



An Analysis of the (In)Security State of the GameHouse Game Installation Mechanism

Carsten Eiram
Chief Research Officer
(Twitter: @CarstenEiram & @RiskBased)



Table of Contents

<u>About Risk Based Security</u>	3
<u>Company History</u>	3
<u>Solutions</u>	3
<u>Preface</u>	4
<u>Introduction</u>	5
<u>The GameHouse Installer</u>	6
<u>Unsafe Methods Provided By ActiveX Controls</u>	7
<u>Use-after-free Vulnerability in RACInstaller.StateCtrl.1 ActiveX Control</u>	8
<u>Unsafe Default Directory Permissions</u>	10
<u>RGI and RGUNINST File Types</u>	11
<u>RGI Files</u>	11
<u>RGUNINST Files</u>	13
<u>Conclusion</u>	14
<u>Timeline</u>	15

About Risk Based Security

Risk Based Security offers clients fully integrated security solutions, combining real-time vulnerability and threat data, as well as the analytical resources to understand the implications of the data, resulting in not just security, but the right security.

Company History

Risk Based Security, Inc. (RBS) was established in early 2011 to better support the many users and initiatives of the Open Security Foundation - including the OSVDB and DataLossDB projects. RBS was created to transform this wealth of security data into actionable information by enhancing the research available, and providing a first of its kind risk identification and evidence-based security management service.

As a data driven and vendor neutral organization, RBS is able to deliver focused security solutions that are timely, cost effective, and built to address the specific threats and vulnerabilities most relevant to the organizations we serve. We not only maintain vulnerability and data breach databases, we also use this information to inform our entire practice.

Solutions

Cyber Risk Analytics - Extensive data breach database including interactive dashboards and breach analytics. Clients are able to gather and analyze security threat and data breach information on businesses, industries, geographies, and causes of loss.

VulnDB - Vulnerability intelligence, alerting, and third party library monitoring and tracking based on the largest and most comprehensive vulnerability database available today. Ability to integrate with products and services via an API or custom export.

YourCISO - Revolutionary service that provides organizations an affordable security solution including policies, vulnerability scans, awareness material, incident response, and access to high quality information security resources and consulting services.

Security Development Lifecycle (SDL) - Consulting, auditing, analysis, and independent verification specifically specialized in breaking code, which in turn greatly increases the security of software products.

Security Program Assessment Services / ISO/IEC 27001:2005 - Customized training, security assessments, security program audits, and gap analysis as well as pre-certification consulting services to both protect organizations with best practice security controls and to prepare for a smooth ISO/IEC 27001:2005 certification audit.

Preface

Recently, there have been a number of reports about vulnerabilities in game clients/installers due to unsafe design.

The most recent one, “EA Origin Insecurity”¹, was published by security researchers Luigi Auriemma and Donato Ferrante of ReVuln in March 2013 and discussed how the origin:// URI handler in the Origin client from EA could be used to exploit issues in installed games that would otherwise only be locally exploitable. Prior to this report, the researchers from ReVuln published an analysis in October 2012, “Steam Browser Protocol Insecurity”², about issues in the popular Steam client.

July 2012, Tavis Ormandy reported³ a vulnerability in the Ubisoft Uplay client within a bundled ActiveX component, allowing execution of arbitrary commands on a user’s system. This vulnerability was patched⁴ in version 2.0.4 of the client.

Going back about two years to April 2011, security researcher rgod reported multiple⁵ vulnerabilities⁶ in ActiveX components provided by RealNetworks GameHouse. While the vendor never provided statements about the vulnerabilities being fixed, one would expect that GameHouse had at some point addressed the vulnerabilities within a two year period.

When Risk Based Security in January 2013, therefore, encountered the latest version of the RealArcade installer provided by GameHouse on a system during an audit, it seemed appropriate to determine the status of the previously disclosed vulnerabilities and perform a more thorough analysis of its design.

The results were hardly impressive...

