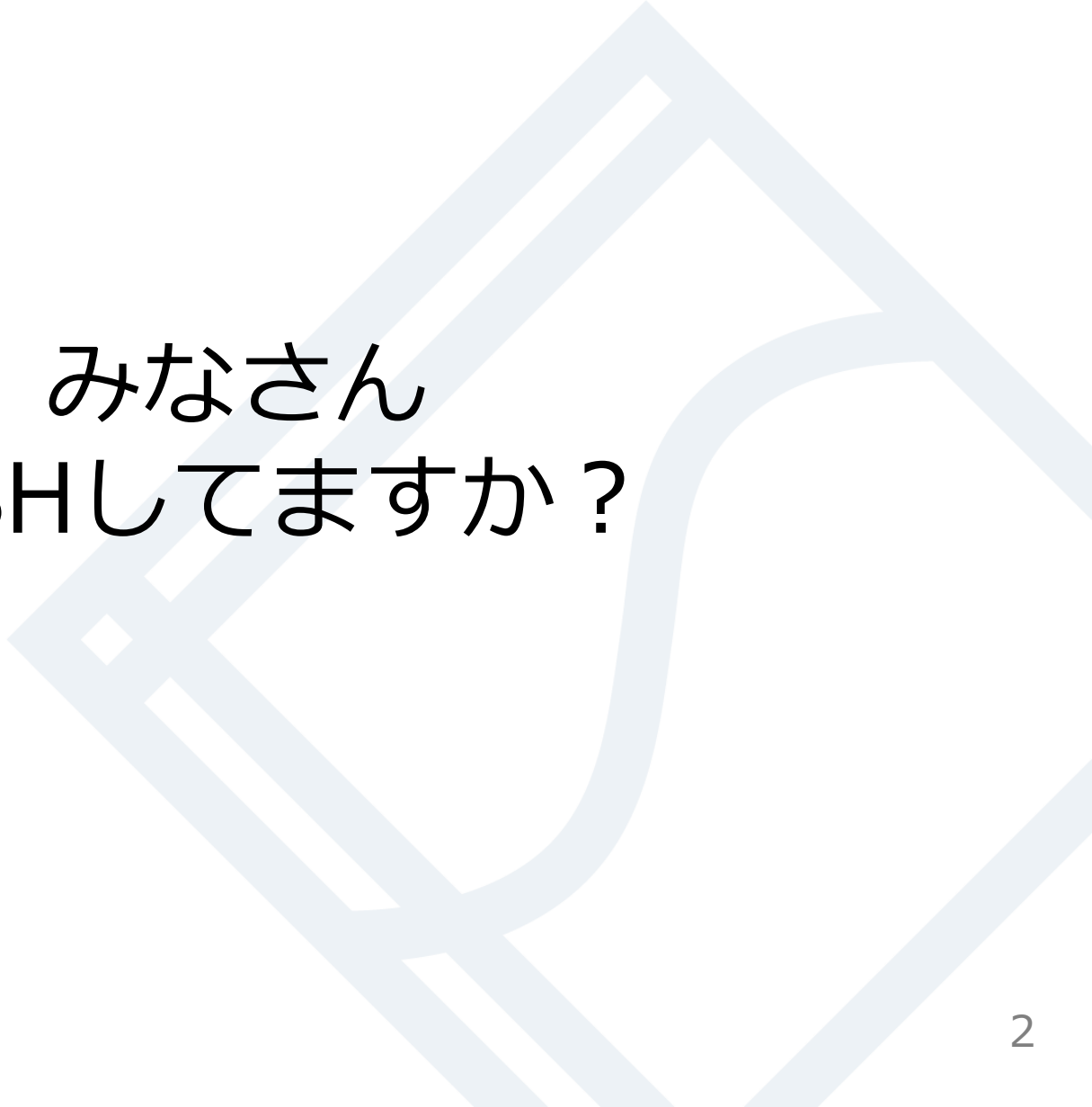


.ssh/configを管理する .ssh/configで管理する

Ken'ichiro Oyama
Fusic Co.,Ltd.
2016.11.3



みなさん
SSHしてますか？



わたしは
SSHしてます



?> NEXT

PHPカンファレンス2016のテーマ

SSHとNEXT

- > FTP
- > SSH + パスワード
- > SSH + 公開鍵認証
- > 「SSHしたら負け」 <- たぶんここがNEXT
 - > Docker、EC2 Run Command、etc

“NEXT”ではないですが、
今日は“SSH + 公開鍵認証”で少しでも役に立ちそうな話をします

A decorative graphic in the bottom right corner consisting of a light blue square frame with a thick border. Inside the frame, there is a stylized, thick blue line that forms a shape resembling a stylized 'P' or a curved path.

Who

k1LoW

- > Ken'ichiro Oyama
 - > @k1LoW
- > Fusic Co.,Ltd. エンジニア
 - > 基盤ユニット テックリード
- > GitHub organizations
 - > fukuokarb / dotcake / emacs-jp / etc.
- > awssecというAWS用のテストツールを作っています
 - > <https://github.com/k1LoW/awssec>

