1. BIND 9.12 Support

BIND 9 is an open source implements of the Domain Name System (DNS) protocols for the Internet that is suitable for use in high-volume and high-reliability applications. Domain Name System Security Extensions (DNSSEC) extends standard DNS to provide a measure of security. BIND supports the full set of DNSSEC standards. [[1]](#endnote-1)

Because BIND is complete software package with extensive documentation, this document will only cover the specific steps for configuring BIND with the CrypTech Alpha and does not provide information on configuring BIND. More information on BIND and DNSSEC can be found at <https://ftp.isc.org/isc/dnssec-guide/html/dnssec-guide.html> which is the official BIND documentation. Please see that documentation for more details and for troubleshooting. BIND is distributed in binary form on Windows, but for other operating systems such as Linux, it needs to be built from source. The following guide was created using BIND 9.12.1-P2 and OpenSSL 1.0.2h.

**OpenSSL-based PKCS#11**

The CrypTech Alpha HSM doesn’t not support Native PKCS#11 because it doesn’t support every cryptographic operation that BIND 9 may need. As mentioned in the BIN documentation, OpenSSL-based PKCS#11 uses a modified version of the OpenSSL library and BIND must be built using that library.

**Updating the Paths**

The following examples will use the following paths:

* Patched Openssl 🡪 /opt/pkcs11/usr
* BIND 🡪 /opt/bind9
* BIND sysconfdir 🡪 /etc/bind

Because the standard install paths for OpenSSL and the BIND tools are not being used, it’s very important to set the PATH environment variable. The secure path should also be set using ‘sudo visudo’. The ‘LD\_LIBRARY\_PATH’ environment variable is the last path to be set as well. It is recommended that these variables be set before making OpenSSL and BIND from source. ‘/opt/bind9’ should be substituted with the path used with --prefix when configuring the BIND build. To make the change permanent, you will also need to add it to either you ~/.profile or ~/.bashrc file.

$ export PATH=/opt/bind9/sbin:/opt/bind9/bin:/opt/pkcs11/usr/bin:/opt/pkcs11/usr/lib${PATH}

$ export LD\_LIBRARY\_PATH=/opt/pkcs11/usr/lib:${LD\_LIBRARY\_PATH}

You will also need to update the secure path for sudo by running the following command.

$sudo visudo

That will open an editor. Find a line with similar to the one below and edit the secure\_path to include ‘/opt/bind9/sbin:/opt/bind9/bin:opt/pkcs11/usr/bin:/opt/pkcs11/usr/lib’ at the beginning of the path.

Defaults secure\_path=”…”

**Troubleshooting**

Regardless of the error given by BIND either during the signing process or while making the project, most errors are actually caused by the environment variables not being set, by the CrypTech Alpha software not being installed, or by cryptech\_muxd, multiplexer daemon, not running correctly in the background.

* 1. Build BIND to work with the CrypTech Alpha
		1. Download Bind

Even though we will install OpenSSL first, downloading BIND is necessary because it contains an important patch.

$ wget http://ftp.isc.org/isc/bind9/9.12.1-P2/bind-9.12.1-P2.tar.gz

$ tar zxf bind-9.12.1-P2.tar.gz

* + 1. Download OpenSSL Source

This manual has been tested using OpenSSL 1.0.2h. That version of OpenSSL has the required patch needed to build OpenSSL for BIND. If you want to use another version, verify that that patch exists in the bind-9/bin/pkcs11 directory.

$ wget http://www.openssl.org/source/openssl-1.0.2h.tar.gz

$ tar zxf openssl-1.0.2h.tar.gz

* + 1. Patch OpenSSL

These directions use OpenSSL 1.0.2h and BIND 9.12.1-P2 for its examples. If other versions are used, that versions in the following examples should change.

$patch -p1 -d openssl-1.0.2h \

 < bind-9.12.1-P2/bin/pkcs11/openssl-1.0.2h-patch

Details about the how OpenSSL is patched to provide PKCS#11 support and the different flavors of support provided has been documented in the BIND 9 Administrator Reference Manual.