¹ http://revuln.com/files/ReVuln_EA_Origin_Insecurity.pdf

² http://revuln.com/files/ReVuln_Steam_Browser_Protocol_Insecurity.pdf

³ <http://seclists.org/fulldisclosure/2012/Jul/375>

⁴ <http://forums.ubi.com/showthread.php/699940-Uplay-PC-Patch-2-0-4-Security-fix>

⁵ http://retrogod.altervista.org/9sg_StubbyUtil.ShellCtl.1_adv.html

⁶ http://retrogod.altervista.org/9sg_StubbyUtil.ProcessMgr.1_adv.html

Introduction

GameHouse is the games division brand of RealNetworks and has previously been known by various names; the most recognized being RealGames and RealArcade.



Fig. 1: GameHouse website

When visiting the main GameHouse website⁷, users are welcomed by the following comforting message:

We practice safe gaming.

All of our game downloads are 100% safe and free of viruses. You can quickly and safely download games to play now or later, even if you are offline.

⁷ <http://www.gamehouse.com/>

Unfortunately for users, while GameHouse may be practising safe gaming, the implementation does not practise safe installation of these games. The installer provided for Windows suffers from serious design flaws and very basic, but critical vulnerabilities - many of which have been publicly known for more than two years.

This paper describes these flaws in the GameHouse game installer implementation for Windows, and how it exposes users' systems to both local and remote attacks. The most severe of the covered vulnerabilities cause any system with the GameHouse installer to be wide open to attack, allowing websites to execute arbitrary commands on the system with the user's permissions.

The GameHouse Installer

GameHouse does not provide a client similar to e.g. Steam and Origin for installation of games. Instead, users visit the GameHouse website from where games are purchased and downloaded.

The first time a user downloads a game from the GameHouse website, the user does not download the game, rather a small executable, containing the main GameHouse installer and details for the installer to identify and download the desired game. On any subsequent visits, attempts to download a game from the website invokes the installer.

The executable installs the installer to the “%ProgramFiles%\RealArcade\Installer” directory. The latest version of the installer at the time of analysis is 2.6.0.481, and some of the most relevant components are:

Uninstall folder

Each time a game is installed on the system, an uninstall script with a RGUNINST file extension is added to this directory. RGUNINST files are described later in this paper.

CheckInst.dll

The `CheckInst.dll` library provides the `InstallerState` ActiveX control, which exposes a single method, `GameInstallerVersion()`, for determining which version of the GameHouse installer is currently on a user's system. This ActiveX control is used by the GameHouse website to confirm that the latest version of the installer is present. As the ActiveX control is marked “safe” and is not `SiteLocked`⁸, it can also be instantiated by any other site to determine the installed version.

⁸ <http://blogs.msdn.com/b/ie/archive/2007/09/18/developing-safer-activex-controls-using-the-sitelock-template.aspx>

[gameinstaller.exe](#)

This is the main executable and is associated with files with the RGI and RGUNINST extensions. It is basically just a Lua 5.0 interpreter loading `installerMain.clf`, which is a compiled Lua file containing the main functionality for parsing associated file types.

[InstallerDlg.dll](#)

The main component providing installation features and serves two purposes. Firstly, it provides the functionality used by `installerMain.clf` when performing operations on the system based on information and commands in RGI and RGUNINST files.

Secondly, it provides three “safe” ActiveX controls (ShellCtl, SlideState, and ProcessMgr) to allow smooth interaction with the GameHouse.com website, when users purchase and download games. Instead of requiring users to download a file, a user simply clicks a download button on the page, invoking the installer to automatically download and install the game.

While the purpose of these ActiveX controls is to manage downloads from the GameHouse website, they are, however, not SiteLocked. This allows any website to repurpose them and make use of the very powerful functionality being provided, which can only be considered extremely unsafe to implement in “safe” ActiveX controls.

Unsafe Methods Provided By ActiveX Controls

April 2011, rgod published advisories for two of the “safe” ActiveX controls in `InstallerDlg.dll` as part of version 2.6.0.445 of the GameHouse installer. The advisories described a number of unsafe methods allowing command execution while also briefly mentioning that other methods could allow information disclosure and file deletion.

