



Connected Sentinel Player Signing Process Guide

Product Version 1.0



copyright

The contents of this documentation are strictly confidential and the receiver is obliged to use them exclusively for his or her own purposes as defined in the contractual relationship. No part of Viaccess-Orca applications or this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage and retrieval system, without permission in writing from Viaccess S.A and/or Orca Interactive.

The information in this document is subject to change without notice. Viaccess S.A nor Orca Interactive warrant that this document is error free. If you find any problems with this documentation or wish to make comments, please report them to Viaccess-Orca in writing at documentation@viaccess-orca.com.

trademarks

Viaccess-Orca is a trademark of Viaccess S.A[®] in France and/or other countries. All other product and company names mentioned herein are the trademarks of their respective owners.

Viaccess S.A and or Orca Interactive may hold patents, patent applications, trademarks, copyrights or other intellectual property rights over the product described in this document. Unless expressly specified otherwise in a written license agreement, the delivery of this document does not imply the concession of any license over these patents, trademarks, copyrights or other intellectual property.

Document reference number: 21716

Document version number: 1.0 Draft

Contents

- Introduction 5
 - Target Audience 5
 - Glossary..... 5
- Signing Process 6
 - Overview 6
 - Generate RSA key pair 6
 - Generate Signature binary 7
- Glossary 8

Introduction

This guide explains the process of signing the SO (shared object) files that are used in parallel to the SP-SDK (SecurePlayer SDK). This procedure is mandatory for using the SP-SDK.

note

OpenSSL should be install on the computer before starting sections 3 and 4.

SDL (Secure dynamic loading) is based on build-time signing of binary modules followed by load-time verification of these modules. By ensuring that only properly-signed modules are loaded and used by the application, the mechanism blocks classes of code-injection attacks that use the dynamic loading interfaces.

As an obvious extension, the secure dynamic loading mechanism also covers signing and verification of developer-controlled configuration files, to prevent those from enabling attacks (by malicious changing of security-relevant configuration parameters).

The secure dynamic loading mechanism integrates with a separate secure runtime code-monitoring mechanism, intended to detect post-loading attempts to modify the code in memory.

Target Audience

This document is intended for developers writing a player application based on the SecurePlayer SDK and need to add an SO file/s to their application.

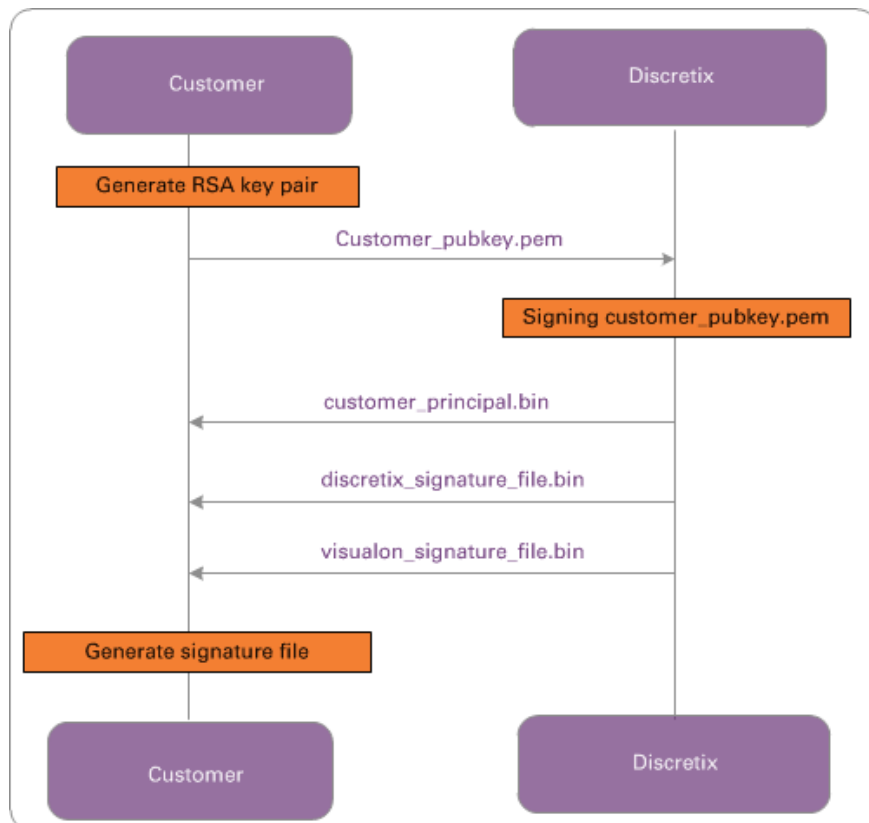
Glossary

This manual contains a lot of acronyms or terms that are specific to the field of the Viaccess-Orca Conditional Access System. If they are not defined within the text, refer to the *Glossary* on page 8 at the end of the manual for a complete definition.

Signing Process

Overview

The diagram below explains the Secure Player signing process:



Generate RSA key pair

The RSA key pair should be generated once as follow:

note

<Name> should be replaced by something that will represent your organization and MUST be consistent throughout the entire process.

```
openssl req -newkey rsa:2048 -keyout <Name>_prikey.pem -passout pass:<choose password> -subj "/CN=<Name>" -out <Name>.pem
openssl req -in <Name>.pem -verify -noout -pubkey -out <Name>_pubkey.pem
```

The command operations yield <Name>.pem, <Name>_pubkey.pem, <Name>_privkey.pem

Only <Name>_pubkey.pem will be send to Discretix (this is the only key that should be sent to Discretix as part of the SDL process).

```
-o <Name>_principal.bin
```

Generate Signature binary

All SO files that the application needs to load must be added to the signature binary.

The signature binary is generated by `DxDlcSignatureFileGeneratorTool.exe` and should be generated as follows:

```
DxDlcSignatureFileGeneratorTool.exe -key <xxx_prikey.pem> -keysig <xxx_principal.bin> -v <SecurePlayer Package Name> -f <First SO path> -f <Second SO path> -f <Third SO path> ..... -sigf discretix_signature_file.bin -sigf visualon_signature_file.bin -o libDxSig.so
```

note

The files <xxx_principal.bin>, discretix_signature_file.bin and visualon_signature_file.bin will be supplied by Discretix after the customer will send to Discretix the <Name>_pubkey.pem file.

For instance:

```
DxDlcSignatureFileGeneratorTool.exe -key C:\Dir\<Name>_prikey.pem -keysig C:\Dir\<Name>_principal.bin -v GENERAL_ANDR_VOP_PROB_RC_02_00_00_0000 -f C:\Dir\lib1.so -f C:\Dir\lib2.so -f C:\NOW_SDL\lib3.so -f -sigf C:\Dir\discretix_signature_file.bin -sigf C:\Dir\visualon_signature_file.bin -o C:\Out_Dir\libDxSig.so
```

The libDxSig.so should be added to the project that use the SP-SDK.

Glossary

Definitions of technical terminology and acronyms are listed in the table below:

Term	Definition
SDP	Session Description Protocol
SO	Shared Object
SP-SDK	Discretix Secure Player SDK

