

信頼できる OpenPGP 公開鍵を提供する公開鍵サーバ OpenPKSD Trusted Keyserver

株式会社 S R A 先端技術研究所
鈴木裕信

Abstract

OpenPKSD Trusted Keyserver (OpenPKSD-TKS) aims that an OpenPGP public key owner can to handle their own public key under OpenPGP public keyserver. Today's public keyserver as know as pksd and OpenPKSD and others can handle representation of public key not only an public key owner but also anyone who have someone's public key even thought a public key owner don't intention to show their own public key. OpenPKSD-TKS is designed to prevent such situation.

はじめに

本プロジェクトは OpenPGP 公開鍵所有者が OpenPGP 公開鍵サーバ上に保持されている自分の公開鍵に対して公開・非公開などのアクセスを自ら設定できる「信頼できる OpenPGP 公開鍵を提供する公開鍵サーバ Trusted Keyserver (正式名称: OpenPKSD Trusted Keyserver 略称: OpenPKSD-TKS)」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プロジェクトである。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。誰でも公開鍵の登録が可能であり、また、Web of Trust 方式を取り入れているため誰でもその公開鍵に署名を付加した上で登録することが可能である。しかし公開鍵所有者が自分の公開鍵をコントロールできない不満があった。それを解決する。尚、開発プロジェクト終了後は OpenPKSD-TKS は OpenPKSD.ORG のサイトで運用される。

背景

インターネット上で最も利用されている暗号技術は OpenPGP(RFC2440)である。

OpenPGPとはPGP互換ソフトウェア間でインターオペラビリティを持たせるための仕様である。OpenPGPはRFC2440という形で文章化されている。IETF RFCとしてまとめられるだけではなく、会社や団体からなる OpenPGP Alliance というグループなどがあり普及が進められている。OpenPGP仕様の暗

号化ツールとしてPGPやGNUPGなどが使われている。

1991年 Philip Zimmermann が作成した暗号ツールで公開鍵暗号、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1セット揃っている。最も名前が知られている暗号ツールである。現在 PGP Corp, Inc が開発を引き継いでおり、非商用目的であれば、自由にダウンロードし利用できる。GNUPGとはGNU Privacy Guard プロジェクトの一環として作られた OpenPGP 仕様の暗号化ツール。GPL Public License に基づき公開されているので、誰でも自由に利用できる。GPL Public Licenseとはフリーソフトウェア財団のGNUプロジェクトで用いられているライセンス条件で、著作権を保持した上で、第三者への公開、第三者の自由な改変・再配布を許すライセンスである。Linuxのカーネルコードなどもライセンスとして採用している。

その OpenPGP のユーザ間で公開鍵を交換するための公開鍵インフラ(PKI)に使われているサーバソフトが PGP Public Keyserver である。

インターネットのような広域で公開鍵を利用する際の問題の1つに多数間で鍵配送を行うと指数的に手数が増える問題がある。そこで自分の公開鍵を「広く公開」することによって鍵配送の負担を減らすことが可能である。その目的のために公開鍵をプールするサーバとして作られたのが PGP Public Keyserver である。誰でも公開鍵の登録が可能であり、また、Web of Trust 方式を取り入れているため誰でもその公開鍵に署名を付加した上で登録することが可能である。最初のバージョンは1994年に作られた。世界に散らばる主要な PGP Public

Keyserver 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 Public Keyserver へ登録されるメカニズムを持っている。尚、本文中では PGP Public Keyserver と鍵サーバは同じ意味で使っている。

運用は世界各地の鍵管理者コミュニティのボランティアによって支えられ、インターネットセキュリティの向上に貢献している。PGP Public Keyserver の管理者はヨーロッパでは電子認証技術研究及びセキュリティ研究関係者が多く、アメリカでは大学のコンピュータセンター関係者が多い特徴がある。現在世界で中心的な PGP Public Keyserver は約 10 程度ある。2000 年 5 月にオランダ SURFNET で第一回 PGP Public Keyserver Manager Symposium が開催された。参加者枠 25 名で世界 8 カ国から関係者が集まったが、アジア圏では唯一、本プロジェクトの提案者である筆者が参加した。

