Dialogic
An Enghouse Company

Brooktrout Fax Products

Documentation Update

SDK Versions 6.17.0

February 2024

Terms of Use

Any software (“Software”) that is made available by Enghouse Interactive Inc. (“Enghouse”), together with any User Documentation (“User Documentation”) is the copyrighted work of Enghouse. Use of the Software is governed by the terms of a Master Purchase Agreement, End User License Agreement, or similar software license agreement (“License Agreement”). End users are not legally authorized to install any Software that is accompanied by or includes a License Agreement unless he or she first agrees to the License Agreement terms.

The Software is made available for installation solely for use by users according to the License Agreement. Any reproduction or redistribution of the Software not in accordance with the License Agreement is expressly prohibited by law and may result in severe civil and criminal penalties. Violators will be prosecuted to the maximum extent possible.

WITHOUT LIMITING THE FOREGOING, COPYING OR REPRODUCTION OF THE SOFTWARE TO ANY OTHER SERVER OR LOCATION FOR FURTHER REPRODUCTION OR REDISTRIBUTION IS EXPRESSLY PROHIBITED, UNLESS SUCH REPRODUCTION OR REDISTRIBUTION IS EXPRESSLY PERMITTED BY THE LICENSE AGREEMENT ACCOMPANYING SUCH SOFTWARE.

THE SOFTWARE IS WARRANTED, IF AT ALL, ONLY ACCORDING TO THE TERMS OF THE LICENSE AGREEMENT. ENGHOUSE HEREBY DISCLAIMS ALL OTHER NON-EXPRESS WARRANTIES AND CONDITIONS WITH REGARD TO THE SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES AND CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE, AND NON-INFRINGEMENT.

Enghouse grants a nonexclusive license to customer for use of the User Documentation. The User Documentation contains copyrighted and other proprietary materials. By accepting the User Documentation, recipients agree that they will not transmit, reproduce, or make available to any external third-party this User Documentation or any information contained herein. Copying, reverse-engineering, or reselling any part of the Software or User Documentation is strictly prohibited.

The information contained in the User Documentation furnished by Enghouse is based on the most accurate information available at the time of printing. No representation or warranty is made by Enghouse as to the accuracy or completeness of such information or any ongoing obligation to update such information. Enghouse reserves the right to change the information contained in this document without notice.

Table of Contents

Purpose

The purpose of this document is to capture the documentation changes since the SDK 6.13 release. Specifically, it provides information on the changes to the SDK 6.13 version of the Brooktrout Documentation.

Technical Support

For Technical Support, see https://mysupport.enghouse.com/ or email Brooktrout.support@enghouse.com.

Product Documentation

For the latest product documentation, see https://www.dialogic.com/manuals/brooktrout/brooktrout.

Brooktrout BFV APIs Reference Manual

This section describes the changes to the Brooktrout BFV API's reference manual based on the SDK version.

SDK 6.17.0

FIPS Support for Windows and Linux

FIPS 140-2 Support for SR140 is provided by the OpenSSL FIPS Provider Module version 3.0.8. This module has the latest Cryptographic Module Validation Program (CMVP) approvals for FIPS 140-2.

The certificate number for the validation is **Certificate #4282**. Cryptographic Module Validation Program | CSRC (nist.gov)

To install the OpenSSL FIPS Provider, an OpenSSL installation step is now required. This step must be executed on each system.

Windows FIPS Provider Installation

Windows PowerShell is required for the installation batch file.

Unzip the four files (fips.dll, installFips.bat, opensll.exe, opensll.cnf), which are in the provided installFips.zip file into the directory identified by the BRKTD_LICENSE_FILE environment variable.
From a Windows Powershell window, change to the created fips directory (cd %BRKTD_LICENSE_FILE%\fips) and run the installFips batch file (installFips.bat). This batch file will run the OpenSSL FIPS Provider installer on the system by performing the following steps:

- Removes old FIPS configuration files (if present).
- Runs the OpenSSL FIPS Module's self-tests.
- Generates the Module config file output containing information about the Module (such as the self-test status, and the Module checksum)

If the self-tests run without issues, it will show an INSTALL PASSED message and create the FIPS configuration files for this system. For more information, please see the OpenSSL fipsinstall manual page.

Linux FIPS Provider Installation

Linux bash shell is required for the installation batch file.

Unzip the LinuxinstallFips.zip file into the directory defined by the BRKTD_LICENSE_FILE environment variable. From a command terminal, change to the created fips directory (cd $BRKTD_LICENSE_FILE/fips) and run the installFips shell file (./installFips). This shell file will determine the version of Linux installed and run the OpenSSL FIPS Provider installer for the system by performing the following steps:

- Removes old FIPS configuration files (if present).
- Runs the OpenSSL FIPS Module's self-tests.
- Generates the Module config file output containing information about the Module (such as the self-test status, and the Module checksum)
- Puts the appropriate configuration files into the $BRKTD_LICENSE_FILE/fips directory.

