OpenSecurity

open source solution for public institutions

Project: OpenSecurity

Windows Implementation Proposal

Document Control Page

Creator	**AIT**
Editor	**Oliver Maurhart**
Subject	OpenSecurity - Windows Implementation Proposal
Meeting date(s)	
Meeting location	
Publisher	OpenSecurity consortium
Type	Text
Format	application/vnd.oasis.opendocument.text
Language	EN-US
Creation date	2013-11-12
Rights	© Copyright “OpenSecurity consortium”.
Audience	☒ internal
Review status	☒ draft ☐ final
Action requested	☐ to be checked by partners present at the meeting

Revision history

Version	Date	Modified by	Comments
0.1	2013-11-13	Oliver Maurhart	Initial version

Table of contents

1 Introduction

This document summarizes the thoughts on the implementation details on the Windows platform. This is the follow-up document of a technical meeting held on Wednesday, 6th November 2013 at AIT at the TechGate Vienna.

As a prerequisite to this document the study of architectural studies and designs within the OpenSecurity project are essential.

2 Goals

The Goals of the Windows implementation are twofold:

a) Implement “safe browsing” on Windows.

b) Provide Virus checking off mass storage device.

These are but two use cases described by the architectural documents of the OpenSecurity project. A description of these goals can be found in “Project: OpenSecurity Architecture” by Michai Bartha.

The implementation concepts here address:

- “1.5 Safe Internet Access”
- “1.6 Interaction with Removable Storage Devices”
- “1.7 Antivirus Tools and Usage Patterns”

in the above-mentioned document.

3 Basic Concept

A key element in the design is the strict segregation of user space applications, user space presentation, security VMs, and security VM administration, extending the "Security by Isolation" principle.

The determining factors of the current approach are:

- Target system is Windows 7 64Bit
- The target system are currently productive operational.
- Thus the OpenSecurity instalments must not compromise any already existing deployed software.
- Target installation covers more than 1K (1000) machines. Returning machines into service stations in order to install the OpenSecurity solution is therefore infeasible. The installation must be either driven remotely or very easy to attend.
- Hardware and software layout and setup may differ from machine to machine.
- In an ideal scene a "OpenSecurity Setup.exe" or "*.msi" likewise installs the whole OpenSecurity system covering the aspects mentioned in section 2.

3.1 Prerequisites

Some OpenSecurity tasks have to run under a privileged user account. These are:

- USB or mass storage driver management
- Disposable Security VM incarnation and destruction on demand.
- Updating of Security VMs

As any security breach on these systems tampers the overall security of the OpenSecurity instalment the user MUST NOT log into the system with administration rights.

OpenSecurity services, daemons and VBoxes run as privileged user are therefore isolated by well defined and well understood open IPC interfaces. These interfaces are:

- HTTP for RESTful interaction between OpenSecuirty Admin and User side services
- SMB/CIFS for data exchange between files hosted within the VBox guests and the Windows Host

- SSH-X11 for rendering the graphical user interface on the Windows Host.