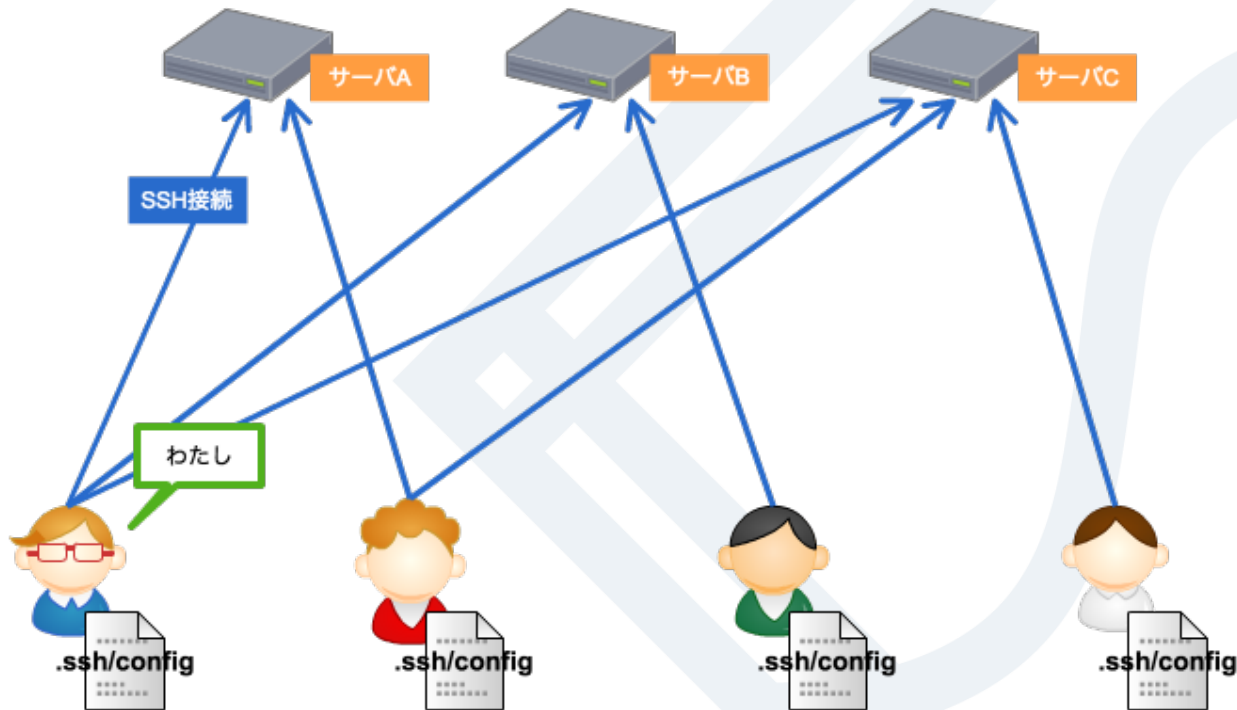


普段のわたし

- > 複数のプロジェクト
 - > 開発中
 - > 運用中
- > 複数のサーバ
 - > プロジェクト単位
 - > ロール単位 (Web、DB、etc)
 - > 本番、ステージング、テスト

イメージ

> 複数のプロジェクトの複数のサーバにSSH



~/.ssh/config

- > SSH接続の管理をするための設定ファイル
 - > ~/.ssh/configはユーザごとの設定ファイル

```
>Host github.com
>  User git
>  Hostname github.com
>  PreferredAuthentications publickey
>  IdentityFile ~/.ssh/github_rsa
```

/home/k1low/.ssh/config

- > 管理しているサーバへSSH接続するための設定を記載
- > 秘密鍵も .ssh/ 以下に配置
 - > プロジェクトごと、ロールごと、サーバごとなどで分けることが多い
 - > id_rsaは使わない
- > 秘密鍵だけでなく、その他の接続設定も重要
 - > HostnameやUser
 - > ProxyCommandとか



`.ssh/config`
どうやって管理していますか？

.ssh/configの管理

- > configのバックアップ/リストア
- > 秘密鍵のバックアップ/リストア
- > サーバ接続情報の追加/削除
- > チームメンバーへの共有
- > 踏み台サーバなど他サーバへの設置

configと秘密鍵に分かれていて 面倒

configは分割できないので
メンバーへの共有も面倒

今のconfigとバックアップした configのマージも面倒

A decorative graphic in the bottom right corner consisting of a light blue diamond shape with a thick border. Inside the diamond is a stylized, thick blue line that forms a partial circle or arc, resembling a stylized letter 'C' or a part of a logo.

sconb

sconb

- > “Ssh CONfig Buckup tool”
 - > .ssh/config専用のバックアップツール
 - > <https://github.com/k1LoW/sconb>
- > `gem install sconb` でインストール
- > ※本日も大変お世話になっているCONBUとは無関係です

使い方

バックアップ

```
>$ sconb dump > ssh_config.json
>$ cat ssh_config.json
>{
>  "github.com": {
>    "Host": "github.com",
>    "User": "git",
>    "Hostname": "github.com",
>    "PreferredAuthentications": "publickey",
>    "IdentityFile": [
>      "~/.ssh/github_rsa"
>    ]
>  },
>  ...
```

絞り込んでバックアップ

```
>$ sconnb dump example > ssh_config.json
>$ cat ssh_config.json
>{
>  "example.com": {
>    "Host": "example.com",
>    "User": "k1low",
>    "Hostname": "dev.example.com",
>    "PreferredAuthentications": "publickey",
>    "IdentityFile": [
>      "~/.ssh/example_rsa"
>    ]
>  }
>}
```

リストア

```
>$ sconb restore < ssh_config.json  
>Host github.com  
>  User git  
>  Hostname github.com  
>  PreferredAuthentications publickey  
>  IdentityFile ~/.ssh/github_rsa  
>...  
  
>$ sconb restore < ssh_config.json > ~/.ssh/sshconfig
```

秘密鍵も含めて1ファイル化

```
>$ sconnb dump --all > ssh_config_with_key.json
>$ cat ssh_config_with_key.json
>{
>  "github.com": {
>    "Host": "github.com",
>    "User": "git",
>    "Hostname": "github.com",
>    "PreferredAuthentications": "publickey",
>    "IdentityFile": [
>      "~/.ssh/github_rsa"
>    ],
>    "IdentityFileContent": [
>      "-----BEGIN RSA PRIVATE KEY-----\nMIIEpXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX..."
>    ]
>  },
>  ...
```

秘密鍵のリストア

```
>$ sconb keyregen < ssh_config_with_key.json  
>Regenerate ~/.ssh/github_rsa ...  
>...
```




つまり ssh_config <-> JSON のコンバータ

パイプで各種UNIXコマンドとも 連携できる

jqを使ったssh_config.jsonのマージ

> jq: JSONをいい感じに操作できるコマンド

```
>$ jq -s '.[0] + .[1]' a.json b.json | scomb restore > ~/.ssh/config
```