If the self-tests run without issues, it will show an INSTALL PASSED message and create the FIPS configuration files for this system. For more information, please see the OpenSSL fipsinstall manual page.

FIPS Provider Self-tests

The FIPS provider power-on self-tests are run upon the initialization of the module and do not require operator intervention. This validation step will be performed during the SR140 startup when configured for FIPS mode. If any of the self-tests fail, the module will not initialize, and all data output is inhibited. If the FIPS installation process is not performed or the FIPS Provider self-tests fail, the Boston Host Service will fail to start when configured for FIPS mode and will note the error in the ECC log file.

Note: The Module requires that the self-tests run, and the Module config file output generated on each platform where it is intended to be used. The Module config file output data should not be copied from one machine to another.

Note: Two integrity checks are performed, the software integrity check during the installation and an additional integrity check post installation of the Module. The software integrity check is performed using HMAC-SHA-256 on the Module file to validate that the Module has not been modified. The integrity value is compared to a value written to the config file during installation.

Additional integrity checks are performed once the Module has been installed using HMAC-SHA-256 on a fixed string to validate that the installation process has already been performed and that the self-tests have been executed. The integrity value is compared to a value written to the config file after successfully running the self-tests during installation.

See the OpenSSL FIPS 140-2 Security Policy document for further details.

When SR140 is configured for FIPS mode, ciphers labeled as deprecated are no longer allowed. The DHE Cipher suites have been removed from SR140 allowed cipher suites to meet FIPS compliance due to being deprecated. The default value for the local_cipher_suite is different when configured with FIPS enabled.
For FIPS:
local_cipher_suite = "ALL:!EXPORT:!LOW:!aNULL:!eNULL:!SSLv2:!DHE"
For non FIPS:
local_cipher_suite = "ALL:!EXPORT:!LOW:!aNULL:!eNULL:!SSLv2"

When configured in FIPS mode, PKCS#12 certificate files must use a signature algorithm encoding method allowed in FIPS mode. Certificates with Signature Algorithm md5WithRSAEncryption or other deprecated ciphers will fail because deprecated ciphers are not allowed.

Minimum CED Tone Detection

(SR140 Only) A new parameter (ced_detect_duration) has been added to the user-defined configuration file to specify the minimum length of time that the CED tone (2100 Hz) must be present before the CED tone detection is reported by call progress. (BRKT-547)

Parameter	Purpose
ced_detect_duration	SR140 only. This value specifies the minimum length of time that the CED tone (2100 Hz) must be present before CED tone detection is reported by call progress. ***Unit:*** 10 ms increments ***Range:*** 5 - 250 (50 ms to 2500 ms inclusive) ***Value Type:*** decimal ***Default:*** 15 (150 ms)

SDK 6.16.0

SIP over TLS

Starting with SDK 6.16.0, the SR140 supports up to TLS Version 1.3 as defined in RFC 8446.

The TLS Configuration File keyword below has been updated to add TLS Version 1.3 support.

Key Name	Description
sip_tls_method	TLS mode version. The following are the allowable parameter values: tls1.3 tls1.2 tls1.1 tls1.0 sslv3 **Default:** tls1.2

SR140 behaves as a TLS server when receiving a SIP over TLS message and acts as a TLS client when placing a SIP over TLS call. The standard TLS connection setup is Client Hello (request), Server Hello with certificate (response), and then key exchange. Once completed, the SIP session will be performed using the exchanged keys.

Ciphersuites

OpenSSL has implemented support for five TLSv1.3 ciphersuites as follows:

- TLS_AES_256_GCM_SHA384
- TLS_CHACHA20_POLY1305_SHA256
- TLS_AES_128_GCM_SHA256
- TLS_AES_128_CCM_8_SHA256
- TLS_AES_128_CCM_SHA256

If TLS 1.3 is configured, the first three of the above TLSv1.3 cipher suites will be enabled by default. This means that if you have no explicit ciphersuite configuration in the TLS config file (tls.cfg), then you will automatically use those three and will be able to negotiate TLSv1.3 when configured to support TLS1.3.

The SR140 TLS local_cipher_suite string lists the cipher suites supported. It is recommended that this value is left blank, and the default values are used. The default value is defined as ALL:!aNULL:!eNULL. The ciphers for the SR140 are provided by OpenSSL library version 1.1.

