PacSec2008 at 青山ダイヤモンドホール

振る舞いに基づく SSHブルートフォース対策

平成20年11月13日

TOMOYO Linux Project

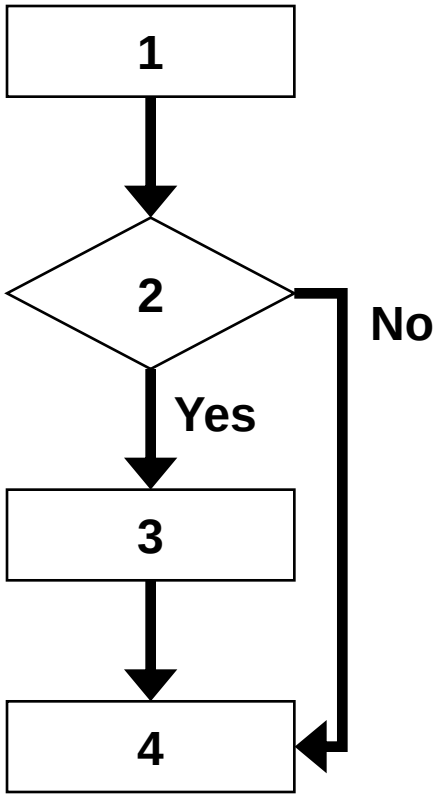
半田　哲夫

はじめに

- 遠隔管理などで利用されるSSHサービスへの不正なログインを許してしまうと、情報漏えいだけでなく、踏み台やトロイの木馬の設置のような被害が生じます。
  - 近年の攻撃手法は高度化・洗練化されてきており、従来の防御手法だけでは対処しきれない可能性が増えてきています。
- 「アクセス制御機能を強化したOS」を用いて、全く新しい防御手法を紹介します。
  - 例として TOMOYO Linux を利用します。

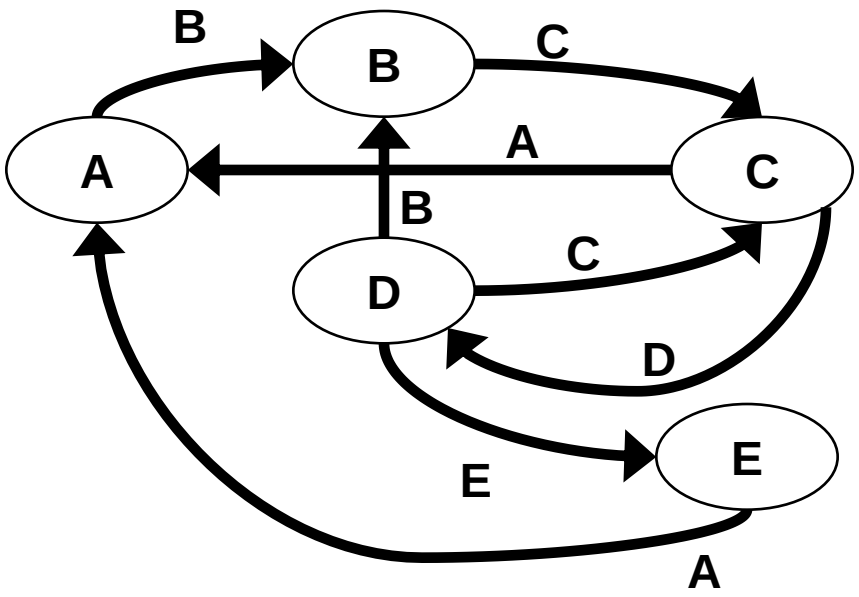
準備:フローチャート

・この図の見方はご存じですよね？



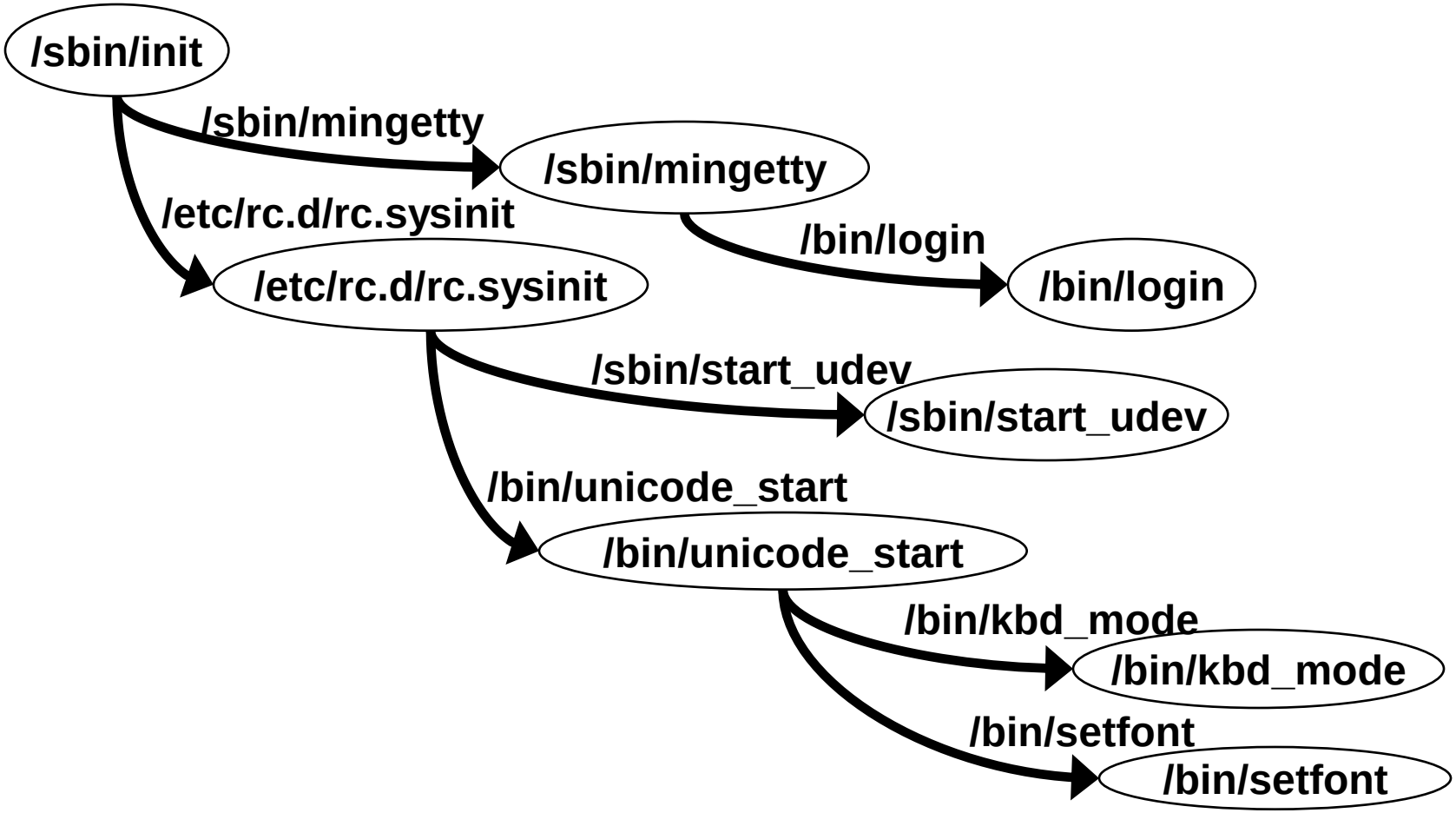
準備:状態遷移図

・この図の見方はご存じですよね？



準備:Linux における状態遷移図の例

・ /sbin/initを起点にツリー状に広がります。

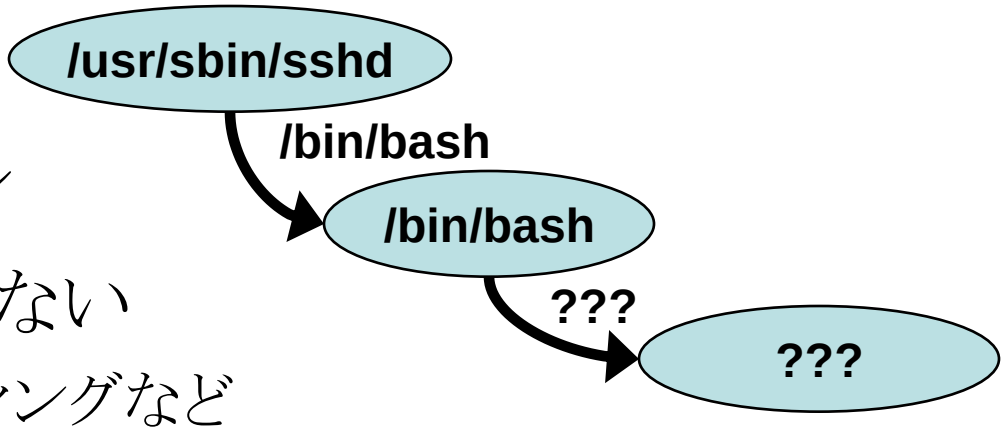


準備：TOMOYO Linuxとは

- 状態遷移をデザインし強制するツール
  - プログラムの中で発生するプログラムの実行要求を監視し、その可否を制限する。
  - プログラムの実行により状態遷移を行う。
- リクエストを観測し制限するツール
  - プログラムの中で発生するファイルの読み書き要求を監視し、その可否を制限する。
  - プログラムの実行要求やファイルのオープン要求によりプロセスの内部状態を更新する。

準備:SSHセッションの種類

- 対話型シェルセッション
  - シェルが提供され、自由にコマンドを入力できる
    - 自由に資源にアクセスできる
- 非対話型シェルセッション
  - シェル起動時に -c で指定されたコマンドが実行される
    - scp や sftp など
- 非シェルセッション
  - シェルが提供されない
    - ポートフォワーディングなど



従来の対策

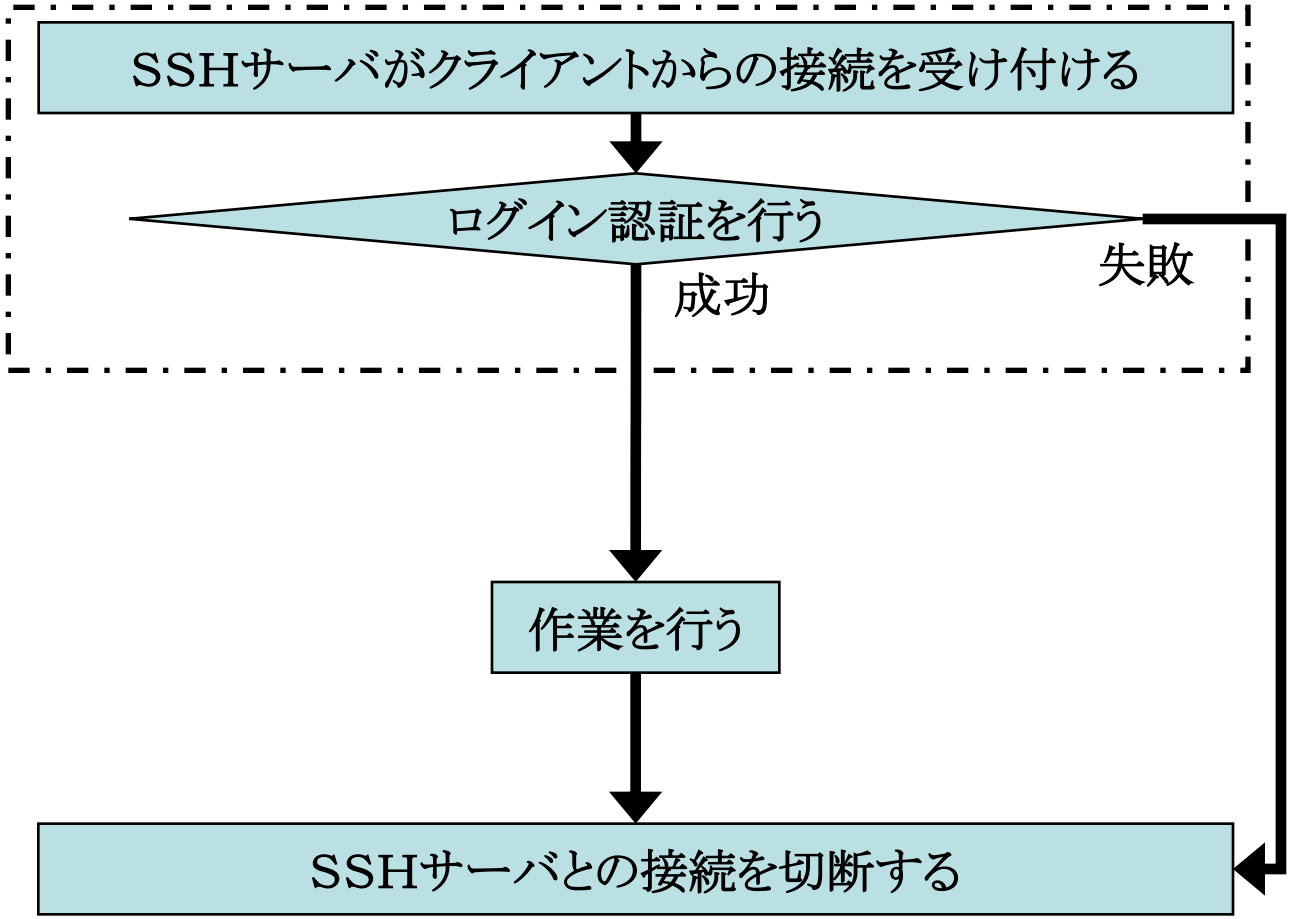
- 知識に基づく認証
  - ブルートフォース攻撃の対象になります。
- ログイン認証を突破されないことを前提とした対策
  - 突破される確率を減らす
    - ファイアウォールと連動して認証に失敗したクライアントを一定期間再接続禁止にする
    - 公開鍵認証を用いる
  - 攻撃の分散化やパスワード・秘密鍵の漏洩が始まっています。

提案する対策

- 振る舞いに基づく認証
  - 状態遷移を活用します。
- 従来のログイン認証を突破されることを前提とした対策
  - 従来のログイン認証から先を制限します。
  - とりあえずログインシェルを与えてみて、期待通りの振る舞いをするかどうかを観察します。
    - 労働契約における試用期間をイメージしてください。