When checking the GameHouse Installer almost two years later as part of this analysis, these old vulnerabilities were still present in the latest version (2.6.0.481). Evidently, GameHouse had for two years failed to address these publicly known vulnerabilities despite the most critical allowing remote code execution in a reliable manner.

The following lists the exposed unsafe methods:

Method	Functionality
CreateShortcut	Create fully controlled shortcut on desktop
DeleteShortcut	Delete arbitrary files (not only shortcuts) on system
SetWorkingDir	Change the working directory to arbitrary directory (useful in combination with other methods)
GetWorkingDir	Get working directory (potentially disclosing username in path)
ShellExec	Execute arbitrary file
ShellExecRunAs	Execute arbitrary file, but prompting RunAs

PlatformInfo	Obtain platform details
CopyDocument	Copy file to/from user's system
DeleteOnReboot	Delete arbitrary file on reboot
KillProcess	Terminate arbitrary running process by name
CreateLUAProcess	Run arbitrary commands as a separate process
Exec	Execute arbitrary commands
ExecLow	Execute arbitrary commands

Fig. 2: Unsafe ActiveX control methods

As most of these were mentioned in the advisories by rgod in 2011 and accompanied by clear examples, they will not be discussed further.

Risk Based Security contacted RealNetworks and GameHouse to ensure they were aware of these vulnerabilities, and that they would be addressed. While not responsive, GameHouse rolled out an update of their site and provided a new version (3.0.7) of the game installer in May 2013, which no longer marks these ActiveX controls as safe-for-scripting.

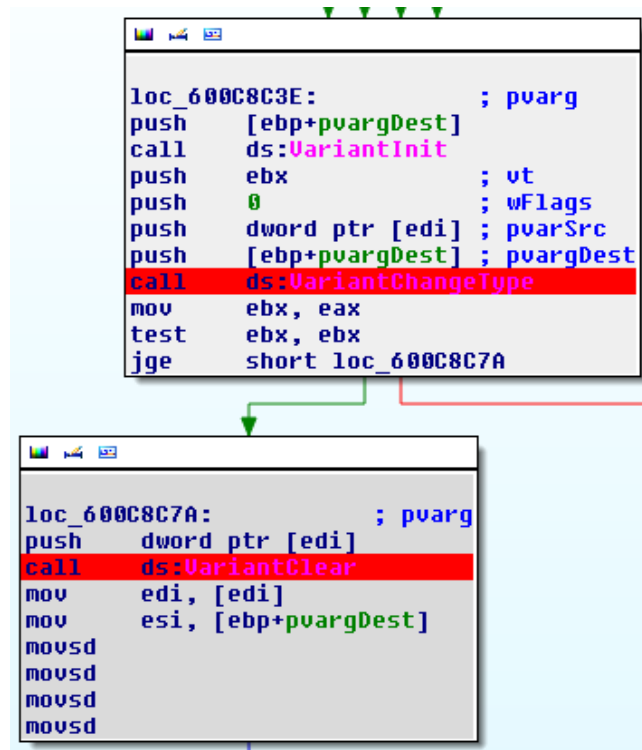
Use-after-free Vulnerability in RACInstaller.StateCtrl.1 ActiveX Control

The RACInstaller.StateCtrl.1 ActiveX control, {C8F76629-E4F4-4646-AFC0-665082D167B1}, provided by `InstallerDlg.dll` uses its own dispatching functionality called from the `InvokeDispatchWithNoThis()` function in `mshtml.dll`. Instead of dismissing invalid parameter types supplied to methods and properties, the dispatcher attempts to convert these into the correct type by changing the variant type. This behaviour makes it possible to trigger a use-after-free condition that allows arbitrary code execution.

