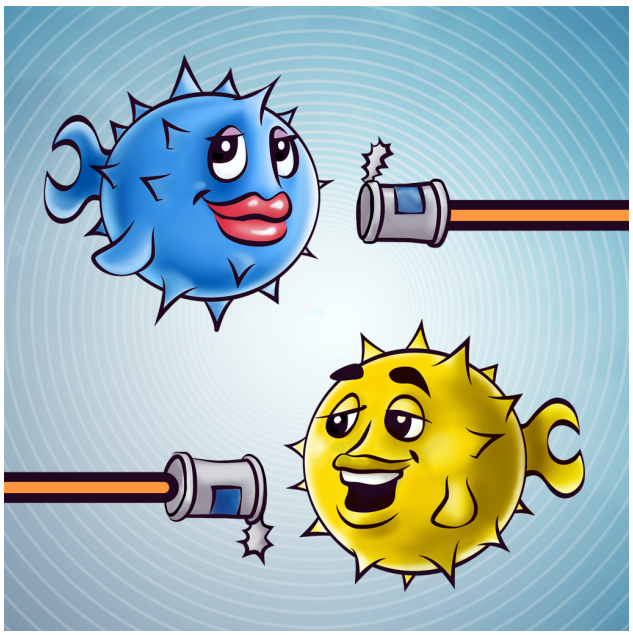
*Open*IKED

Reyk Floeter (reyk@openbsd.org)

February 2013

Abstract

This paper introduces the OpenIKED project[14], the latest portable subproject of OpenBSD[13]. OpenIKED is a FREE implementation of the most advanced Internet security "Internet Key Exchange version 2 (IKEv2)" Virtual Private Network (VPN) protocol using the strongest security, authentication and encryption techniques. The project was born in need of a modern Internet Protocol Security (IPsec) implementation for OpenBSD, but also for interoperability with the integrated IKEv2 client since Windows 7 and to provide a compliant solution for the US Government IPv6 (USGv6) standard. The project is still under active development; it was started by Reyk Floeter as "iked" for OpenBSD in 2010 but ported to other platforms including Linux, FreeBSD and NetBSD in late 2012 using the "OpenIKED" project name.



1 Introduction

This paper provides a brief description of the ***Open*IKED** project, including technical background, history and my personal motivation of creating it. It is inteded to provide some complementary information for my talk at AsiaBSDCon 2013 .

The project aims to provide a free implementation of the Internet Key Exchange (IKEv2) protocol which performs mutual authentication and which establishes and maintains IPsec VPN security policies and Security Associations (SAs) between peers. The IKEv2 protocol is defined in Request for Comments (RFC) 5996, which combines and updates the previous standards: Internet Security Association and Key Management Protocol (ISAKMP)/Oakley (RFC 2408[8]), Internet Key Exchange version 1 (IKE) (RFC 2409[3]), and the Internet DOI (RFC 2407[17]). ***Open*IKED** only supports the IKEv2 protocol; support for ISAKMP/Oakley and IKE is provided by OpenBSD's isakmpd(8) or other implementations on non-OpenBSD platforms.

It is intended to be a lean, clean, secure, better configurable and interoperable implementation that focusses on supporting the main standards and most important features of IKEv2. I primarily wrote ***Open*IKED** with significant contributions from Mike Belopuhov and various contributing OpenBSD hackers.

***Open*IKED** is developed as part of the OpenBSD Project. The software is freely usable and re-usable by everyone under an Internet Systems Consortium (ISC) license. The OpenBSD project sells CDs, T-Shirts and Posters. Sales of these items help to fund development.

2 Background & History

2.1 Why another VPN protocol?

There are a number of free implementations of VPN protocols available that can provide a sufficient security for private network connections. But these protocols and implementations have different use cases, limitations, benefits, and drawbacks. Most of them are specialized in one aspect but cannot provide a solution in another area.

