IPA
INFORMATION-TECHNOLOGY PROMOTION AGENCY, JAPAN

2004 情財第 987 号

信頼できる OpenPGP 公開鍵を提供する公開鍵サーバ
OpenPKSD Trusted Keyserver　開発成果報告書

2005 年 8 月

独立行政法人 情報処理推進機構

目　次

１．プロジェクト概要

１．１ 背景

インターネット上で最も利用されている暗号技術の一つに OpenPGP (RFC2440)がある。OpenPKSD に代表される OpenPGP 公開鍵サーバは、インターネット上で公開されている OpenPGP フォーマット(RFC2440)の公開鍵を交換するためのサーバである。
PGP の起源は 1991 年に遡る。Philip Zimmermann が 1991 年に暗号ツール PGP を作成・公開しインターネット上に広まった。その後、GNU Privacy Guard Project からリリースされた GnuPGP などのフトウェアが開発され、さらに普及が加速し、OpenPGP (RFC2440)として共通の仕様が出来るまでに至った。PGP が広まる一方で、その公開鍵を交換するインフラストラクチャのニーズが高まり、1994 年ごろ MIT の学生が PGP コミュニティの中で公開鍵を交換するための公開鍵サーバを作成した。インターネット利用者の増大と同時に PGP ユーザも増大し、初期のきわめてシンプルな公開鍵サーバでは処理しきれなくなった。1997 年当時 MIT の大学院生だった Marc Horowitz が本格的な公開鍵サーバ(PKSD:PGP Public Keyserver)を作成し、さらに公開鍵サーバとダイレクトにデータをやり取りする hkp (Horowitz Keyserver Protocol)を考案した。この Horowitz 版 PKSD は改良され現在でも利用されている。
Horowitz 版 PKSD が開発された当時、登録されていた公開鍵は 5 万鍵程度であり、莫大な数のインターネットユーザを想定して作られたわけではなかった。現在、Horowitz 版 PKSD は、200 万鍵を越える公開鍵を保持しているが、基本設計が非常に優れていたため、十分に実用に耐える状態ではある。しかしながら、インターネットユーザの数は増える一方であり、また、セキュリティに対する関心も以前よりはるかに高い。日本国民一人一人が OpenPGP 公開鍵を登録し、日々利用するような時代が到来した場合、現在の Horowitz 版 PKSD では限界に達してしまう。また、Horowitz 版 PKSD は、データベースの運用と並行してバックアップを取れず、大規模データベース運用のメンテナンス性にも問題があった。
そこで高負荷耐性・高信頼性・大量公開鍵保持のための次世代 OpenPGP 公開鍵サーバを提案し、平成 13 年度、14 年度に IPA の情報セキュリティ対策事業として採択され、次世代 OpenPGP Public Keyserver (OpenPKSD)の開発を行った。
OpenPKSD は、Horowitz 版 PKSD で実現されている要件に加え、将来、新しいサービスが必要になったときに柔軟に拡張を行えるように設計されている。オブジェクト指向言語 RUBY を使い OpenPGP クラスライブラリ、OpenPKSD クラスライブラリを作成し、それをベースとして用い、システム全体を構築した。
平成 13 年度事業では、Horowitz 版 PKSD 互換の公開鍵サーバを作成し、平成 14 年度事業ではクラスタ拡張ライブラリの作成と、それを用いたシステムの実証を行った。これらの成果により理論値においては、登録される公開鍵が一億鍵を突破しても問題なく稼動できるシステムを構築できるようになった。OpenPKSD は、現在 OpenPGP 公開鍵サーバとして OpenPKSD.ORG サイト(http://openpksd.org)にて非クラスタ版が稼動し、公開鍵サーバのサービスを提供している。このサーバ上に登録されている公開鍵数は平成 16 年末で 200 万鍵を超え、現在も日々増加していっている。
しかし、その一方で、ユーザは OpenPGP 公開鍵サーバへの鍵登録を積極的行わないという状況もある。OpenPGP 公開鍵サーバは公開鍵を利用するためのインフラストラクチャという意味では PKI (Public Key Infrastructure)であるが、認証局的機能があるわけではなく、誰でも公開鍵を登録でき、登録している公開鍵を自由にダウンロードできる。
公開鍵の信頼性（本当に目指す相手のものかどうか）を確認する方法としては、ユーザが自分自身で確認する方法の他に、信頼できる第三者の署名を信頼する「信頼の輪」（Web of

Trust）方式を取り入れている。そのため、公開鍵に第三者の署名をつけ、公開鍵サーバにアップロードすると、元の鍵にその署名が付加される。誰でも公開鍵に署名を付加できるため、いわゆる「ゴミ」署名が増えてしまい、鍵サーバを使う限り、自分の意思とは関係なく第三者に勝手に登録され、自分の公開鍵が自分自身でコントロールできないという問題が発生している。そのため、公開鍵サーバは、信頼のおける公開鍵入手場所とはされず、正しい OpenPGP 公開鍵は鍵所有者自身の Web サイト上に置かれる場合が多くなってきている。これでは、「アクセスすると誰の公開鍵でも得られる」という公開鍵サーバの本来の目的が達成できない。大規模、高負荷性、高信頼性に優れた OpenPGP 公開鍵サーバを作っても、OpenPGP のユーザが OpenPGP 公開鍵サーバを信頼せずに使わなくなっては無意味になってしまう。