When called, the dispatcher checks the number of arguments supplied before determining which method was called or property specified. If either of the methods listed below are invoked, the function determines if the supplied argument is a string by confirming if the variant type is `VT_BSTR`.

```
* QueueRemove ()
* QueuePause ()
* QueueTop ()
* Ping ()
* message ()
* AddTag ()
* RemoveTag ()
* TagRemoved ()
```


Instead of dismissing other input types as invalid and returning an error, as is customary, a function is called to convert the variant type. This is done by eventually calling `VariantChangeType()` after which `VariantClear()` is called to clear the originally supplied variant. In case the variant was an object, this is released, causing its reference count to be erroneously decremented, which may later cause the object to be prematurely freed.



```
loc_600C8C3E:          ; pvarg
push    [ebp+pvargDest]
call    ds:VariantInit
push    ebx             ; vt
push    0              ; wFlags
push    dword ptr [edi] ; pvarSrc
push    [ebp+pvargDest] ; pvargDest
call    ds:VariantChangeType
mov     ebx, eax
test    ebx, ebx
jge     short loc_600C8C7A

loc_600C8C7A:          ; pvarg
push    dword ptr [edi]
call    ds:VariantClear
mov     edi, [edi]
mov     esi, [ebp+pvargDest]
movsd
movsd
movsd
movsd
```

Fig. 3: Erroneously decrementing object reference count, leading to premature destruction of the object.

On way of triggering this is by supplying the 'window' object to one of the affected methods e.g. `Ping()`. Calling it multiple time, it is possible to decrement the `CComWindowProxy` object's reference count to zero, causing it to be prematurely freed and then later dereference the memory.

Version 3.0.7 of the game installer as released in May 2013, no longer marks the ActiveX control as safe-for-scripting. The vulnerability is covered by CVE-2013-2603⁹ and OSVDB ID 96919¹⁰.

⁹ <http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2013-2603>

¹⁰ <http://osvdb.org/96919>

Unsafe Default Directory Permissions

When installing games, the default proposed installation directory is “%HOMEDRIVE%\GameHouse Games\”. While it may be common knowledge in the IT security industry that installing outside %ProgramFiles% may open applications up to trojan-style attacks due to unsafe default permissions, most regular users are not aware of this and are prone to select whichever installation directory is suggested by the installer.

Any application installed within %ProgramFiles% inherits permissions that only grant members of the “Users” group permission to “Read & Execute”, “List Folder Contents”, and “Read”. However, installing outside %ProgramFiles% as suggested by the GameHouse installer, causes the installation directory to inherit permissions that grant members of the “Users” group two additional special permissions, specifically:

- Create Files / Write Data
- Create Folders / Append Data

While unprivileged users are not permitted to manipulate or delete existing files, any unprivileged user may create arbitrary files within the installation directory, unless the installer specifically changes the default permissions. The GameHouse installer does not, allowing creation of arbitrary files within the “GameHouse Games” folder and subfolders.

The combination of unsafe permissions, allowing unprivileged users to create arbitrary files, and Windows’ search order, which specifically looks for libraries loaded with relative paths within the application directory before other directories, may allow any user on the system to execute arbitrary code with the privileges of another user running a game.

This attack has been successfully demonstrated with the game “Zuma Deluxe”. By placing a library named `DDRAW.DLL` containing “malicious” code in the “%HomeDir%\GameHouse Games\Zuma Deluxe” directory, it was possible for an unprivileged, local user to cause this code to be executed with another user’s privileges, when that user attempted to run the game. This allows for privilege escalation.

GameHouse has not addressed this vulnerability in the latest version (3.0.7). Users are encouraged to specify an installation path within %ProgramFiles% instead of opting for the default suggested path. The issue is covered by CVE-2013-2604¹¹ and OSVDB ID 96918¹².

¹¹ <http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2013-2604>

¹² <http://osvdb.org/96918>

RGI and RGUNINST File Types

By default, files with the extensions RGI and RGUNINST are associated with `gameinstaller.exe`, which in turn executes `installerMain.clf`. These files can be double-clicked to run and while not necessarily a vulnerability, the file types do pose a security concern, and users should consider them unsafe. RGI files allow installing arbitrary applications from GameHouse and potentially arbitrary code execution, while RGUNINST files allow deleting arbitrary files, directories, and registry data on a user's system without further interaction.

