Program storage device verification

Version 1.8.1

Queensland Government

For further information, please contact the Office of Liquor and Gaming Regulation on 13 QGOV (13 74 68) or visit www.business.qld.gov.au/industry/liquor-gaming

Contents

1 Introduction

Policy:

Devices such as Electronic Gaming Machines (EGMs), Jackpot Triggering Devices (JTDs) and any other electronic Gaming Equipment as directed by the OLGR must be able to have the software and data contained in their Program Storage Devices securely authenticated on demand. It is also desirable to minimise the resources required in order to do this.

Devices which use Public Key Encryption techniques to securely implement remote software upgrades to the OLGR's satisfaction and that also implement the OLGR's "Electronic Seal Minimum Requirements" are a special case. Refer to Appendix A for the applicable requirements regarding PSD verification on these devices.

If any manufacturer of a Device is having difficulty meeting these requirements then contact the OLGR to discuss alternative ways to satisfy the intent of these requirements.

Please refer to the revision history for any policy in regards to the incept date of the requirements in this version of the document.

Scope:

These requirements are applicable to EGM manufacturers, manufacturers of Jackpot Triggering Devices and any other electronic Gaming Equipment as deemed as applicable by the OLGR.

2 Glossary

Child PSD(s)

The Child PSDs are defined as any PSDs in a 'Device' that do not qualify to be a Parent PSD. Refer 'Parent PSD' below.

CRC

Cyclic Redundancy Check

Device

An EGM, Jackpot Triggering Device (JTD) or any other electronic Gaming Equipment as deemed as applicable by the OLGR.

Digital Signature

A secure hash value over the data to be signed, signed via a Public Key Encryption Algorithm.

EGM

Electronic Gaming Machine

Gaming Equipment

Refer QLD gaming legislation.

JTD

Jackpot Triggering Device.

Refer OLGR's Jackpot System Minimum Requirements document.

LSB

Least Significant Byte

MSB

Most Significant Byte

PSD

Program Storage Device.

Refers to all possible 'Program Storage Devices' including but not limited to hard drives, CD-ROMs, PROMs, EPROMS and flash devices. This also may include data storage devices if they could potentially be used by the Device to store or alter program or other sensitive data.

PSDV

Program Storage Device Verification.

PSDV Function

The PSDV Function is a user interactive function implemented in a Device which allows the user to input an arbitrary seed value and then using the seed, calculates and outputs a hash result of all the data on its PSDs.

Parent PSD(s)

The Parent PSD(s) in a Device are defined as the first executed PSD(s) on power up of the Device, up to and including all PSDs that must have program code executed off them until a PSD hash can be calculated and presented as specified by these requirements. The Parent PSD must also have full read access of the entire storage space in the Child PSDs, else it is deemed a Child PSD.

Note, it is acceptable for a Parent PSD to read data off a Child PSD in order to calculate the PSD hash and the Child PSD not be disqualified from being classified as a Child PSD, on the condition that the data read off the child is not executed (e.g. in the case of a display font stored on a Child PSD).

OLGR

Queensland Office of Liquor and Gaming Regulation.

3 Introduction

These requirements specify a hybrid method for the secure, in-field verification of PSDs used in Devices. The method uses a combination of isolated PSD verification (e.g. using EPROM readers and other stand alone PSD readers) and PSD hashes, calculated and presented by the Device to securely authenticate its software. The main benefit of this method is that it can reduce the time and the amount of hardware required in performing secure program verifications, particularly in devices which have many PSDs.

Traditionally, to perform a secure, in-field program check over a Device, the Device must have all its PSDs removed and verified in isolation via a compatible PSD reader. However, new Devices are more often making use of multiple technologies for the purpose of storing program data (such as in combinations of EPROMs and flash devices). As many of the flash readers are custom made it can become costly and inconvenient to carry all the equipment required to be able to read all types of PSDs used in these Devices. The authentication process in many cases can be made less hardware intensive while maintaining high security and integrity by using the following assumption regarding the device's PSD hash verification:

If all the 'Parent PSDs'[i] in a Device have been verified in isolation (i.e. by removing them from the Device and using a compatible reader) and the expected object code on the parent/s is trusted[ii], then it is acceptable to trust a hash reported by program code executed off Parent PSDs for any 'Child PSDs'[i] in the Device.

The goal is to have as many PSDs categorised as Child PSDs in the Device as possible because of the following benefits:

- Less hardware will be required for the verification process because Child PSDs do not have to be removed from the Device for in-field verifications. (I.e. hardware in the form of flash readers etc). Refer to section 4.2.