１．２ 目的

上記の問題を解決するため、今回、OpenPKSD Trusted Keyserver を開発した。
本プロジェクトでは OpenPKSD で開発したクラスライブラリや知見を生かし、公開鍵所有者が自分の公開鍵の公開・非公開のコントロールができる、信頼できる OpenPGP 公開鍵を提供する公開鍵サーバ（OpenPKSD Trusted Keyserver）を開発し、かつ、開発したソフトウェアをフリーソフトウェアとして公開することを目的とする。

１．３ 用語の定義

OpenPGP	PGP 互換ソフトウェアでデータの互換性を持たせるための使用である。OpenPGP は RC2440 という形で文章化されている。
PGP	1991 年 Philip Zimmermann が作成した暗号ツール。公開鍵暗号、共通鍵暗号、電子署名など暗号ツールとして必要な機能が搭載されている。現在世界中で最も利用されている暗号ツールとしても知られている。現在は米 PGP Corp が開発・販売を引き継いでおり、非商用目的であれば自由にダウンロードし利用できる。
GPG	GNU Privacy Guard プロジェクトの一環として作成された OpenPGP 仕様の暗号化ツール。GPL ライセンスにもとづき公開されているので誰でも自由に利用できる。
GPL ライセンス	米国 Free Software Foundation 財団が GNU プロジェクト推進のために作成したフリーソフトウェアのためのライセンス。著作権を保持した上で、第三者への公開、自由な変更・再配布を許すライセンス。GPL ライセンスのソースコードを改変・再配布した場合、その改変ソースコードも GPL で公開しなければならない。Linux のカーネルも GPL ライセンスを採用している。
OpenPGP 公開鍵サーバ	広域で公開鍵を利用する際の問題の１つに多数間での鍵配送を行うと指数的に配送数が増える問題がある。そこで自分の公開鍵を広く公開すること、あるいは他者の公開鍵を探すためのの負荷を低減する必要性が生じる。そこで公開鍵をプールするサーバーとして作られたのが PGP Public Keyserver である。1994 年に最初のバージョンが作られた。世界に散らばる主な PGP 公開鍵サーバはお互いが同期して

	いるので、どこに登録しても自動的に他の PGP 公開鍵サーバへと登録されるメカニズムを持っている。
OpenPKSD	平成１３、１４年 IPA 情報セキュリティ対策事業で開発を行った次世代 OpenPGP 公開鍵サーバ。OpenPKSD.ORG サイトにて稼動し、現在 200 万を超える公開鍵を保持している。文中では OpenPKSD 公開鍵サーバと呼ぶ場合もあるが、同じ意味。
OpenPKSD.ORG	OpenPKSD の保守や、OpenPKSD を使い公開鍵サーバをサービスしているコミュニティ。OpenPGP の利用方法や OpenPGP 公開鍵サーバの利用方法などを紹介するために多数の公開ドキュメントなどを用意している。 ● 代表：鈴木裕信 ● OpenPKSD.ORG サイト：http://openpksd.org

２．ソフトウェア概要

２．１ 背景説明

公開鍵が増えていっている一方で、誰でも署名を公開鍵に付加できるため、公開鍵所有者が意図しない署名がついた公開鍵が所有者の意思とは関係なく登録できる。いわゆる「ゴミ署名問題」である。このため公開鍵サーバは、公開鍵の署名交換には使われるが、信頼のおける公開鍵入手場所とはされず正しい OpenPGP 公開鍵は自分の Web サイト上に置かれる場合が多くなってきている。このように OpenPGP 公開鍵サーバの利用を妨げている一面がある。あるいは第三者が勝手に登録し、意図せずに自分の公開鍵が公に提供されてしまう問題もある。これでは公開鍵サーバの目的である、「公開鍵サーバをアクセスすると公開鍵が得られる」ということが達成できない。

そこで既存の OpenPKSD をベースにさらに発展させた「所有者が自身の公開鍵の提供をコントロールでき、所有者・利用者のどちらからも信頼される公開鍵サーバ」を開発し上記の問題を解決を目指した。インターネット上の OpenPGP のユーザは、あちらこちらの Web サイトを一々探して必要な公開鍵をダウンロードする必要がなくなり、OpenPKSD Trusted Keyserver へアクセスするだけで入手できるようになる。公開する側も安心して自分の公開鍵を OpenPKSD Trusted Keyserver へ登録できるようになる。これらによりインターネット上で OpenPGP の利用を促進する手助けができ、安全なメッセージ交換が普及することを期待する。