An overview of VPN protocols

SSL-VPN is using the Secure Sockets Layer (SSL)/Transport Layer Security (TLS) protocols to securely tunnel network traffic on the "application layer". It is especially designed for "road-warriors" and is typically running on top of HyperText Transfer Protocol (HTTP) to pass web proxies in restricted corporate networks or public hotspots. There is no "SSL-VPN" standard, there are many different vendor-specific implementations, and it is suffering from some significant protocol overhead due to the SSL and HTTP encapsulation. SSL-VPN is typically not sufficient for high-performance VPN connections and its security is questionable due to its proprietary and vendor-specific nature.

OpenVPN is a well-known, open source SSL-VPN, that does not use an HTTP layer and can run on either TCP or UDP. It became very popular because it was easier to use with clients running various operating systems and because of its user-friendly Graphical User Interface (GUI) for most of these systems. OpenVPN made open source VPN useable for a huge user base, but it also had an negative impact on use and development of IPsec. And this is still the case: whenever someone asks deployment questions about (IPsec) VPN on the OpenBSD mailing lists, you can almost expect that someone will refer to OpenVPN instead. OpenVPN still suffers from some of the problems of SSL-VPN, it is not standards-based and all the portable versions are still based on a single implementation that is licensed under the terms of the GNU Public License version 2 (GPLv2). The "copy-left" limits the use in BSD environments and traditional BSD-based products. For the sake of IPsec, we're going to ignore OpenVPN here.

IPsec does a better job for performance, stronger security, and is very well reviewed by the security community. It is completely open and based on some RFC standards in the Internet community which enables interoperability between many different implementations. It is a real protocol stack and not just a strategy or "buzzword" like SSL-VPN. The biggest problems of IPsec are the limitations of its older IKE protocol and some obscurities and incompatible vendor-specific extensions in existing implementations.

VPN protocols in OpenBSD

OpenBSD already supports some different VPNs in the default installation of its "base" system, and there are many more options in its ports collection of 3rd party open source software.

npppd is a server-side Point-to-Point Protocol (PPP) implementation. The development was started as an internal project by the Japanese company Internet Initiative Japan Inc. (IIJ) and later continued in OpenBSD's source tree since January 2010. It was not enabled in the default OpenBSD build until recently and it will be included in the upcoming OpenBSD 5.3 release for the first time, that is expected around May 2013. It provides support for a number of different protocols that allow tunneling and private networking, including Layer 2 Tunneling Protocol (L2TP), Point-to-Point Tunneling Protocol (PPTP), and PPP over Ethernet (PPPoE).

The PPP protocols are generally used for Remote Access Service (RAS) solutions by Internet service providers with dial-in and mobile access offerings. The protocol stack provides a very flexible support for Authentication, Authorization and Accounting (AAA) that is critical in such environments. The protocols also provide great interoperability with a number of different platforms, including Microsoft Windows, Apple OS X, iOS, Google's Android and all free operating systems.

The PPTP and L2TP can be extended to provide additional VPN capabilities such as encryption and authentication of the tunnelled data. Nevertheless, PPTP has some serious security vulnerabilities and weaknesses that have been found in many security analyses. While it might still be a choice to connect mobile devices that do not support any other security protocols, it cannot be considered as a sufficiently secure VPN protocol.

The L2TP protocol is typically used as L2TP/IPsec to encapsulate the link layer 2 tunneling of the protocol in a "secure channel" that is provided by IPsec and negotiated by IKE. The major problem is the complexity of its protocol stack and the overhead that is caused by the encapsulation of different protocol layers.

isakmpd is a BSD-licensed implementation of the IKE protocol, the predecessor of IKEv2. It was written in 1998 by Niklas Hallqvist and Niels Provos for OpenBSD; an effort that was sponsored by Ericsson Radio Systems. It is widely used in the OpenBSD community and one of the major IKE implementations that was also ported to many other platforms and is the foundation of many proprietary implementations. ISAKMP/Oakley provides a protocol layer that was designed to allow multiple key exchange protocols, but IKE was the only protocol that ever implemented and supported by the daemon. The basic protocol is based on the standards RFC 2407, 2408, 2409 and many additional extensions.