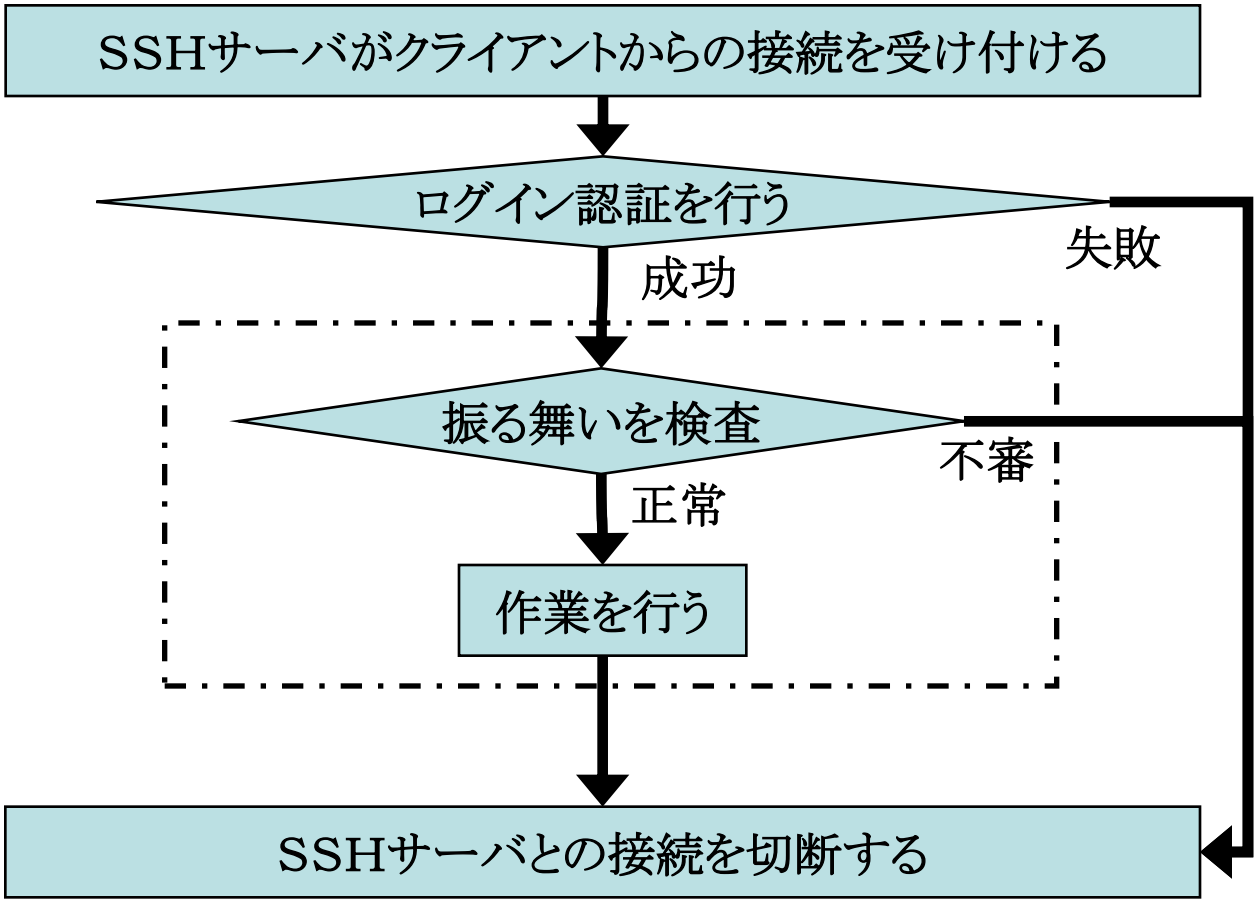
従来の対策のフロー

・ ログイン認証までをカスタマイズする



提案する対策のフロー

・ ログイン認証から先をカスタマイズする

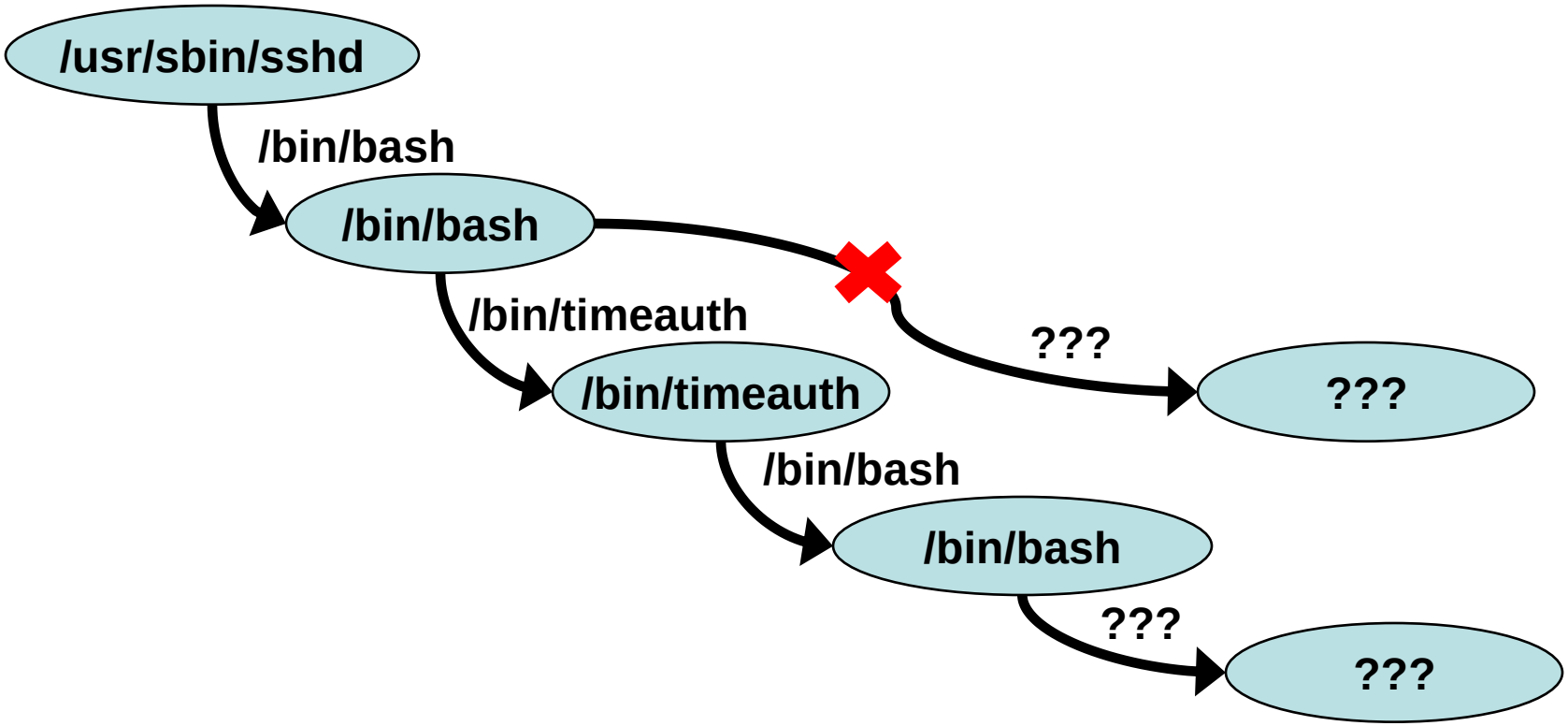


扉をあけて

- 常識は捨ててください。
  - 「なんでもあり」の世界です。
  - あなた自身のアイデアを実装してください。
- 新しい世界の始まりです。
  - セキュアな世界へようこそ！