OpenSSL-based PKCS#11 mode uses a modified version of the OpenSSL library; stock OpenSSL does not fully support PKCS#11. ISC provides a patch to OpenSSL to correct this. This patch is based on work originally done by the OpenSolaris project; it has been modified by ISC to provide new features such as PIN management and key-by-reference.

There are two "flavors" of PKCS#11 support provided by the patched OpenSSL, one of which must be chosen at configuration time. The correct choice depends on the HSM hardware:

* Use ’crypto-accelerator’ with HSMs that have hardware cryptographic acceleration features, such as the SCA 6000 board. This causes OpenSSL to run all supported cryptographic operations in the HSM.
* Use ’sign-only’ with HSMs that are designed to function primarily as secure key storage devices, but lack hardware acceleration. These devices are highly secure but are not necessarily any faster at cryptography than the system CPU --- often, they are slower. It is therefore most efficient to use them only for those cryptographic functions that require access to the secured private key, such as zone signing, and to use the system CPU for all other computationally-intensive operations. The AEP Keyper is an example of such a device.

The modified OpenSSL code is included in the BIND 9 release, in the form of a context diff against the latest versions of OpenSSL. OpenSSL 0.9.8, 1.0.0, 1.0.1 and 1.0.2 are supported; there are separate diffs for each version. In the examples to follow, we use OpenSSL 0.9.8, but the same methods work with OpenSSL 1.0.0 through 1.0.2.

* BIND 9 Administrator Reference Manual
	+ 1. Build OpenSSL

Per the BIND 9 Administrator Reference Manual, when building OpenSSL, place it in a non-standard location so that it does not interfere with other OpenSSL libraries on the system. This manual will use the location in the BIND 9 reference material, "/opt/pkcs11/usr" and will be used when configuring BIND. After configuring the OpenSSL build, you will need to run ‘make depend’. Pay careful attention to the output of ‘make’ and ‘make test’ to make sure OpenSSL has been built correctly.

$ cd openssl-1.0.2h

$ ./Configure -fPIC linux-x86\_64 \

 --pk11-libname=/usr/lib/libcryptech-pkcs11.so \

 --pk11-flavor=sign-only \

 --prefix=/opt/pkcs11/usr

$ make depend

$ make

$ make test

$ sudo make install

* + 1. Build BIND 9

Once OpenSSL has been made and installed, BIND 9 must be installed with the following configuration.

$ cd bind-9.12.1-P2

$ ./configure --enable-threads \

 --with-openssl=/opt/pkcs11/usr \

 --with-pkcs11=/usr/lib/libcryptech-pkcs11.so \

 --sysconfdir=/etc/bind \

 --prefix=/opt/bind9

$ make

$ make test

$ sudo make install

The sysconfdir is set to ‘/etc/bind’ in this example because it’s used later in this document; however, ‘/etc/bind’ does not need to be used.

PKCS#11 Tools

BIND 9 includes a minimal set of tools to operate the HSM, including pkcs11-keygen to generate a new key pair within the HSM, pkcs11-list to list objects currently available, pkcs11-destroy to remove objects, and pkcs11-tokens to list available tokens.

In UNIX/Linux builds, these tools are built only if BIND 9 is configured with the --with-pkcs11 option. (Note: If --with-pkcs11 is set to "yes", rather than to the path of the PKCS#11 provider, then the tools will be built but the provider will be left undefined. Use the -m option or the PKCS11\_PROVIDER environment variable to specify the path to the provider.)

* BIND 9 Administrator Reference Manual

The PATH environment variable will have to be updated if you change the default location using --prefix. See **Updating the Paths** section above.

* 1. Complete Zone Signing Example

Once BIND has been successfully built, you can use the PKCS#11 tools to generate keys and sign zones. This example will go through the signing of a specific zone file and show the expected output. The following zone file will be used for an example domain, ‘example.net’. All commands use ‘sudo’ so the BIND pkcs#11 tools can use the file '/usr/lib/libcryptech-pkcs11.so’.