.ssh/configを管理する
sconbはおいしい

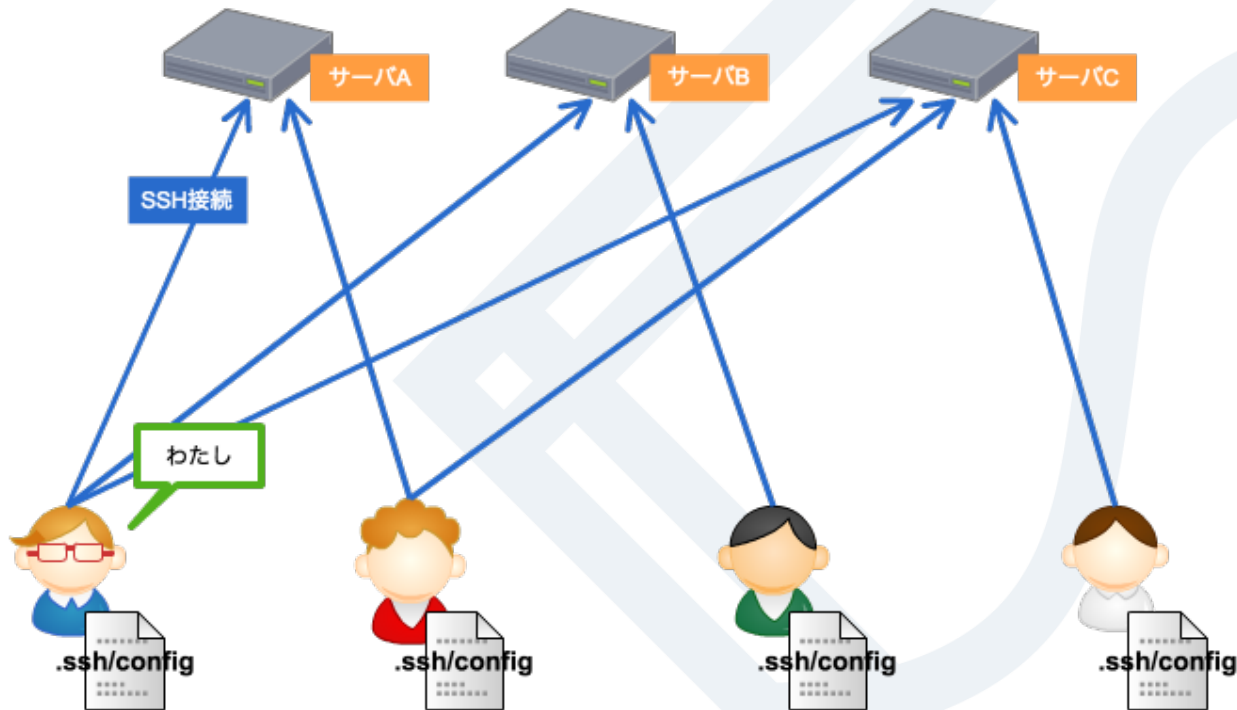
別の視点から

普段のわたし（再掲）

- > 複数のプロジェクト
 - > 開発中
 - > 運用中
- > 複数のサーバ
 - > プロジェクト単位
 - > ロール単位（Web、DB、etc）
 - > 本番、ステージング、テスト

イメージ（再掲）

- ＞ 複数のプロジェクトの複数のサーバにSSH



突然のOpenSSLの脆弱性発表

インベントリ情報

- > 「サーバにインストールされているパッケージのバージョン一覧を再度確認して欲しい」
 - > 現在稼働しているサーバ数十台
 - > まったく別の時期にまったく別のサービスを構築している
- > 全部SSHでログインして確認。。。??そしてその後まとめるの。。。?
- > ツラみがある。。。

A small, fluffy white poodle dog is the central focus of the image. It has a perfectly rounded, pom-pom-like head and large, dark, expressive eyes. The dog is looking directly at the camera with a calm expression. It is wearing a dark, patterned bow tie around its neck. The dog is standing on a brown rug that features a repeating pattern of green stylized leaves or ferns. The background is a warm-toned wooden wall or paneling, with a striped fabric hanging down on the left side. The overall lighting is soft and indoor, creating a cozy atmosphere.

koma

koma

- > エージェントレスでリモートホストのインベントリ情報を収集するツール
 - > <https://github.com/k1LoW/koma>
 - > 類似ツール: ohai / facter
- > `gem install koma` でインストール
- > `ssh` を `koma ssh` に変えるだけで、リモートホストのインベントリ情報をJSONで取得できる

koma ssh

```
>$ koma ssh example.com
>{
>  "memory": {
>    "swap": {
>      "cached": "652kB",
>      "total": "2097148kB",
>      "free": "2091376kB"
>    },
>    "total": "1019956kB",
>    "free": "255104kB",
>    "buffers": "310644kB",
>    "cached": "185748kB",
>    ...
>
```

koma keys (収集できるインベントリ情報)

- > memory
- > ec2 (disabled)
- > hostname
- > domain
- > fqdn
- > platform
- > platform_version
- > filesystem
- > cpu
- > virtualization
- > kernel
- > block_device
- > package
- > user
- > group
- > service

koma run-command

```
>$ koma run-command example.com,example.jp uptime
>{
>  "example.com": {
>    "uptime": {
>      "exit_signal": null,
>      "exit_status": 0,
>      "stderr": "",
>      "stdout": " 00:18:10 up 337 days,  4:51,  1 user,  load
average: 0.08, 0.02, 0.01\n"
>    }
>  },
>  "example.jp": {
>    "uptime": {
>...
```

そう、`.ssh/config`が設定されて
いけばね！



`.ssh/config`で管理する



ssh_config形式で管理する
~~.ssh/configで管理する~~

koma with ssh_config

- > komaはssh_config形式のフォーマットをSTDINから読み込んで実行することができる
- > Vagrant上のOSのcpu情報を取得

```
>$ vagrant ssh-config | koma ssh --key cpu
```

- > sconbをでHostをフィルタリング

```
>$ sconb dump dev.* | sconb restore | koma ssh
```

Vuls



Vuls

- > “VULnerability Scanner”
 - > バルス
- > Linux/FreeBSD向けの脆弱性スキャンツール
 - > Ubuntu、Debian、CentOS、Amazon Linux、RHELに対応
 - > 標準のTUIだけでなくVulsRepoというWebビューワもある
 - > NVDやJVNの脆弱性情報などを使用してスキャン
- > スキャン対象のサーバーへは“SSH”で接続
 - > TOMLで設定を記述