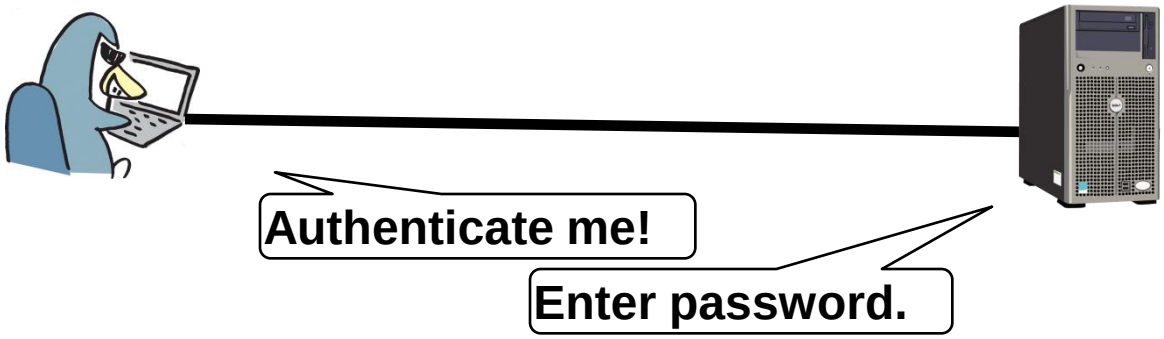
ケース1：対話型シェルセッション

- 打鍵タイミングを利用します。
  - 利用するもの
    - 自作プログラム /bin/timeauth

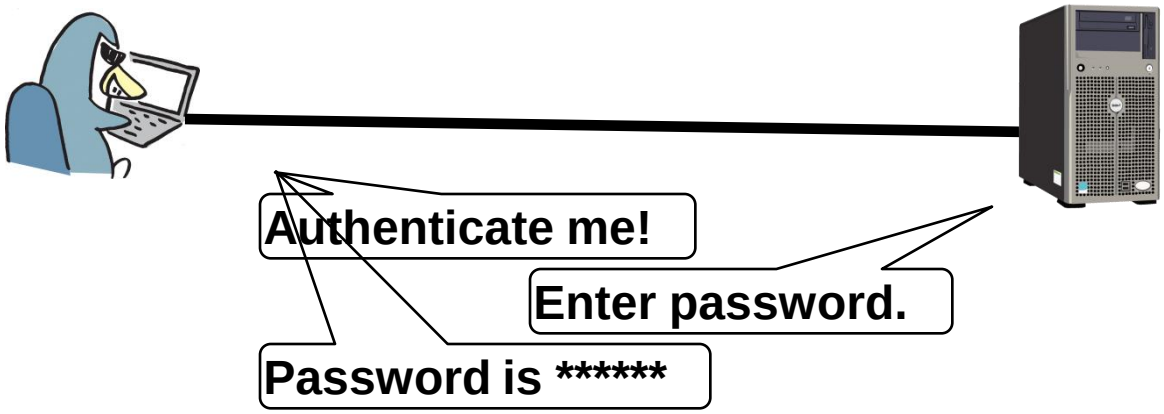


ケース1：対話型シェルセッション

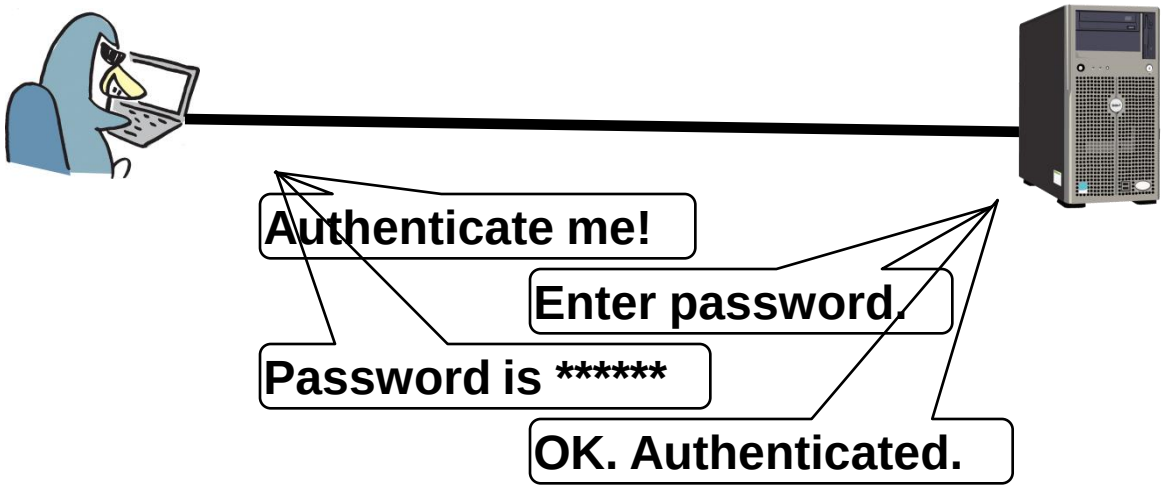
ケース1：対話型シェルセッション



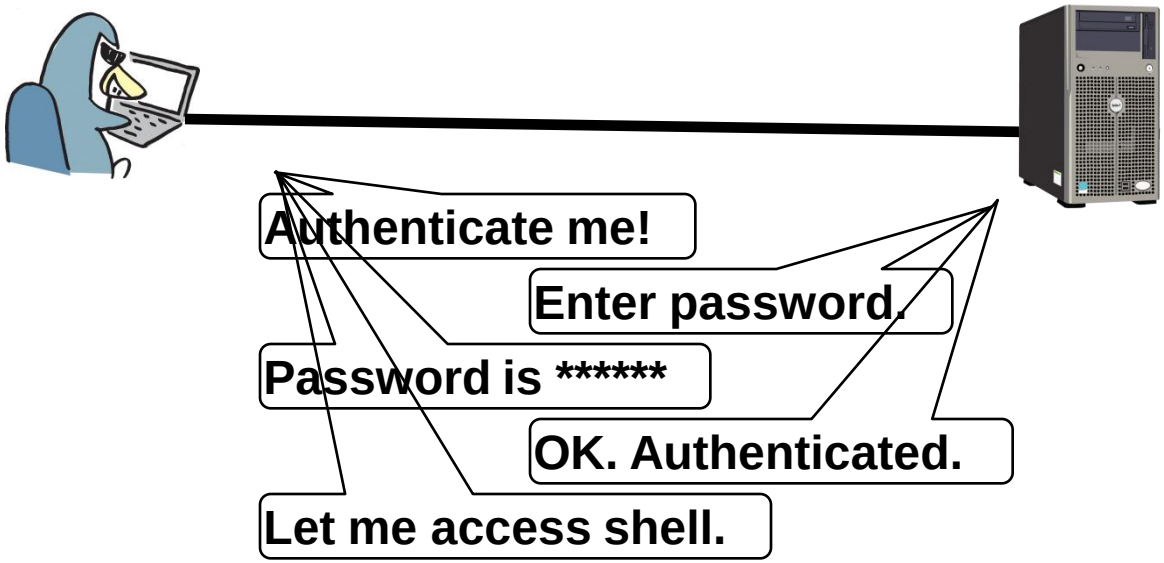
ケース1：対話型シェルセッション



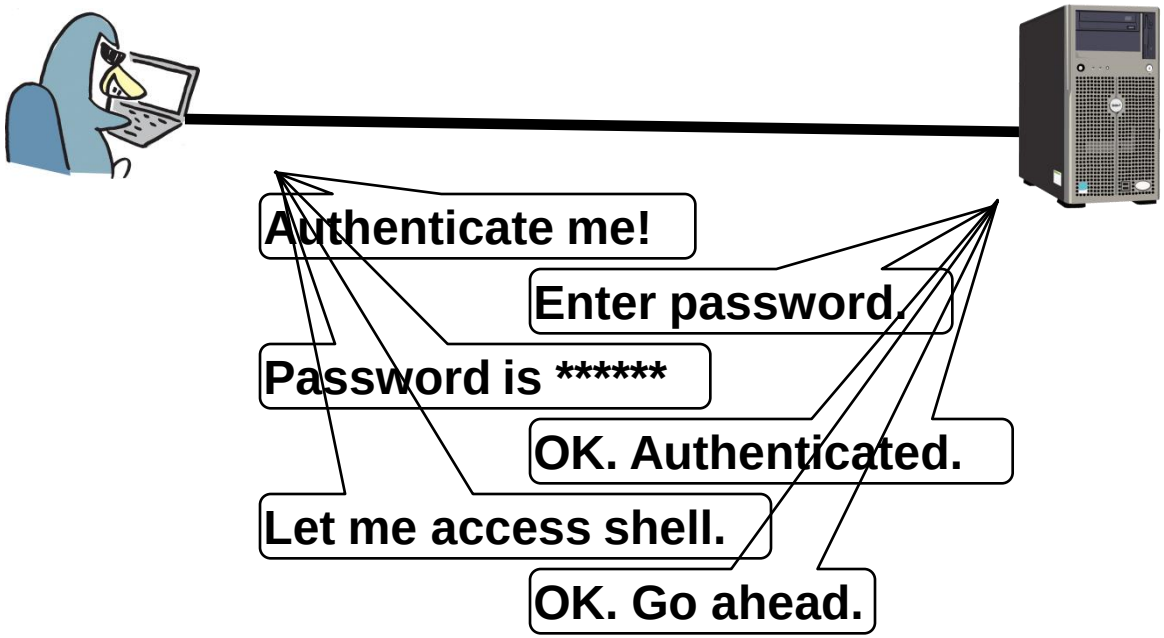
ケース1：対話型シェルセッション



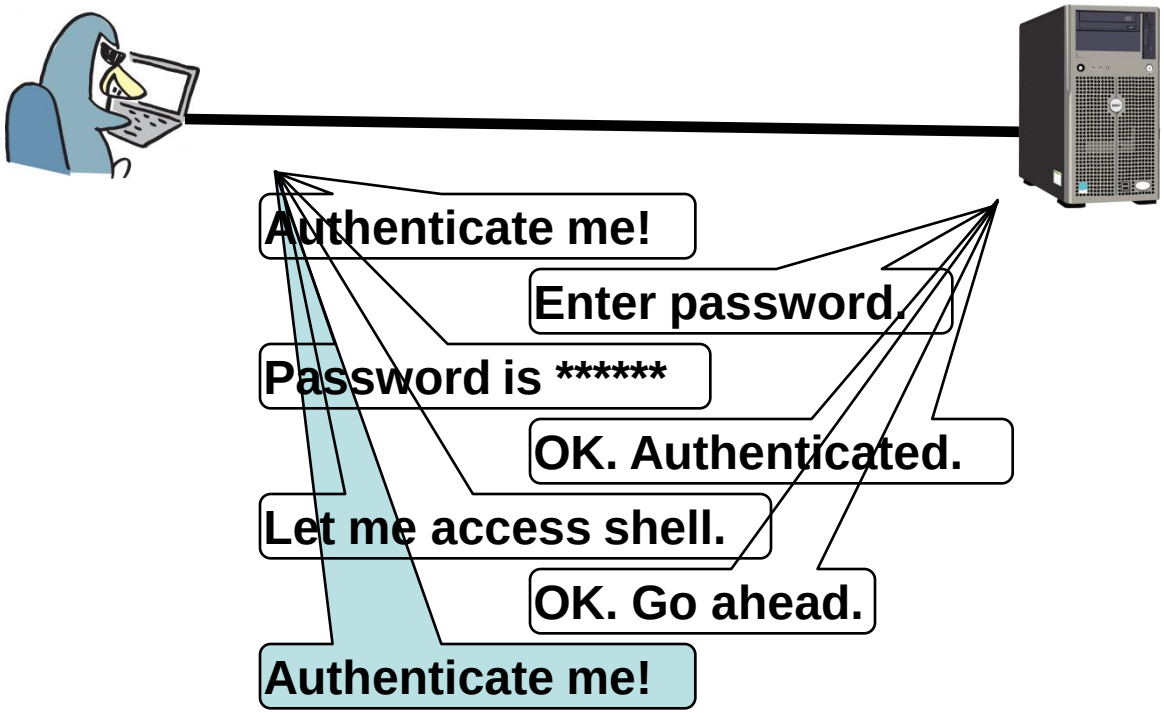
ケース1：対話型シェルセッション



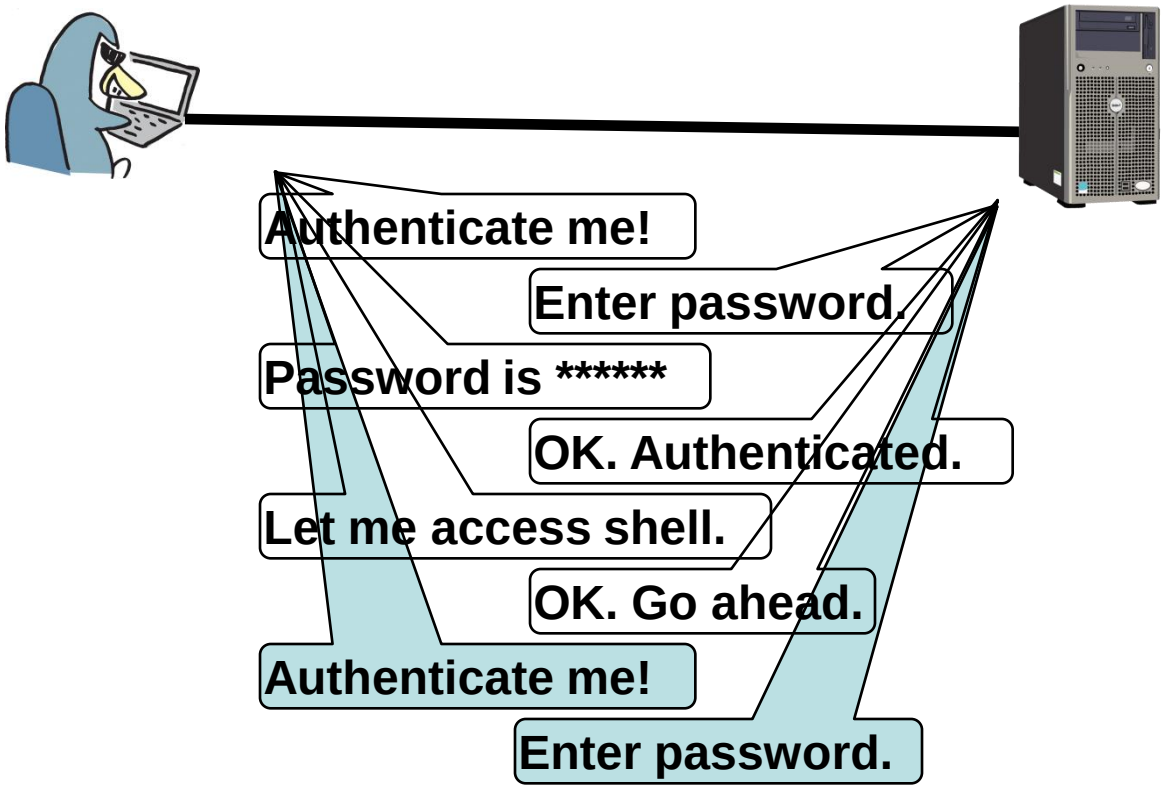