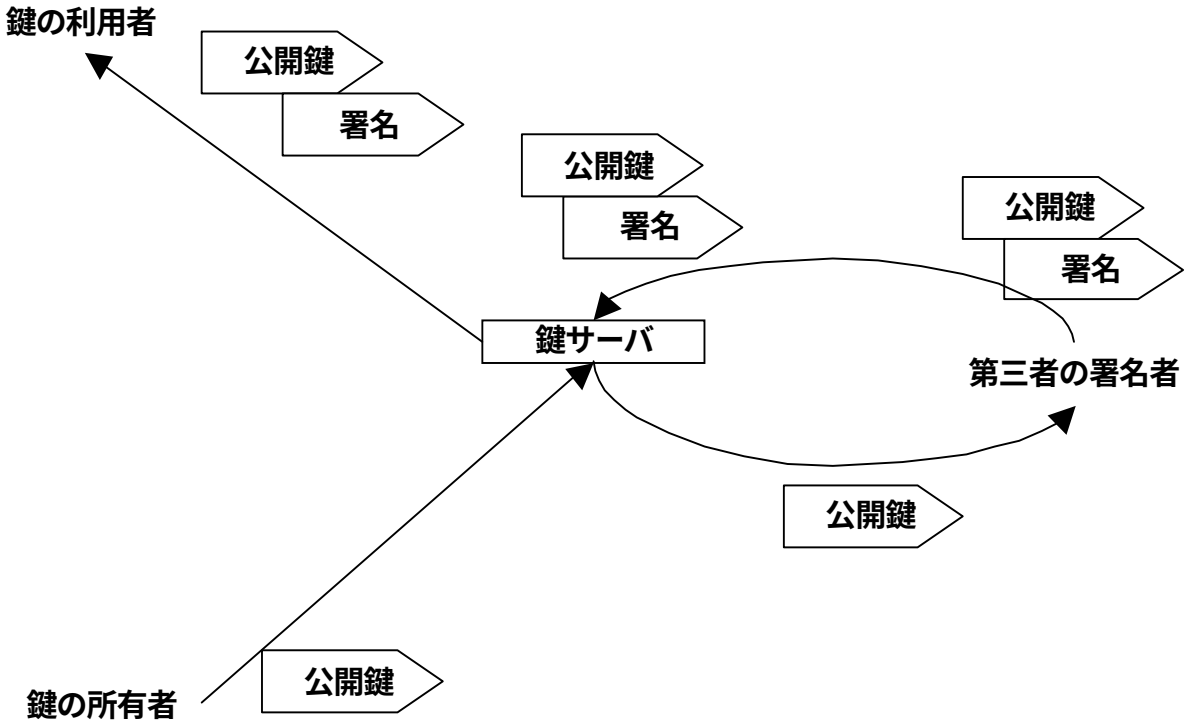

図１　既存サーバにおける鍵の流れ

２．２ システムの概念

達成できる安全性の性質の議論

本システムを理解する上で基本事項をまず説明したい。以下に暗号技術の３つの利益を上げる。

秘匿性	交換する情報の内容を秘密にする。
完全性	交換する情報の内容が書き換えられていない。
認証性	情報を交換する相手を認証する。

さてこれを踏まえたうえで、本システムではどのように、これらの性質に関与できるかを議論したい。

OpenPGP の使う電子署名法と公開鍵暗号法では、秘匿鍵（＝署名鍵）と公開鍵（＝検証鍵）のペアを使う。このペアが一致しない限り、公開鍵で暗号化した内容は復号できないし、署名鍵で署名した内容を改ざんすると検証鍵で検知できる。したがって、公開鍵（＝検証鍵）を持っていれば、この秘匿鍵（＝署名鍵）を持つ者と安全にやり取りできる。

公開鍵サーバは、公開鍵を持っているので公開鍵所有者（厳密にはその対になる秘匿鍵・署名鍵を所有する者であるが、ここでは公開鍵所有者と呼ぶ）からの署名は確認でき、また公開鍵所有者に秘密のメッセージを送ることができる。しかし認証性については、このシステムでは曖昧であることを議論しなければいけない。公開鍵と秘匿鍵は対であり、この組み合わせ以外は使えない。よって秘匿鍵を持つ者は公開鍵の正当な所有者であることがわかる。ただしそれ以上の情報はない。通常、認証性と呼ぶ場合、暗黙のうちに所有者

を特定できる(Identify)できる情報を信頼のおける形で付与し、その情報と公開鍵とをリンクさせている。このシステムでは所有者を特定できる情報を管理はしない。つまり所有者を特定できる情報は匿名であるか、あるいは自称でしかない。唯一、メールアドレスが到達可能である可能性が非常に高い、というだけである。あくまでこの Trusted というのは公開鍵所有者側から見た Trusted であって、この公開鍵を取り出す側からの視点ではないことに注意したい。下に最も基本的なアイデアを示す。

- 公開鍵所有者は公開鍵を鍵サーバに公開鍵を預ける。
- 鍵サーバは秘密の情報を安全に公開鍵所有者に送ることができるので秘密の情報を共有できる。
- 公開鍵所有者が署名した情報を鍵サーバは確認できるので情報が正しいことがわかる。