The drawbacks of `isakmpd` are the limitations of the old IKE protocol and the complex configuration of the daemon itself. It uses an `.ini`-style configuration file and an additional KeyNote policy file that have to include a number of irritating and non-default

options. Configuring certificate-based IKE authentication was so difficult, that most users and tutorials simply used the lesser-secure pre-shared key authentication that was easier to configure with `isakmpd` and provided better interoperability with other IKE implementations.

In 2005, the `ipsecctl` tool was introduced to simplify the configuration of `isakmpd` in OpenBSD. `ipsecctl` is providing a single alternative configuration file, `/etc/ipsec.conf`, with a human-readable and modern grammar. Additionally, the tool is smart about some configuration defaults and dependencies. It parses its own configuration file and generates a more complex `.ini`-style configuration for `isakmpd` that is feeded into the running daemon through its First In - First Out (FIFO)-socket. It added another layer to the implementation but greatly reduced the complexity of deploying IPsec with `isakmpd` and significantly reduced the number of users in OpenBSD. However, it could not fix any of the limitations of the IKE protocol.

2.2 Internet Key Exchange version 2

IKEv2 is an important VPN protocol because it was designed for the strongest security requirements, the basic protocol is based on a single RFC 5996 standard[5], and it attempts to eliminate some of the worst limitations and misconceptions of IKE. It also aims to reduce the vendor-specific extensions by including them in a standardized way. Its importance in the "real world" is also increasing since Microsoft included it in its Windows 7 operating system, many other vendors started to migrate to IKEv2, and it became mandatory for compliance by standards like USGv6.

Changes in IKEV2

The IKEv2 protocol got improvements based on the experiences with the previous protocol version. It actually became a single protocol in contrast to IKE that was based on an protocol stack with the additional ISAKMP and DOI layers that have been removed in IKEv2. The format of encrypted messages has been modified to match Encapsulating Security Payload (ESP) and the initial handshake has been reduced to a single 4-way handshake that is replacing the previous "main" and "aggressive" modes. Support for road warriors has been greatly improved by adding more flexibility and reorganizing the protocol; road warriors with dynamic IP addresses can also use preshared keys now. Dynamic configuration of the clients is part of the standard using an integrated configuration payload that is inspired by the former IKE-CFG extension.

3 Design & Implementation

The implementation of ***Open*IKED** was originally started to get a future-proof IPsec keying daemon for OpenBSD. Only the IKEv2 protocol is implemented to get all the benefits of the improved version and to avoid the additional complexity of its predecessor IKE.

The daemon is using a layout that is used by OpenBSD's modern networking daemons. Additionally, further improvements of the layout and the related `privsep` and `imsg` frameworks are regularly merged between these daemons. The existing `isakmpd` still exists in OpenBSD and can be used for most legacy IPsec configurations.

3.1 Project Goals

The following statements have been picked to describe the project goals of ***Open*IKED** .

Lean: Provide a small and monolithic architecture that supports the main standards and most important features of IKEv2. Monolithic means that we do not even try to put lots of features in lots of dynamic libraries.

Clean: Write readable and clean code following strict coding style(9)[12] guidelines.

Secure: Implement secure code with strict validity checking, bounded buffer operations, and privilege separation to mitigate the security risks of possible bugs. Use strong cryptography with sane but secure defaults.

Interoperable: Provide good interoperability with other IKEv2 implementations, support non-standard extensions if it is required to interoperate with other major implementations.

Configurable: Make the configuration easy and nice with sane defaults, minimalistic configuration files and good documentation in the manual pages. Avoid the headaches of past and other IKE implementations.

Strong Crypto

***Open*IKED** supports strong crypto using modern cryptographic ciphers and algorithms that provide state-of-the-art security, performance, and possibly optimization for modern hardware. The implementation supports modern ciphers for IKESA (IKEv2 messages) and CHILDSAs (IPsec messages, e.g. ESP) including authentication and pseudorandom function with the SHA2 family and additional AES modes like AES-CTR or AES-GCM. The Diffie-Hellman key agreement

protocol has been extended with additional modes includes support for latest elliptic curve groups.

