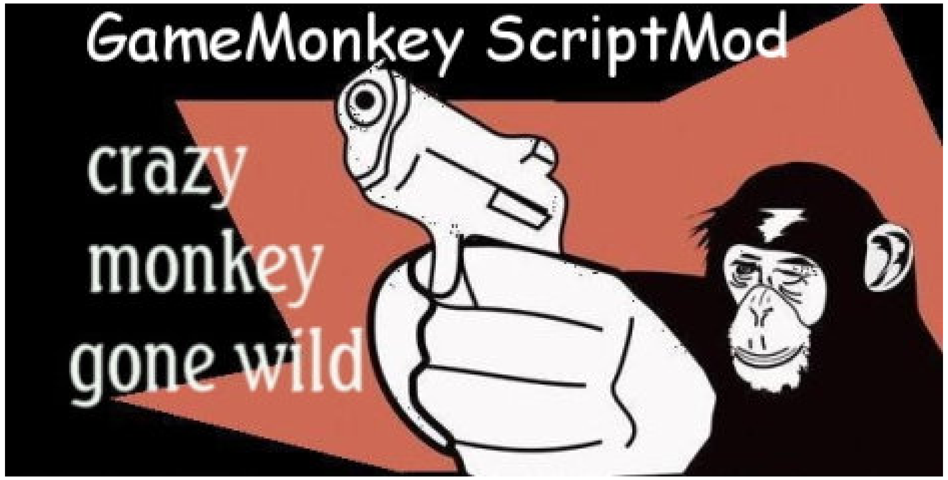


GameMonkey ScriptMod 1.1

What is GameMonkey Script?

GameMonkey is a embedded scripting language that is intended for use in game and tool applications. GameMonkey is however suitable for use in any project requiring simple scripting support. GameMonkey borrows concepts from Lua (www.lua.org), but uses syntax similar to C, making it more accessible to game programmers. GameMonkey also natively supports multithreading and the concept of states.

What is GameMonkey ScriptMod?

GameMonkey ScriptMod is a QMM Plugin, which allows Server Admins to define the Server behavior by Script.

What is QMM?

Quake3 Multi Mod (short: QMM) is a hook to the Q3 Engine and allows you to extend the Quake3 Engine functions. Check http://qmm.planetquake.gamespy.com/ for more details about QMM.

Can I use GameMonkeyScriptMod on other Q3 Engines?

Yes, and No. Currently we support only Enemy Territory. But we think about a release for all QMM Supported Games. QMM currently supports Quake 3 Arena, Jedi Knight II: Jedi Outcast, Jedi Knight:

Academy, Return to Castle Wolfenstein, and RtCW: Enemy Territory.
Can I use GameMonkeyScriptMod on my Windows Server?
Yes, and No. The gmScriptMod is written in Platform independent Code. But we haven't compiled a Windows Version yet. Most Gameservers are running Linux, so atm. we don't see the need for a Windows Version.
Where can i get some Informations about GameMonkey?
Here: http://www.somedude.net/gamemonkey/

How do i write a Script Function?

```
global PlayerList={
{0,"0",},
{1,"0",},
{2,"0",},
{3,"0",},
{4,"0",},
{5,"0",},
{6,"0",},
{7,"0",},
{8,"0",},
{9,"0",},
{10,"0",},
{11,"0",},
{12,"0",},
{13,"0",},
{14,"0",},
{15,"0",},
{16,"0",},
{17,"0",},
{18,"0",},
{19,"0",},
};

global GAME_CLIENT_COMMAND= function(int_value, string_value, concat_value)
{
   if(string_value=="gmsm" && concat_value=="FRIENDPASS" ){PlayerList[int_value][1]=1;
    say("Login from Client: "+int_value);
   }

};
```

gmScriptMod Features

- 100% scriptable Admin Interface
- scriptable Rcon-like Interface
- Interface for Clan Members. (Beside Ref login)
- Interface for Clan Friends. (Beside Ref login)
- Access to your FileSystem by .gm Script Functions
- Works with ALL Enemy Territory Mods (etmain/etpro/etbub/shrubmod/tce/etc.)

- Easy to modify. No C/C++ knowledge neccessary.
- Change your ServerMod Scriptfile without Server restart.
- Easy Script Syntax. Not as complicated as LUA.
- A lot of Script Examples.
- Fast. GameMonkey is one of the fastest Script Languages.
- Define your own "/" commands. Example: "/ban ID", "/kick ID", "drop ID"
- Create an individual Gameserver Mod. Personalize your Server.
- Exchange your Script Files with other Serveradmins.
- Client to Client Private Chat
- IRC Like Channel Chat

What is a minimal NeelixScript.gm Script?

This is a valid mini NeelixScript.gm Script

```
global GAME_CLIENT_COMMAND= function(int_value, string_value, concat_value)
{

};
global GAME_CLIENT_CONNECT = function(clientNum, firstTime)
{
};

global GAME_CONSOLE_COMMAND = function(command, concat_value){

};
```

---

gmScriptMod-1.0.zip

gmScriptMod-1.1.zip

GMSM 1.1 src (linux).rar

qmm.ini

GameMonkeyScriptReference.pdf

GameMonkey ScriptMod Command Reference

What kind of functions come with GMScriptMod?