The following three steps will be considered:

1. Create the KSK (key signing key) and ZSK (zone signing key) key pairs
2. Add the public key to the zone file
3. Sign the zone

For more information on other BIND topics such as creating a DS-set and replication, please see the BIND documentation.

The following zone file will be used in this example. The filename is ‘example.com’.

$ORIGIN example.net.

$TTL 86400

@ IN SOA dns1.example.net. hostmaster.example.net. (

 2001062501 ; serial

 21600 ; refresh after 6 hours

 3600 ; retry after 1 hour

 604800 ; expire after 1 week

 86400 ) ; minimum TTL of 1 day

 IN NS dns1.example.net.

 IN MX 10 mail.example.net.

dns1 IN A 127.0.0.1

server1 IN A 127.0.0.1

mail IN CNAME server1

www IN CNAME server1

* + 1. Creating the Key-Signing Key and Zone-Signing Key

Before proceeding, please make sure the the CrypTech Alpha is connected to the computer and that cryptech\_muxd is running without any errors. To create the keys, run the following commands in the terminal. In the folder of your zone.

$ sudo pkcs11-keygen -b 1024 -l example-net-ksk

$ sudo pkcs11-keygen -b 2048 -l example-net-zsk

Depending on the size of the key, it may take a few minustes to generate. Each command should return ‘Key pair generation complete.’ The private key will remain on the HSM, but you’ll have to add the public keys to the zone file.

To confirm that the key exists:

$ sudo pkcs11-list

Enter PIN:

object[0]: handle 2147483658 class 3 label[8] ’example-net-ksk’ id[0]

object[1]: handle 2147483657 class 2 label[8] ’example-net-ksk’ id[0]

dnssec-keyfromlabel generates a key pair of files that referencing a key object stored in a cryptographic hardware service module (HSM). The private key file can be used for DNSSEC signing of zone data as if it were a conventional signing key created by dnssec-keygen, but the key material is stored within the HSM, and the actual signing takes place there. [[2]](#endnote-2) The algorithm must be set using the ‘-a’ flag and the engine must also be set using ‘-E pkcs11’.

$ sudo dnssec-keyfromlabel –a RSAMD5 –E pkcs11 -l example-net-zsk example.net

Enter PIN:

Kexample.net+001+37518

$ sudo dnssec-keyfromlabel –a RSAMD5 –E pkcs11 -l example-net-ksk -f KSK example.net

Enter PIN:

Kexample.net+001+57242

It’s important to save the values returned from ‘dnssec-keyfromlabel’ because it tell the name of the key file for the particular key pair. ‘dnssec-keyfromlabel’ will create a set of two file. One will have the extension .key and the other .private. The .key file should be included in the zone file as it the public key. The .private file is not the actual private key because the private key is on the HSM, but it gives information on how to sign using the HSM.

* + 1. Adding the Public Key to the Zone File

The public key needs to be added to the zone file. This can be done using BIND’s ‘$include’ statement as shown below.

$ORIGIN example.net.

$TTL 86400

@ IN SOA dns1.example.net. hostmaster.example.net. (

 2001062501 ; serial

 21600 ; refresh after 6 hours

 3600 ; retry after 1 hour

 604800 ; expire after 1 week

 86400 ) ; minimum TTL of 1 day

 IN NS dns1.example.net.

 IN MX 10 mail.example.net.

dns1 IN A 127.0.0.1

server1 IN A 127.0.0.1

mail IN CNAME server1

www IN CNAME server1

$include /etc/bind/zones/example.net.test/Kexample.+001+37518.key

$include /etc/bind/zones/example.net.test/Kexample.+001+57242.key

* + 1. Signing the Zone

Asdsad dnssec-signzone is used to sign the zone. The ‘-S’ option is for smart signing and will create the appropriate .signed file.

$sudo dnssec-signzone -S -E pkcs11 example.net

Enter PIN:

Verifying the zone using the following algorithms: RSAMD5.