AES-GCM combines the authentication and encryption steps in the same AES block operation and allows to leave out any expensive HMAC calculation. Mike Belopuhov added support for AES-GCM and AES-GMAC (the authentication-only version) to ***Open*IKED** and the OpenBSD kernel and the ability to accelerate it on modern CPUs using Intel's AES New Instructions (AES-NI). This enables a significant performance improvement compared to traditional software-based AES-CBC-128 + HMAC-SHA2-256.

3.2 iked(8)

The `iked` program is the ***Open*IKED** daemon itself, accompanied by the `ikectl` control utility. It includes the IKEv2 implementation based on RFC 5996[5]. The implementation was created to get an IKEv2 IPsec daemon for OpenBSD based on the modern privilege separation model and the `imsg` framework, a well-defined configuration grammar for the `/etc/ipsec.conf` file and many other improvements over `isakmpd` like support for stateful configuration reloads, the ability to control the running daemon with the `ikectl` utility instead of a FIFO socket, better and more scalable support for gateway to gateway and especially road warrior scenarios, improved X.509 Certificate Authority (CA) usability and proper use of the OpenSSL API functions instead of custom crypto code.

Example Configuration

The grammar is based on `/etc/ipsec.conf` of `ipsecctl`, which loads its configuration and translates it into the `.ini`-style grammar of `isakmpd`. But `iked` is able to load and understand the grammar of the `/etc/iked.conf` configuration file directly without the need for an additional tool like `ipsecctl`. `iked`'s built-in support resolves many problems that appeared with the `ipsecctl` approach and allows features like stateful config reload.

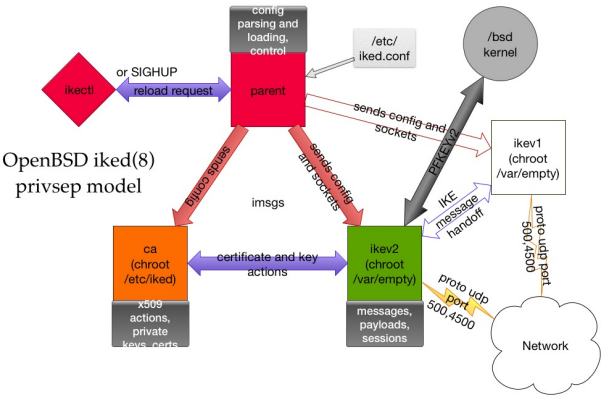
This is a "complex" `/etc/iked.conf` configuration file example for `iked`:

```
user "user1" "password123"
user "user2" "password456"

ikev2 "win7" passive esp \
        from 10.1.0.0/24 to 10.2.0.0/24 \
        local any peer any \
        eap "mschap-v2" \
        config address 10.2.0.1 \
        config name-server 10.1.0.2 \
        tag "$name-$id"

ikev2 esp \
        from 10.3.0.0/24 to 10.1.0.0/24 \
        from 10.5.0.0/24 to 10.1.0.0/24 \
        from 10.5.0.0/24 to 172.16.1.0/24 \
        local 192.168.1.1 peer 192.168.2.1 \
        psk "mekmitasdigoat"
```

Privilege Separation Model

The privilege separation model[18] was first defined for OpenSSH to restrict the effects of attacks and programming errors. A bug in the unprivileged child process does not compromise the security of the privileged part and the operating system itself. It is a similar approach to the modern "sandboxing" techniques but without the need for any custom or non-standard kernel extensions.