ServerCommands:

rslap
example: rslap 2 700
that will slap the player with clientnumber 2, 700 up, and his sideward, and forward motion will be randomly increased.

slap

example: slap 2 700
that will slap the player with clientnumber 2, 700 up

ClientCommands:

hjump
example: /hjump
This will push the player into the sky, as a 'high jump'. The height is defined by a scriptfunction called 'GET_VELOCITY_Z_IMPACT'
So you can change the ammount of units the players goes up in the air.
Why are some script functions in upper case?

The upper case script functions are all called by GMSM.
What gmScriptMod functions can i use?

Here is a short list with (nearly) all available Script functions. We offer you some C++ written functions, which you can use inside your script.

index:

GAMINGGONE
GamingGoNe::sscanf
GamingGoNe::search
GamingGoNe::strip
GamingGoNe::Exec
GamingGoNe::say
GamingGoNe::echo
GamingGoNe::set
GamingGoNe::cp
GamingGoNe::cpm
GamingGoNe::cprint
GamingGoNe::GetValueForKey
GamingGoNe::GetUserInfo
GamingGoNe::getStringCvar
GamingGoNe::getIntCvar
GamingGoNe::strlen
GamingGoNe::logWrite
GamingGoNe::playsound
GamingGoNe::include
GamingGoNe::getConfigString
GamingGoNe::registerCvar
GamingGoNe::setCvar
GamingGoNe::atoi

GM
gm::debug
gm::gmVersion
gm::typeId
gm::typeName

gm::typeRegisterOperator
gm::typeRegisterVariable
gm::sysCollectGarbage
gm::sysGetMemoryUsage
gm::sysSetDesiredMemoryUsageHard
gm::sysSetDesiredMemoryUsageSoft
gm::sysGetDesiredMemoryUsageHard
gm::sysGetDesiredMemoryUsageSoft
gm::sysSetDesiredMemoryUsageAuto
gm::sysGetStatsGCNumFullCollects
gm::sysGetStatsGCNumIncCollects
gm::sysGetStatsGCNumWarnings
gm::sysIsGCRunning
gm::sysTime
gm::doString
gm::globals
gm::threadTime
gm::threadId
gm::threadAllIds
gm::threadKill
gm::threadKillAll
gm::thread
gm::yield
gm::exit
gm::assert
gm::sleep
gm::signal
gm::block
gm::stateSet
gm::stateSetOnThread
gm::stateGet
gm::stateGetLast
gm::stateSetExitFunction
gm::tableCount
gm::tableDuplicate
gm::print
gm::format
sscanf
Brief: Allows you to have a ssanf like function
Param: char string
Param: char searchPattern
Param: int varType
Return: char scannedVar
search
Brief: Allows you to search a string in a string
Param: char searchIn
Param: char searchString

Return: char match
strip
Brief: removes the color code from a player/string
Param: char string
Return: char stripped_string
Exec
Brief: Exec will execute a system command
Param: string params will be concatinated together with a single space to form the final system command string
Return: integer value returned from system exec call, -1 on error
say
Brief: say will send a console say message
Param: char string
Return: null
echo
Brief: echo will echo a string to the serverconsole output
Param: char string
Return: null
set
Brief: set param will be concatinated and piped to the gameserver using EXEC_APPEND
Param: string
Return: null
cp
Brief: this function will send string as a Center Print message to 'clientnum'
Param: int clientNum
Param: char string
Return: null
cpm
Brief: this function will send string to clientnum in CPM mode
Param: int clientNum
Param: char string
Return: null
cprint
Brief: this function will send string to clientnum in print mode(console)
Param: int clientNum
Param: char string
Return: null
GetValueForKey
Brief: this function will get a certain value from the userinfo string, eg 'GetValueForKey(UserInfoString,"ip")'
Param: char userinfostring
Param: char value
Return: char value
GetUserInfo
Brief: this function will return a userinfostring
Param: char string
Return: char userinfostring
getStringCvar
Brief: this function will return the string of the cvar 'var_name'

Param: char string
Return: char cvar_value
getIntCvar
Brief: this function will return the integer of the cvar 'var_name'
Param: char string
Return: integer cvar_value
strlen
Brief: this function will return the length of the string
Param: char string
Return: int length
logWrite
Brief: this function will write 'string' in the game logfile
Param: char string
Return: null
playsound
Brief: this function will play the sound 'string' using the clientcommand mu_play
Param: char pathToSound
Return: null
include
Brief: This function will include a file with 'string' as the filename
Param: char fileName
Return: null
getConfigString
Brief: This function gets the config string of a client (CS_PLAYERS + clientNum)
Param: int clientNum
Return: null
registerCvar
Brief: This function registers a server Cvar
Param: char cvarName
Return: null
setCvar
Brief: This function sets a server Cvar
Param: char cvarName
Param: char Value
Return: null
atoi
Brief: This function converts a string to an integer
Param: char string
Return: integer
gm

Brief: functions in the gm lib are all global scope
debug
Brief: debug will cause a the debugger to break at this point while running.

gmVersion
Brief: gmVersion will return the gmMachine version string. version string is major type . minor type as a string and was added at version 1.1