It should be noted that similar concerns¹³ about arbitrary file deletion were raised by researcher Luigi Auriemma in 2005 for RealArcade, the precursor to GameHouse, when handling RGP (RealArcade Game Package) files. This specific file format is, however, not supported by the latest version of the GameHouse RealArcade Installer.

As users may not be aware that these file types are unsafe, it is problematic that GameHouse associates RGI and RGUNINST files with `gameinstaller.exe` in a manner in which they are executed without warning when simply double-clicked by a user. At a minimum, GameHouse should ensure a warning dialog is displayed, clearly informing users about the impact of running an RGI or RGUNINST file.

RGI Files

RGI files are installation files in an XML format containing information required to download a specific game from GameHouse.com. When an RGI file is opened, an HTTP HEAD request is sent to `logging.gamehouse.com/arcade/sites.html` with some of the details in the file and system specific information (e.g. platform, architecture, locale) as parameters.

An HTTP GET request is then sent to `switchboard.real.com/arcade/feeds.html`, also providing some of the information in the RGI file as parameters. Requests are redirected to `install.real.com/bundleInstall.packages`, supplying the initially sent information from the RGI file as parameters. This page responds with another RGI file named `gameInstall.rgi`, which contains the full installation details.

It should be noted that all communication is over HTTP and not HTTPS. Intercepting and manipulating traffic would, therefore, be trivial for any MitM (Man-in-the-Middle) attacker, allowing installation of malicious files and execution of arbitrary commands by manipulating the contents of the returned `gameInstall.rgi` file.

¹³ <http://osvdb.org/show/osvdb/13653>

Your game is currently being transfered from our secure servers to your computer.



When the download is finished the install screen will open and you can install your game.

Fig. 4: The definition of "secure servers" is debatable...

RGI files support the `<Payload>` tag, which allows specifying a destination path ("Target" parameter) to download a file to from a specified source path ("Src" parameter). RGI files also support a `<ConditionDef>` tag that allows supplying and executing Lua code by specifying a function name in the "id" parameter, setting the "type" parameter value to "LuaBlock", and supplying the actual code in the "var" parameter. Using this, an attacker can download arbitrary files to a user's system and execute arbitrary commands on it.

It should be noted that these tags do not immediately seem to be parsed in the initially executed RGI file, but only the downloaded RGI file from GameHouse i.e. requiring the ability to intercept or redirect the traffic.

As the GameHouse installer registers RGI files with the "application/vnd.rn-realarcade-rgi" MIME type, these can be automatically executed by simply getting a user to visit a malicious web page.

The updated version (3.0.7) of the game installer still supports this file type and executes files without warning once double-clicked or supplied using the registered MIME type. The RGI files in turn trigger downloads of the main executable and/or RGA files (RAR-compressed files containing the required game resources). While the download process seems to have changed, downloading of the RGI, executable, and RGA files still occurs over HTTP seemingly without any mechanisms to prevent a MitM attacker from manipulating content.

RGUNINST Files

RGUNINST files are uninstallation script files used to remove installed games on a user's system. These are in a simple text format containing commands separated by newlines and allow e.g. deleting arbitrary files and registry keys.

A list of some of the more interesting commands that may be present in an RGUNINST file follows:

RM: The command accepts a single argument: A path to a file or directory that is deleted on the system. An example of a valid command, which would delete `MyFile.txt` from `C:\MyDir` is: `"RM,C:\MyDir\MyFile.txt"`.

REG: The command accepts two or three arguments: Subtree to access, registry key, and optionally registry value. This is used to delete arbitrary registry keys or values. An example of a valid command, which would delete the "MyVal" registry value from the "MyKey" registry key in the HKCU subtree is: `"REG,HKEY_CURRENT_USER,MyKey,MyVal"`.

UNINSTSCR: The command accepts a single argument: A path to a RGUNINST file. The command is typically used to reference the path of the executing uninstall script to ensure it's deleted from the "Uninstall" directory. It should be noted, though, that the command accepts any directory or file as path to delete and is not limited to RGUNINST files. As such, this command works much in the same manner as the "RM" command.