***Open*IKED** uses an emerged privsep model with the `imsg` message passing framework that was first implemented for OpenBGPD, copied between multiple daemons, and later imported into OpenBSD's `libutil`.



parent The parent process runs with full privileges to execute privileged operations for the children. It opens and binds to privileged sockets (e.g. UDP port 500), opens the PF_KEY Key Management API, Version 2 (PFKEYv2) socket, and loads the configuration file before sending the required information and resources to the unprivileged children.

ca The semi-privileged CA process handles everything related to certificates and private keys. The process runs in a chroot-environment of the `/etc/iked` directory (or ***Open*IKED** 's default configuration directory of the portable version) and accepts requests from the other processes. The idea behind CA is to isolate the private keys and any other confidential information in a dedicated process. The other processes can request CA lists and send signing or verification

requests of payload that is exchanged via `imsg` messages.

`iked`'s CA process has recently been adopted by OpenSMTPD[15], making it an SMTP implementation that strictly separates private keys from connection handling or email processing.

ikev2 The unprivileged IKEv2 process is the main actor in `iked`. It listens to requests from the network on User Datagram Protocol (UDP) ports 500 and 4500, parses and handles messages, handles IKEv2 sessions and PFKEYv2 communication with the kernel. It can forward IKE messages, version 1, to the IKEv1 process.

ikev1 The unprivileged IKEv1 process is currently an empty stub that does not implement the IKE protocol. It can accept messages, check the version, and forward IKEv2 messages to the IKEv2 process. The design intends to allow operation of both protocol versions on the same host, IP addresses and ports. In the future, the process could either use an implementation of the first protocol version or forward IKEv1 messages to `isakmpd` over an internal communication socket that could be added to both daemons.

ikectl The `ikectl` control utility communicates with the `iked` parent process by sending `imsg` messages over an UNIX socket attached to `/var/run/iked.sock`. It is used for status, reset and configuration reload commands.

3.3 ikectl(8) CA

In addition to status and reset commands, `ikectl` includes an isolated tool to simplify maintenance of the X.509 PKI and to set up a simple CA for `iked` and its peers. It is not intended to be a fully-featured CA toolset, but a set of commands that can create and maintain keys and certificates that are sufficient for medium or small installations of `iked`.

Example configuration of a local CA with two peers:

```
$ ikectl ca test create
$ ikectl ca test install
$ ikectl ca test cert 10.1.1.1 create
$ ikectl ca test cert 10.1.1.1 install
$ ikectl ca test cert 10.1.1.2 create
$ ikectl ca test cert 10.1.1.2 export
```

4 Portable version

Portability of networking software to different operating systems is a complicated task. Every system comes with differences in system APIs, headers, libraries, linker options and file locations. Even standards like POSIX only provide some limited portability if the target system follows that standard and is referring to the same revision. Additionally, some system features like cryptography and networking do not provide a fully-standardized API. The most common approach is to use automatic build tools like GNU automake and autoconf and many OS-specific `#ifdef`s in portable code.

4.1 OpenBSD's Portability Approach

There is a very important design decision in OpenBSD: *OpenBSD software is specifically written for OpenBSD*. It is not intended to have compatibility glue in the source code, neither OS-specific `#ifdef`s, abstraction layers or complex make files. Only external projects that get imported into the base system might use GNU autoconf tools, but software that was written for OpenBSD keeps on using the system's variant of BSD make. This makes the source code very clean and helps readability, maintenance, auditing and security.

OpenBSD distinguishes between "core" and "portable" versions of its subprojects. The main development usually happens in OpenBSD's CVS repository of the base system where the core version is specifically written for OpenBSD. The "portable" version adds a set of patches, a compatibility library, and portable build infrastructure using the GNU autoconf tools and is developed and maintained in different places outside of OpenBSD's CVS repository.

This approach was defined with the development of OpenSSH-portable and is described in Damien Miller's "Secure Portability"[10] paper. All the other portable OpenBSD projects, including OpenNTPD, OpenBGPD, OpenSMTPD and ***Open*IKED** are sharing the same principles and some autoconf and compatibility code that was created by the OpenSSH-portable team.

4.2 The Portable *Open*IKED Project

The project website is available at www.openiked.org[14]. In difference to the core version that is located in OpenBSD's CVS repository, the source code of the portable version is currently hosted at GitHub. Any changes on the portable code are pushed to GitHub from a private Git repository.