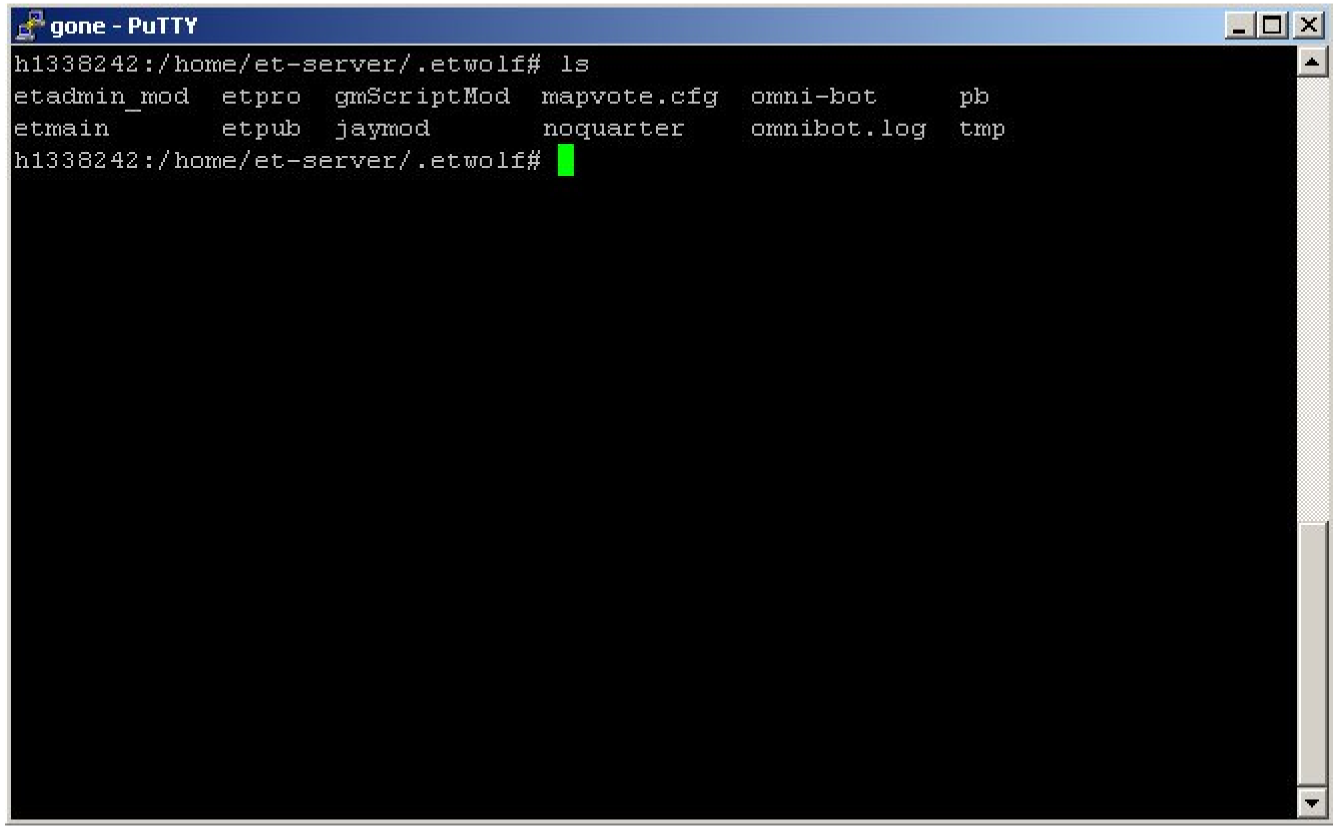
Return: string
typeId
Brief: typeId will return the type id of the passed var
Param: var
Return: integer type
typeName
Brief: typeName will return the type name of the passed var
Param: var
Return: string
typeRegisterOperator
Brief: typeRegisterOperator will register an operator for a type
Param: int typeid
Param: string operator name is one of "getdot", "setdot", "getind", "setind", "add", "sub", "mul", "div", "mod", "inc", "dec", "bitor", "bitxor", "bitand", "shiftleft", "shiftright", "bitinv", "lt", "gt", "lte", "gte", "eq", "neq", "neg", "pos", "not"
Param: function
Return: 1 on success, otherwise 0
typeRegisterVariable
Brief: typeRegisterVariable will register a variable with a type such that (type).varname will return the variable
Param: int typeid
Param: string var name
Param: var
Return: 1 on success, otherwise 0
sysCollectGarbage
Brief: sysCollectGarbage will run the garbage collector iff the current mem used is over the desired mem used
Param: forceFullCollect (false) Optionally perform full garbage collection immediately if garbage collection is not disabled.
Return: 1 if the gc was run, 0 otherwise
sysGetMemoryUsage
Brief: sysGetMemoryUsage will return the current memory used in bytes
Return: int memory usage
sysSetDesiredMemoryUsageHard
Brief: sysSetDesiredMemoryUsageHard will set the desired memory useage in bytes. when this is exceeded the garbage collector will be run.
Param: int desired mem usage in bytes
sysSetDesiredMemoryUsageSoft
Brief: sysSetDesiredMemoryUsageSoft will set the desired memory useage in bytes. when this is exceeded the garbage collector will be run.
Param: int desired mem usage in bytes
sysGetDesiredMemoryUsageHard
Brief: sysGetDesiredMemoryUsageHard will get the desired memory useage in bytes. Note that this value is used to start garbage collection, it is not a strict limit.
Return: int Desired memory usage in bytes.
sysGetDesiredMemoryUsageSoft
Brief: sysGetDesiredMemoryUsageSoft will get the desired memory useage in bytes. Note that this value is used to start garbage collection, it is not a strict limit.
Return: int Desired memory usage in bytes.
sysSetDesiredMemoryUsageAuto
Brief: sysSetDesiredMemoryUsageAuto will enable auto adjustment of the memory limit(s) for subsequent garbage collections.
Param: int enable or disable flag
sysGetStatsGCNumFullCollects
Brief: sysGetStatsGCNumFullCollects Return the number of times full garbage collection has occured.
Return: int Number of times full collect has occured.

