




Table of Contents

What This Shortcut Will Cover	3
A Brief History of Cheating	4
Defeating Piracy by Going Online	5
Or Not.....	6
The Lawyers Have Landed Bearing EULAs	7
The Rise of MMORPGs	9
The WoW Warden Is Watching	9
Cheating Is Quick and Easy	13
Grinding Is Boring and Dull	13
Farming Makes Things Easy	14
Virtual-World Economics	25
Farming Hurts the Virtual Economy	27
Games as Reality	28
Cracking Down on Farming	29
Online Game, Real-World Cheating	30
Defeating Cheaters and Crossing the Line	31
The Governor Watches the Watcher	32
Where Do You Stand?	43
The Tip of the Online Gaming Iceberg	44

Cheating Online Games

Gary McGraw and Greg Hoglund

Cheating Online Games, an original Addison-Wesley Digital Short Cut, contains information that will appear in ***Exploiting Online Games*** (0132271915).



Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this work, and the publisher was aware of a trademark claim, the designations have been printed with initial capital letters or in all capitals.

The author(s) and publisher have taken care in the preparation of this work, but make no expressed or implied warranty of any kind and assume no responsibility for errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of the use of the information or programs contained herein.

Visit us on the Web: www.awprofessional.com

Copyright © 2007 Pearson Education, Inc.

All rights reserved. This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. For information regarding permissions, write to:

Pearson Education, Inc.
Rights and Contracts Department
One Lake Street
Upper Saddle River, NJ 07458
United States of America
Fax: (201)236-3290

ISBN 0-321-46072-3

First release, July 2006

Cheating Is Quick and Easy

Are backdoor programs, rootkits, and invasive scanners really necessary for piracy detection? When does such a countermeasure cross the line between legitimate copy protection and invasion of privacy?

Consider, for example, that Blizzard's Warden scanning program is not really used for detecting pirates—it is really used to detect “cheaters.” Blizzard, of course, gets to define precisely what “cheating” entails—it is their game after all. Among other things, cheating in this case means any use of software automation tools to play the game.

Suppose, for example, that the game has three buttons (A, B, and C) and you are only allowed to click these manually yourself. If you were to install a software automation tool (such as a QA testing tool) that automatically clicks X/Y mouse coordinates to drive these buttons, you would be cheating by Blizzard's definition. As you can see, EULA allowances and their associated enforcement mechanisms are not only being used to curb piracy but also *to restrict how you use the software*.

Grinding Is Boring and Dull

Using invasive technology such as a rootkit to control someone's use of a game seems rather extreme until you consider the economic impact of automated game play. In most cases, automated game play is realized using special tools and scripts typically referred to as *macros*. For example, in *WoW*, monsters appear at specific locations on a periodic basis. You can easily write a macro that causes the in-game character to stand in that location and automatically kill the monster every time it appears (thus gaining experience points and virtual gold). Of course, you can

do this manually yourself, waiting around all day for the monster to appear; given that the monster only appears every 10 minutes, however, that plan will commit you to a long and boring night. Why not write a macro to wait around for you? Ultimately, the question is, how can automating such a boring and repetitive activity be considered cheating?

WoW and many MMORPGs like it are so afflicted with boring repetitious game play that players have invented a term to describe it: *grinding*. That is, doing awful repetitive things all day with your character just to gain experience is likened to a mule going around and around on a treadmill, grinding grain into flour day in and day out. For some reason, players *enjoy* this self-inflicted misery and will pay \$14 a month for the privilege of doing it. Why?

As it turns out, a deep-seated human psychology is at play here, and it has to do with living a double life, and the fact that grinding away like this brings economic reward. Whenever you kill that monster, it drops in-game play money and gives you other rewards, such as more experience, skills, and ultimately levels.

Farming Makes Things Easy

If you write a macro to do this grinding for you, you can go away to work, or sleep, and come back later and have the sum of all gold pieces and experience for all the repetitive boring monster kills waiting for you. Thus, the macro earns you in-game money and simultaneously increases your character's power—but without the associated boredom of actually paying attention. What a great idea! It is so great, in fact, that thousands of players do it all the time. There is even a special term used to describe players who play this way: They are called *farmers*.

SECTION 10

Farming Makes Things Easy

Here is a simple grinding/farming macro (called hoglund's WoW_Agro Macro) designed to be used with the tool called ACTool—a popular (and free) macro writing program. This macro controls a character who camps out waiting for monsters to appear and kills them when they do. We interleave commentary with the code so that you can understand what is happening.

```
// _ _ _ _ _
// hoglund's WoW_Agro Macro
// _ _ _ _ _
//RESOLUTION: 1024x768
//PUT ALL FILES IN YOUR ACTOOL\MACROS FOLDER.

SetActiveWindow World of Warcraft
delay 4 sec

// put all your 'globals' here
Constants
    gPCHealth = "NoValue"           // the current health of the
                                   // PC ( HIGH | MEDIUM | LOW | CRITICAL )
    gMobHealth = "NoValue"          // the current health of the
                                   // current targetted mob ( HIGH | MEDIUM | LOW | CRITICAL )
    gPCPosture = "Standing"         // current/starting posture of the PC character
    gMachineState = "START"         // global machine state, core of the system
    gSelectedTarget = "NoTarget"
    pc_name = xanier
```

SECTION 10

Farming Makes Things Easy

The first part of the code sets up global variables, most importantly the currently selected target, the name of the character being automated, and the current state of the state machine. This script implements a classic architecture that many macros follow—that is, it is designed to operate as a state machine. This means that the script maintains a single “state” variable while automating the character. States can have mnemonic labels such as “attacking,” “healing,” or “running away for dear life.” Only one state can be active at any one time, and rules apply to transitioning from one state to another.

```
// hot keys
keyExecAttack = 1           // set your F1 key to attack before using this MACRO
keyExecPickup = {F2}
```

The preceding part of the script indicates which keys must be used to automate the game. When the F2 key is pressed, the character picks up items, and when the number 1 is pressed, the character attacks. This kind of key binding is common in macros such as this. Simple macros operate the software using only keystroke and mouse movement—nothing invasive exploits the game software directly. Advanced methods of cheating are nowhere near as simple.

```
// screen coordinates
coord_MobHPMin           = 262, 50
coord_MobHPFull          = 370, 50

coord_SafeHP             = 200, 50 //if green at or above this mark, you're fine
coord_HalfHP             = 146, 50 //half hp if lower than this mark
coord_LowHP              = 116, 50 //low hp if at or lower than this mark
coord_DeadHP             = 96, 50  //you're dead :(
```

SECTION 10

Farming Makes Things Easy

```
coord_temp                = 0,0

// colors
color_AliveGreen          = 150
color_AttackRed           = 147

// temp stuff
TempTarget                = NoValue //holds the value for target before placing into list
tCount                    = 1        //used to traverse the targetList index
Total                     = 1        //used to find the ListCount
redDifference              = 0
blueDifference             = 0
greenDifference            = 0
numAttacks                = 0
Returned                  = 0
aCount                    = 0
```

End

The preceding section of the script is quite fascinating. It designates X/Y coordinates on the screen. Furthermore, it designates exact color thresholds for pixels. This information is used to sample pixels on the screen at precise X/Y coordinates. The pixels in question happen to correspond to locations where health and magical power display onscreen for the game user. This information, along with the color threshold, is then used to determine whether the character is at full