Zone fully signed:

Algorithm: RSAMD5: KSKs 1 active, 0 stand-by, 0 revoked

 ZSKs: 1 active, 0 stand-by, 0 revoked

example.net.signed

This is the resulting zone file.

example.net.signed:

; File written on Wed Jun 20 07:01:50 2018

; dnssec\_signzone version 9.12.1-P2

example.net. 86400 IN SOA dns1.example.net. hostmaster.example.net. (

 2001062501 ; serial

 21600 ; refresh (6 hours)

 3600 ; retry (1 hour)

 604800 ; expire (1 week)

 86400 ; minimum (1 day)

 )

 86400 RRSIG SOA 1 2 86400 (

 20180720110145 20180620110145 37518 example.net.

 MU3U0aMVBuz1YzaZfbBhIlmqr2/yBvJEUqqC

 Ha1uGV0c2ghcH39NYfxxiQpbhdnNBgZrMY2D

 0fr1igsET59TvQ5qopxbTsWshnsmHE0+JiG7

 zT1aUei5Iqste+lA/6y7YBoeoqRQ8qum7yY3

 DkZyv8DdqyqU4ZPL6IqzHWvdSp4= )

 86400 NS dns1.example.net.

 86400 RRSIG NS 1 2 86400 (

 20180720110145 20180620110145 37518 example.net.

 mB2DNklqKxCAD1ysXkyXs8iqEOPRu1yADQ5t

 b6x8nPkHMBR0cHZcz64WvxuGBomUksN40uOl

 rYWw7uGQdQOu+fBa/qO6FDLMKGte4UHw58Ic

 M4/kjyOii4L+D2UCZSwF5UtJrrW2l+XUfyvv

 fVjpyWHfnrCHdJc0YPT7htP739I= )

 86400 MX 10 mail.example.net.

 86400 RRSIG MX 1 2 86400 (

 20180720110145 20180620110145 37518 example.net.

 G/1eQldAcctWoPgiPMvvNFIFjmjWYMTsUIcC

 dR41qsEn7CTHchu0FmWt7x2gOcLW+1xSkQ6K

 gTLQQf5EQGT7+bRrHkKip/cqWU9zL+6mF+jM

 yLAwMhWuT7zl3ug779X6EY8GRignqSELyCa4

 e3Rzp6aQwolQ+7SXHwDVx7vEUpY= )

 86400 NSEC dns1.example.net. NS SOA MX RRSIG NSEC DNSKEY

 86400 RRSIG NSEC 1 2 86400 (

 20180720110145 20180620110145 37518 example.net.

 G0EpXW59+VM7hq9Q1abcSsVAHEs6EIJ6H/r5

 AIKbdXHXdmWV2u1SjqQ8UeCks9RXk0LWeeUq

 v5RZAFpJpVmXpislOduAk5uzj3W07oF0Xnmp

 XIykg9VAcp/TBek2Dtol3j6CL6Yl9SYKL0VM

 AV1lnXwxUM5fG50gqxFDnFj+Aes= )

 86400 DNSKEY 256 3 1 (

 AwEAAeKkz7moqoWX8dx3K5iOr5YeQnrIOLfE

 4OUtep23G+NmMcOawDELBMphBlCAcPjtkAvF

 QOdty6K+OsCeRh/+0PKX+cTisbV/HdjmJ58/

 gSFckeepiueoomkzewdJoNPGW7O//3rn8dqz

 1QJett+QHQHh0qH6hSe1FT4mVOkfko6h

 ) ; ZSK; alg = RSAMD5 ; key id = 37518

 86400 DNSKEY 257 3 1 (

 AwEAAcJ03xso21S5H4AxCd4AEAptff2EcL2e

 G4bYgMINAVYSla28Y6MbUDFg5wUXGHAbc5cz

 Tjml4SiReALoiKLQ8Httiq1Iwxcj4fzaYhV3

 4BIy9fqGGalY2TFHtmcxCXSMCFPO7nihkzIn

 O4koi9fGGkSeUn8LhT5n8TgbD0FssBLud1X1

 hddy4Fky6LMZWsri5veF150ZdgszDFqcB2UG

 BH1fyJzETKlR7DgHgbsgh3B8L8Ya7q98iydk

 orO6nJe4HdDsS9egDdfra9pW71skDUcyye4K

 WGQ8/x7wUHsJ3joFrI7CzFDn1q+Nq4Z4Z+Wt

 sB00noQaiopMv7u57gHfmt8=

 ) ; KSK; alg = RSAMD5 ; key id = 57242

 86400 RRSIG DNSKEY 1 2 86400 (

 20180720110145 20180620110145 37518 example.net.