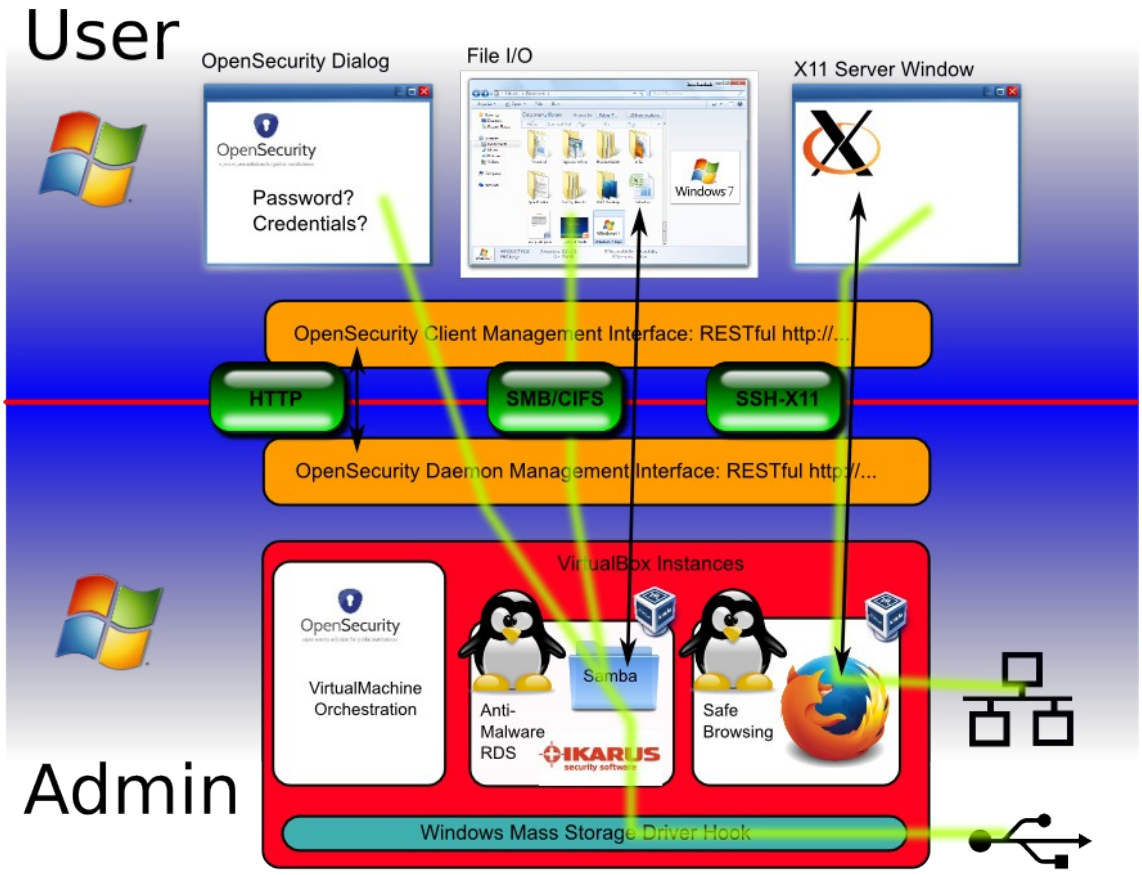

Figure 3.1: Sketch of OpenSecurity Service and Daemon interactions

3.2 Tasks of the OpenSecurity Admin

The OpenSecurity Admin has these challenges:

- Orchestrate VirtualBoxes: create, start, stop and destroy of VirtualBoxes. Update of VirtualBox templates.
- Listen on system events related to mass storages (mostly USB devices): capture USB device events, launch appropriate disposable VMs, redirect USB events and I/O to VM
- Manage addressing of running VMs: IP addressing, NAT, Host Only Networks.

The OpenSecurity Admin acts as daemon aka Windows Service started at boot. It provides a RESTful API in order to manage OpenSecurity Admin service. The concrete API of the service is to be defined but examples are[1]:

1 Port 801 seems currently not to be assigned and used by the IANA. As these service is security

http://127.0.0.1:801/	GET	return current status, version number
http://127.0.0.1:801/vms	GET	return list of all VMs registered in the system
http://127.0.0.1:801/vms/SafeBrowsing	GET	return status of VM named "SafeBrowsing"
http://127.0.0.1:801/vms/SafeBrowsing	POST	With the argument "action=start" starts a new instance of the VM
http://127.0.0.1:801/vms/SafeBrowsing	POST	With the argument "action=clone" clones the VM assuming the original VM served as a template.
http://127.0.0.1:801/vms/SafeBrowsing-Clone-0/home/user/.ssh/authorized_keys	PUT	Places the accompanying data as file "/home/user/.ssh/authorized_keys" into the VM
http://127.0.0.1:801/vms/SafeBrowsing-Clone-0/usr/bin/iceweasel	POST	start command "/usr/bin/iceweasel" within the VM "SafeBrowsing-Clone-0". Command arguments could be hold in the "args" POST argument. The result of this call (stdout and stderr or the command invocation) are returned.

Table 3.1 Sample API for the OpenSecurity Admin

3.3 Tasks of the OpenSecurity User

The OpenSecurity User side services ensure for a working SSH-X11 environment to forward the graphical representations of applications running inside a OpenSecurity VM. For this task the OpenSecurity Admin must have means to iterate and hand out the available running VMs and their IP addresses to connect to.

On mounting some RDS within the OpenSecurity Admin service, the OpenSecurity Admin must have means to query the current[2] user to provide passwords and/or credentials to the system.

sensitive it MUST run at a port number <1024. This ensures that it has been started with privileged rights is not a fake user service mimicking the real service.

2 In a multi-user environment this is a problem not tackled yet: if more than 1 user are currently logged in, which one to ask for credentials/password when mounting an encrypted RDS?

As for the OpenSecurity Admin also the OpenSecurity User API is to be defined. Examples are:

http://127.0.0.1:8801/	GET	return current status, version number
http://127.0.0.1:8801/password	GET	get a password of the user. An optional argument “text” may hold a descriptive text shown to the user reasoning the query.
http://127.0.0.1:8801/credentials	GET	return username and password for an action. Again the optional argument “text” may hold a descriptive text for reasoning this action.

Table 3.2: OpenSecurity User API