２．３ 公開鍵の流れ

公開鍵サーバを中心にしてすべての公開鍵の流れがリンク（図 1）している。これを次の二つに分解して考える。

- 公開鍵所有者が登録した公開鍵を第三者署名が署名し鍵サーバに登録する流れ。
- 公開鍵所有者が登録した公開鍵を利用者が鍵サーバから入手する流れ。

この２つの流れの中間に入るのが公開鍵サーバと公開鍵所有者である。この処理の流れを順を追って説明する。

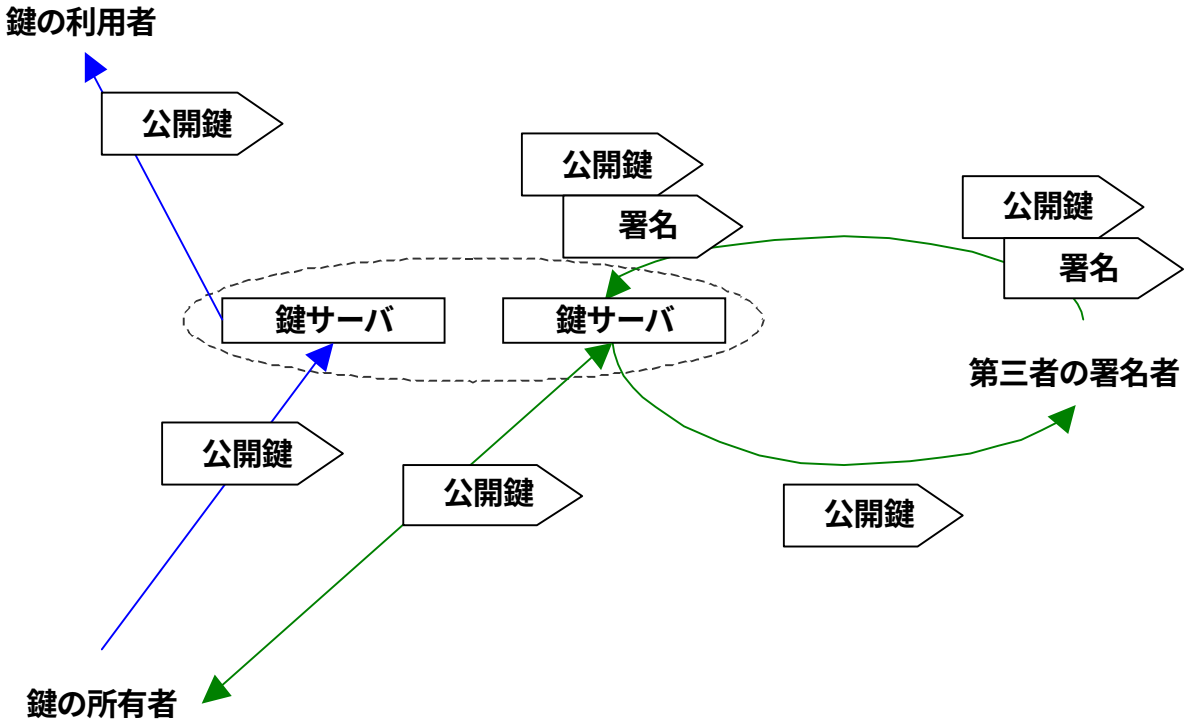

図　概念的な鍵サーバの分離

２．４ 公開鍵の初期登録

公開鍵を登録するところから始まる。公開鍵を登録することによって公開鍵所有者と鍵サーバ間で秘匿性と完全性を確保する通信が行えるようになる。

Step 1:
公開鍵所有者は安全な経路で OpenPKSD-TKS へ公開鍵を送る。

この時点では鍵サーバに初期登録されただけで誰も公開鍵を取り出すことはできない。これを取り出せるようにできるようにすることを、ここでは「公開する(show)」と呼ぶことにする。逆に取り出せなくすることを「非公開(hide)」と呼ぶことにする。デフォルトは hide である。show や hide に設定する要求を出すことをここではリクエスト(request)と呼ぶことにする。尚、ここでの安全な経路というのは SSL を想定するが、認証可能で安全な経路が確保できれば手動でもかまわない。これは運用でカバーする。

２．５ 公開鍵の公開・非公開化

Step 1:
公開鍵所有者は鍵サーバに show リクエスト、KeyID と、自分のメールアドレスを送る。
Step 2:
ワンタイムパスワードを鍵サーバ中の公開鍵を使い暗号化し公開鍵所有者にメールで送る。
Step 3:
ワンタイムパスワードを鍵サーバへ送る。

Step 1 でのリクエストは Step 2 と Step 3 で送られるワンタイムパスワード(One-Time Password)により承認(confirm)される。ワンタイムパスワードは 128 ビット以上の乱数から出来ていて、その値は登録されたリクエストへのインデックスとなっている。鍵サーバの発行したワンタイムパスワードを鍵サーバに戻すことで始めて予約しているリクエストが承認され処理される。Step 1 で指定するメールアドレスは任意のあて先である。

Step 2 でワンタイムパスワードは公開鍵により暗号化され公開鍵所有者の指定したメールアドレスへ送付する。ワンタイムパスワードは暗号化されているため公開鍵所有者しか知ることができない。またワンタイムパスワードの乱数は十分に大きいために推定は困難であるので Man-In-The-Middle 攻撃への耐性がある。