- Time savings, as only the Parent PSDs have to be removed from the Device in order to complete a verification of the Device (once verified, the Parent PSDs can be trusted to perform and report a hash value over the Child PSDs).

Analysis:

The strength of this method is only as good as the strength of the hash algorithm used for program verification. Weak hash algorithms that are susceptible to result rigging (such as CRCs) should not be used. The algorithm used is also strengthened by the ability to vary the seed.

[i] Refer Glossary.
[ii] There is also an assumption that the source code of the Parent PSDs has been examined and verified with the object code on the Parent PSDs and therefore the Parent PSD may be trusted once it has been verified in isolation.

4 Requirements

4.1 Classification of all PSDs in the Device as either 'Parent' PSDs or 'Child' PSDs.

The Device manufacturer must classify all PSDs in the Device as either Parent PSDs, or Child PSDs (as defined in this document) and provide this information to the OLGR. The classification of PSDs is primarily determined by the location in software of the Device's PSDV Function. Refer to the Glossary and Introduction above to understand how to categorise PSDs.

Device manufacturers must make every attempt to have as many PSDs categorised as Child PSDs in their Device as possible by locating the PSDV Function as early in the boot process as possible. **(Benefits of this are discussed in the Introduction, but the earlier the PSDV Function can be located in the boot process, then the more PSDs that can be classified as Child PSDs)**

Reason: For secure validation, all Parent PSDs must be verified in isolation (i.e. removed from the device), where as Child PSDs can be verified simply via the PSDV Function. So the less PSDs that have to be removed for isolated validation, the better, as the less time and resources it takes to perform the verification. Requirements for the PSDV Function are contained later in this document.

(The OLGR or Approved Evaluator will use source code checks, emulators and possibly special Child PSDs filled with garbage to verify that the Parent PSDs are capable of performing a hash calculation without executing any code off a designated Child PSD).

4.2 PSD Readers

Compatible PSD readers must be supplied to the OLGR (at the Device manufacturer's expense) for the isolated verification of all PSDs in the Device.

This may not be required in all cases, for example, where a compatible reader is already in the OLGR's possession (such as EPROM compatible readers). The OLGR will advise when a reader is not required.

Note, for PSDs classified as Parent PSDs in the Device, more than one reader may be requested depending on the number of intended markets for the Device (e.g. club, hotels & Casinos), and the total number of Devices supplied to Queensland.

To minimise potential costs, it would be desirable in the design of a Device, if the PSDs that will be classified as Parent PSDs, are verifiable in a cheap and readily available reader device.

4.3 All PSDs in the Device must be socketed (i.e. easily removable) and verifiable.

Rationale: This is to allow all the PSDs to be verified in isolation outside of the Device where necessary for even greater security, or incident investigations.

Note, it is acceptable for a group of PSDs mounted together to be considered as a single PSD under the following conditions:

1. There is full and direct read access of the PSDs at the PSD interface, and
2. A compatible reader is available.

For example, a number of flash chips mounted on a plug-in circuit board may be considered a single PSD so long as the above conditions are met.

Exemption: If it proves too difficult to socket a PSD and there exists a method in which the PSD can still be read directly (e.g. via a port or JTAG connector) then an exemption on socketing that PSD may be granted.

4.4 Devices which must offer a PSDV Function as specified by these requirements

EGMs, JTDs and other electronic Gaming Equipment as directed by the OLGR.

4.5 PSDV Function Format

The PSDV Function must be "on-screen" for Devices that have a built-in or attached display device; otherwise a suitable communications port and protocol must be used.

4.6 The PSDV Function must be available for activation at power up of the Device.

The following requirements apply:

4.6.1 The PSDV Function must be offered as early as possible in the Device's power up procedures and before it executes code off as many other PSDs as physically possible.

Rationale: If the hash calculation can be performed solely within the code of a boot ROM(s), then only that PSD needs to be removed from the Device and verified by a suitable reader. Once verified, then the hash calculation performed over all PSDs by the Boot ROM(s) may be trusted without having to remove the remaining PSDs for isolated verification. Refer to the Introduction for a detailed explanation.

Benefit: This will reduce the need for a wide variety of PSD readers to be required for in-field verification of Device PSDs in the field.

4.6.2 The PSDV Function must be activated by some specific user action on Device power up (eg. holding down of a button(s) or key-switch or a combination of switches and buttons). The action must be available to the user, without the user having to access the Device's sealed logic area. The action must not be available to the general public (eg. a key may be required to open a door or turn a switch).