ケース1：対話型シェルセッション



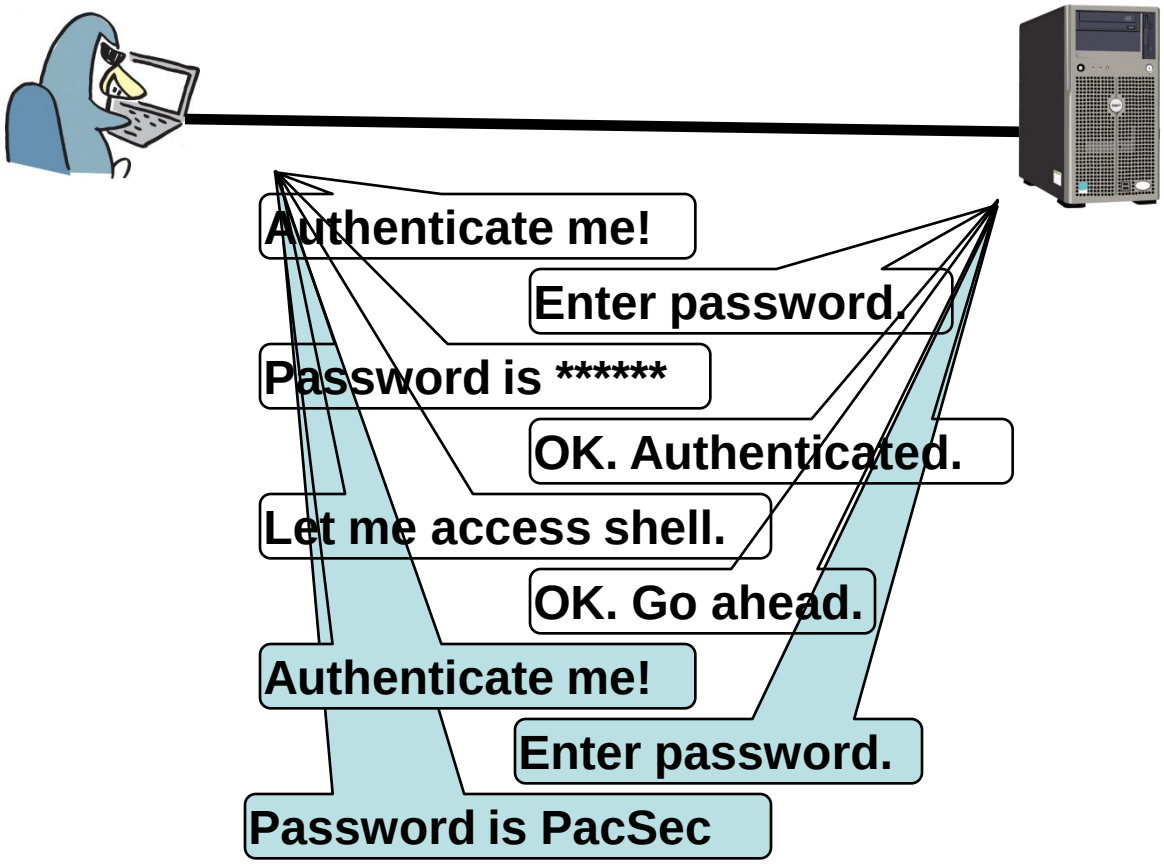
ケース1：対話型シェルセッション



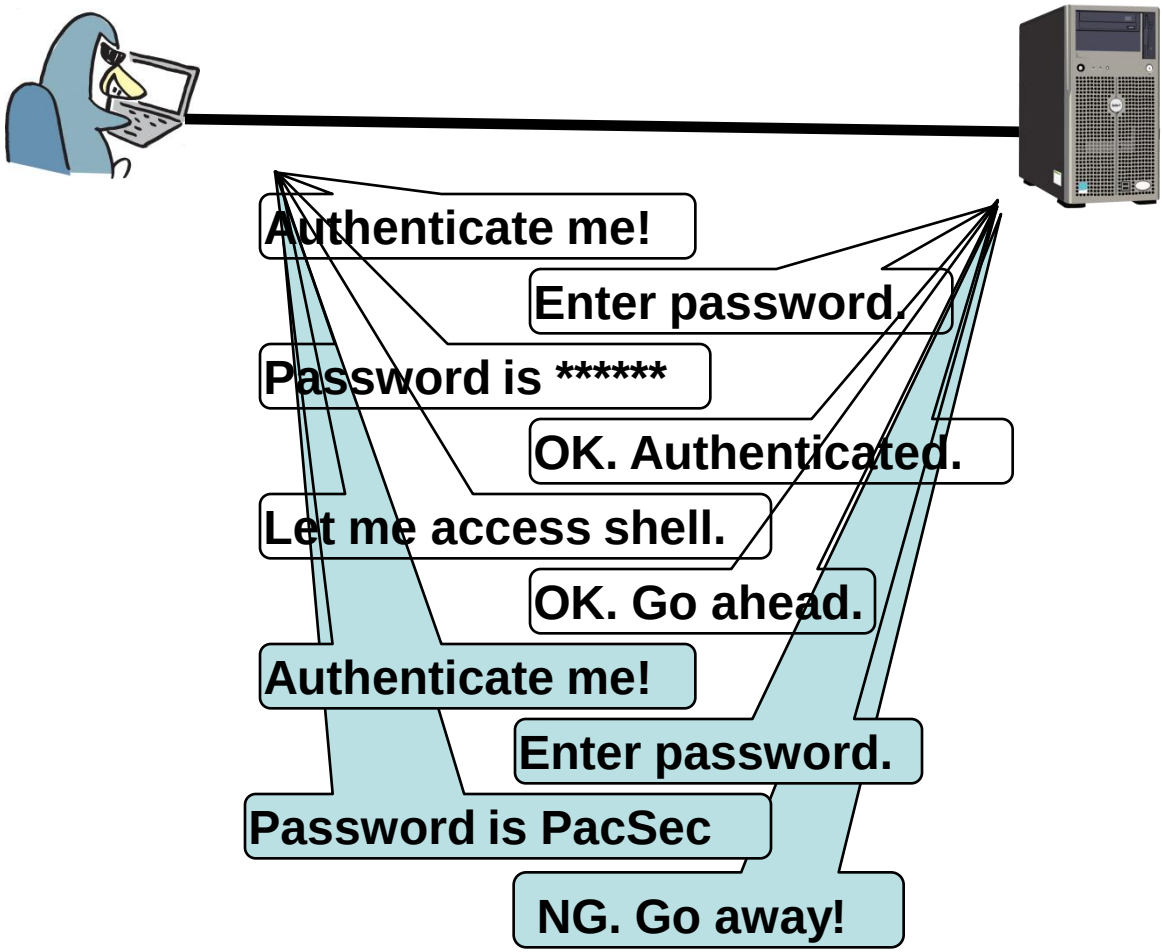
ケース1：対話型シェルセッション



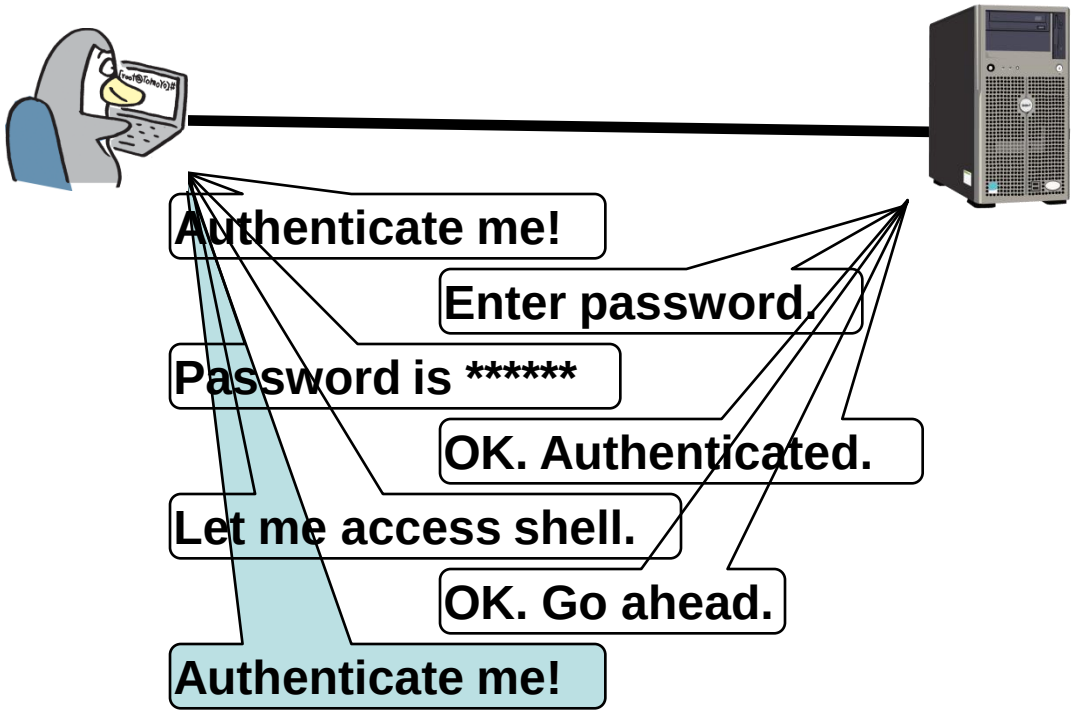
ケース1：対話型シェルセッション



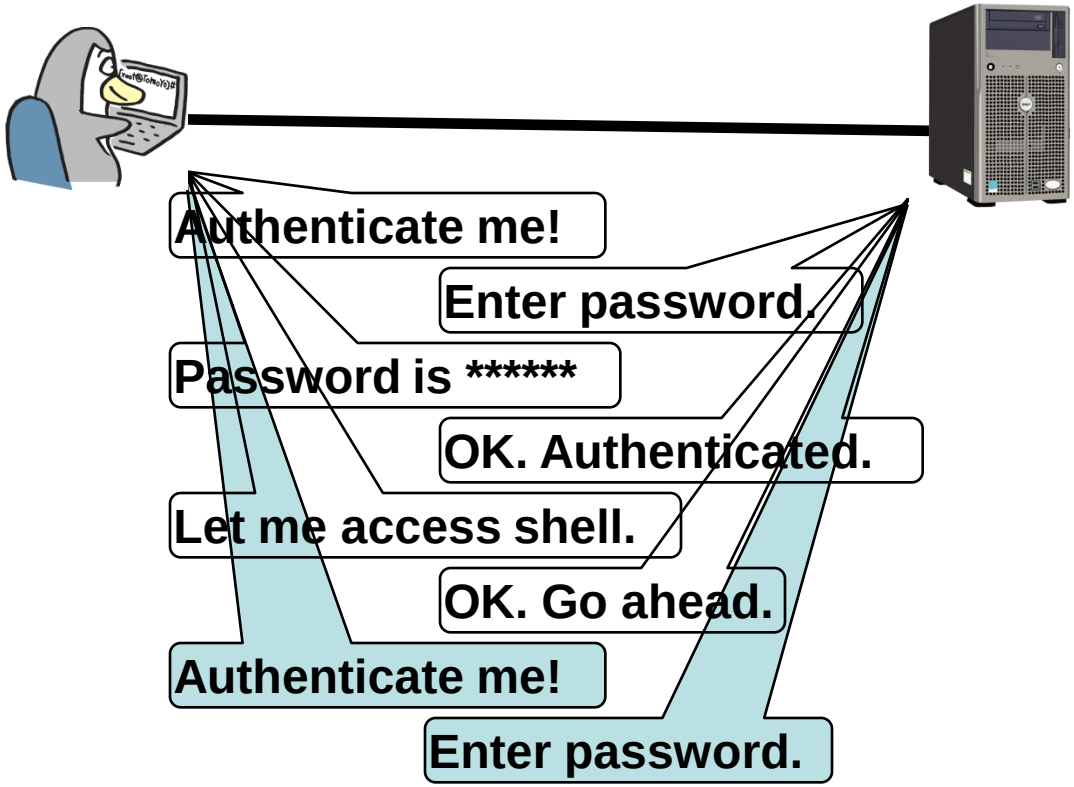
ケース1：対話型シェルセッション



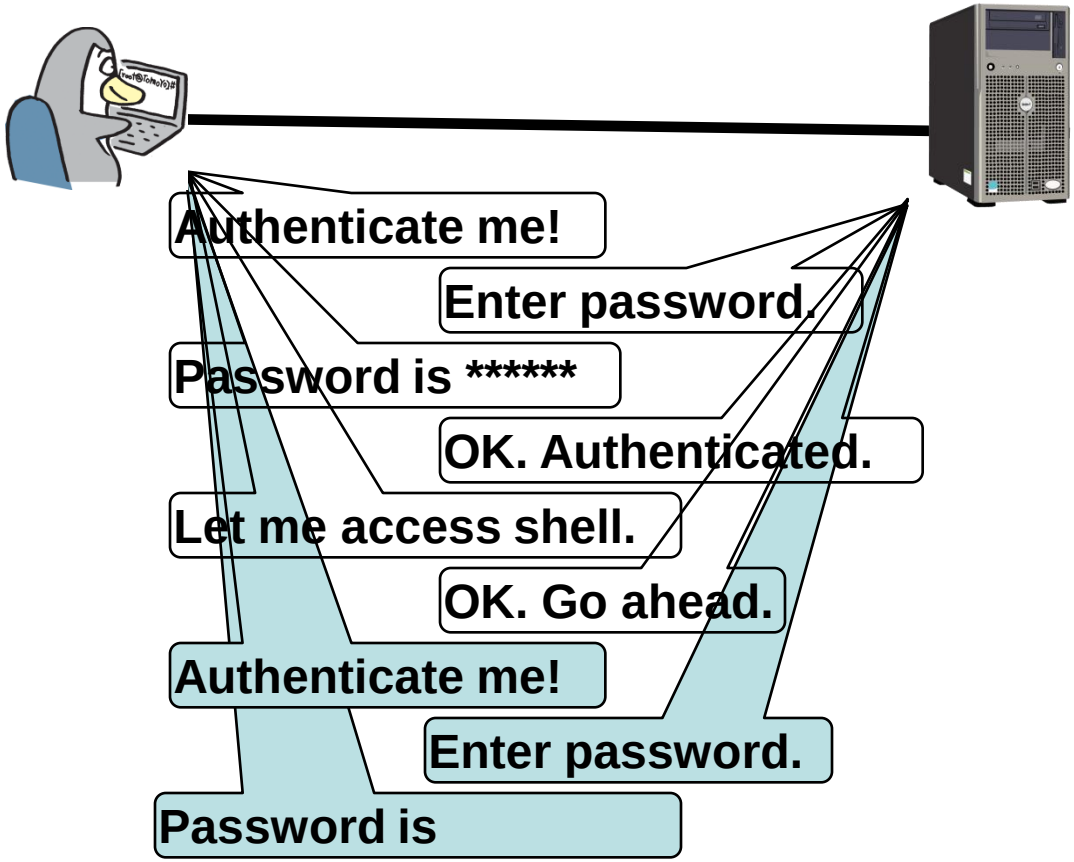
ケース1：対話型シェルセッション



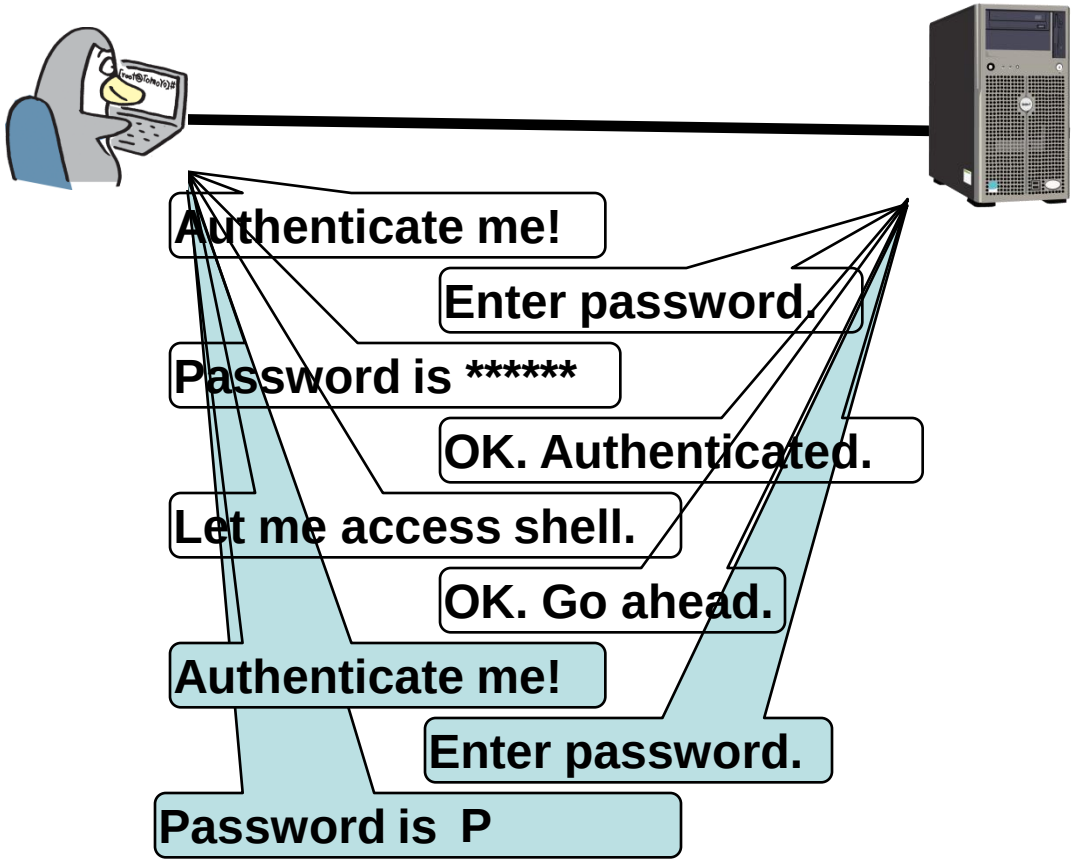
ケース1：対話型シェルセッション



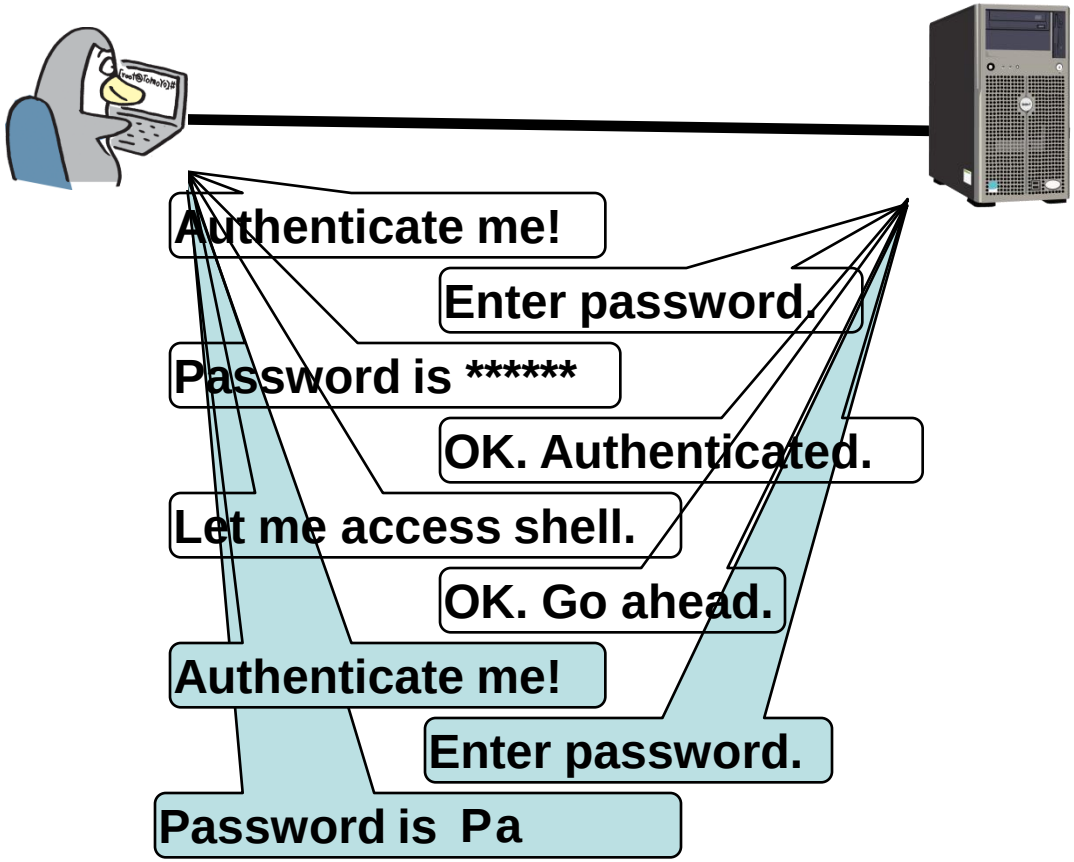
ケース1：対話型シェルセッション