Step 1 と Step 3 は Web サーバとブラウザ間で行う。このシーケンスにおいては SSL は必須ではない。

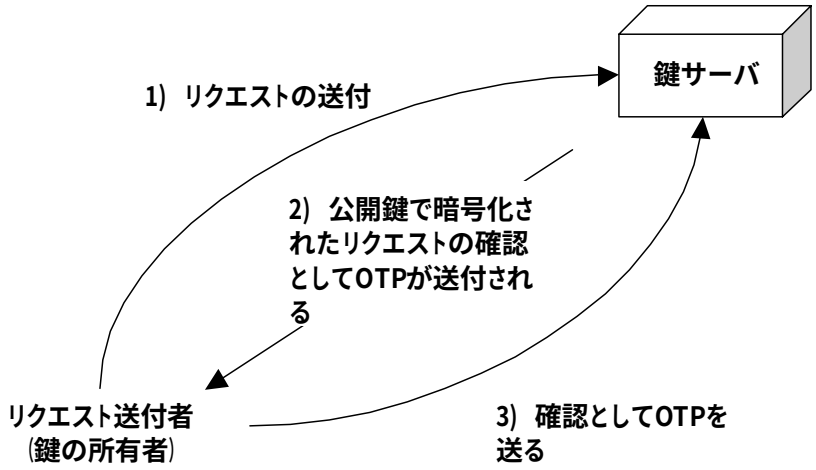

図 2 OTP を使う処理

第三者による署名による登録

Step 1: 鍵サーバに鍵を登録する。

ここでの第三者署名とは公開鍵に第三者の署名を付加することを指す。以降、署名を行う第三者をここでは単に署名者と呼ぶ。署名者は鍵サーバから任意の公開鍵を入手する、あるいは鍵サーバではなく別の経路で公開鍵を入手していることとする。そこで、自分の署名をその公開鍵に付加し鍵サーバに登録する。このときに次のような問題が発生する。

- 公開鍵所有者が署名者を選択できない。意図しない署名が付加され配布されてしまう。
- 公開鍵所有者の意思に関係なく鍵サーバに公開鍵が登録され公開鍵が配布されてしまう。

これを解決する方法は公開鍵を公開する流れから外してしまうことである。公開鍵所有者のみが署名者により鍵サーバに登録された公開鍵を入手することができるようにする。第三者による鍵サーバへの公開鍵の登録は既存の手順と同じである。

公開鍵所有者による公開鍵の入手

公開鍵所有者が公開鍵を入手する手順は、公開鍵の公開・非公開化と同じ手順で入手(get)リクエストを送り、ワンタイムパスワードを得て、ワンタイムパスワードを使い公開鍵を入手する。

入手した公開鍵を実際に公開鍵所有者が受け入れる、つまり自分の公開鍵に第三者の署名を受け入れるかどうかは公開鍵所有者が決める。

Step 1:
公開鍵所有者は鍵サーバに get リクエスト、KeyID と、自分のメールアドレスを送リクエスト、KeyID と、自分のメールアドレスを送る。
Step 2:
ワンタイムパスワードを鍵サーバ中の公開鍵を使い暗号化し公開鍵所有者に送る。
Step 3:
ワンタイムパスワードを鍵サーバへ送る。
Step 4:
第三者の署名がついた公開鍵が出力される。

公開鍵所有者による公開鍵の更新

公開鍵所有者は公開鍵をアップデートするために鍵サーバに公開鍵を送る。その際に、公開鍵に署名をつける。鍵サーバは、既に登録されている公開鍵を使いその署名が正しいかどうかを検証する。正しければ公開鍵を入れ替える。本システム以外の鍵サーバでは鍵をマージする。自分で公開鍵内部の各種の情報を追加、削除できる。追加の場合は、鍵マージで問題はないが、削除の場合鍵マージをするといつまでも古い情報が残ってしまう。このような問題を回避するために本システムでは公開鍵をまるごと入れ替えてしまう。