 gyUFDNjJERDKcxOmzJpNJZp1jrcsf+q2wAeM

 KMEMFx0GKUy3jTk97D5JMbjAbfQMC/uFu6km

 3uXS9XPec0tjBHBR7TOmd2nTi67ggZz2OYBW

 pGxX723t9SRF5XnAB07C4ucQZa6goSs74q3Y

 ioFPl+sjTkVXmtGh7Bc35LlR9zk= )

 86400 RRSIG DNSKEY 1 2 86400 (

 20180720110145 20180620110145 57242 example.net.

 KdeLT56n/CxjZrcfBY4l2982wRX4LHpssvMg

 t2FOF/rVv2v3/KfZuBTKqW6kwnut5X9ioLD4

 qBH4MVBu5Q0Tzrmiggwo+0xR9oDIyDakK6JJ

 cfpVvgKcMz2rb2Y55Xrp0pM7FCqdrTS003i8

 3RJGFzLfkec59l6lMOz16m3ZoJGxG8NCdqhv

 c7/+VNhdkcMkEKFdRzPG+UJD8cboraZjrEV4

 /tzgYIA5Z9ipimCcqvCLac3AAkS4TA/R5hVn

 NDyUnSOLtH/uxC/RWI6m2Se8JOBbQh2ti1Zh

 /bphBmUBumfuCshq60x7bN0smEcaS1Ph+I43

 eWjX5/Wt0VHkTVrefw== )

dns1.example.net. 86400 IN A 127.0.0.1

 86400 RRSIG A 1 3 86400 (

 20180720110145 20180620110145 37518 example.net.

 gYJl92B+fOMZlP8r1pEObbF738KH/MkiMfyg

 p85ruUtT1H5TXPZJFaTBoL881v25K7eunyOR

 qAriFE7NAWzhV7ubuWZB8F1E1dMaVL9TWKJF

 wXMsCPIH6jHNdsodlUyV+CGeJgxOxKc77KUf

 rgTPwh2uMF3wbxfU1HXP+0aVaHU= )

 86400 NSEC mail.example.net. A RRSIG NSEC

 86400 RRSIG NSEC 1 3 86400 (

 20180720110145 20180620110145 37518 example.net.

 ulLpUUkbDC27qMcgTlvc0rIm7SZR22RL1G+S

 R8VZldkeFF8uJ3PYnfKe+ExEsCMB3F5tMNhT

 3YRJf1v2ES6D+YPT7yEV6xBxFg08oIwqUDVd

 5gnhwobH/4aGkMlZTJyWyqlAgIMQEHyR1+z6

 iRah6RE/1kmE4S0M9gIksLbcgFg= )

www.example.net. 86400 IN CNAME server1.example.net.

 86400 RRSIG CNAME 1 3 86400 (

 20180720110145 20180620110145 37518 example.net.

 ekjwXqc/+VyDqGWiF/Y8ThDKoZDNNvQGbLdu

 wzBbJVX8juCvCJmi+qK7p15Dj4lC2fmSeyML

 zaMz6rMh5Ynl1jozLw06yfaEpT/41P3VohVK

 bJpMdy1IqP4bbyw4j/GS8kSj3Kk6jGfBxZ+6

 2oWH7mo33TO2cA8uuIkTT7zx87s= )

 86400 NSEC example.net. CNAME RRSIG NSEC

 86400 RRSIG NSEC 1 3 86400 (

 20180720110145 20180620110145 37518 example.net.

 dq8O5chwXvgsI055AgPNSQaGbjDO5XmnENpP

 ZTC3B4lwhVf09noM+k3l4HJRew//xPk2RSE1

 Wf76gyq+CGKqrjci2jRmt+4EedTrn6nrnkxs

 EHrqd//4ZV5wz0AoDKwudPzHwb0yrBqPfH12

 f6C87sxqE/OI7yrihqQtl87XfTQ= )

mail.example.net. 86400 IN CNAME server1.example.net.