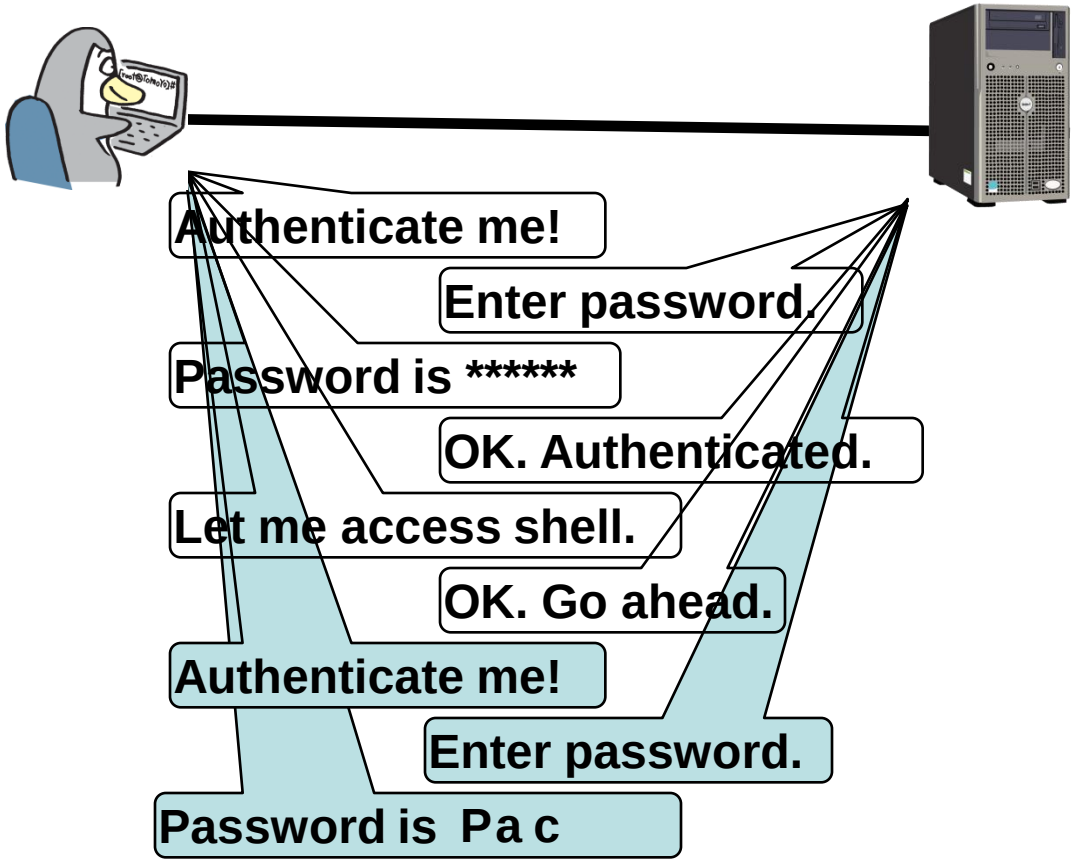
ケース1：対話型シェルセッション



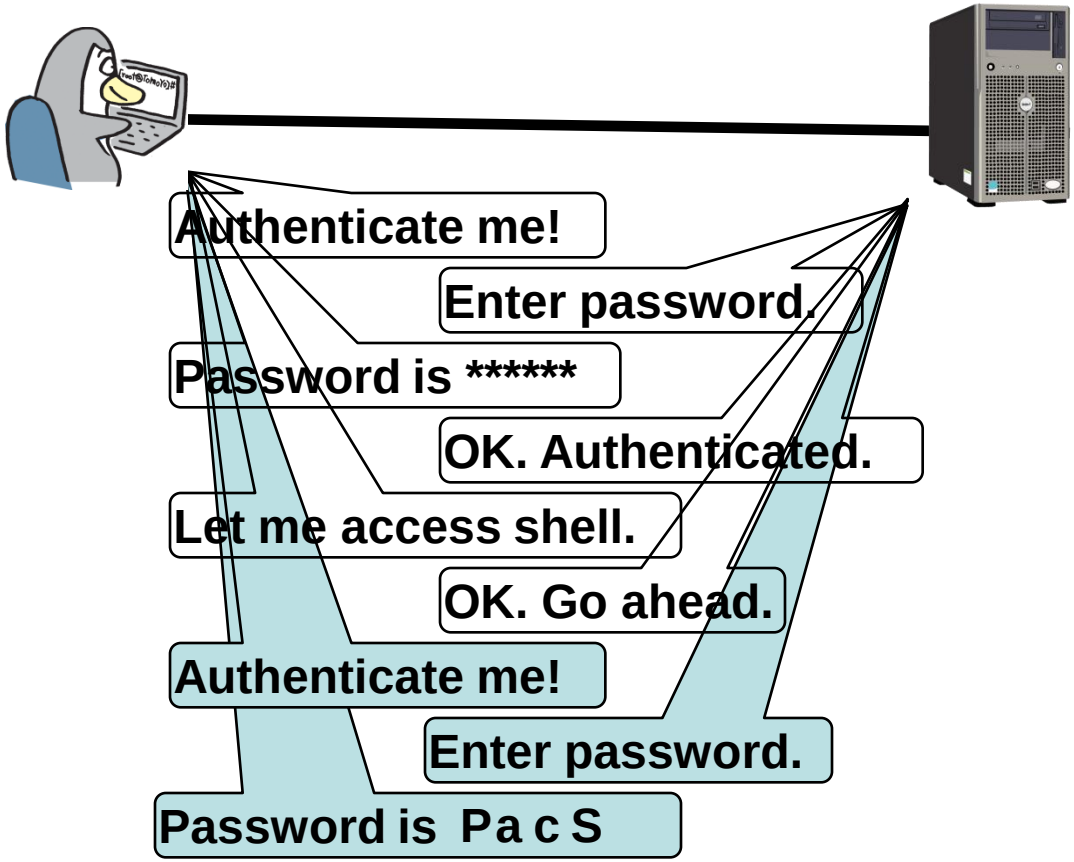
ケース1：対話型シェルセッション



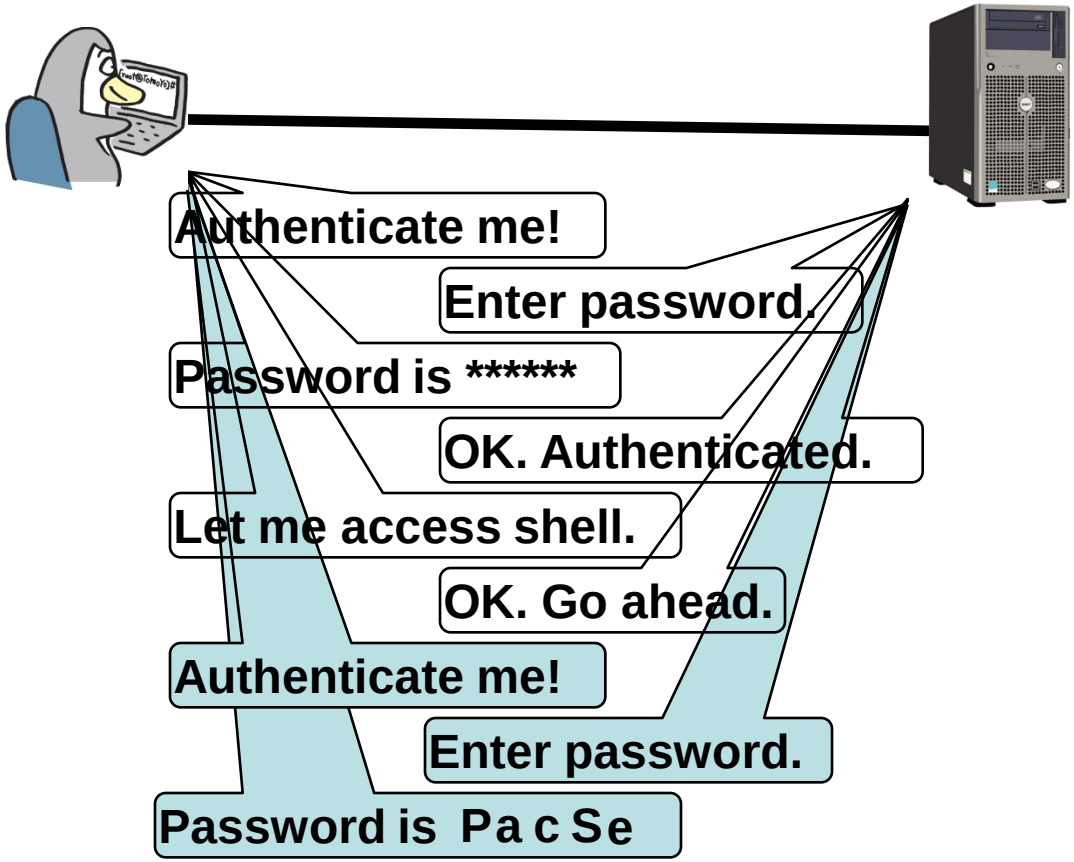
ケース1：対話型シェルセッション



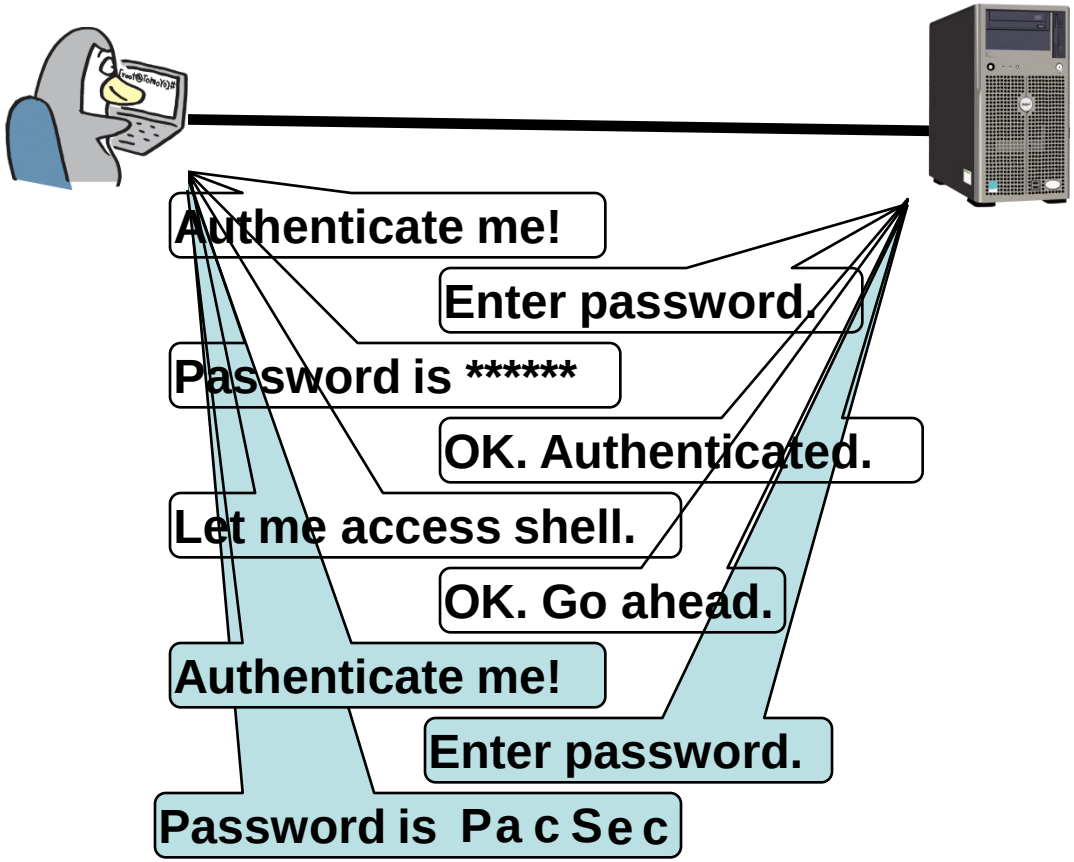
ケース1：対話型シェルセッション



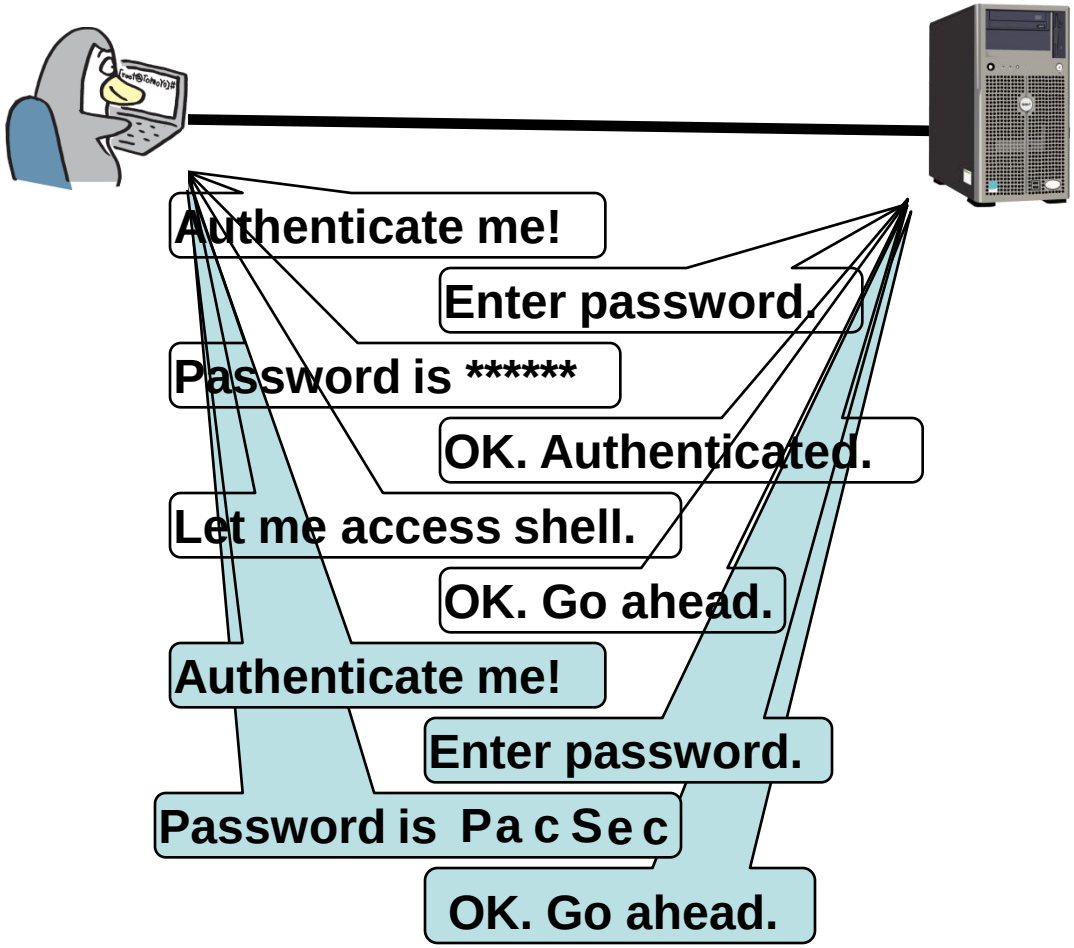
ケース1：対話型シェルセッション



ケース1：対話型シェルセッション



ケース1：対話型シェルセッション



ケース1：対話型シェルセッション

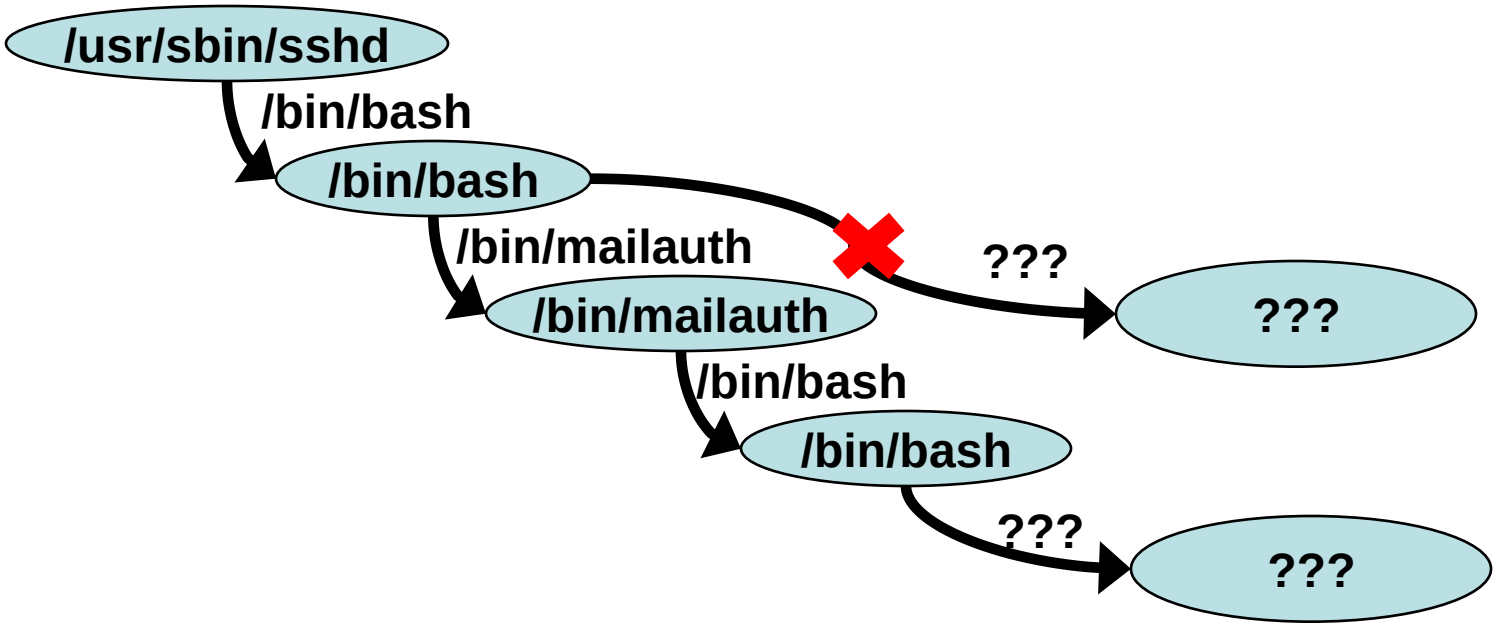
- 利点
  - 使える要素が制限されない
    - RFCなどの標準に従う必要がない