REGSRVR: The command accepts a single argument: A path to a file to unregister as a command component in the registry. An example of a command unregistering `CheckInst.dll`: `"REGSRVR,C:\Program Files\RealArcade\Installer\CheckInst.dll"`.

SHORTCUT: This command accepts a single argument: The path to a shortcut (.lnk) file. The command is typically used to reference the game shortcut to remove. It should be noted, though, that the command accepts any directory or file as path to delete and is not limited to shortcut files. As such, this command works much in the same manner as the "RM" command.

As evident from the functionality supported by RGUNINST files, these should be considered unsafe and never run if not fully trusted. They allow deleting arbitrary directories, files, and registry information on the system with the privileges of the user opening the file.

The updated version (3.0.7) of the game installer still supports this file type and executes when double-clicked without warning. While this should not be considered a vulnerability, users should be aware of the implications of running an untrusted RGUNINST file.

Conclusion

GameHouse being unresponsive to security researchers contacting them, as well as the critical, publicly documented vulnerabilities allowing remote code execution via a simple web page remaining unpatched for two years, makes it clear that GameHouse needs to take security more seriously.

While GameHouse was not responsive to the initial report provided by Risk Based Security, a site redesign was at least completed about 2½ months later, providing a new installer from Zylom.com (also a GameHouse site) referred to as ActiveMARK Game Installer version 3.0.7.

Testing in July 2013 by Risk Based Security confirmed that the unsafe ActiveX controls are no longer marked safe-for-scripting. This addresses the most serious concerns (unsafe methods and use-after-free vulnerability), but other issues remain unpatched.

Local users can still gain escalated privileges due to the unsafe default games installation path and a Man-in-the-Middle attacker may still be able to manipulate the contents of the `gameInstaller.rgi` files transmitted over HTTP to execute arbitrary code.

Users can prevent the local privilege escalation by not selecting the default games installation location suggested by the installer. By changing it to a subfolder within `%ProgramFiles%`, appropriate permissions will be set. Limited testing has not determined any adverse effects from doing so.

Preventing the threat of MitM attacks is, unfortunately, not equally straightforward. Users can remove support for the RGI (and optionally RGUNINST) file extension and the “application/vnd.rn-realarcade-rgi” MIME Type, but doing so prevents game installation from GameHouse.com. It should, however, not impact the ability to play already installed games. Hopefully, GameHouse will ensure that communication to/from their “secure servers” in the future occurs over HTTPS instead of HTTP.

Timeline

2011/04/03	Unsafe methods in ActiveX controls originally discovered and reported by rgod in version 2.6.0.445.
2013/01/26	Previously reported vulnerabilities confirmed in latest version (2.6.0.481) along with new security concerns.
2013/02/13	RealNetworks contacted (clientsecurity@real.com) to obtain GameHouse security contact.
2013/02/15	Security contact requested from GameHouse via their online form.
2013/02/15	Random, off-topic response received from GameHouse customer support: <i>"I am sorry that you are receiving virus alert when attempted to download games. GameHouse games are tested before release they are safe. If you're receiving a warning message, you may simply need to update your anti-virus software"</i> .
2013/02/15	Clarifying mail sent to GameHouse customer support. No response.
2013/02/21	RealNetworks security contact provides appropriate security contact (DL-games-security@realnetworks.com) for reports related to RealArcade and GameHouse.
2013/03/11	Details sent to GameHouse / RealArcade security contact (DL-games-security@realnetworks.com).
2013/05/29	While not being responsive, GameHouse completes a major site update and provides a new version of the game installer.
2013/07/17	RBS notices the site update and examines the latest version of the game installer. RBS concludes that while the unsafe ActiveX controls are no longer marked safe-for-scripting, other security concerns still persist.
2013/08/13	RBS sends an update to GameHouse to inform about issues still affecting the latest version (DL-games-security@realnetworks.com).
2013/09/05	No response from vendor. Report published.