The source tree of OpenIKED contains the following directories:

- `openiked/`: Build scripts for automake/autoconf and README files.
- `openiked/ikectl/`: The control and status utility for iked.

- `openiked/iked/`: The IKEv2 daemon itself and some files that are shared with ikectl.

- `openiked/openbsd-compat/`: Portability glue and API functions for non-OpenBSD platforms.

Any changes in the core version are regularly merged into the portable version using a combination of CVS and Git. It is possible to compare the differences between these versions, by cloning the portable code from GitHub:

```
$ git clone git://github.com/reyk/openiked.git
```

And comparing the contents of the iked and ikectl directories in the git repository with the original sources in OpenBSD's (Anon)CVS repository by running the cvs diff command in these subdirectories. This will show the differences between to the core version:

```
$ cd openiked/iked/
$ cvs diff -Nup | tee ../iked.diff
$ cd ../ikectl/
$ cvs diff -Nup | tee ../ikectl.diff
```

4.3 Supported Operating Systems

***Open*IKED** currently runs on only a few major operating systems ("ports"), but it should theoretically run on any Unix-like system with kernel-based IPsec and PFKEYv2[9]. It should specifically run on all systems that are based on the Internet Protocol Version 6 (IPv6)/IPsec reference implementation of the KAME project[4], including all major BSD operating systems and their offsprings.

Nevertheless, the RFC document of the PFKEYv2 standard only specified an API for maintaining IPsec SAs, but it didn't specify an API for maintaining IPsec security policies. OpenBSD was the first open source operating system that supported IPsec and PFKEYv2 by default and the developers created its own API extensions in need for specifying security policies, or "flows". The KAME project implemented it differently using its own API of an "Security Policy Database (SPD)". Most other systems are using the KAME-based PFKEYv2 variant with minor individual differences, most notably the differences in supported crypto algorithms and extensions like NAT Traversal (NAT-T). Support for this variant has been added to the portable ***Open*IKED** version making it compatible with non-OpenBSD ports.

The software additionally depends on two libraries that are widely available for all of these operating systems: OpenSSL[16] 1.0 or later and libevent1[19] (libevent2 should theoretically work but was not tested as it is not used in OpenBSD).

Apple OS X (Darwin)

The Darwin operating system is originally based on FreeBSD, which includes a BSD-like system and a KAME-based IPsec stack and PFKEYv2 interface. Darwin was the first port because of its practical importance to run ***Open*IKED** on MacBooks with Apple OS X.

Since Apple deprecated OpenSSL in recent versions of Darwin/OS X, and their default installation only ships a pre-1.0 version, it is required to install OpenSSL 1.0 manually. I decided to use the MacPorts[7] system to install OpenSSL and libevent1 from its community-based collection of open source packages.

On the kernel side, Darwin provides support for NAT-T, but it is officially marked as "private" and hidden from the official `pfkeyv2.h` header file. The related information is found in the publicly available sources of the "XNU" Darwin kernel and NAT-T is supported by the port. Apple changed some of the standard BSD kernel APIs and header files that are related to networking and packet level bit and endianess operations. For example, FreeBSD's "htobe64" byte conversion macros has been replaced by "OSSwapHostToBigInt64" in a non-standard header file.

NetBSD & FreeBSD

Both systems provide OpenSSL and libevent1 in their package repositories and they are using an KAME-based IPsec stack. The NAT-T extension is currently not supported by the ports to these systems.

My biggest surprise was that they do not support IPsec in their default "GENERIC" kernels, not even as a loadable kernel module. It is required to compile a custom kernel with some additional options to enable IPsec and PFKEYv2.

For FreeBSD[2]:

```
options IPSEC
#options IPSEC_DEBUG
device crypto
```

For NetBSD[11]:

```
options IPSEC
options IPSEC_ESP
```

DragonflyBSD

I started looking into a DragonflyBSD port, but I gave up quickly. Any efforts from the DragonflyBSD community would be appreciated.