sysGetStatsGCNumIncCollects
Brief: sysGetStatsGCNumIncCollects Return the number of times incremental garbage collection has occured. This number may increase in twos as the GC has multiple phases which appear as restarts.
Return: int Number of times incremental collect has occured.
sysGetStatsGCNumWarnings
Brief: sysGetStatsGCNumWarnings Return the number of warnings because the GC or VM thought the GC was poorly configured. If this number is large and growing rapidly, the GC soft and hard limits need to be configured better. Do not be concerned if this number grows slowly.
Return: int Number of warnings garbage collect has generated.
sysIsGCRunning
Brief: Returns true if GC is running a cycle.
sysTime
Brief: sysTime will return the machine time in milli seconds
Return: int
doString
Brief: doString will execute the passed gm script
Param: string script
Param: int optional (1) set as true and the string will execute before returning to this thread
Param: ref optional (null) set 'this'
Return: int thread id of thread created for string execution
globals
Brief: globals will return the globals table
Return: table containing all global variables
threadTime
Brief: threadTime will return the thread execution time in milliseconds
Return: int
threadId
Brief: threadId will return the thread id of the current executing script
Return: int
threadAllIds
Brief: threadIds returns a table of thread Ids
Return: table of thread Ids
threadKill
Brief: threadKill will kill the thread with the given id
Param: int threadId optional (0) will kill this thread
threadKillAll
Brief: threadKillAll will kill all the threads except the current one
Param: bool optional (false) will kill this thread if true
thread
Brief: thread will start a new thread
Param: function entry point of the thread
Param: ... parameters to pass to the entry function
Return: int threadid
yield
Brief: yield will hand execution control to the next thread
exit
Brief: exit will kill this thread
assert
Brief: assert

Param: int expression if true, will do nothing, if false, will cause an exception
sleep
Brief: sleep will sleep this thread for the given number of seconds
Param: int\float seconds
signal
Brief: signal will signal the given variable, this will unblock dest threads that are blocked on the same variable.
Param: var
Param: int destThreadId optional (0) 0 will signal all threads
block
Brief: block will block on all passed vars, execution will halt until another thread signals one of the block variables. Will yield on null and return null.
Param: ... vars
Return: the unblocking var
stateSet
Brief: stateSet will collapse the stack to nothing, and push the passed functions.
Param: function new state function to execute
Param: ... params for new state function
stateSetOnThread
Brief: stateSetOnThread will collapse the stack of the given thread id to nothing, and push the passed functions.
Param: int thread id
Param: function new state function to execute
Param: ... params for new state function
stateGet
Brief: stateGet will return the function on the bottom of this threads execution stack iff it was pushed using stateSet
Param: a_threadId Optional Id of thread to get state on. \reutrn function \ null
stateGetLast
Brief: stateGetLast will return the last state function of this thread
Param: a_threadId Optional Id of thread to get last state on. \reutrn function \ null
stateSetExitFunction
Brief: stateSetExitFunction will set an exit function for this state, that will be called with no parameters if this thread switches state
Param: function
tableCount
Brief: tableCount will return the number of elements in a table object
Param: table
Return: int
tableDuplicate
Brief: tableDuplicate will duplicate the passed table object
Param: table
Return: table
print
Brief: print will print the given vars to the print handler. passed strings are concatinated together with a seperating space.
Param: ... strings
format
Brief: format (like sprintf, but returns a string) %d, %s, %f, %c, %b, %x, %e
Param: string