    - どんな要素を使っているかを秘匿できる
  - 想定外の認証方式
    - 侵入者が思いも付かない方法を使える
    - パスワード文字数が少なくても、タイミング情報と組み合わせることでブルートフォースを無意味に

ケース1：対話型シェルセッション

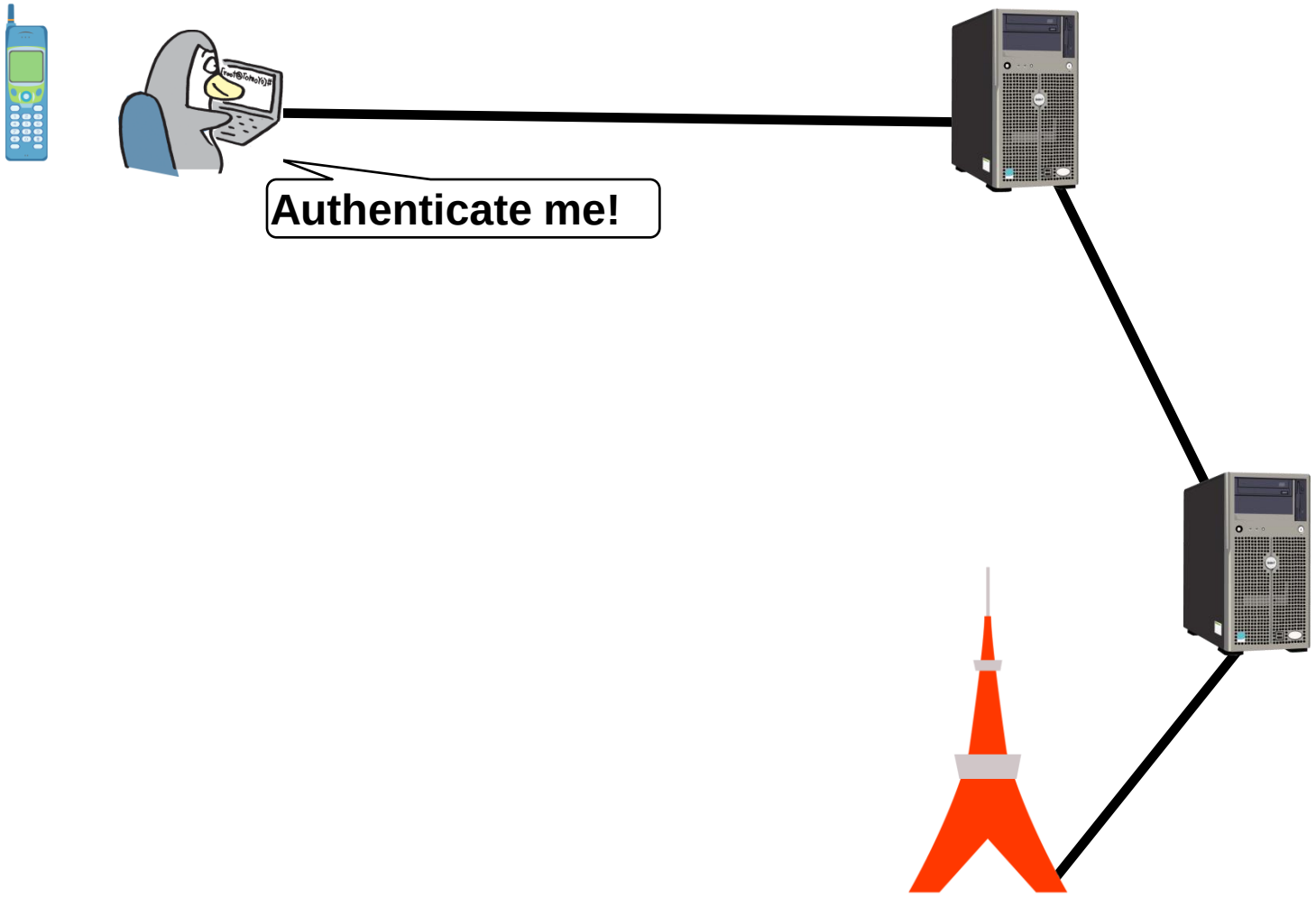
- 難点
  - 「アクセス制御機能を強化したOS」が必要
    - ログインシェルから実行できるコマンドを制限するため
    - MAC（強制アクセス制御）と呼ばれる機能を利用
  - Round Trip Timeが大きいと使いにくい
    - 外国からのアクセスを防ぐのには好都合？

ケース2：対話型シェルセッション

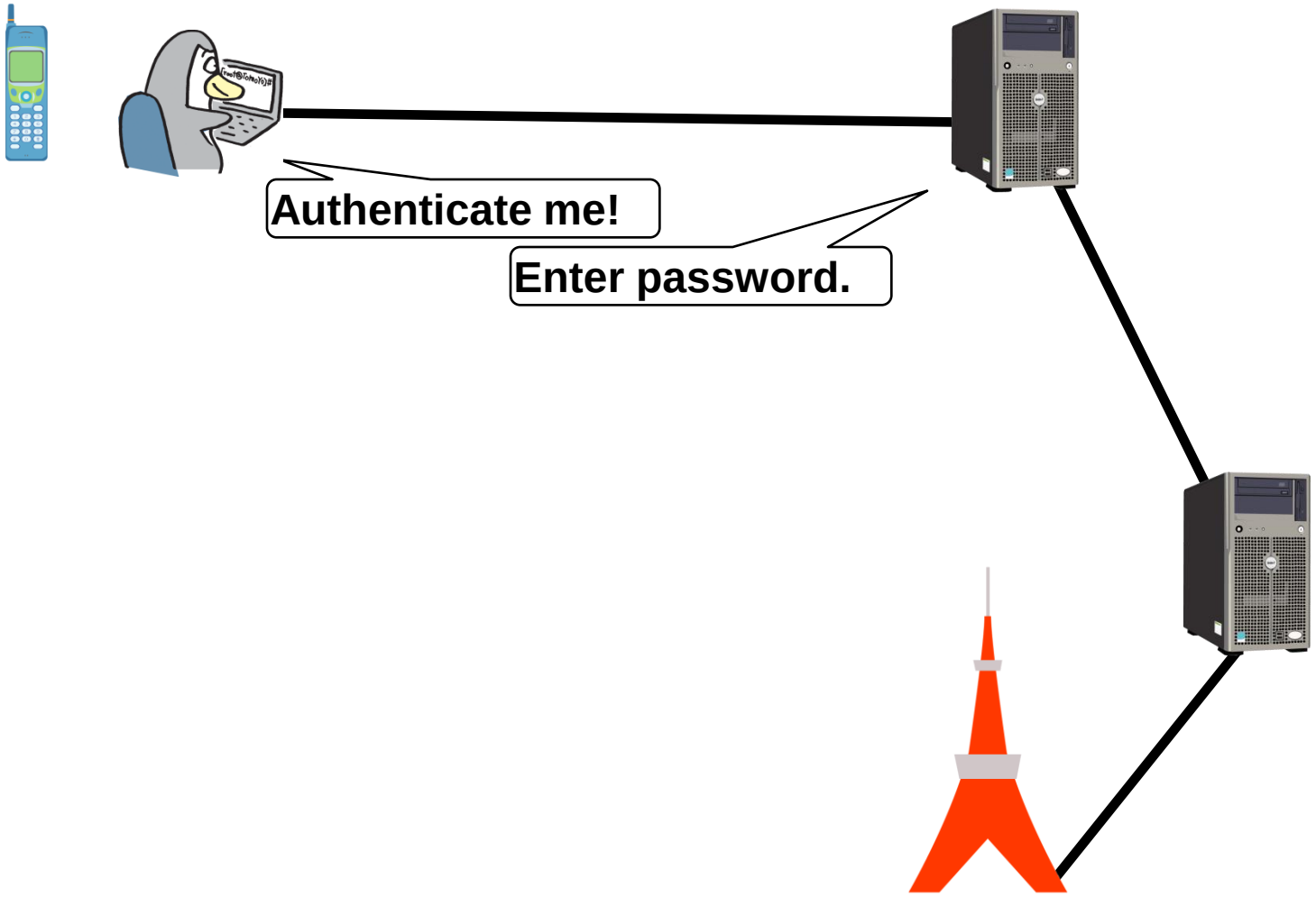
- ワンタイムパスワードとメールを利用します。
  - 利用するもの
    - SMTPサーバ
    - 自作プログラム /bin/mailauth