GNU/Linux

Linux provides its own PFKEYv2 implementation that is mostly compatible with the KAME variant.

The autoconf framework and compatibility library that is based OpenSSH's portable version made it surprisingly easy to port ***Open*IKED** to Linux.

A drawback is the fact that the Linux kernel developers invented their own non-standard "XFRM" kernel API that is intended to replace PFKEYv2, which is considered to be obsolete. The PFKEYv2 interface still exists but is poorly maintained and lacks some features, like working support for HMAC-SHA2-256 HMAC authentication for IPsec. Linux originally added HMAC-SHA2-256 support based on the pre-standard specification with a truncation length of 96 bits that is incompatible to the standard length of 128 bits that is described in RFC 4868[6]. PFKEYv2 uses pre-defined identifiers and attributes for algorithms, e.g. `SADB_X_AALG_SHA2_256` for HMAC-SHA2-256 with 128 bits truncation. The Linux kernel recognizes the `SADB_X_AALG_SHA2_256` identifier but assumes 96 bits truncation. The kernel developers never fixed this obvious bug to keep compatibility with one or two other implementations that use the pre-standard version. They refer to the "XFRM" API that allows to set the algorithm, and the key and truncation lengths individually.

4.4 User-friendly GUI

There is actually no need to use a GUI to set up gateway to gateway connections. The configuration file, `iked.conf`, uses a well-defined grammar that is easy to understand for system administrators and most users of ***Open*IKED** . But when connecting mobile users, or road warriors, to the gateway, an easy GUI is an important requirement for deploying the VPN. These users are most commonly non-technical laptop users that connect to a VPN gateway of their organization, university or company. It is most desirable that they can set up and debug the client-side of the VPN connection without much interaction from the IT department.

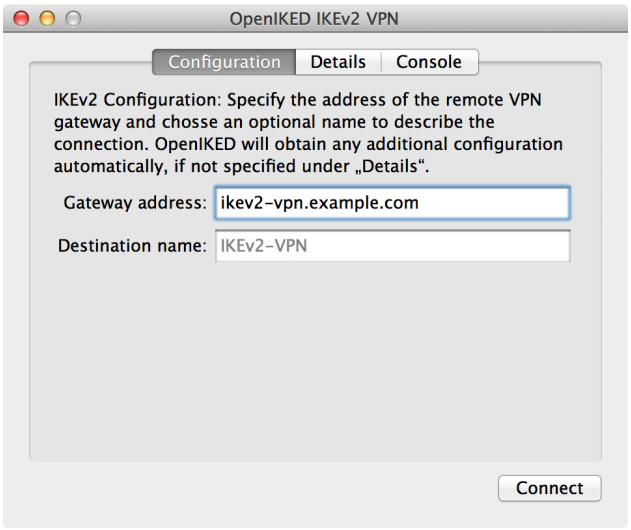
Microsoft Windows

Microsoft Windows 7 introduced an integrated IKEv2 client configuration dialog that is surprisingly easy to use for standard users. The configuration of traditional IPsec/IKE used to be very difficult under Windows but the IKEv2 client only requires installing the required certificates, setting the remote gateway address, and specifying a user name and password for additional EAP-MSCHAPv2 authentication. And of course, ***Open*IKED** is a compatible gateway that can be used with the built-in Windows client.

OpenIKED.app

To get a similar level of IKEv2 user-friendliness on OS X, I started working on a simple GUI that is inspired by the Windows client. Accordingly, the goal is to provide a very simple tool that allows to set up IKEv2 client connections from Mac-based road warriors. The dynamic negotiation of IKEv2 and the secure defaults of ***Open*IKED** allows to reduce the required configuration settings to the minimum on the client side: remote gateway address and optional connection name. Other optional settings can be configured in the "Details" tab. In difference to the Windows client, additional user-based authentication is currently not supported as EAP-based authentication is only implemented for the server (responder) side.

The current version is a working proof of concept that requires manual installation of keys and certificates into the configuration directory.