筆者は 2001 年に PGP Public Keyserver (Horowitz 版)互換の OpenPKSD システムを構築した。このシステムを使い公開鍵サーバを OpenPKSD.ORG サイト (<http://openpkds.org>) にて一般に公開している。OpenPKSD.ORG 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つている。

OpenPKSD.ORG サイト

- <http://openpkds.org>

公開鍵が増えていつている一方で、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の解決を目指した。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。

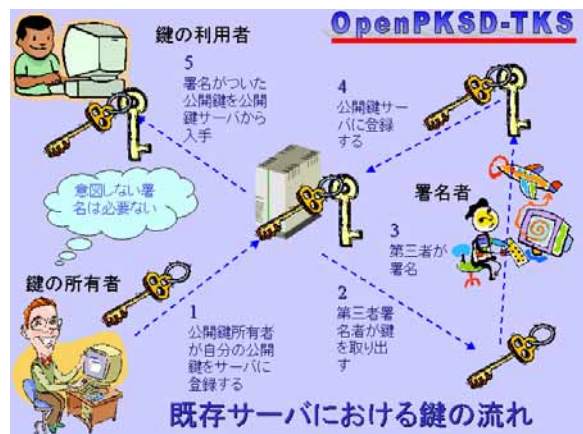


図 1：既存サーバにおける鍵の流れ

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とえ、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には使えない。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。

公開鍵の流れ

公開鍵サーバを中心に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けて考える。

- 公開鍵所有者が登録した公開鍵を第三者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。

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。

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。

OTPを使う安全なリクエスト処理

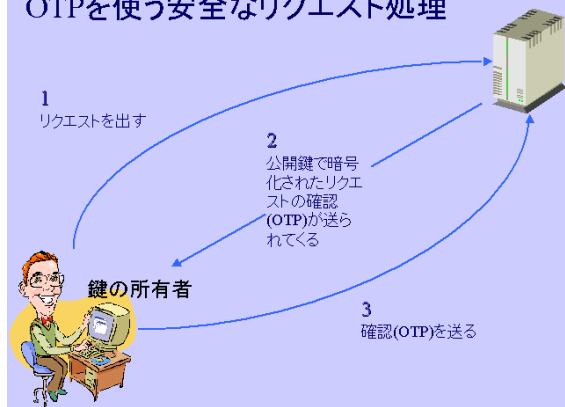


図 2 OTP を使う処理

- 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。

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。

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しない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。

システム構成

本システムは以下のコンポーネントから構成されている。

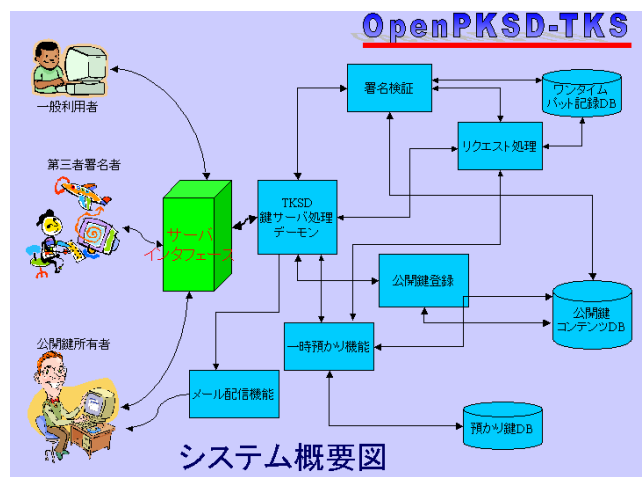


図 3 システム構成図

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、今後、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。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が行われる。