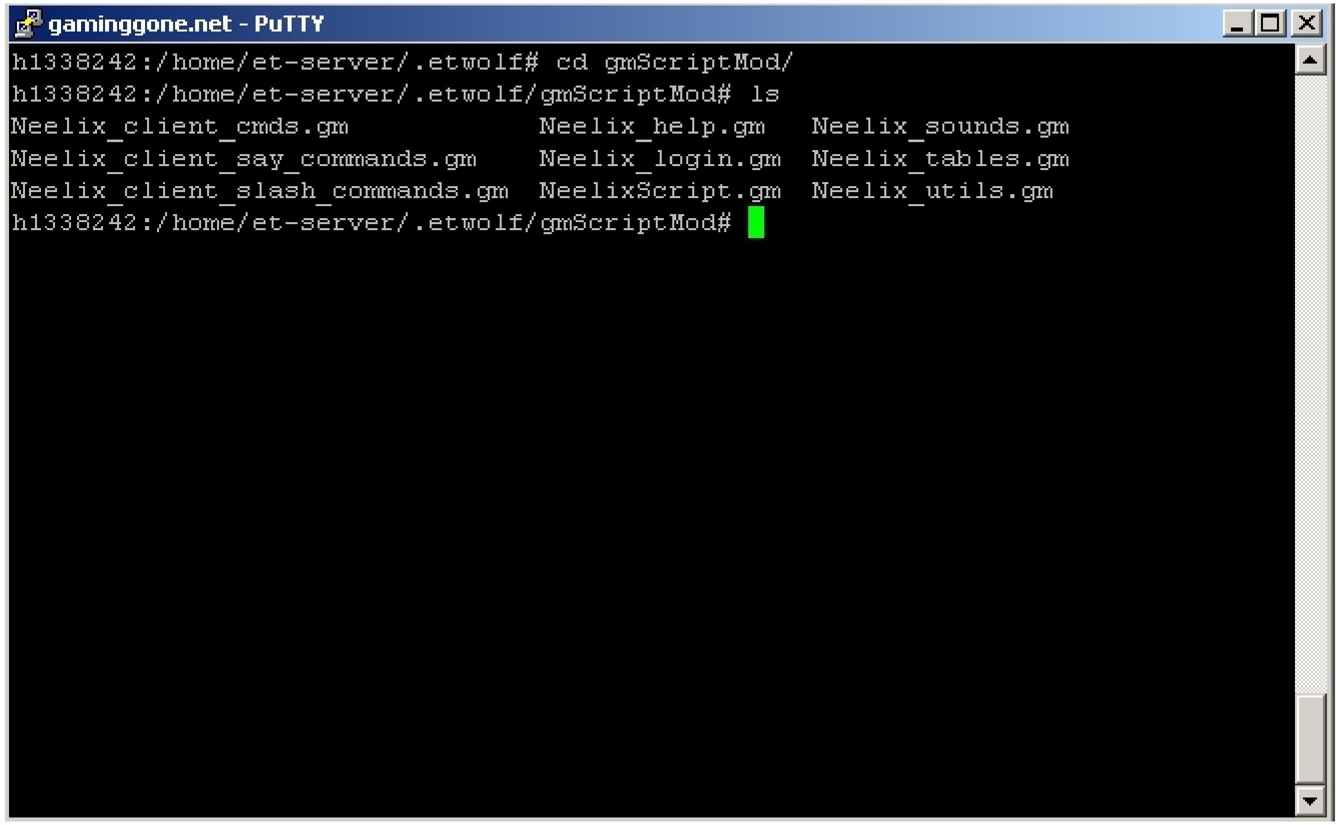
Step by Step Help

1. Go to your .etwolf folder and create a new folder called: "gmScriptMod"
!ATTENTION! The Folder Name is case sensitiv!



```
h1338242:/home/et-server/.etwolf# ls
etadmin_mod  etpro  gmScriptMod  mapvote.cfg  omni-bot     pb
etmain       etpub  jaymod       noquarter    omnibot.log  tmp
h1338242:/home/et-server/.etwolf#
```