4.5 The Artwork

OpenBSD and its subprojects aren't just known for security, they're also known for their comic-style artwork. Each of these projects has a theme that is including the OpenBSD-styled logo and "Puffy" the pufferfish in some action. The artwork is used on T-Shirts, posters and CD covers and was originally designed by the Canadian artist Ty Semaka and some other artists today.

When I decided to turn iked into a portable project, it was clear that I needed a matching artwork. I had the idea of using a "tin can telephone" as a theme that represents VPN communication in an obscure way.

But I needed an artist to create a real Puffy theme and found Markus Hall from Sweden who kindly designed the logo and artwork for ***Open*IKED** . He also contributed the idea that puffy is talking to the cute blue pufferfish girl over the tin can phone.

5 Appendix

5.1 About the Author

Reyk Floeter[1] works as a freelance consultant and software developer with a focus on OpenBSD, networking, and security. He lives in Hannover, Germany, but works with international customers like IIJ in Tokyo. As a member of the OpenBSD project, he contributed various features, fixes, networking drivers and daemons since 2004, like OpenBSD's ath, trunk, vic, hostapd, relayd, snmpd, and iked. For more than nine years and until mid-2011, he was the CTO & Co-Founder of .vantronix where he gained experience in building, selling and deploying enterprise-class network security appliances based on OpenBSD.

References

[1] Reyk Floeter, *Reyk Floeter Consulting*, `http://www.reykfloeter.com/`.

[2] FreeBSD, *VPN over IPsec*, `http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/ipsec.html`.

[3] Dan Harkins and Dave Carrel, *RFC 2409 - The Internet Key Exchange (IKE)*, `http://www.ietf.org/rfc/rfc2409.txt`, November 1998.

[4] KAME, *The KAME Project*, `http://www.kame.net/`, April 1998.

[5] Charlie Kaufman, Paul Hoffman, Yoav Nir, and Parsi Eronen, *RFC 5996 - Internet Key Exchange Protocol Version 2 (IKEv2)*, `http://www.ietf.org/rfc/rfc5996.txt`, September 2010.

[6] Scott Kelly and Sheila Frankel, *RFC 4868 - Using HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 with IPsec*, `http://www.ietf.org/rfc/rfc4868.txt`, May 2007.

[7] MacPorts, *The MacPorts Project*, `http://www.macports.org`.

[8] Douglas Maughan, Mark Schertler, Mark Schneider, and Jeff Turner, *RFC 2408 - Internet Security Association and Key Management Protocol (ISAKMP)*, `http://www.ietf.org/rfc/rfc2408.txt`, November 1998.

[9] Daniel McDonald, Craig Metz, and Bao Phan, *RFC 2367 - PF_KEY Key Management API, Version 2*, `http://www.ietf.org/rfc/rfc2367.txt`, July 1998.

[10] Damien Miller, *Secure Portability*, `http://www.openbsd.org/papers/portability.pdf`, October 2005.

[11] NetBSD, *Configuring IPsec kernel*, `http://www.netbsd.org/docs/network/ipsec/#config_kernel`.

[12] OpenBSD, *style(9) - Kernel source file style guide (KNF)*, `http://www.openbsd.org/cgi-bin/man.cgi?query=style`.

[13] ———, *The OpenBSD Project*, `http://www.openbsd.org/`.

[14] ———, *The OpenIKED Project*, `http://www.openiked.org/`.

[15] ———, *The OpenSMTPD Project*, `http://www.opensmtpd.org/`.

[16] OpenSSL, *The OpenSSL Project*, `http://www.openssl.org`.

[17] Derrell Piper, *RFC 2407 - The Internet IP Security Domain of Interpretation for ISAKMP*, `http://www.ietf.org/rfc/rfc2407.txt`, November 1998.

[18] Niels Provos, Markus Friedl, and Peter Honeyman, *Preventing Privilege Escalation*, `http://www.citi.umich.edu/u/provos/papers/privsep.pdf`, August 2003.

[19] Niels Provos and Nick Mathewson, *libevent - an event notification library*, `http://libevent.org`.