Exhibit 15

```
/*----------------------------------------------------------------------
 * key.h :     Declarations and Definitions for Key Engine for BSD.
 *
 * Copyright 1995 by Bao Phan, Randall Atkinson, & Dan McDonald,
 * All Rights Reserved.  All rights have been assigned to the US
 * Naval Research Laboratory (NRL).  The NRL Copyright Notice and
 * License Agreement governs distribution and use of this software.
 *
 *  Patents are pending on this technology.  NRL grants a license
 *  to use this technology at no cost under the terms below with
 * the additional requirement that software, hardware, and
 * documentation relating to use of this technology must include
 * the note that:
 *      This product includes technology developed at and
 *      licensed from the Information Technology Division,
 *  US Naval Research Laboratory.
 *
 ----------------------------------------------------------------------*/
/*----------------------------------------------------------------------
#   @(#)COPYRIGHT   1.1a (NRL) 17 August 1995

COPYRIGHT NOTICE

All of the documentation and software included in this software
distribution from the US Naval Research Laboratory (NRL) are
copyrighted by their respective developers.

This software and documentation were developed at NRL by various
people.  Those developers have each copyrighted the portions that they
developed at NRL and have assigned All Rights for those portions to
NRL.  Outside the USA, NRL also has copyright on the software
developed at NRL. The affected files all contain specific copyright
notices and those notices must be retained in any derived work.

NRL LICENSE

NRL grants permission for redistribution and use in source and binary
forms, with or without modification, of the software and documentation
created at NRL provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright
   notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
   notice, this list of conditions and the following disclaimer in the
   documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software
   must display the following acknowledgement:

     This product includes software developed at the Information
     Technology Division, US Naval Research Laboratory.

4. Neither the name of the NRL nor the names of its contributors
   may be used to endorse or promote products derived from this software
   without specific prior written permission.

THE SOFTWARE PROVIDED BY NRL IS PROVIDED BY NRL AND CONTRIBUTORS ``AS
IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED
TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A
PARTICULAR PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL NRL OR
CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

```

```
The views and conclusions contained in the software and documentation
are those of the authors and should not be interpreted as representing
official policies, either expressed or implied, of the US Naval
Research Laboratory (NRL).

-----------------------------------------------------------------------*/


/*
 * PF_KEY messages
 */

#define KEY ADD         1
#define KEY DELETE      2
#define KEY UPDATE      3
#define KEY GET         4
#define KEY ACQUIRE     5
#define KEY GETSPI      6
#define KEY REGISTER    7
#define KEY EXPIRE      8
#define KEY DUMP        9
#define KEY_FLUSH       10

#define KEY VERSION     1
#define POLICY_VERSION  1

/*
 * Security association state
 */

#define K USED          0x1    /* Key used/not used */
#define K UNIQUE        0x2    /* Key unique/reusable */
#define K LARVAL        0x4    /* SPI assigned, but sa incomplete */
#define K ZOMBIE        0x8    /* sa expired but still useable */
#define K DEAD          0x10   /* sa marked for deletion, ready for
reaping */
#define K_INBOUND       0x20   /* sa for inbound packets, ie. dst=myhost
*/
#define K OUTBOUND      0x40   /* sa for outbound packets, ie.
src=myhost */

/*
 * Structure for key message header.
 * PF KEY message consists of key msghdr followed by
 * src sockaddr, dest sockaddr, from sockaddr, key, and iv.
 * Assumes size of key message header less than MHLEN.