2. Save in the gmScriptMod folder the .gm files.

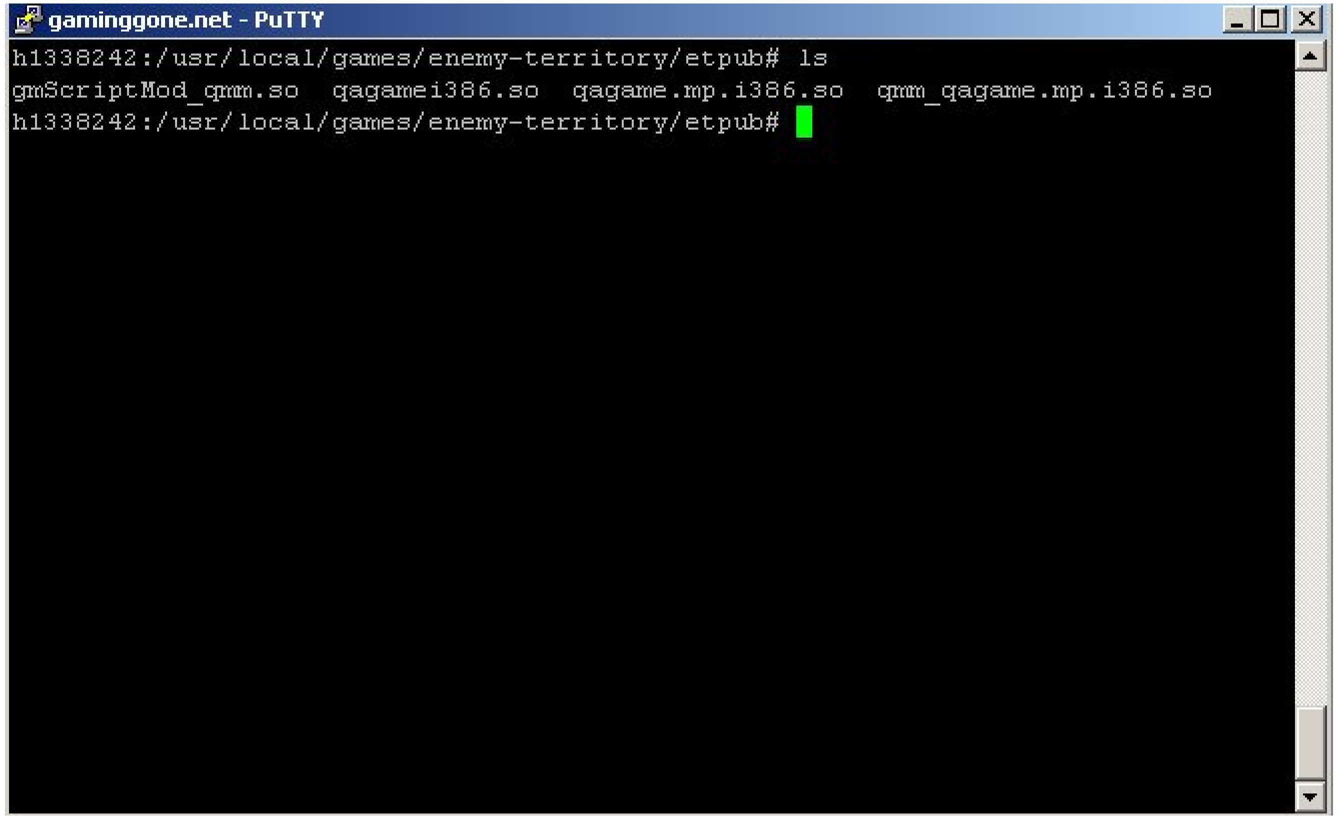


```
h1338242:/home/et-server/.etwolf# cd gmScriptMod/
h1338242:/home/et-server/.etwolf/gmScriptMod# ls
Neelix_client_cmds.gm            Neelix_help.gm   Neelix_sounds.gm
Neelix_client_say_commands.gm    Neelix_login.gm  Neelix_tables.gm
Neelix_client_slash_commands.gm  NeelixScript.gm  Neelix_utils.gm
h1338242:/home/et-server/.etwolf/gmScriptMod# 
```