Advanced SIP IP Call Control Stack Parameters

The following items are to be added to Table 27 on page 1276.

Key Name	Description
sip_registration_use_options	This parameter is used to indicate that the SR140 should send SIP option messages to the registration server. When set to TRUE, this parameter assumes that at least one registration server is present in the configuration and that default gateways are not configured. The following are the allowable parameter values: TRUE Send SIP options to registration server FALSE No SIP options to registration servers and will send options to SIP gateways. ***Default***: FALSE
sip_multiple_m_lines	Allows the user to choose between combining the SDP media audio descriptors (m: blocks) and keeping them separate. With this parameter set to False, two or more "m:" blocks will be combined if **ALL** of the following is true: • All are of type "audio" • In "Connection" lines("c:"), all elements match • Port numbers are the same • Types (e.g. "RTP/AVP") match The following are the allowable parameter values: TRUE Keep media audio descriptors separate. FALSE Combine server media audio descriptors into one block. ***Default***: TRUE

Current behavior with *sip_multiple_m_lines* set to TRUE
Initial SDP fragment having 3 "m:" lines (blocks):

```
m=audio 56004 RTP/AVP 0
c=IN IP4 192.168.5.149
a=rtpmap:0 pcmu/8000
m=audio 56004 RTP/AVP 8
c=IN IP4 192.168.5.149
a=rtpmap:8 pcma/8000
m=audio 56004 RTP/AVP 101
c=IN IP4 192.168.5.149
a=rtpmap:101 telephone-event/8000
```

Behavior with sip_multiple_m_lines set to FALSE
Transformed SDP fragment having 1 combined “m:” line (block):

```
m=audio 56004 RTP/AVP 0 8 101
c=IN IP4 192.168.5.149
a=rtpmap:0 pcmu/8000
a=rtpmap:8 pcma/8000
a=rtpmap:101 telephone-event/8000
```

Configuring T.38 Fax Transport Parameters

The text of the *t38_fax_max_datagram_recv* parameter starting on page 1239 to be replaced with the text below:

Parameter	Value
t38_fax_max_datagram_recv	Specifies size of maximum datagram that can be transmitted for T.38. This value or the maximum datagram size exchanged in the SDPs via T38FaxMaxDatagram will be used to set the maximum datagram size that can be transmitted for T.38, whichever value is smaller. See the t38_fax_max_datagram_send parameter. **Note**: In SDK 6.14.0, a change was implemented to the packaging of T.38 packets when they included redundant packets. This change was a reimplementation of the calculation to determine the number of bytes included in a T.38 UDPTL packet, when the packet contained a number of redundant packets, along with taking the negotiated max datagram setting into account. Starting with SDK 6.16.0, setting t38*_fax_max_datagram_recv* to 0 will revert SR140 to send T.38 packets in the method used prior to this change. ***Unit:*** octets ***Range:*** 0, 1 through 65535 ***Value Type:*** decimal ***Default:*** 125

SDK 6.15.0

Advanced SIP IP Call Control Stack Parameters

The following items are to be added to Table 27 on page 1276.

Key Name	Description
ignore_non_initial_record_route	This parameter is used to ignore or recognize non-initial record routes from incoming SIP messages. The following are the allowable parameter values: TRUE Ignore non-initial record routes FALSE Recognize non-initial record routes **Note:** This parameter should only be used when directed to do so by Technical Services and Support. ***Default***: FALSE
sip_use_any_reg_contact_expire	This parameter is used determine the behavior of the session timeout relating to the REGISTER message. The following are the allowable parameter values: TRUE The URI(s) that appear in the Contact: must match on the Contacts in the REGISTER message for the "expires=" value to be accepted FALSE The URI(s) do not need to match **Note:** This parameter should only be used when directed to do so by Technical Services and Support. ***Default***: TRUE
nat_sip_address	This parameter specifies the Network Address Translation (NAT) SIP IPv4 address. If this parameter is defined, the private IP addresses in the SIP messages for the From (REQUEST only), Via (REQUEST only), Contact and RFC3325 P-Asserted-Identity and P-Preferred-Identity fields will be set to the nat_sip_address. Also, the origin (o=) IP address in the SDP will be set to the nat_sip_address. ***Range:*** 0 – 255 for each dotted decimal position of the IP address. ***Value Type:*** Dotted decimal **Note:** This keyword only supports an Ipv4 address. Typically, this parameter is used in conjunction with nat_media_address and both are set to the same Ipv4 address. ***Default:*** <blank> (empty string indicating no NAT SIP address defined)