SSH。。。だと。。。？



ssh_config_to_vuls_config

ssh_config_to_vuls_config

- > ssh_config形式をVuls用のTOMLファイル形式に変換
 - > sc2vcコマンド
 - > https://github.com/k1LoW/ssh_config_to_vuls_config
- > `gem install ssh_config_to_vuls_config` でインストール

```
>$ cat ~/.ssh/config | sc2vc > vuls_config.toml
```

- > (当然) sconbでフィルタリングも可能

```
>$ sconb dump dev.* | sconb restore | sc2vc > filtered_config.toml
```

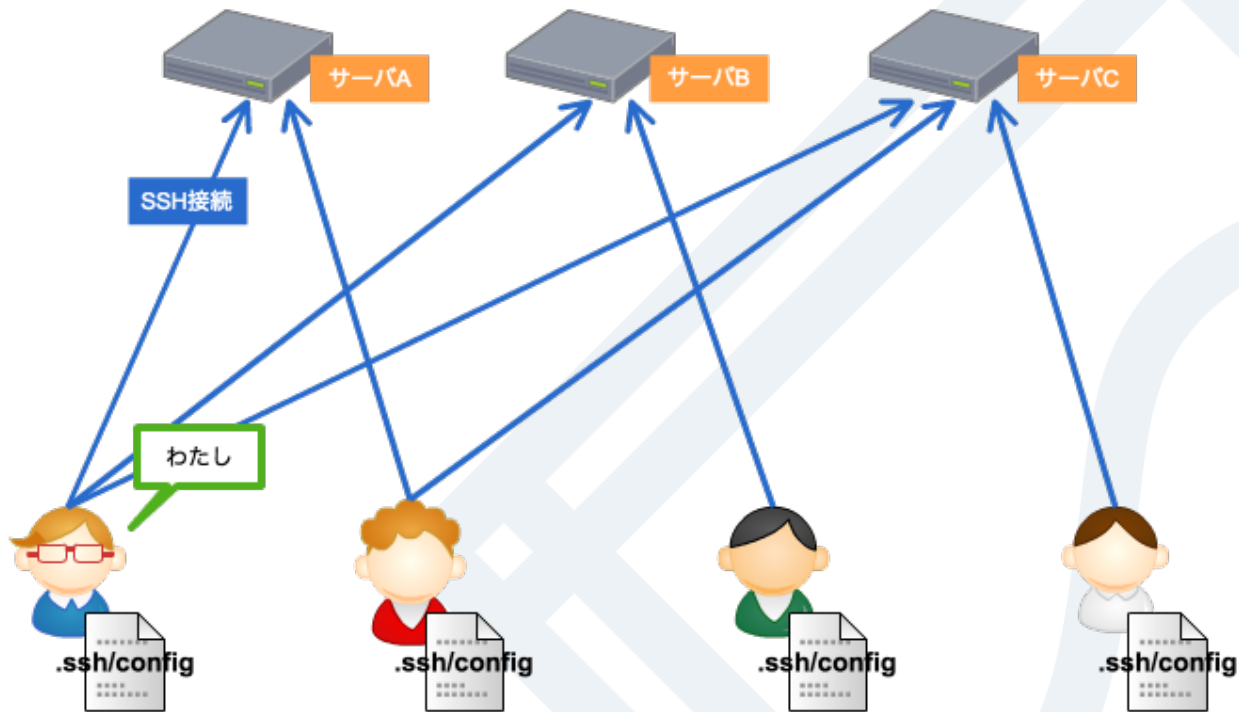
.ssh/config -> JSON
JSON -> .ssh/config
JSON -> koma
JSON -> Vuls



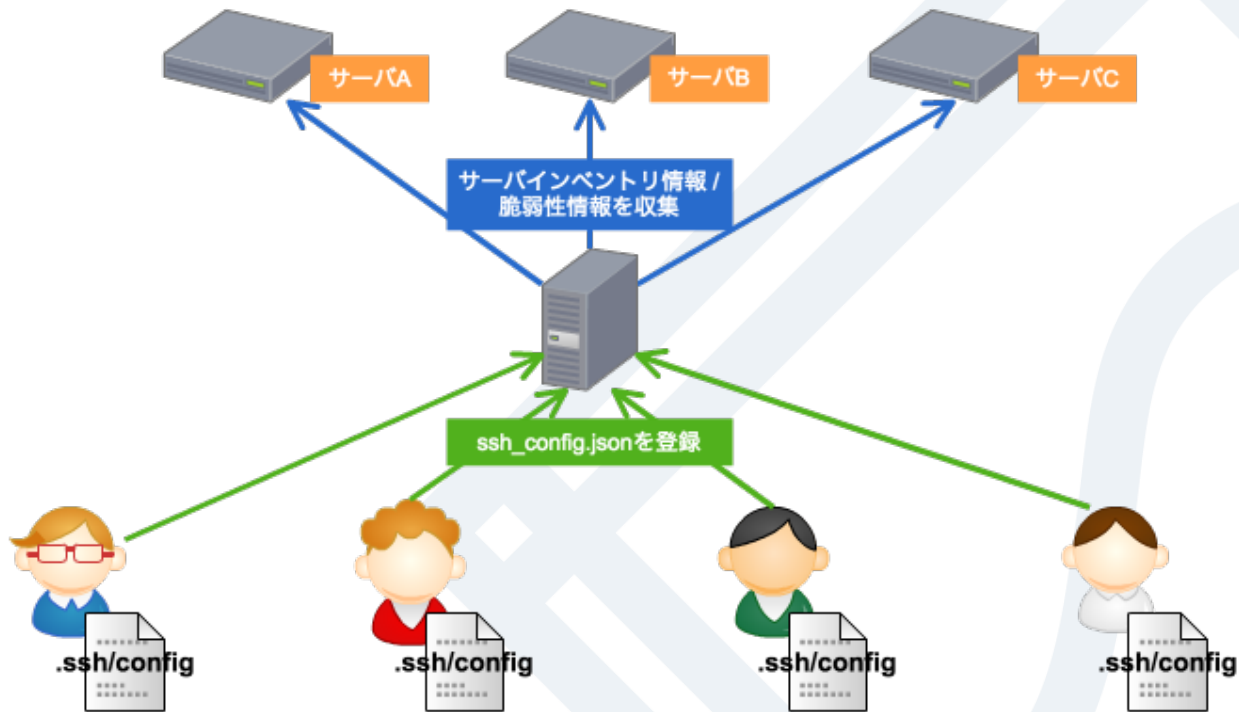
NEXT

ssh_configファイルを集約すれば
サーバインベントリ情報も
脆弱性情報も
管理できるのでは？

こんなサーバを



.ssh/configで管理



komad (仮)