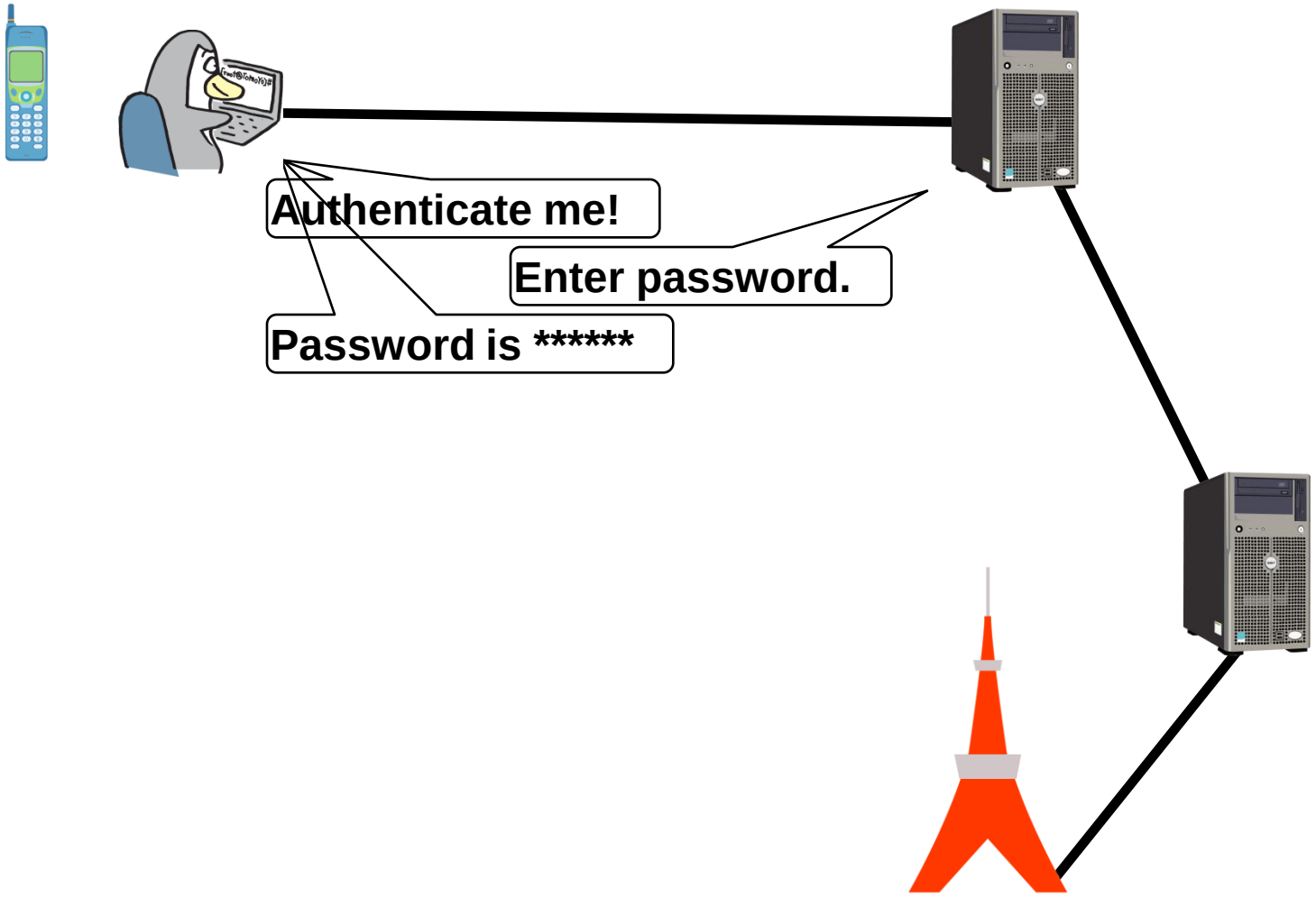
ケース2：対話型シェルセッション



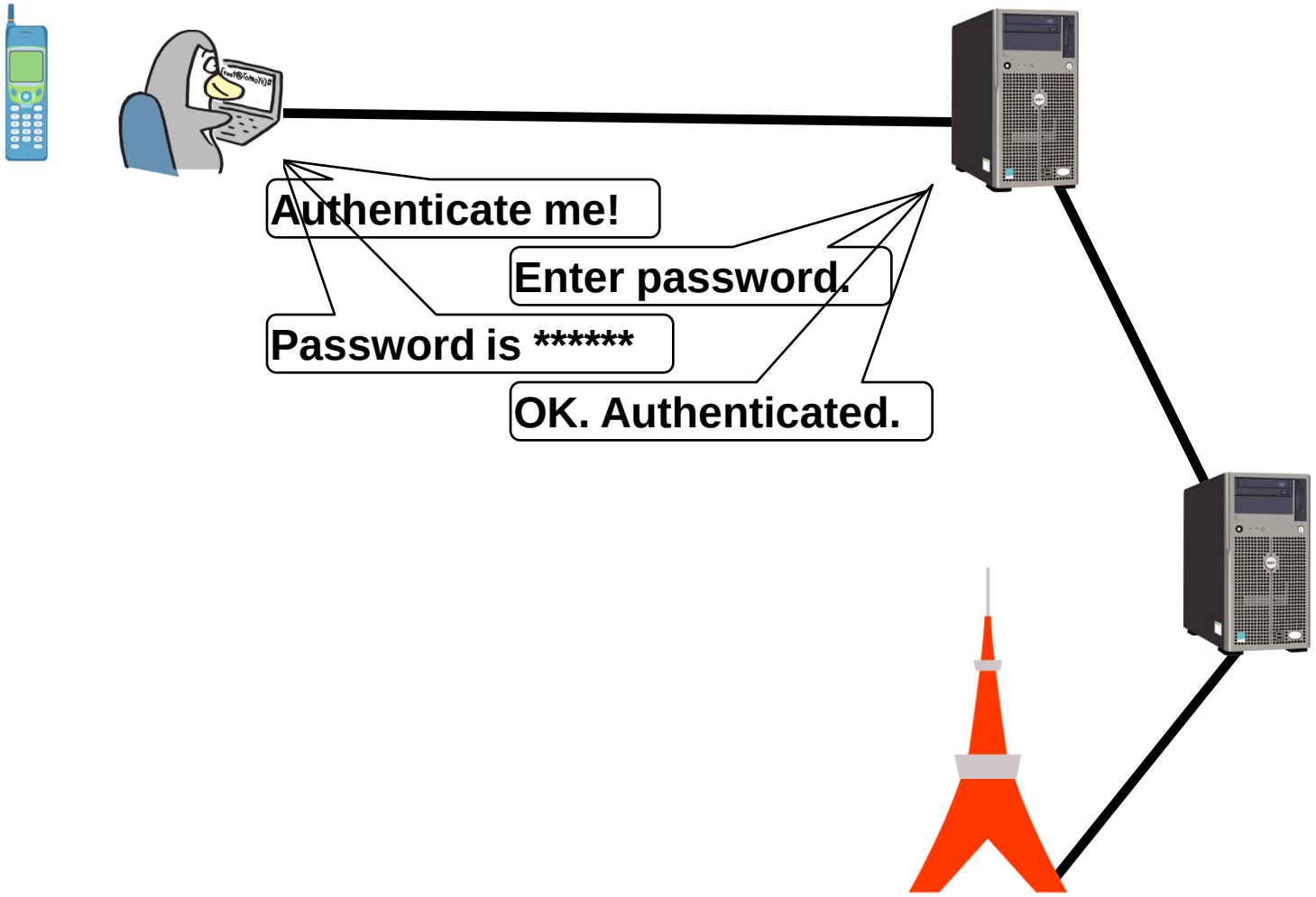
ケース2：対話型シェルセッション



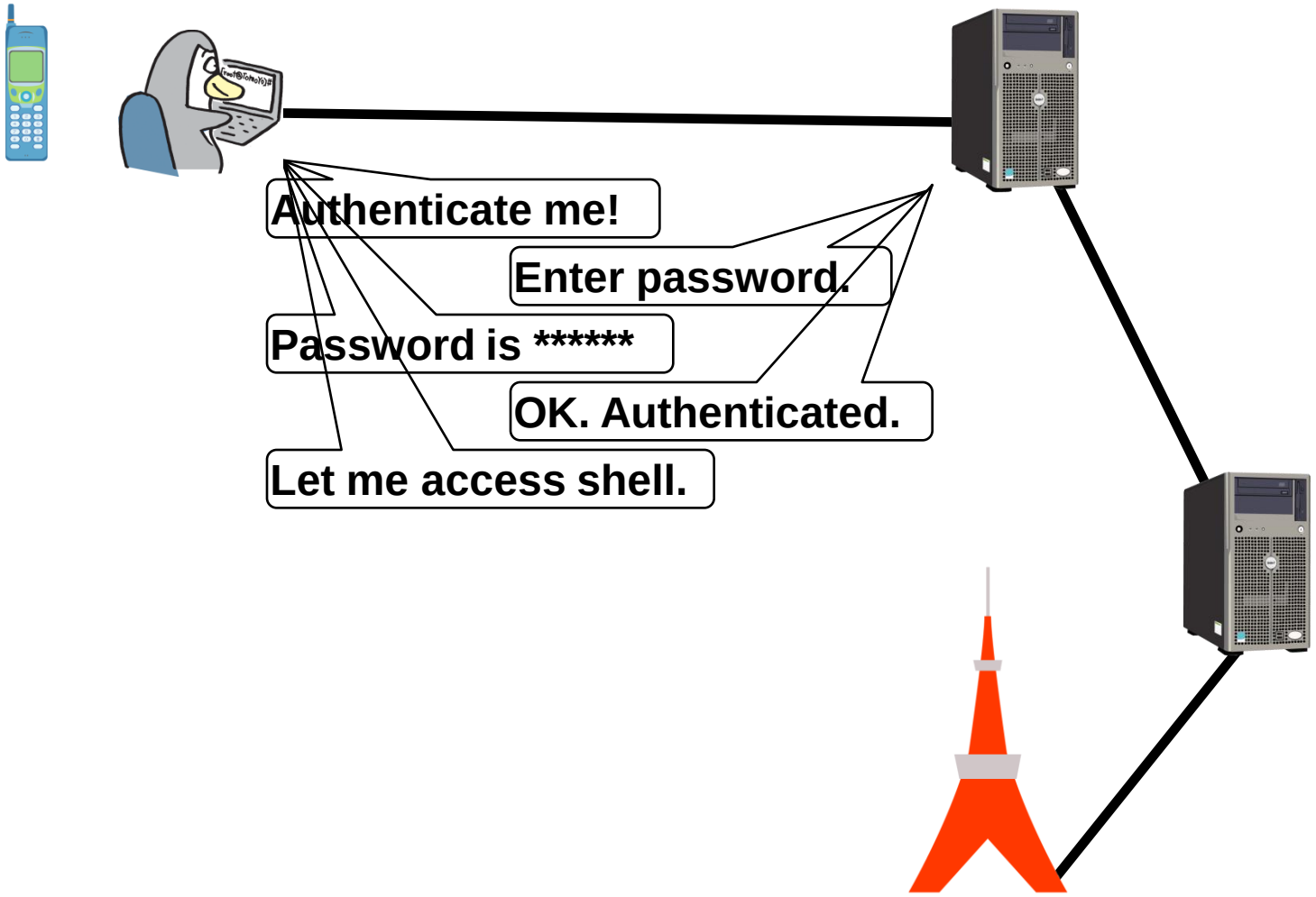
ケース2：対話型シェルセッション



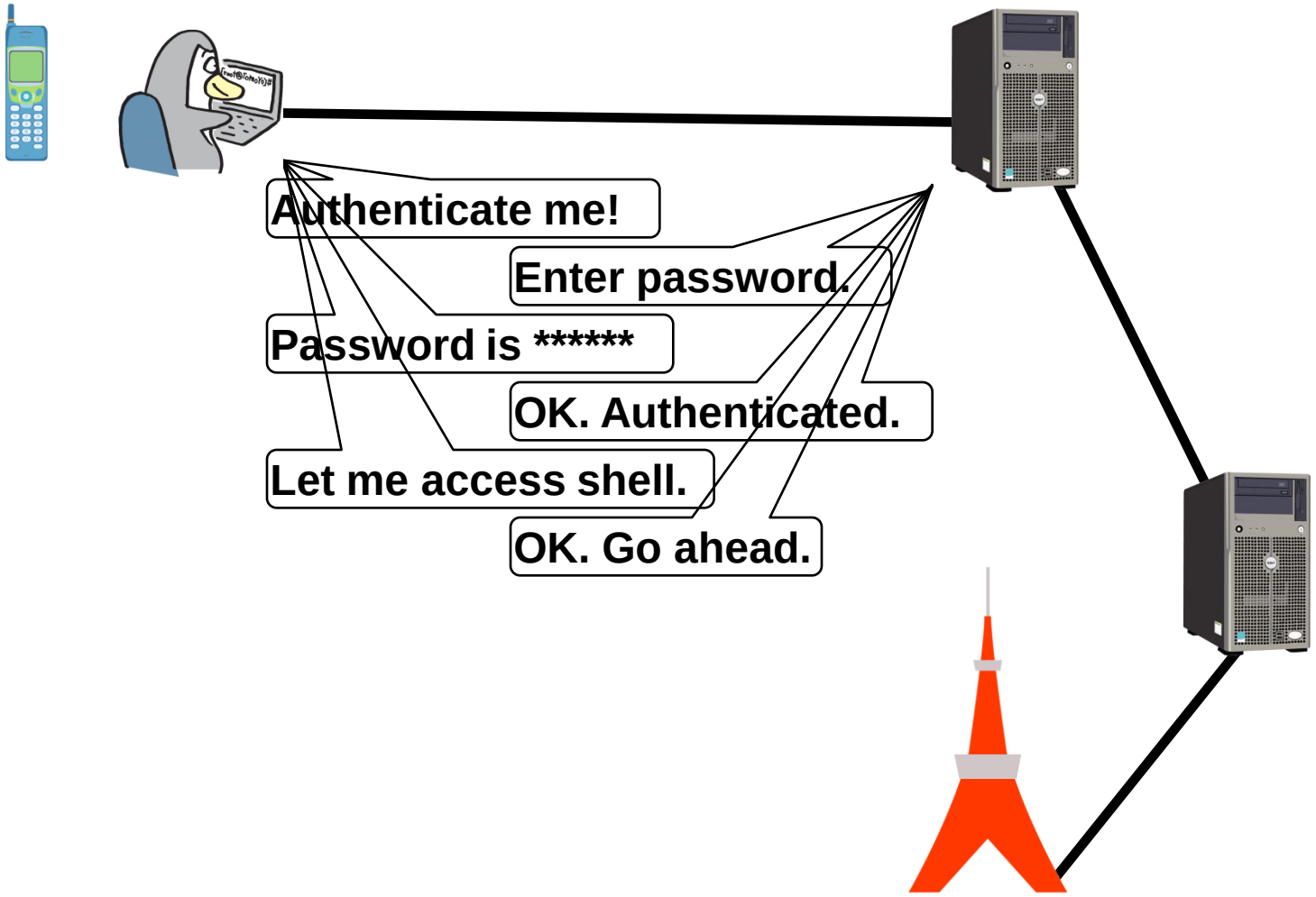
ケース2：対話型シェルセッション