nat_media_address	This parameter specifies the Network Address Translation (NAT) media Ipv4 address. If this parameter is defined, the connection (c=) IP address in the SDP will be set to the nat_media_address. ***Range:*** 0 – 255 for each dotted decimal position of the IP address. ***Value Type:*** Dotted decimal **Note:** This keyword only supports an Ipv4 address. Typically, this parameter is used in conjunction with nat_sip_address and both are set to the same Ipv4 address. ***Default:*** <blank> (empty string indicating no NAT SIP address defined)
sips_sip_uri_scheme	This parameter sets the "sips" or "sip" URI scheme for TLS. The parameter ***sip_transport_protocol*** must be set to TLS and both ***sip_tls_enabled*** and ***sip_tcp_enable*** must be set to TRUE for the ***sips_sip_uri_scheme*** parameter to have an effect. When this parameter is set to SIPS the SIP messages will have "sips:" in the SIP message fields, and when this parameter is set to SIP the SIP messages will have "sip:" in the SIP message fields. The following are the allowable parameter values: SIPS The SIP messages will have a "sips:" in the SIP message fields SIP The SIP messages will have a "sip:" in the SIP message fields ***Value Type:*** Character string ***Default:*** SIPS

SDK 6.14.0

BfvFaxT30Holdup

In the last paragraph on page 758, the following text:

The receiver responds with a signal which is typically MCF, though it could also be RTN, RTP, PIN, or PIP.

To be replaced with:

The receiver responds with a signal which is typically MCF, though it could also be ERR, RTN, RTP, PIN, or PIP.

BfvLineOriginateCall and BfvCallSetup

The following Input Field to be added to BfvLineOriginateCall on page 374 and BfvCallSetup on page 303.

Enum TperCallFaxTransportProtocol per_call_fax_transport_protocol;

per_call_fax_transport_protocol – Allows restriction of the fax transport protocol on a per-call basis.

The value can be one of the following:

PER_CALL_FAX_TRANSPORT_PROTOCOL_DEFAULT (0) – The fax transport protocol will be determined based on the call control configuration file settings.

PER_CALL_FAX_TRANSPORT_PROTOCOL_RTP_PREFERRED (1) – The fax transport protocol will be G.711 (RTP), if available, regardless of the call control configuration file settings; if not, it will be T.38.

BfvLineOriginateCall and BfvCallWaitForComplete

The Output fields for BfvLineOriginateCall on page 374 and BfvCallWaitForComplete on page 377 to be changed to:

```
int cause;
int subcause;
int cause_location;
RES res;
CALL_RES cres.name_ident;
CALL_RES cres.name_char_set;
CALL_RES cres.connected_num;
CALL_RES cres.sip_call_id
enum TrxTransportType call_transport;
unsigned sip_header_list_len;
BT_SIP_HEADER_LIST *sip_header_list;
```

In the description section, the following text to be added:

args.cres.char sip_call_id

Returns text in the sip_call_id field of the CALL_RES structure (see Volume 6, Appendix B, CALL_RES Structure Parameters), indicating the SIP Call-ID when using the SIP protocol. The field allows a maximum of 256 characters(MAX_SIP_CALL_ID).

In **Appendix B – Bfv API Structures** section, in "**Result Structures**" section on page 1335 the CALL_RES structure to be updated to the following:

```
typedef struct {
       int call_type;
       char dest_id[MAX_DID];
       /* The rest are ISDN only */
       #define called_party_number dest_id
       char called_party_subaddress[MAX_DID];
       char calling_party_number[MAX_DID];
       char calling_party_subaddress[MAX_DID];
       char redir_number[MAX_DID];
       int redir_reason;
       char name_ident;
       int name_char_set;
       char connected_num[MAX_CONN_NUM];
       char sip_call_id[MAX_SIP_CALL_ID]
} CALL_RES;
```

In "**CALL_RES Structure Parameters**" on page 1343, the following to be added:

sip_call_id

A null-terminated ASCII string that identifies the SIP Call-ID. This value is only indicated for the SIP protocol.

BfvLineAnswer and BfvCallAccept

The following Input Field to be added to BfvLineAnswer on page 362 and BfvCallAccept on page 280.

```
Enum TperCallFaxTransportProtocol per_call_fax_transport_protocol;
```

per_call_fax_transport_protocol – Allows restriction of the fax transport protocol on a per-call basis.

The value can be one of the following:

PER_CALL_FAX_TRANSPORT_PROTOCOL_DEFAULT (0) – The fax transport protocol will be determined based on the call control configuration file settings.

PER_CALL_FAX_TRANSPORT_PROTOCOL_RTP_PREFERRED (1) – The fax transport protocol will be G.711 (RTP), if available, regardless of the call control configuration file settings; if not, it will be T.38.