- > sconbなJSONをAPI or SCP経由で登録
- > ssh_config + komaでサーバインベントリ情報を収集管理
- > ssh_config + Vulsで脆弱性情報を収集管理
- > koma run-commandでSSHを通じた外形監視
- > ちょっと変わったサーバ管理/監視システムができそう

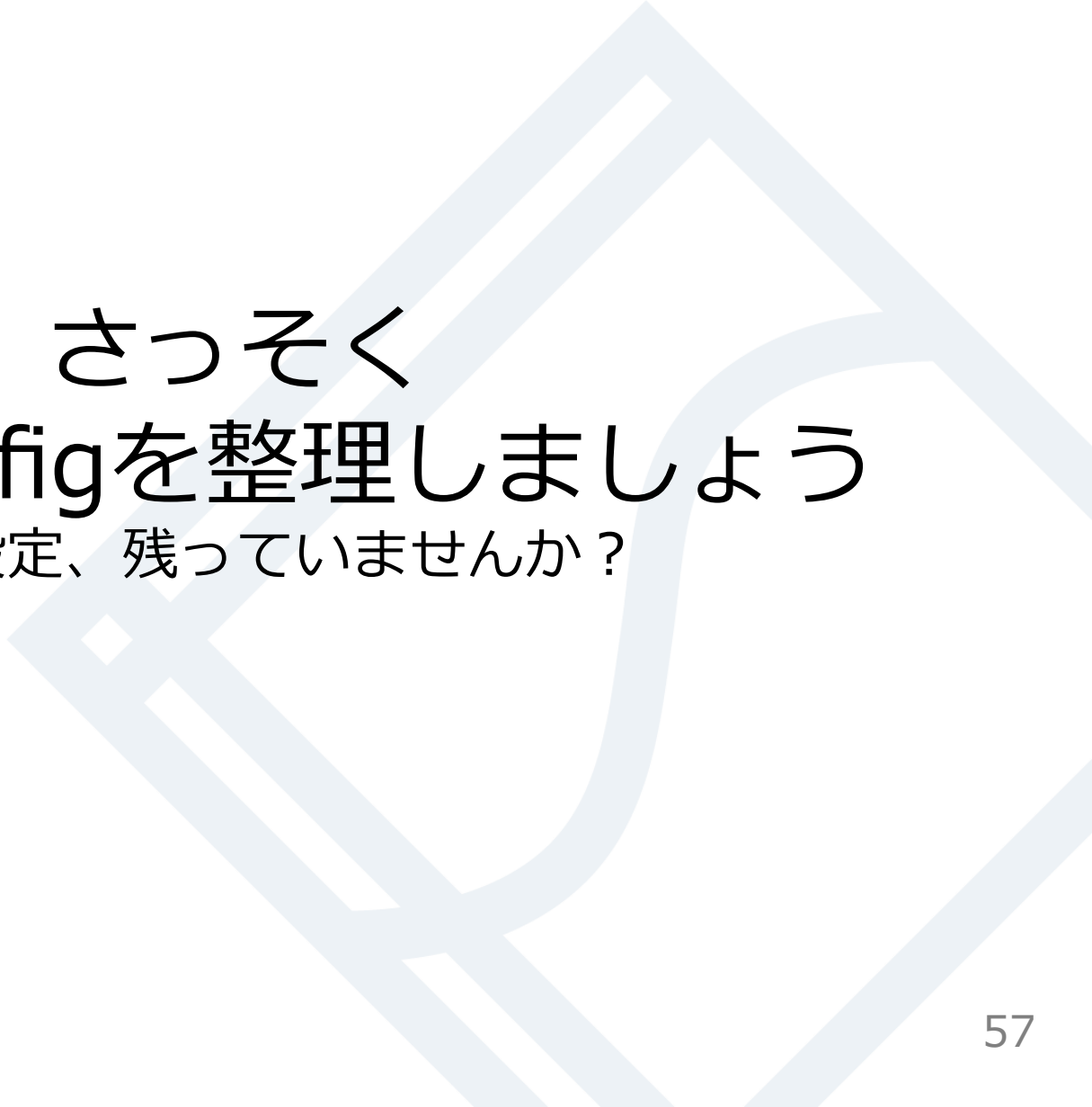


.ssh/configで管理する

まとめ

まとめ

- > .ssh/configをsconbでJSON形式にすることで管理容易性、再利用性を向上
- > .ssh/configやssh_config形式のフォーマットを入力として動くエージェントレスなサーバーインベントリ収集ツール koma
- > Vulsの設定の元として.ssh/configの活用
- > ssh_configを活用したサーバ管理/監視システムkomad（仮）の可能性



さっそく
.`ssh/config`を整理しましょう
古い設定、残っていませんか？

Thank you!