２．６ システム構成

本システムは以下のコンポーネンツから構成されている。

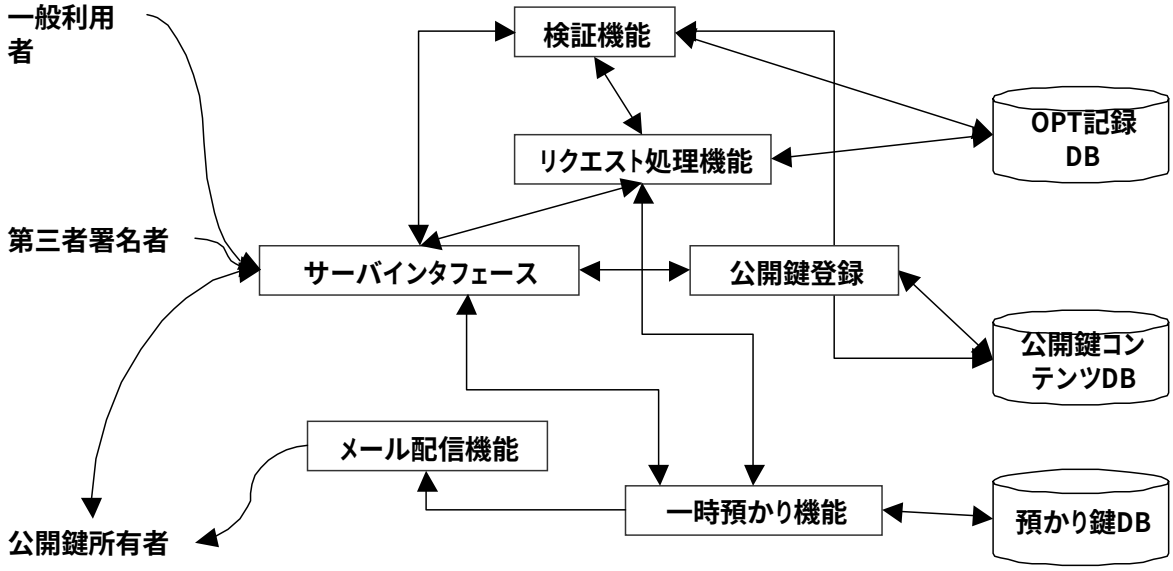

図３ システム構成図

サーバ・インタフェース

サーバ・インタフェースは基本的には http ベースのインタフェースでネットワークを介してやり取りする。OpenPKSD では cgi-bin ベースでウェブブラウザ経由でのアクセスと hkp ベース gnupg ベースの両方を用意している。本システムではまず利用度の高いウェブブラウザ経由のアクセスを提供している。将来は hpk ベースのアクセスもできるように拡張していく。

TKSD 鍵サーバ処理デーモン

すべての処理はこのデーモンによって切り分けられ処理が行われるためのものである。処理内容により子プロセスが生成され、その子プロセス下で処理のためのインスタンスが生成され処理が行われる。子プロセスは処理の終了とともに消滅する。

署名検証

署名が追加されたデータに対して署名検証を行う。必要な公開鍵を公開鍵コンテンツ DB から KeyID を使い検索する。署名検証は gpg を呼び出し、検証用公開鍵、検証対象を与え署名が正しいかどうかチェックする。

リクエスト処理

ユーザからのリクエストの処理を行う。ワンタイムパスワード（ワンタイムパッド）の生成もリクエスト処理の中で行い、データベースへ登録される。公開鍵コンテンツ DB からリクエスト要求者の公開鍵を取得し暗号化を行う。

公開鍵登録

公開鍵の登録及び入れ替えを行う。一般利用者からの公開鍵を参照する要求もこちらで処理する。この機能を呼ぶときは既に検証等が終了していることが前提で単純に登録、入れ替え、取り出しを行うだけである。

一時預かり機能

第三者からの署名がついた公開鍵を保管する機能。公開鍵所有者からリクエストがあれば一時的に預かっている鍵を送る。一時預かりは第三者からの処理を行うのみで、完全に公開鍵登録とは独立した預かり鍵 DB にアクセスする。

データベース

PostgreSQL 上に作られたテーブル群である。機能的には三つに表現されているが実装では表記されていない内部的なテーブルがいつくもあり連動して動いている。

３．外部設計

３．１ システム概要

OpenPKSD Trusted Keyserver の開発においては、OpenPKSD サーバをベースに署名認証による処理を導入する。既存の公開鍵サーバには「公開鍵サーバにアップロードした自分の公開鍵を自分自身でコントロールできるように署名認証処理を導入する」という考え方自体がなかった。また、公開鍵サーバのような大量の処理が発生する地点で署名認証処理を行うと、サーバに対する新たな負荷が発生するため、大規模、高負荷性、高信頼性に優

れた鍵データベースを必要とする。OpenPKSD はクラスタ処理ができる唯一の公開鍵サーバであり、署名認証処理に計算資源を必要としたとき、クラスタマシンを追加することにより、処理が可能となる。
今回の開発には、クラスタ化部分は含まれていないが、OpenPKSD にはクラスタ機能があり、将来クラスタ化が必要になった場合、簡単に導入ができるという利点がある。

表 1　OpenPKSD Trusted Keyserver の 5W1H

だれが (WHO)	OpenPGP 公開鍵を公開する人が
何を (WHAT)	自分の正当な OpenPGP 公開鍵を
どこに (WHERE)	OpenPKSD Trusted Keyserver 上へ
いつ (WHEN)	自分の望むときに
なぜ (WHY)	自分のコントロール下にある自分の OpenPGP 公開鍵をインターネットを介して他者へ提供することによって安心を得るため
どのように (HOW)	OpenPKSD Trusted Keyserver に対する自分のメッセージ（命令）にはすべて OpenPGP 公開鍵による署名をつけることによって