Advanced TIFF Parameters

In the Brooktrout Bfv API Reference Manual in Appendix A under the User-Defined Configuration File there is a new parameter.

Parameter	Value
allow_zero_lines	Specifies whether BfvFaxRcvPageTiff should allow pages with zero-length scan lines. The TIFF-F specification does not allow this, so if such a page occurs, it is normally treated as an error. Setting the value of this parameter to non-zero will cause this to be treated as if it were a normally received page. Note: Enabling this feature may cause invalid TIFF-F files to be created. ***Value Type:*** decimal ***Default:*** 0 (disabled)

Configuring T.38 Fax Transport Parameters

The text of the *media_renegotiate_delay_inbound*, *media_passthrough_timeout_inbound*, *media_renegotiate_delay_outbound* and the *media_passthrough_timeout_outbound* parameters starting on page 1239 to be replaced with the text below:

Parameter	Value
media_renegotiate_delay_inbound	Controls media renegotiation to image (T.38) on inbound calls. If the gateway is responsible for media renegotiation, set this parameter to -1 and [fax_transport_protocol] to t38_only to disable initiating the media renegotiation to image. Setting this parameter to -1 and [fax_transport_protocol] to t38_first will cause the inbound side renegotiation to T.38 being controlled by [media_passthrough_timeout_inbound]. If the inbound side is responsible for media renegotiation to image, set this parameter to a value between 0 and 60000. Numbers from 0 to 3000 will cause the inbound side to renegotiate to T.38 after 3 seconds. Numbers from 3000 to 60000 indicate the number of milliseconds to delay before the inbound side attempts media renegotiation to T.38. Set this parameter to: -1 Disables media renegotiation on inbound calls if [fax_transport_protocol] is t38_only. See [media_passthrough_timeout_inbound] if [fax_transport_protocol] is t38_first. 0 to 3000 Waits 3 seconds before attempting to renegotiate the media to T.38. 3000 to 60000 Waits this number of milliseconds before attempting to renegotiate the media to T.38. ***Unit:*** ms ***Range:*** -1 and 0 to 60000 ***Value Type:*** decimal ***Default:*** 3000 (3 seconds)
media_passthrough_timeout_inbound	Sets the time before media renegotiation to image (T.38) will be attempted before doing fax passthrough on inbound calls. This timer is active only when [media_renegotiate_delay_inbound] is set to -1, [fax_transport_protocol] is set to t38_first and the module supports fax passthrough.

	Setting this parameter to -1 will suppress the sending of a REINVITE to T.38 and fax passthrough. Numbers from 0 to 3000 will cause the inbound side to wait 3 seconds before attempting media renegotiation to image (T.38). Numbers from 3000 to 60000 indicate the number of milliseconds to delay before attempting media renegotiation to image (T.38). Set this parameter to: -1 Suppress renegotiation to T.38 or fax passthrough. 0 to 3000 Waits 3 seconds before attempting renegotiation to T.38 and fax passthrough. 3000 to 60000 Number of milliseconds to wait before attempting renegotiation to T.38 and fax passthrough. ***Unit:*** ms ***Range:*** -1 and 0 to 60000 ***Value Type:*** decimal ***Default:*** 3000 (3 seconds)
media_renegotiate_delay_outbound	Controls media renegotiation to image (T.38) on outbound calls. If the gateway is responsible for media renegotiation, set this parameter to -1 to disable initiating the media renegotiation to image. If the outbound side is responsible for media renegotiation to image, set this parameter to a value between 0 and 60000. Numbers greater than 0 indicate the number of milliseconds to delay before attempting media renegotiation. A value of 0 will cause an immediate renegotiation, while -1 will wait for a renegotiation to image. Set this parameter to: -1 Disables media renegotiation on outbound calls. 0 Does not delay before attempting to renegotiate the media to image (T.38). >0 Waits this number of milliseconds before attempting to renegotiate the media to image (T.38). ***Unit:*** ms ***Range:*** -1 and 0 to 60000 ***Value Type:*** decimal ***Default:*** -1

media_passthrough_timeout_outbound	Sets the timer to fail over to fax passthrough when no T.38 is negotiated on outbound calls. This timer is active only when [media_renegotiate_delay_outbound] is set to -1, [fax_transport_protocol] is set to t38_first and the module supports fax passthrough. Numbers greater than 0, indicate the number of milliseconds to wait for T.38 negotiation before performing fax passthrough. A value of 0 will cause an immediate renegotiation to passthrough, while -1 will suppress renegotiation to fax passthrough. Set this parameter to: -1 Suppress renegotiation to fax passthrough. 0 Cause an immediate renegotiation to passthrough. >0 Number of milliseconds to wait for T.38 negotiation before performing fax passthrough. ***Unit:*** ms ***Range:*** -1 and 0 to 60000 ***Value Type:*** decimal ***Default:*** 4000

Brooktrout Fax Products Linux End User Guide

This section describes the changes to the Brooktrout Fax Products End user Guide based on the SDK version.