3. Change the Default Passwords in the NeelixScript.gm File.

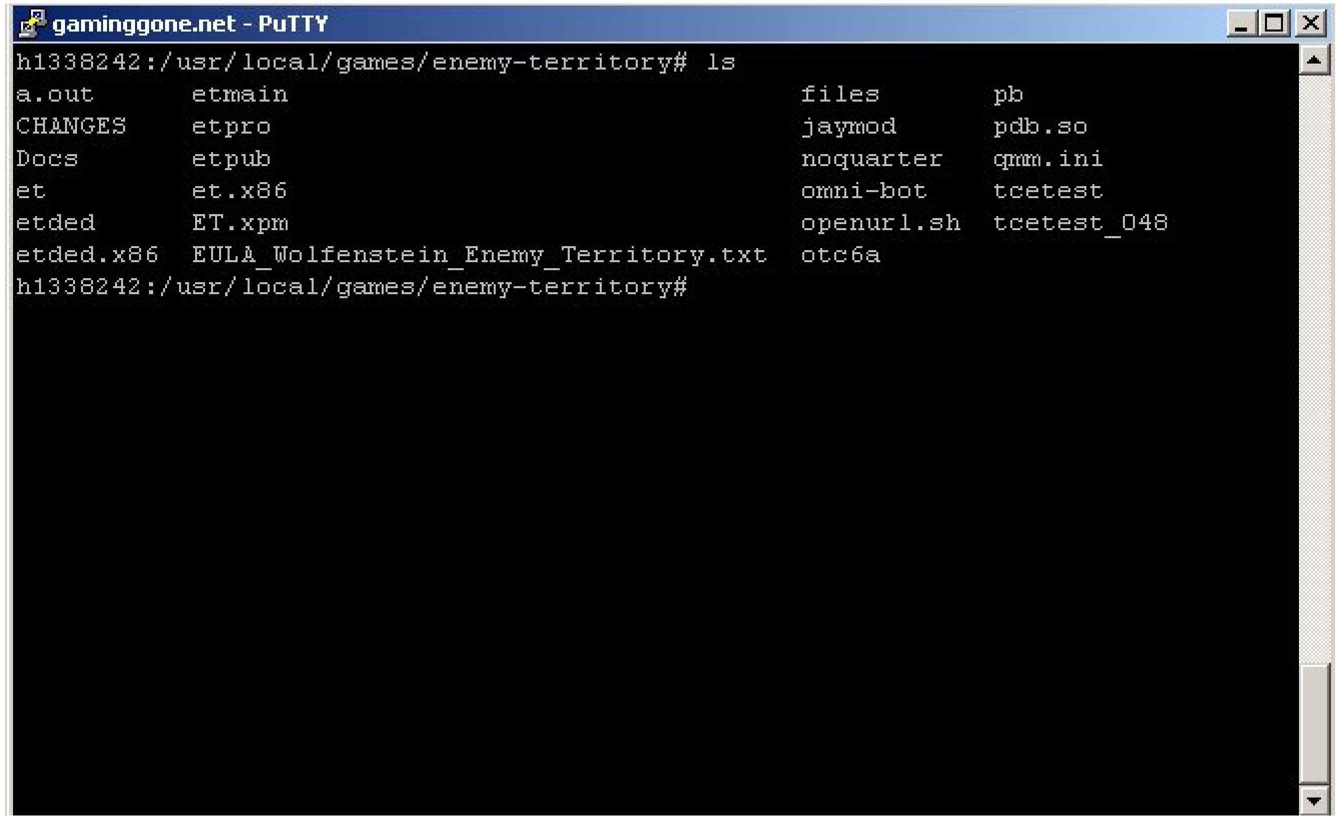
4. Install QMM

4. a) Rename the existing qagame.mp.i386.so file to qmm_qagame.mp.i386.so.
4. b) Place qmm.so from the .tar.gz you downloaded into the mod directory and rename it to qagame.mp.i386.so.
Your Folder should now look like this:



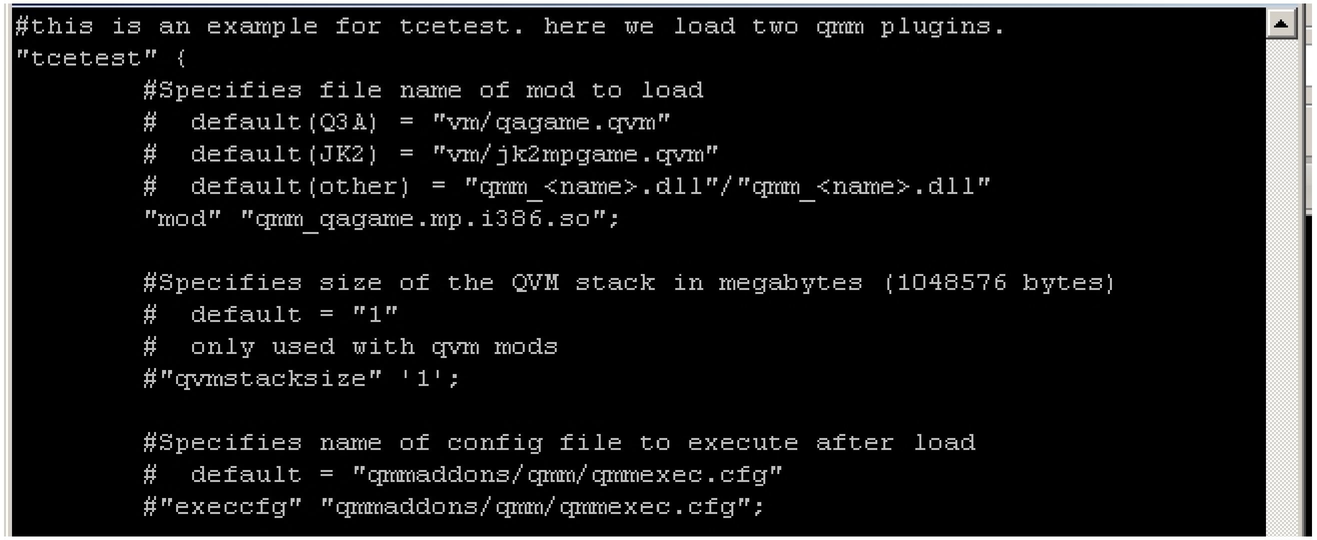
```
h1338242:/usr/local/games/enemy-territory/etpub# ls
gmScriptMod_qmm.so  qagamei386.so  qagame.mp.i386.so  qmm_qagame.mp.i386.so
h1338242:/usr/local/games/enemy-territory/etpub#
```

4. c) Place pdb.so and qmm.ini from the .zip you downloaded into the root game directory (where the server binary is located).



```
h1338242:/usr/local/games/enemy-territory# ls
a.out       etmain                                   files       pb
CHANGES     etpro                                    jaymod      pdb.so
Docs        etpub                                    noquarter   qmm.ini
et          et.x86                                   omni-bot    tcetest
etded       ET.xpm                                   openurl.sh  tcetest_048
etded.x86   EULA_Wolfenstein_Enemy_Territory.txt     otc6a
h1338242:/usr/local/games/enemy-territory#
```