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。

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。

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。

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くつもあり連動して動いている。

ソフトウェアの公開について

本システムは準備が整い次第、GNU General Public License に基づき公開される予定である。

しかし前回の OpenPKSD の経験から、ソフトウェアを開発・改善するだけでなく、ソフトウェアを使うコミュニティ、そしてソフトウェアを改良していくコミュニティの両方をエンカレッジする必要性に気づいた。

OpenPKSD の開発・公開と同時期に SKS という公開鍵サーバが開発されて公開された。ほぼ同じような機能と性能であるが今や SKS の知名度の方が大きい。利用者に関しては計測

がむずかしいので判定はできないが、知名度 = google で発見できるページ数の違いという形で観察することができる。” OpenPKSD Keyserver ”の検索結果が 445 件であるのに対し” SKS Keyserver ”は 14,000 件である。

機能でいえば、ほぼ互角である。OpenPKSD は Ruby で書かれているが SKS も Haskell という言語で書かれているので開発バックグラウンドとしては同じレベルと考えてよい。

知識共有プラットフォームの必要性

圧倒的に知名度が低いのは何故かを考えてみた。SKS の方は利用者コミュニティ、開発者コミュニティの場が用意されており、OpenPKSD の方が用意されていないという部分に大きく関係しているという結論に達した。

オープンソース開発は、少し前までは Web サイトにありソフトウェアが公開されれば良い程度であった。今では“リポジトリ”・“アーカイブされたメーリングリスト”・“CMS (Content Manage System)”が三種神器となっている。

リポジトリは sourceforge や sabanna のようなソースコードリポジトリを管理するサイトや、あるいは CVS サーバを用意し利用することである。アーカイブされたメーリングリストとは mailman やあるいは fml などメーリングリストを維持しつつ、ウェブサーバなどで参照できるようにして、メーリングリスト上での知識を共有することである。CMS は Wiki や Zope/Plone あるいは XOOPS などで構築されたサイトで、これもまた知識を蓄積し共有するためのものである。

今日において普及を進めるということは利用者同士や開発者同士がコミュニケーションを取る場を維持することと同意語になりつつあると考える。この点を把握した上でソフトウェアの公開と普及の戦略を立てなければならない。

今後の予定

ここに当初の目的である公開鍵所有者が自分で公開鍵をコントロールするための仕組みは出来上がった。実装においては現在バージョンレベルで基本的な仕様をクリアしたレベルである。このままでは本格的なサーバ運用までは耐えられないレベルである。外部へサービスを提供するまでのレベルには達しているとは言えない。

またサーバ管理をするための周辺運用ツールや DoS に対する機能など現実的な運用に必要な機能まで整っていない。また、このような新しい仕組みが本当にユーザに受け入れられるかどうかの運用実験を行う必要がある。現状

で考えている次の6ヶ月間の計画は以下の通り。

- 三ヶ月を目処にベータバージョンの完成及びソースコードの公開。
- このシステムを使い6ヶ月以内にサービスを提供開始。
- tks.openpkd.org サイト専用のSSL証明書を取得。
- OpenPKSD.ORG にあったメールアドレスによる検索や表示などをマージ。
- 利用者コミュニティ向け情報サイトの構築。
- 開発コミュニティの構築。

まとめ

本システムは公開鍵所有者が自らの鍵サーバ上にある公開鍵の配布・非配布をコントロールできるシステムを作成した。

既存の公開鍵サーバは公開鍵所有者のコントロールが効かなかったものを出来るようにしたため、公開鍵所有者が望む形の公開鍵を配布することが可能になる。

既存のシステムでは公開鍵所有者の関与ができないため今までウェブサイト上においていたり、あるいはメールでやり取りしていたユーザに対しての解決法を提示できる。

公開鍵を入手する先として公開鍵サーバを統一的に利用できるようになる。

今後はベータバージョンレベルまでもっていき tks.openpkd.org を立ち上げ運用を行い、またソースコードを公開する。