SDK 6.16.0

Linux UEFI Secure Boot

Redhat and other distributions have added support for Unified Extensible Firmware Interface (UEFI) secure boot. This functionality is available if the system includes the appropriate hardware and UEFI is supported by the system BIOS. UEFI with secure boot is designed to protect a system against malicious code being loaded and executed early in the boot process before the operating system has been loaded or in the kernel.

If UEFI with secure boot is enabled, Linux automatically runs in a mode where drivers must be signed with a key enrolled in the appropriate system Machine Owner Key (MOK). The point of a MOK is to give users the ability to run locally compiled kernels, boot loaders and modules/drivers not delivered by the distribution maintainer, and so on.

To support this, the first time the Brooktrout driver is installed on a system (using dinstall or dinstlib), the user has to create a locally self-signed certificate (X.509) key with a password that will be used to sign a rebuilt Brooktrout Linux driver module. Following this, the system must be rebooted, and the system MOK manager entered from the UEFI console. This will prompt for the password and allow you to complete enrollment of the new key in the system keyring.

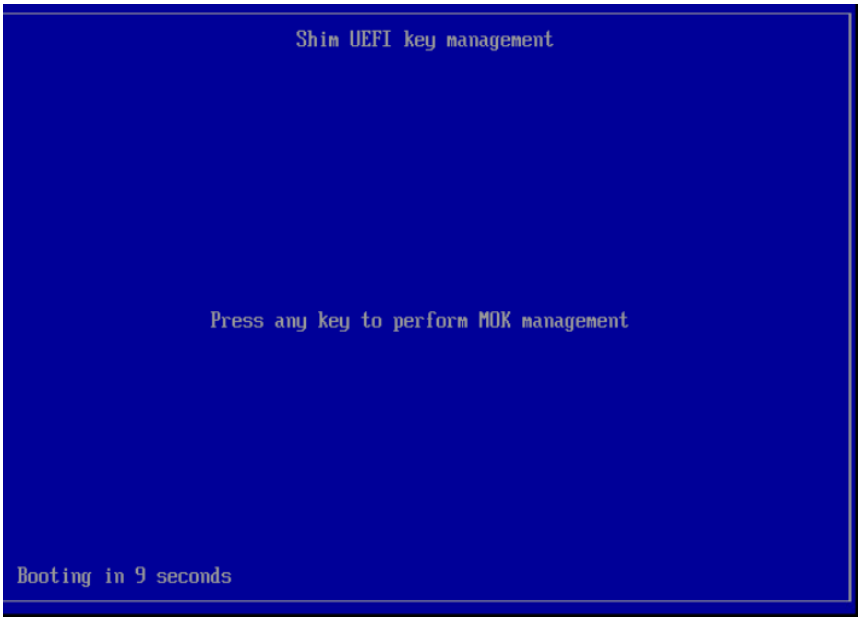
This process only needs to be performed one time, once enrolled in to the MOK, the key is available to use on all future system boots and driver rebuilds.

Initial installation asking for user password.