 86400 RRSIG CNAME 1 3 86400 (

 20180720110145 20180620110145 37518 example.net.

 c836bzTL5DrvUxNjwRT2/bPScv7nLMtVZcAk

 w3XaFUJlwjGkRvcImyIVAr0d/+/VnjtKlfE3

 /miwcYJgoteCrTZx5xSFFSetSDob/sZqCwvm

 yeCAwCOSoFEiToms1Prb+NKpkGuNySTg3TrW

 BL1n+HCJ2A8VqoaciYuk4cwNlpM= )

 86400 NSEC server1.example.net. CNAME RRSIG NSEC

 86400 RRSIG NSEC 1 3 86400 (

 20180720110145 20180620110145 37518 example.net.

 JH5q4sxWFlDTFixo2gcAB78eR7O3TkTtwXAU

 4278jQlasCGy2g2PcmGYhpAhP0JWfBEd/22M

 vCZLCkIxDSZnBerntCUIYpxYs5tOWdYwo55p

 G34xPAZI1JVjmZ2ctwO7zgq4jqDU+pQmv7EW

 v2jQG2fBIZyNaRnvdNjBtAqLLhg= )

server1.example.net. 86400 IN A 127.0.0.1

 86400 RRSIG A 1 3 86400 (

 20180720110145 20180620110145 37518 example.net.

 bTexyOvXy8igKm2HP9GAALvfZ1lEk0wYINp4

 9nqogyUU74dZchOOmkos0ZOWvvqr1kZ3PR8X

 WD/JzavV7N7Q1iw7Um2rZrCFCKepXJ9Gfban

 dQe1rP5N648Pb2Q+zmdx5hE4FrLGYUOOZH9/

 H1p36vxSq1pP2XPRE3MueopKcNY= )

 86400 NSEC www.example.net. A RRSIG NSEC

 86400 RRSIG NSEC 1 3 86400 (

 20180720110145 20180620110145 37518 example.net.

 KcWhgubez09s298vN7WfOoHiY18ROxsBZCB2

 HUn3z0KHWB5W2vjZxInH8QcrBF+KctJemOlP

 xSzPkO+lvbGgnRDSH+RWsvXfSVz5veHSmrQn

 tqd9lBjpjwfMykiRNq2jA2gYf2Bl5HbddPPh

 EfVgFglQDHvKA2PW4qJ3JSmD/r8= )

* + 1. Running named with automatic zone re-signing

If you want named to dynamically re-sign zones using HSM keys, and/or to sign new records inserted via nsupdate, then named must have access to the HSM PIN. In OpenSSL based PKCS#11, this is accomplished by placing the PIN into the openssl.cnf file (in the above examples, /opt/pkcs11/usr/ssl/openssl.cnf).

The location of the openssl.cnf file can be overridden by setting the OPENSSL\_CONF environment variable before running named.

Sample openssl.cnf:

openssl\_conf = openssl\_def

[ openssl\_def ]

engines = engine\_section

[ engine\_section ]

pkcs11 = pkcs11\_section

[ pkcs11\_section ]

PIN = <PLACE PIN HERE>

This will also allow the dnssec-\* tools to access the HSM without PIN entry. (The pkcs11-\* tools access the HSM directly, not via OpenSSL, so a PIN will still be required to use them.)

Placing the HSM’s PIN in a text file in this manner may reduce the security advantage of using an HSM. Be sure this is what you want to do before configuring the system in this way.

* BIND 9 Administrator Reference Manual
1. References and Endnotes
1. <https://ftp.isc.org/isc/dnssec-guide/html/dnssec-guide.html#hardware-security-modules> [↑](#endnote-ref-1)
2. <https://ftp.isc.org/isc/bind9/cur/9.10/doc/arm/man.dnssec-keyfromlabel.html> [↑](#endnote-ref-2)