- Fusicはテクノロジーが
- 好きなエンジニアを募集しています
- <https://fusic.github.io>



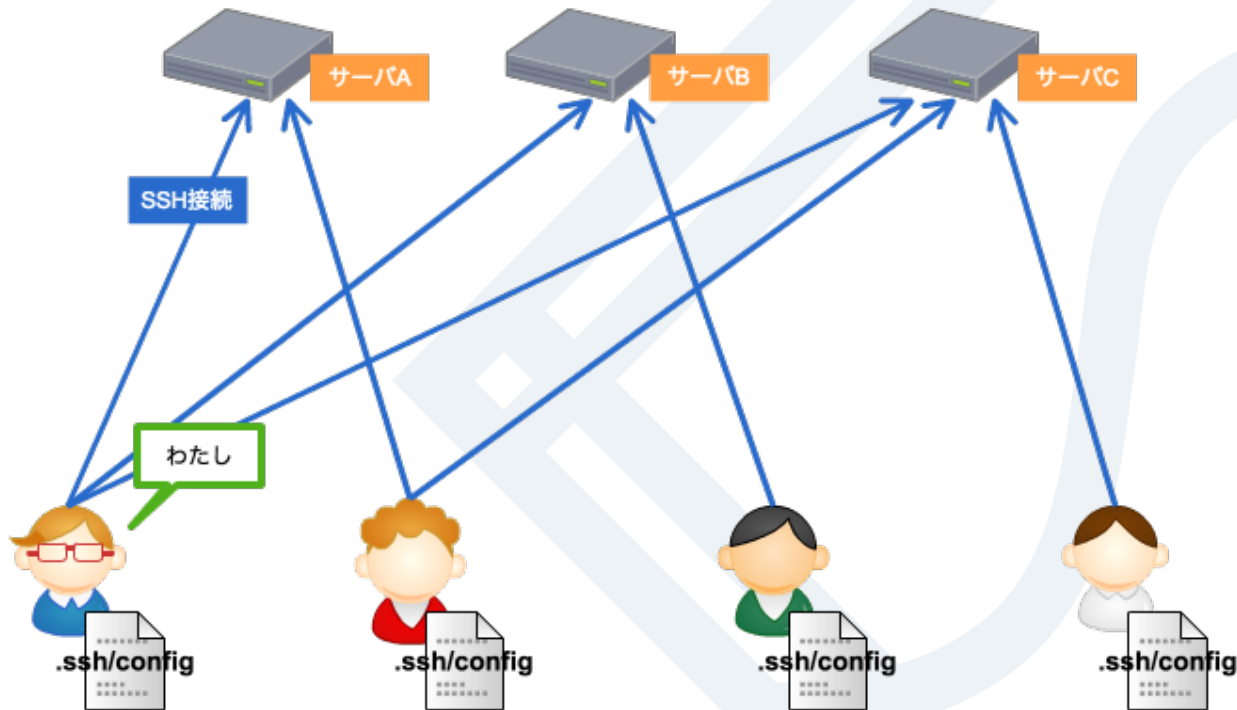
?>



NEXT

イメージ（再掲）

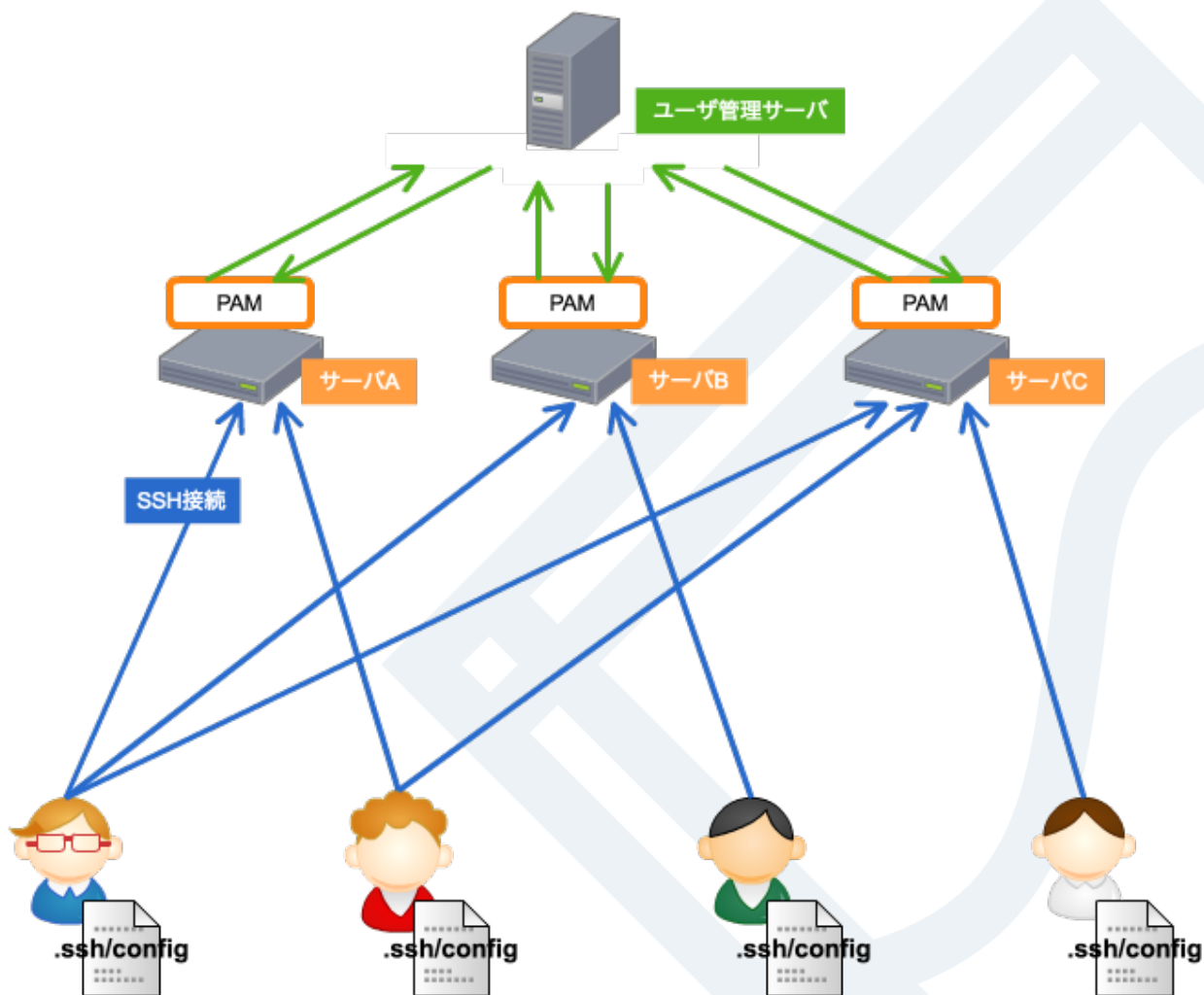
- ＞ 複数のプロジェクトの複数のサーバにSSH



サーバのユーザを管理したい

- > どのサーバにどのユーザがSSHログインできるのか
 - > 簡単なものでいい
 - > ある程度でいい
- > もしくはSSHログインできる情報をもらえるようにしたい
- > 公開鍵とか秘密鍵とかの管理