If the specific user action does not occur then the Device must display (if a display device is attached), a short message on how to activate the PSDV Function. The message must be displayed for sufficient time during the boot process to enable the user to read it. This is to ensure access to the PSDV Function is possible without requiring the user to memorise the access procedure across all brands of Devices.

Rationale: The function is not required for every power up and because manual entry of a seed is required for the function, it would be inconvenient if the Device halted every power up requesting a seed.

Exemption: If all PSDs in the Device are classified and approved as Parent PSDs, then the PSDV Function does not have to be a part of the Device's boot up sequence.

4.7 PSDV Title Display

The PSDV Function while active must display the following title lines at all times:

Program Storage Device Verification
(Hash Alg: xxxx)

Where "xxxx" = is the hash algorithm used. E.g. "**HMAC-SHA-1**" for a QCOM v1.6 EGM

4.8 Seed Entry

The Device must have provision for the manual entry of a hexadecimal seed for the hash algorithm used by the PSDV Function. Rationale: A hash algorithm with an arbitrary seed is more secure than one without.

One of the following proposed methods must be used to enter the seed.

Method 1

As seeds are becoming quite lengthy, the preferred methodology for seed entry is the use of a dongle such as a USB flash storage device, or equivalent readily accessible and portable device acceptable to the OLGR.

In this method for example, the seed is saved onto a file on the USB flash storage device prior to insertion in the Device. Then the user inserts the USB flash storage device into the Device. When the Device detects the USB flash storage device's presence, it looks for a seed file[iii], imports the seed if one is found, performs a hash calculation and displays the seed and results to the user (if available) and also writes the result back to the USB flash storage device to a uniquely identified file.

While the Device is importing a seed from a file, the Device must be immune to all possible malformed file system and seed file format attacks. Refer to appendix B for more information regarding the proposed seed & hash file formats.

Variants of the above method may also be considered acceptable upon application.

Method 2

Seed entry must be via a button combination on the Device and there must be an on-screen button legend constantly displayed for all available button commands at any time.

To expedite seed entry on Devices such as EGMs that have at least two rows of five buttons plus at least one extra button (2x5+1 buttons), the seed entry button configuration and methodology must be as follows: The top row of buttons must correspond to the input of values 1…5, the second row must correspond to 6…9 & 0. One spare button must act as a dedicated 'shift' key which while being pressed, converts the top row to buttons to values 0xA

[iii] Refer to appendix B for more information.

… 0xE and the second row to 0xF, ‘Next Digit’, ‘Previous Digit’, ‘Accept’ and ‘Accept’[iv]. Once a digit value button is pressed, the cursor must automatically move to the next digit to the right. The cursor must wrap if moved past either end of the seed. Any spare buttons may be assigned to functions at the manufacturer’s discretion.

Button Panel Seed Entry Layouts

Without “shift” pressed:

1	2	3	4	5
6	7	8	9	0

With “shift” pressed:

A	B	C	D	E
F	Prev Digit	Next Digit	Accept	Accept

For EGMs that may be supplied with less than 2x5+1 buttons, then an alternative minimal seed entry button input methodology must also be provided. If the Device cannot auto-detect its button panel configuration then the Device must allow the user to select the appropriate button seed input methodology.

For Devices with touch screens, the following on-screen hexadecimal keypad would be the preferred method for seed entry with the following mandatory keypad layout:

Touch Screen Seed Entry Keypad Layout

A	B	C	D
7	8	9	E
4	5	6	F
1	2	3	Enter*
0	Bck	Fwd	

(* ‘Enter’ must be used to accept the seed, and should not be required to be pressed on a per-byte, or per-digit basis. Once a digit has been pressed, it must overwrite the current digit and the cursor must automatically move to the next digit to the right)

<u>General</u>

Devices with no built in display or inputs, may use for example, a RS-232, USB or Ethernet communications port for manual seed entry. If any special software is required to communicate with the Device for this purpose, then the Device manufacturer must provide this software.

Other types of removable Human Interface Devices may be considered acceptable for seed entry, such as keyboards and mice. However, any Human Interface Device used to enter the seed must not be able to be used to compromise the security of the device in any way.

To avoid confusion, any previously calculated hash result must no longer be displayed once a new seed is being entered / edited.

[iv] I.e. Both ‘Accept’ buttons and the ‘shift’ button must be pushed simultaneously in order to confirm the seed and commence the PSDV hash calculation.