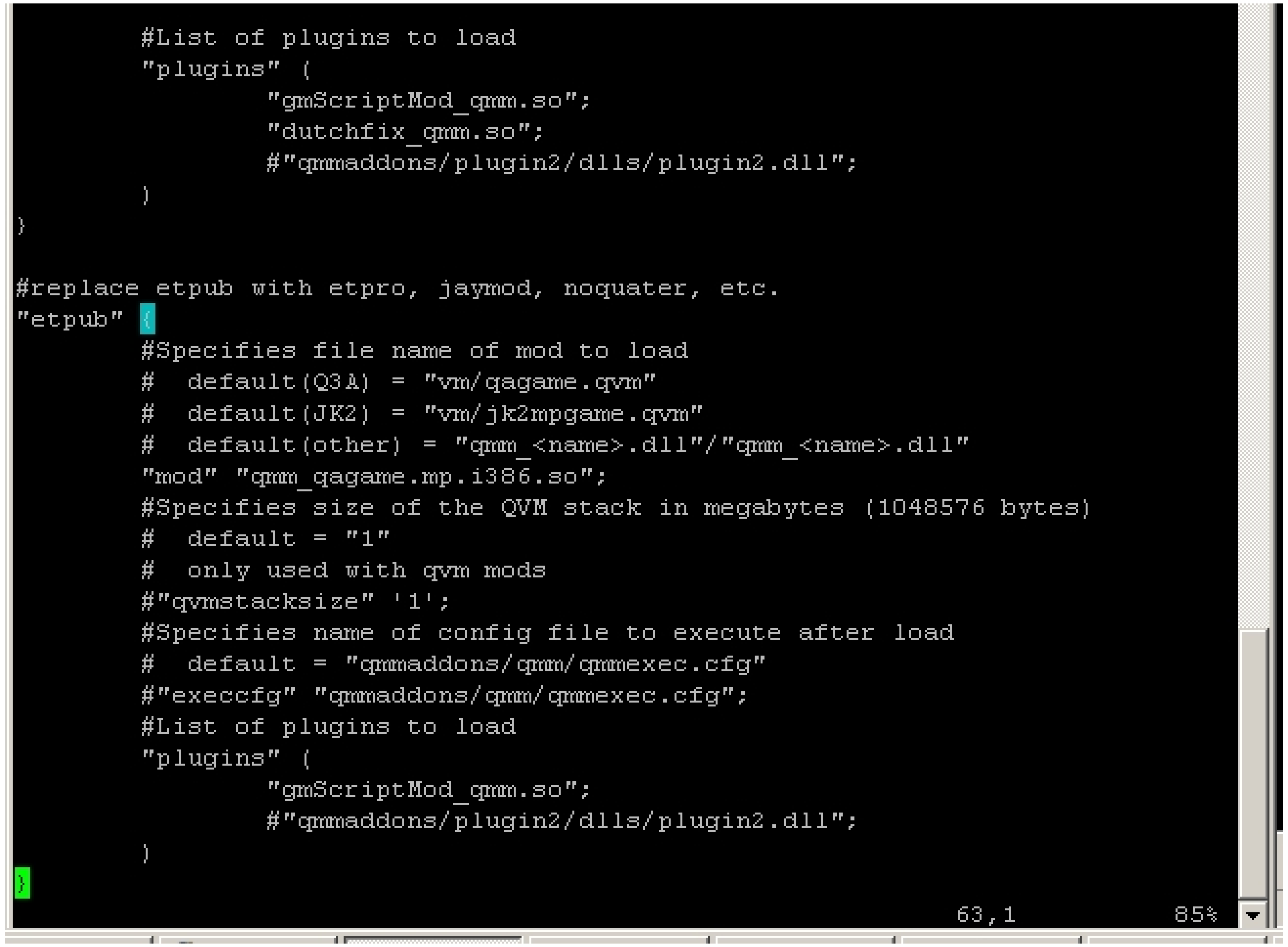
4. d) Configure QMM (see Configuration File (qmm.ini) regarding a setting that must be set for Enemy Territory).



```
#this is an example for tcetest. here we load two qmm plugins.
"tcetest" {
        #Specifies file name of mod to load
        #  default(Q3A) = "vm/qagame.qvm"
        #  default(JK2) = "vm/jk2mpgame.qvm"
        #  default(other) = "qmm_<name>.dll"/"qmm_<name>.dll"
        "mod" "qmm_qagame.mp.i386.so";

        #Specifies size of the QVM stack in megabytes (1048576 bytes)
        #  default = "1"
        #  only used with qvm mods
        #"qvmstacksize" '1';

        #Specifies name of config file to execute after load
        #  default = "qmmaddons/qmm/qmmexec.cfg"
        #"execcfg" "qmmaddons/qmm/qmmexec.cfg";
```



```
        #List of plugins to load
        "plugins" (
                "gmScriptMod_qmm.so";
                "dutchfix_qmm.so";
                #"qmmaddons/plugin2/dlls/plugin2.dll";
        )
}

#replace etpub with etpro, jaymod, noquater, etc.
"etpub" {
        #Specifies file name of mod to load
        #  default(Q3A) = "vm/qagame.qvm"
        #  default(JK2) = "vm/jk2mpgame.qvm"
        #  default(other) = "qmm_<name>.dll"/"qmm_<name>.dll"
        "mod" "qmm_qagame.mp.i386.so";
        #Specifies size of the QVM stack in megabytes (1048576 bytes)
        #  default = "1"
        #  only used with qvm mods
        #"qvmstacksize" '1';
        #Specifies name of config file to execute after load
        #  default = "qmmaddons/qmm/qmmexec.cfg"
        #"execcfg" "qmmaddons/qmm/qmmexec.cfg";
        #List of plugins to load
        "plugins" (
                "gmScriptMod_qmm.so";
                #"qmmaddons/plugin2/dlls/plugin2.dll";
        )
}
                                                                  63,1          85%
```

Source