表 2　OpenPKSD と OpenPKSD Trusted Keyserver の違い

OpenPKSD	優位性	OpenPKSD Trusted Keyserver
第三者が勝手に公開鍵を登録し一般公開してしまう。	◎ ×	公開鍵所有者が一般公開を停止することができる。
第三者が勝手に署名をつけて公開鍵サーバに公開鍵を登録してしまうのでゴミ署名がついた公開鍵が一般公開される。	◎ ×	第三者が勝手に署名をつけて公開鍵サーバに公開鍵サーバに登録しても、それは公開鍵所有者のみに渡るので一般公開されない。
公開鍵所有者の望む内容の公開鍵を一般公開することはできない。	◎ ×	公開鍵所有者の望む公開鍵を一般公開することができる。
公開鍵サーバを使わずに自分の Web サイトに登録してしまうので統一した公開鍵検索先がなくなりユーザは公開鍵を見つけるのに煩雑さが増す。	◎ ×	OpenPKSD Trusted Keyserver を統一した公開鍵検索先に指定できるので簡単に目指す公開鍵を探せる。
電子署名技術などを使わず単純にデータをデータベースへ格納する。	◎ ×	電子署名技術を使い検証した後にデータをデータベースへ格納する。
大量の公開鍵を管理できる。	◎ ◎	OpenPKSD のクラスライブラリを使うので同様に大量の公開鍵を管理できる。

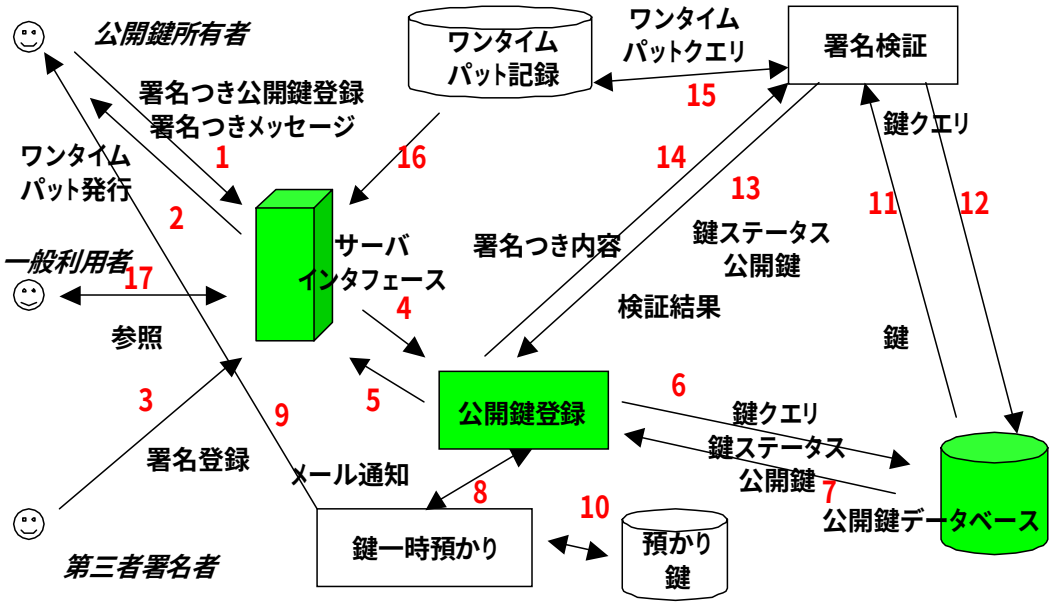

図 1 OpenPKSD Trusted Keyserver 構成図

※　緑の部分が既存の OpenPKSD 開発時に作成したクラスライブラリを利用する。他は新規作成部分。

３．２ 稼働環境

開発するソフトウェアは以下の環境で動作するものとする。

３．２．１　ハードウェア

CPU アーキテクチャー/CPU 種類	IA32 / Intel Pentium 3.0GHz 相当以上
メモリサイズ	512MB 以上 / 1024MB 標準
ハードディスク	空き容量 40GB 以上
ネットワーク	ギガビットイーサーネット

※平成１７年における標準的 IA32 アーキテクチャーマシンを想定

３．２．２　ソフトウェア

OS	Debian GNU/Linux（※）
データベース	PostgreSQL 8.0

※PostgreSQL8.0 を除くその他のソフトウェアは Debian GNU/Linux のディストリビューションに含まれるものを使用する

３．３ 使用する他のシステム

OpenPKSD version 0.2.8	OpenPKSD は OpenPGP(RFC2440)の仕様を満たす公開鍵をプールするための公開鍵サーバである。平成１３年度、１４年度の IPA 情報セキュリティ対策事業にて開発を行った。現在 OpenPKSD.ORG サイト(http://openpksd.org)ではソースコードを公開していると同時に、本システムを稼動させ OpenPGP 公開鍵サーバーのサービスを提供している。OpenPKSD 公開鍵サーバはオブジェクト指向言語 Ruby で書かれている。Pure Ruby 実装もあるが、OpenPKSD.ORG サイトでは全ソースコードの 2%を C 言語で書き換えたチューンされたバージョンが利用されている。

３．４ 機能概要

３．４．１　公開鍵バージョンコントロール機能

(ア) 鍵登録機能:　公開鍵の所有者は、自分自身の公開鍵で署名をつけた公開鍵を登録することができる。鍵は Owner Submitted Key (所有者が預け入れる鍵)として登録される。

① 登録者は Owner Submitted Key のみの設定を鍵サーバに通知すれば、一般への提供される公開鍵は Owner Submitted Key のみとなる。