Seed entry must be in hexadecimal, LSB first and a space must be automatically inserted every 4 characters when the seed is displayed to the user. In addition, at all times while visually editing or displaying the seed value, a 2 byte MSB first hexadecimal 16 bit CCITT-CRC of the current hexadecimal seed (calculated LSB first) must also be displayed in real time just to the right of the seed. (This value must only be appended to seeds and never hash results.)

Examples (NB for development purposes, the (CRC's) in the examples below are correct with respect to the given seed):

"Seed: 1234 5678 1234 5678 1234 5678 1234 5678 1234 5678 (8F06)"
"Seed: 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 (0000)"
"Seed: 1234 5678 9098 7654 3212 3456 7890 9876 5432 1234 (286E)"

This allows the user to quickly verify they have entered the correct seed.

4.9 Default Seed.

Upon seed entry, the user must also be able to select from at least three default seeds:

1. All zero.
2. The last manually entered seed in the on-screen function (if available)
3. The last received seed* via QCOM (if applicable and available).

*The last received seed received by the Device must be stored in NV RAM for this purpose.

4.10 Data to be included in the PSDV Function's hash calculation.

Refer to the OLGR Minimum Requirements Document entitled 'Approved Hashing Algorithms'.

Acceptance of the data and PSDs to be included in the hash calculation is at the discretion of the CEO OLGR.

4.11 Hash Algorithm to implement for the PSDV Function.

For EGMs, unless specified otherwise in this document, the hash algorithm must be the same hash algorithm as implemented in its QCOM protocol implementation (i.e. the same hash result must be received for the same seed in the protocol and the PSDV Function).

For all other Devices, unless specified otherwise, the hash algorithm must be HMAC-SHA-1.

4.12 Progress Indicator

While calculating the PSDV Function's hash, a hash calculation progress indicator must be displayed.

4.13 Display of Seed and Hash Results

The PSDV Function's hash calculation's seed & hash result must be clearly displayed.

The Device must be able to display (or transmit if no display) a hash result along with the corresponding seed.

The hash result display must be able to be paused indefinitely in order to verify the displayed hash value.

The seed and hash result must be displayed in hexadecimal, LSB first in groups of 4 characters. E.g.

Seed: 1234 5678 1234 5678 1234 5678 1234 5678 1234 5678 (8F06)
Hash: 1234 5678 1234 5678 1234 5678 1234 5678 1234 5678

Devices without a built in display may use for example a RS-232, USB or Ethernet port for the display of the hash seed and result. If special software is required to communicate with the Device, then the Device manufacturer must provide this software.

4.14 Table of PSDs

Accessible within the PSDV Function and in the Device's audit mode it must be possible to display (or download if no display) a list of every possible physical PSD in the Device (including spare or unused PSD sockets) with information under at least the following headings:

Description/Type	Location	Parent/Child	Version	Size
E.g.				
BIOS EPROM	U12 Main-board	Parent	1.6.2	64Kb
Spare	U88 Main-board	NA	NA	0Kb
Application Flash	U44 CPU board	Parent	4.7	256Mb
Game Flash	U61 I/O Board	Child	1.2.3.4	1GB
Game Flash aux	U62 I/O Board	NA	NA	0Kb
…				
Etc.				

A SHA-1 hash result (either the HMAC or non-HMAC version may be utilised in this table; whichever is easier, however NS mandates HMAC-SHA1 here) for each PSD in the device must also be displayed (or downloaded). E.g:

Description/Type	PSD [HMAC-]SHA-1 Hash Result:
Master Result*	(Combined XOR of below as per National Standards)
BIOS EPROM	1234 5678 1234 5678 1234 5678 1234 5678 1234 5678
Spare	1234 5678 1234 5678 1234 5678 1234 5678 1234 5678
Application Flash	1234 5678 1234 5678 1234 5678 1234 5678 1234 5678
Game Flash	1234 5678 1234 5678 1234 5678 1234 5678 1234 5678
…	
Etc.	

If the table is too long to display on-screen, then it may be split across multiple screens.

If the HMAC-SHA-1 algorithm is used in the above PSD table, then the EGM must also display the seed that was used for producing the hashes in the table on the same screen.

Given the appropriate PSD reader/s, the PSD hash results in the above table must be able to be reconciled with the corresponding PSD. Procedures on how to do this (re bit/byte order etc) must be provided.