一般的な方法



PAM

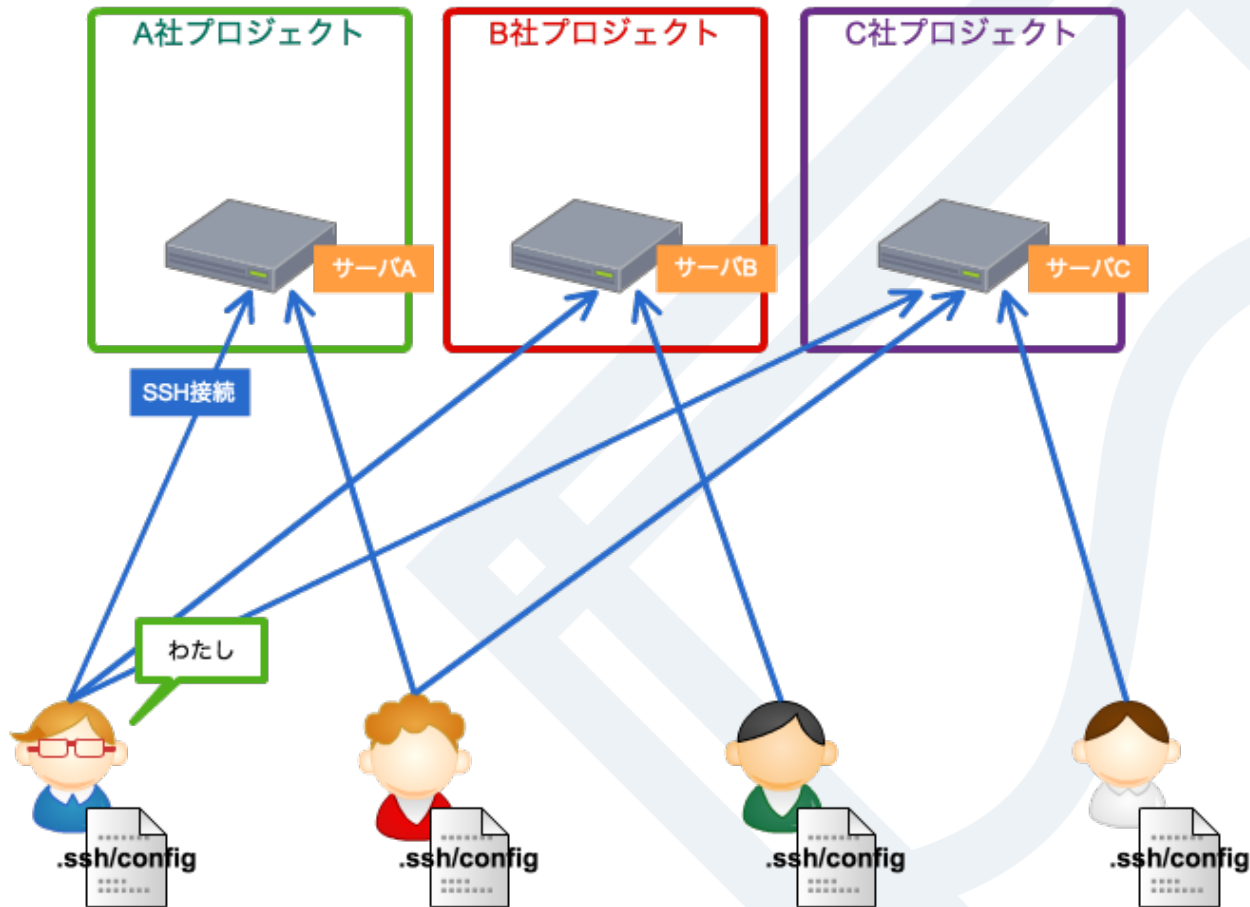
- > Pluggable Authentication Module
- > PAMを使ってユーザ管理サーバ（認証基盤）に認証の要求をもらう
 - > ADとかLDAPが一般的らしい
- > ただ個人的にはSTNS推し
 - > シンプル
 - > 設定がTOML
 - > 認証サーバ側をサーバレスにできそう
 - > <http://qiita.com/shogomuranushi/items/f09fcdeb146b45452403>
 - > 新イケメンことP山さん(@pyama86)が開発している