 */

struct key msghdr {
  u short key msglen;   /* length of message including
  src/dst/from/key/iv */
  u char  key msgvers;  /* key version number */
  u char  key msgtype;  /* key message type, eg. KEY ADD */
  pid_t   key pid;      /* process id of message sender */
  int     key seq;      /* message sequence number */
  int     key errno;    /* error code */
  u int8  type;         /* type of security association */
  u int8  state;        /* state of security association */
  u int8  label;        /* sensitivity level */
  u int32 spi;          /* spi value */
  u int8  keylen;       /* key length */
  u int8  ivlen;        /* iv length */
  u int8  algorithm;    /* algorithm identifier */
  u int8  lifetype;     /* type of lifetime */
  u int32 lifetime1;    /* lifetime value 1 */
  u_int32 lifetime2;    /* lifetime value 2 */
};
```

```

struct key msgdata {
  struct sockaddr *src;    /* source host address */
  struct sockaddr *dst;    /* destination host address */
  struct sockaddr *from;   /* originator of security association */
  caddr t iv;              /* initialization vector */
  caddr t key;             /* key */
  int ivlen;               /* key length */
  int keylen;              /* iv length */
};

struct policy msghdr {
  u short policy msglen;   /* message length */
  u char  policy msgvers;  /* message version */
  u char  policy msgtype;  /* message type */
  int     policy seq;      /* message sequence number */
  int     policy_errno;    /* error code */
};


#ifdef KERNEL

/*
 * Key engine table structures
 */

struct socketlist {
  struct socket *socket;     /* pointer to socket */
  struct socketlist *next;   /* next */
};

struct key tblnode {
  int alloc count;               /* number of sockets allocated to
  secassoc */
  int ref_count;                 /* number of sockets referencing secassoc
  */
  struct socketlist *solist;     /* list of sockets allocated to secassoc
  */
  struct ipsec assoc *secassoc; /* security association */
  struct key_tblnode *next;      /* next node */
};

struct key allocnode {
  struct key tblnode *keynode;
  struct key_allocnode *next;
};

struct key so2spinode {
  struct socket *socket;         /* socket pointer */
  struct key_tblnode *keynode;  /* pointer to tblnode containing secassoc
  */
                  /*  info for socket  */
  struct key_so2spinode *next;
};

struct key registry {
  u int8 type;            /* secassoc type that key mgnt. daemon can
  acquire */
  struct socket *socket;  /* key management daemon socket pointer */
  struct key_registry *next;
};

struct key acquirelist {
  u int8 type;                    /* secassoc type to acquire */
  struct sockaddr_in6 target;    /* destination address of secassoc */
  u int32 count;                  /* number of acquire messages sent */
  u_long expiretime;              /* expiration time for acquire message */
  struct key_acquirelist *next;
};

struct keyso cb {
  int ip4 count;         /* IPv4 */
  int ip6 count;         /* IPv6 */
  int any_count;         /* Sum of above counters */
};

#endif

/*
 * Useful macros
 */

#ifndef KERNEL
#define K Malloc(p, t, n) (p = (t) malloc((unsigned int)(n)))
#define KFree(p) free((char *)p);
#else
#define K Malloc(p, t, n) (p = (t) malloc((unsigned long)(n), M_SECA,
M DONTWAIT))
#define KFree(p) free((caddr_t)p, M_SECA);
#endif /* KERNEL */

#ifdef KERNEL
void   key init  _P((void));
void   key cbinit   P((void));
void   key inittables   P((void));
int    key_secassoc2msghdr __P((struct ipsec_assoc *, struct key_msghdr
*,
                struct key_msgdata *));
int    key_msghdr2secassoc __P((struct ipsec_assoc *, struct key_msghdr
*,
                struct key_msgdata *));
int    key add   P((struct ipsec assoc *));
int    key delete   P((struct ipsec assoc *));
int    key_get   P((u int, struct sockaddr *, struct sockaddr *, u_int32,
            struct ipsec assoc **));
void   key flush   P((void));
int    key dump  _P((struct socket *));
int    key_getspi   P((u int, struct sockaddr *, struct sockaddr *,
               u int32 *));
int    key update  _P((struct ipsec assoc *));
int    key register  _P((struct socket *, u_int));
void   key unregister   P((struct socket *, u int, int));
int    key acquire  _P((u int, struct sockaddr *, struct sockaddr *));
int    getassocbyspi   P((u int, struct sockaddr *, struct sockaddr *,
                  u int32, struct key tblnode **));
int    getassocbysocket   P((u_int, struct sockaddr *, struct sockaddr *,
                 struct socket *, u int, struct key_tblnode **));
void   key free  _P((struct key tblnode *));
int    key output   P((struct mbuf *, struct socket *));
int    key_usrreq __P((struct socket *, int, struct mbuf *, struct mbuf
*,
               struct mbuf *));
#endif
```