構成図での処理の流れ １－４－６－７－１４－１３－５

公開鍵の所有者が署名をつけた公開鍵を受け取り、それを検証した後、データベースに登録する。

- 入力：署名つきの所有者の公開鍵
- 処理：公開鍵は所有者によって署名されているか検証する
- 出力：データベースへの登録

(イ) 提示コントロール機能:　Owner Submitted Key に対し所有者は次のことができる。

① 所有公開鍵のアップデート

構成図での処理の流れ １－４－６－７－１４－１２－１１－１３

② 所有公開鍵の一般への提供停止と再開

構成図での処理の流れ １－１６－２－１－４－１４－１５－１２－１１－１３

所有公開鍵のアップデート、一般への提供、停止、再開を指定する。

- 入力：公開鍵所有者の署名つき提示コントロールメッセージ

- 処理：データベース内にある公開鍵によって署名を検証する
- 出力：コントトールメッセージの実行（公開鍵の提供・停止）

(ウ) メール通知機能：　Owner Submitted Key への第三者署名が登録されたとき、メールで通知が行われる

① Owner Submitted Key 登録時、通知先メールアドレスを登録しておけば公第三者署名つき公開鍵が登録されたとき、通知が行われる。

構成図での処理の流れ
３－４－６－７－８－１０－９

② 通知登録の解除と再開

構成図での処理の流れ
１－１６－２－１－４－１４－１５－１２－１１－１３

第三者の署名つき公開鍵が登録されたとき、事前に指定していたメールアドレスへ登録されたことを通知する。

- 入力：公開鍵への第三者署名が登録された通知
- 処理：署名された公開鍵の所有者のメールアドレスへメールを送る
- 出力：通知メール

(エ) 預かり鍵取り寄せ機能：　Owner Submitted Key に対して新しい第三者署名がつけられた公開鍵が公開鍵サーバにある場合、その公開鍵をダウンロードできる。

① ダウンロードする。

構成図での処理の流れ
１－４－６－７－８－１０－８－１４－１３－５

② ダウンロード後は一定時間後に消去される。

構成図での処理の流れ
１０

所有公開鍵に署名が付加され、公開鍵サーバに第三者の署名つき公開鍵がアップデートされ、かつ確保されているとき、その公開鍵を所有者のリクエストにより提供する。

- 入力：公開鍵署名つき取り寄せメッセージ
- 処理：預かっている第三者署名つきの公開鍵を検索し取り出す
- 出力：預かっている公開鍵

３．４．２　公開鍵署名一時預かり機能

（ア）　一時預かり機能：第三者によって署名された公開鍵が登録されたとき、それが Owner Submitted Key であり、かつ所有者から第三者からのアップデートが許されない設定がされている場合、その公開鍵は一時預かり状態になり、公開鍵所有者のみが参照できる。

構成図での処理の流れ
３－４－６－７－８－１０

第三者によって署名された公開鍵が登録されたとき、それが既に所有者により済みであり、かつ第三者からのアップデートが許されていないとき、公開鍵を一時預かり状態にし、鍵所有者からの指示を待つ。一定時間処理されない場合は破棄される。

- 入力：第三者署名がついた公開鍵
- 処理：所有者の公開鍵に制限が設定されている場合に預かる
- 出力：なし（メール通知機能へ連動）

３．４．３　署名検証用公開鍵一括ダウンロード機能

（ア）一括ダウンロード機能：　目的の公開鍵につけられている第三者の署名に使われた公開鍵すべてを含めて公開鍵を一括でダウンロードする。

構成図での処理の流れ １７－４－６－７－５－３

３．５ 利用概要

ロール（利用者）	利用概要
公開鍵所有者 OpenPGP 公開鍵を所有しており OpenPKSD Trusted Keyserver へ登録を行う者	● 自分の公開鍵を公開鍵サーバへ登録する ● 公開鍵サーバ上にある公開している自分の公開鍵をアップデートする ● 自分の公開鍵を一般公開する（しない）メッセージに署名をつけて送付 ● 自分の公開鍵に第三者署名がつけられたものが公開鍵サーバに登録されたら所有者は通知を受け取る（一般には公開されない） ● 通知を受けたらアップデートされている自分の公開鍵をダウンロードする
一般利用者 OpenPKSD Trusted Keyserver から目的の公開鍵をダウンロードする者	● OpenPGP KeyID を指定し公開鍵を入手する ● 入手対象となる公開鍵に署名されている一部、もしくは複数の公開鍵を選択し一度に入手する
第三者署名者 公開鍵所有者の OpenPGP 公開鍵へ署名を行いOpenPKSD Trusted Keyserver へ登録を行う者	● 署名を行った OpenPGP 公開鍵を公開鍵サーバへ登録する

４．取扱い説明

以下の添付資料参照：

	ファイル名	内容
添付資料０１	manual.txt	OpenPKSD-TKS マニュアル インストール方法から内部プロトコル、データベーステーブルの内容等の説明
添付資料０２	draft-ietf-openpgp-rfc2440bis-14.pdf	OpenPGP の規格書

以上