* As the physical PSDs may not be the same as the logical PSDs in the device, there is no expectation that the combined (XOR) PSD hash results (i.e the Master Result) will equal the overall hash result returned by the PSDV calculation function.

Appendix A

Devices which use Public Key Encryption techniques to securely implement remote software upgrades (to the OLGR's satisfaction) and that also implement the OLGR's "Electronic Seal Minimum Requirements" are a special case and only the following requirements apply regarding PSD verification:

1. Sections: 4.2, 4.3, 4.10, **Error! Reference source not found.** & 4.11.

2. There must be a secure and authenticated method (to the OLGR's satisfaction) of ensuring that upon initial commissioning and sealing of the Device, that the device is initially installed with only approved bios/base/application software.

3. During operation, there must be a secure and authenticated* method (to the OLGR's satisfaction) of verifying the current software on the device. I.e. to ensure a device is running the latest approved s/w and not an un-approved, or older approved version of s/w.

4. During operation, there must be a secure and authenticated (to the OLGR's satisfaction) method to verify the Device's current Public Key on demand.

*e.g. a hash over application files, (possibly signed with a Digital Signature of the device, to ensure that the verification can also be performed remotely) is one good method. (Of course if an inspector can interrogate the device directly, then it may be acceptable to trust what is received from the device without the need for a Digital Signature.)

Consequently an in-field verification would only entail a seal inspection and #3 above.

Appendix B

Seed File format (refer 4.8)

Filename: psdvseed.xml

Contents:

```
<?xml version="1.0"?>
<seed>
      <hexstring length=“20” byteorder=”lsb”>
            12345678123456781234567812345678
      </hexstring>
</seed>
```

At this time a file-system for the USB storage device has not been specified, however the FAT or FAT32 file systems are preferred as they are natively readable by both Windows and Linux operating systems.

Hash File format (refer 4.8)

Filename: psdvhash-<*Deviceserialno*>.xml

Contents:

```
<?xml version="1.0"?>
<seed>
      <hexstring length=“20” byteorder=”lsb”>
            12345678123456781234567812345678
      </hexstring>
</seed>
<hash alg=”HMAC-SHA1”>
      <hexstring length=“20” byteorder=”lsb”>
            12345678123456781234567812345678
      </hexstring>
</hash>
```

Revision History

Version	Release Date	QIR	Who	What
1.8.1	19/04/2016		JG	• Updated to new DJAG report document template
1.8	20/8/2010		RLL	• Re hash xml format: Moved “alg” to <hash> so <hexstring> is consistent • Updated to new DEEDI report document template
1.7	19/03/2007		RLL	• See 1.7 draft below • Replaced section 4.10 on ‘data to include...’ with reference. • Added hash file format to Appendix B
For new or updated requirements in the above version of the document, the incept date is 6 months from the document’s release date and is applicable to base upgrades or new base software submissions only.				
1.7 draft	6/12/06	NA	RLL	• Bring into line with National Standards requirements. E.g. 4 chars/space re seed/hash display and hash type for PSD table • Proposed the use of seed dongles such as USB storage devices. • Added appendix B
1.6.2	06/04/2006	365	RLL	• Added requirement for the display of on-screen PSDV title. • Added display of hash values for each PSD and added display of unused/spare PSDs. • Clarified use of ‘on-screen’ • Generalised for non-EGM/JTD use. • Fixed issues with standard 2 row x 5 button seed entry.
The HMAC-SHA-1 algorithm must be implemented in all QCOM v1.6.x gaming machines as the hashing algorithm pertaining to this requirements document. For all other new or updated requirements in this version of the document, the incept date is 6 months from the document’s release date and is applicable to base upgrades or new base software submissions only.				
1.6.1	29/11/05 released for internal review only.		RLL	• 2nd draft • Removed all references to ‘Signatures’ as this term is now too easily confused with ‘Digital Signatures’ as used in cryptography. • Short message on how to access the during the boot process must be displayed (refer 4.6.2) • Added appendix A • Added PSD display list (4.14) • Added mandatory and preferred methods of seed entry (4.8)
1.6	28/09/05 released for internal & external review only		RLL	• General Review. • Incorporated requirements for the upcoming use of HMAC-SHA-1 in Devices, e.g. QCOM v1.6 EGMs. I.e. 3 types of default seeds, LSB seed & hash display (was previously MSB) and 16 bit CRC on seed display.

Version	Release Date	QIR	Who	What
1.1	03/06/02		RLL	Initial Release
1.0	08/05/02		RLL	1st Draft