だかしかし

プロジェクトを横断できない

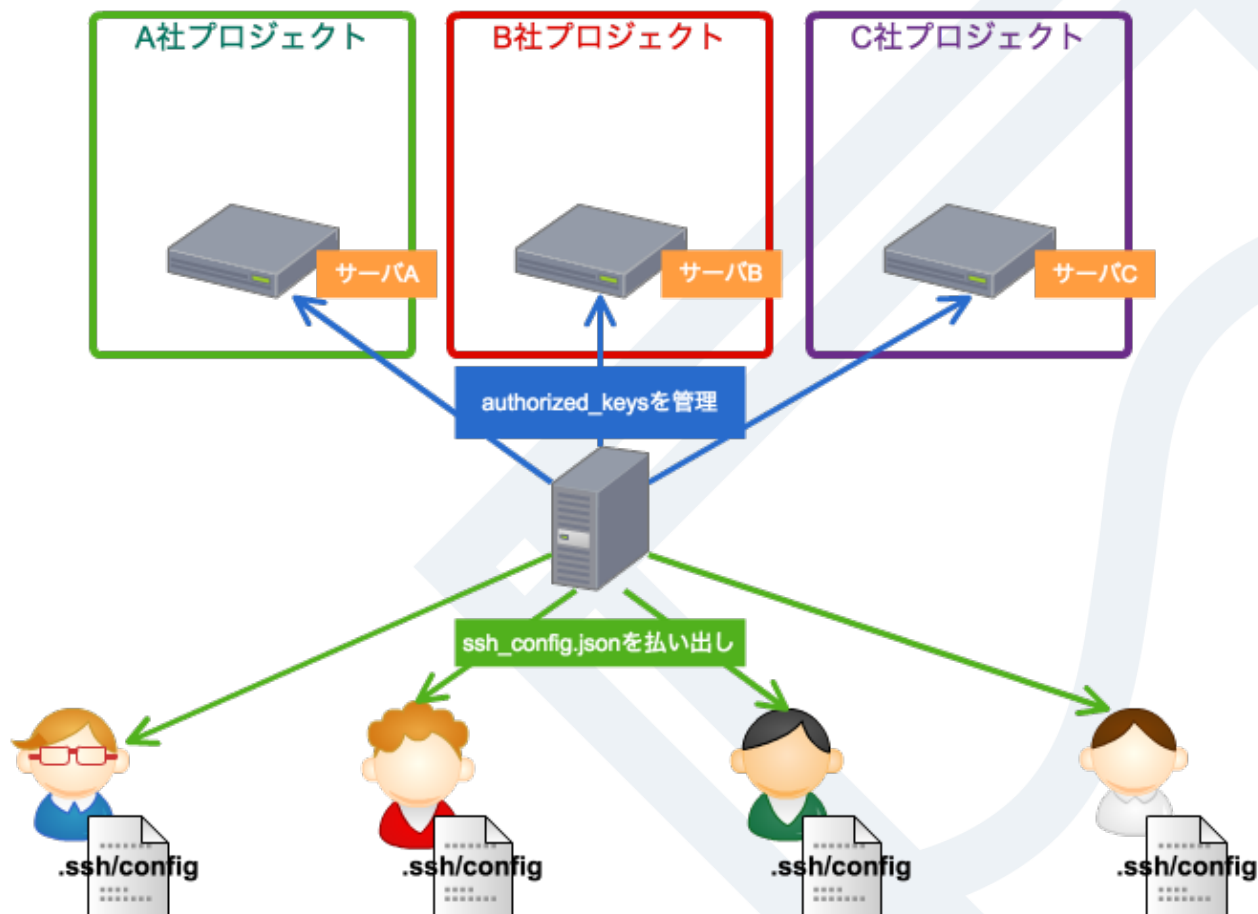
> 完全弊社都合



komad（仮）がssh_configを
払い出したらどうだろう

komad (仮) にもらう

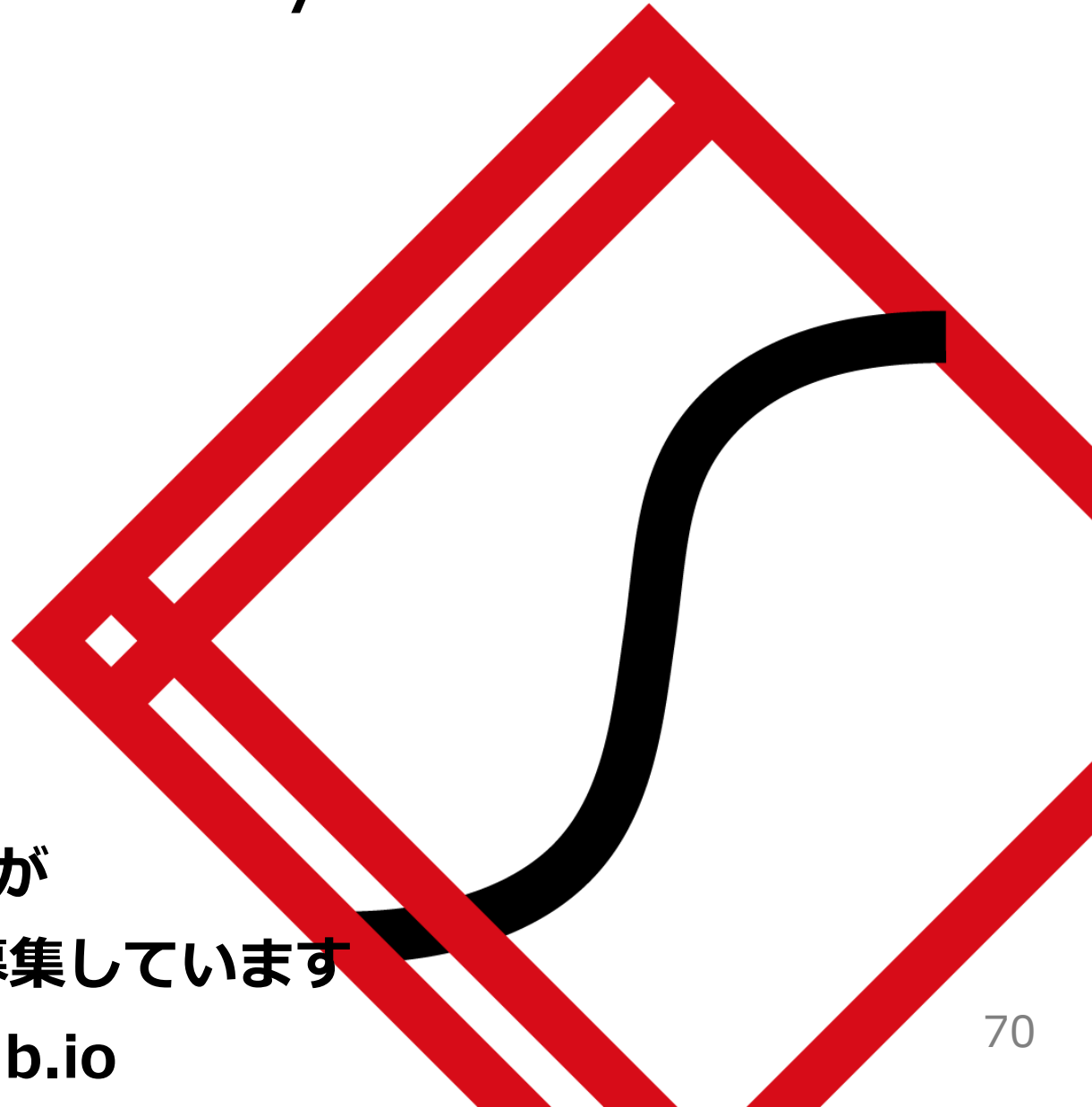
- > API経由でssh_config.jsonをもらう



どうでしょう。。？

- > イビツなのは承知
- > komad（仮）サーバの運用。。。
- > 無理があるか。。。

Thank you!



- > Fusicはテクノロジーが
- > 好きなエンジニアを募集しています
- > <https://fusic.github.io>