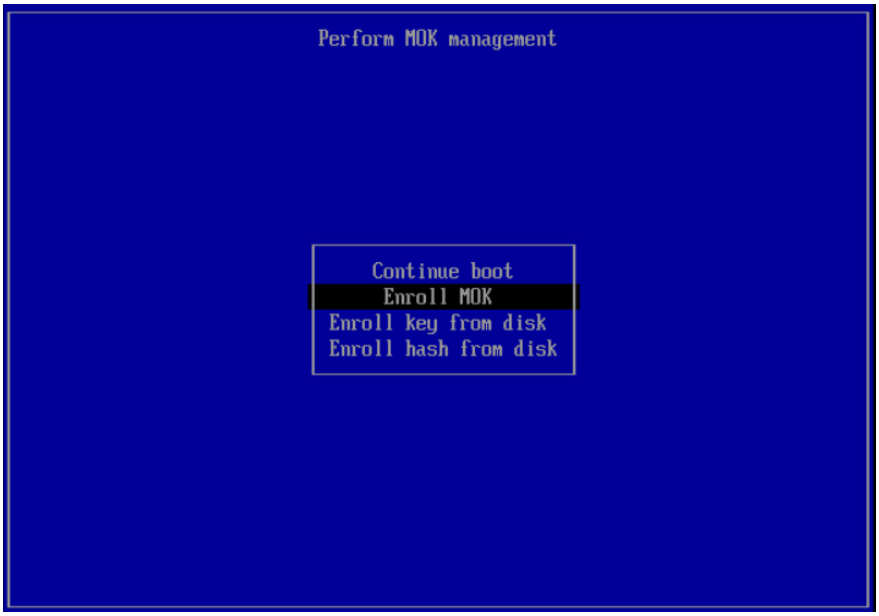
```
Installing Dialogic(R) Brooktrout(R) Device Driver Version 6.16.0
Maximum number of PCI/cPCI hardware modules (default 16): Maximum number of PCI/
cPCI hardware modules (default 16): Physical buffer size (default 32768): Applic
ation buffer size (default 10240): Machine ID, in hex (default 1): Do initial re
set (default 1): History enable (default 0): Configure advanced parameters (n)?
Configuring Boston driver:
    Max PCI hw modules 16
    Phys buf size 32768
    App buf size 10240
    Machine ID 1
        History disabled
Creating new Brooktrout key for secure boot.

A new key is being enrolled in the system keyring. Please enter
a password when prompted. To complete the enrollment request,
you must reboot the system, enter the MOK manager from the
UEFI console, and confirm the request by supplying the password.

The Dialogic Brooktrout driver will not function until these
steps are performed.

These steps only need to be performed once on this system.

input password:
```

```
input password:
input password again:

The enrollment of the new key has been requested. Before the
driver will function, you must complete the enrollment request by
rebooting the system, entering the MOK manager from the
UEFI console, and supplying the password just entered.

The Dialogic(R) Brooktrout(R) Driver has been installed on your system.
Driver installed and configured.

The Brooktrout Boston SDK Package (BRKTBOSsetup version 6.16.0)
is now installed.
```

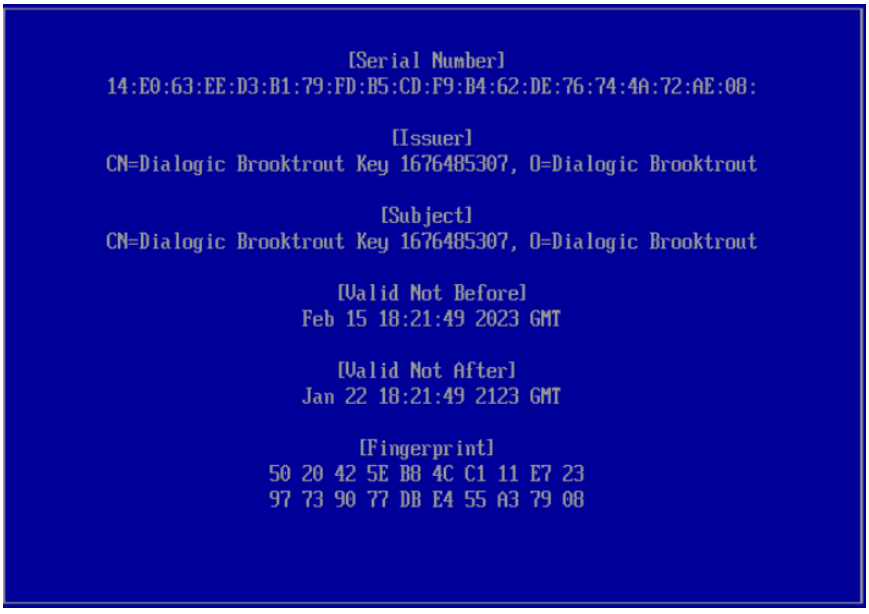
After rebooting MOK management screen.



Select Enroll MOK to add new key.

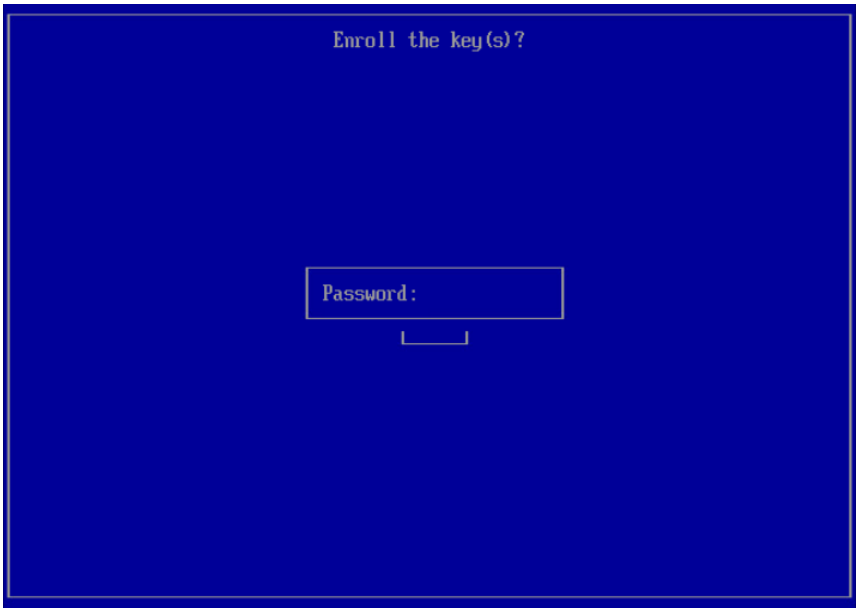


View key will show new key information to be enrolled into MOK.

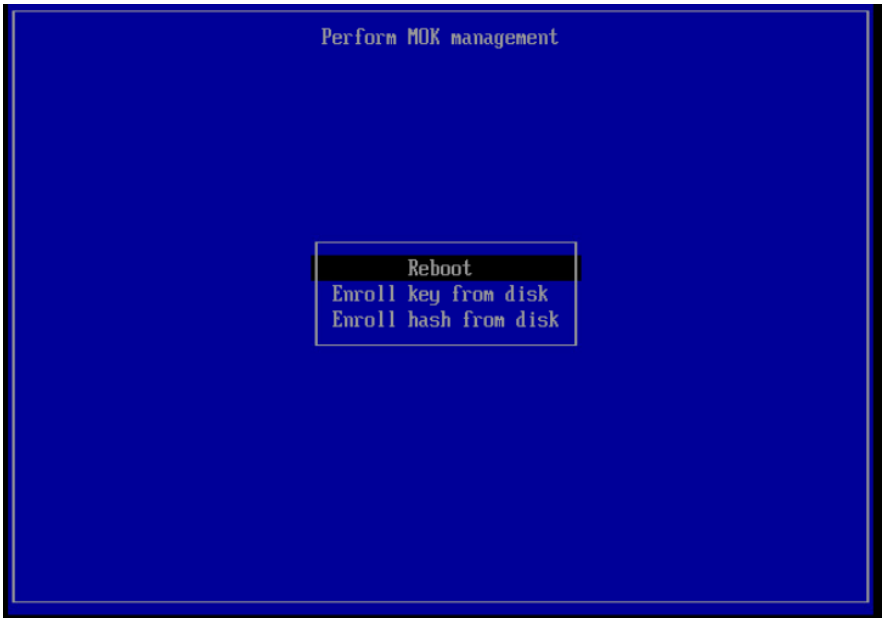


Selecting Continue will enroll key into MOK.

Enter Password used to create key in complete enrollment.



Once key is enrolled, system will reboot and have key going forward.



To view enrolled keys, the following commands can be used:

- On RH 7: `keyctl list %:.system_keyring`
- On RH 8 and RH 9: `keyctl list %:.platform`

To determine whether a driver is signed, you can view the output of the following command after locating the appropriate boston.ko driver file.

```
/usr/sbin/modinfo boston.ko
```

Driver Rebuilding for Linux Kernels Patches

This section provides an update to page 50 of the Fax Products SDK Install and Config guide.

The instructions describe how to re-compile the Dialogic Brooktrout driver on supported Linux platforms so that the driver can operate with any kernel patch version. Dialogic only supports official kernel patches as released by Red Hat. After you follow the procedure, the driver will support the exact version of the kernel currently running on your system, including architecture and variant.

Note: This feature only provides support for the Dialogic Brooktrout driver, the kernel mode code. Other parts of the Brooktrout SDK (the user mode code) might also need rebuilding and this feature will not help in these situations.

Supported versions include:

- Red Hat Enterprise Linux 9.0 (5.14.0-70.el9), 64-bit
- Red Hat Enterprise/CentOS Linux 8.0 (4.18.0-80.el8), 64-bit
- Red Hat Enterprise/CentOS Linux 7.0 (3.10.0-123.el7), 64-bit

Additional Packages Required for Red Hat Linux 7.0 and Later x64

The following packages are required for the Dialogic Brooktrout SDK to be used on Red Hat Linux 7.0 and later x64 systems. These packages are on your Linux installation media in the Packages directory.

Each of the packages listed has a version number associated with it following the name of the package before additional text about the Linux version and the architecture. The package version will depend on the exact version of Linux being used.

Below, the packages are listed without the exact version number, and "*" is used instead. The version number can be determined by examining the installation media.

Runtime-only systems	Development systems
`libstdc++-*.i686`	`libstdc++-devel-*.i686` `glibc-devel-*.i686`
Required for SR140 only	
`perl-Digest-MD5`	`perl-Digest-MD5`