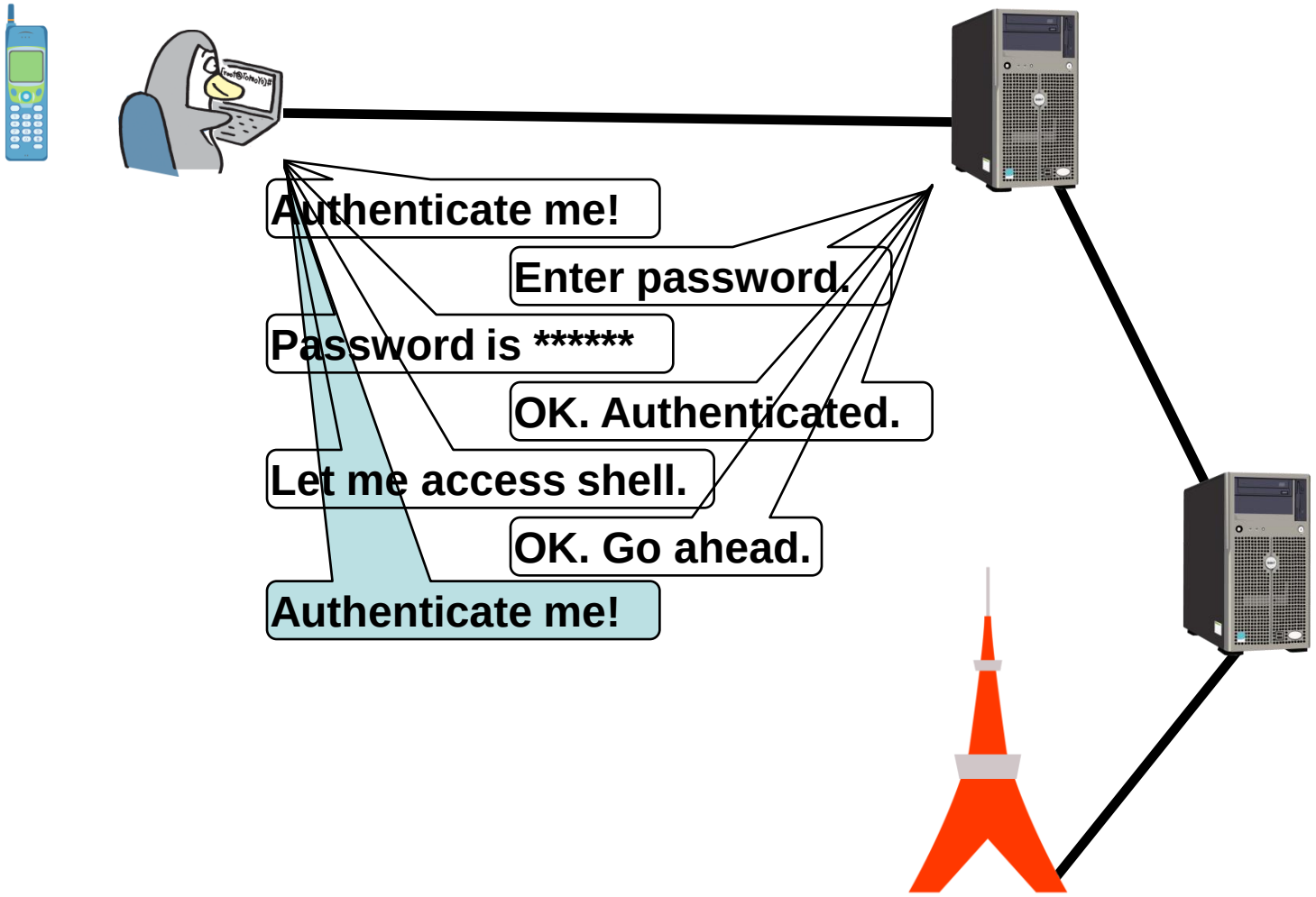
ケース2：対話型シェルセッション



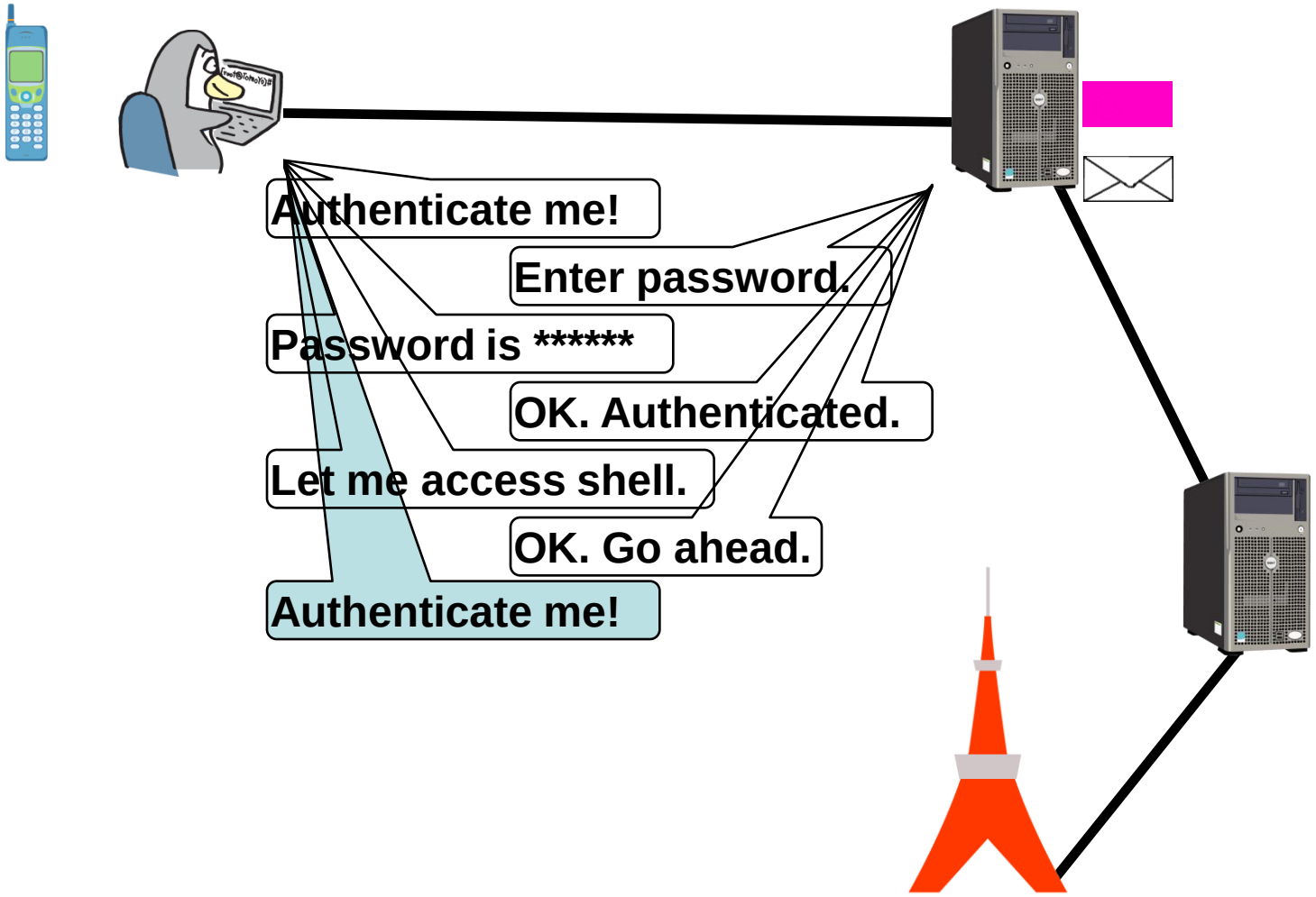
ケース2：対話型シェルセッション



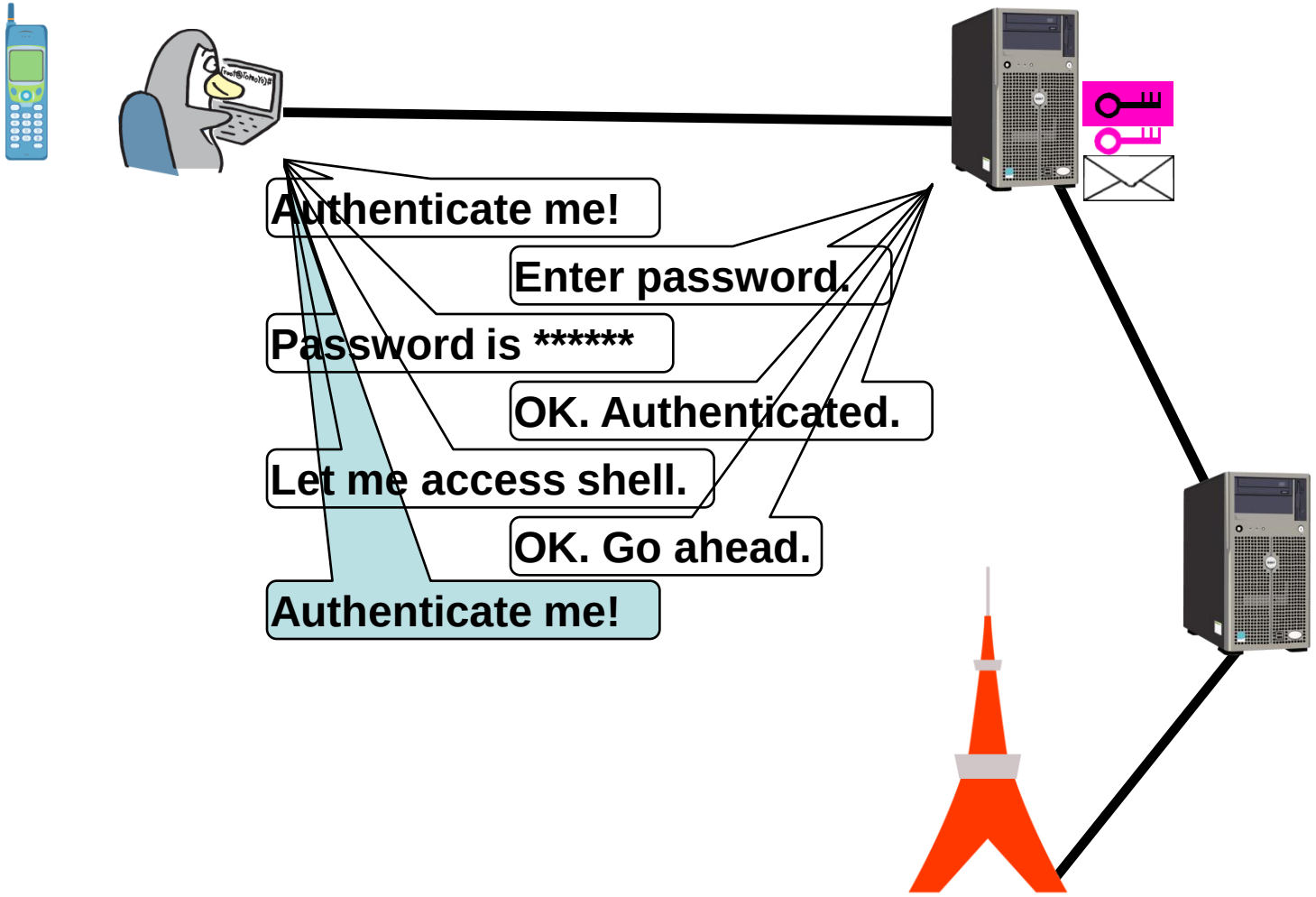
ケース2：対話型シェルセッション



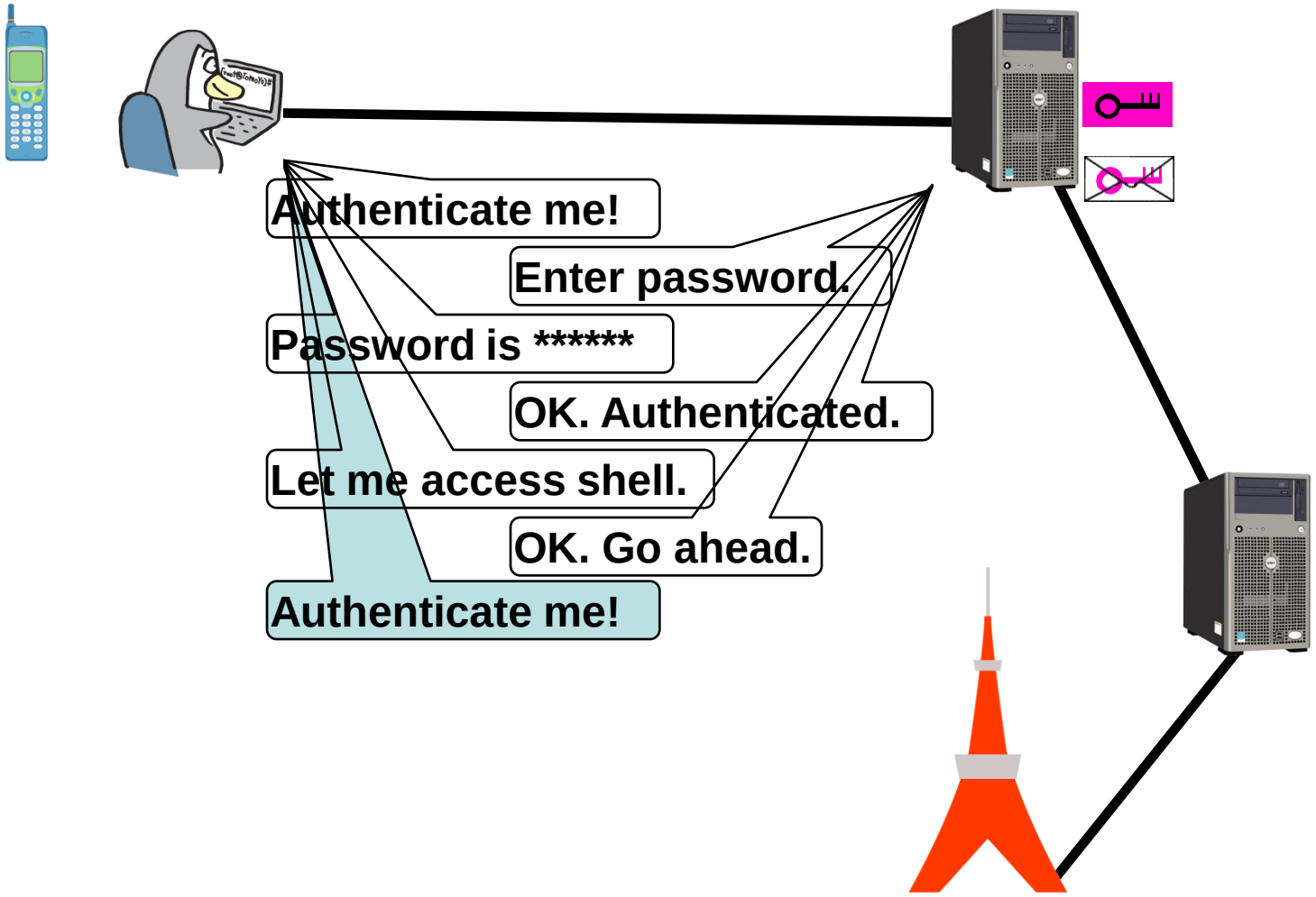
ケース2：対話型シェルセッション



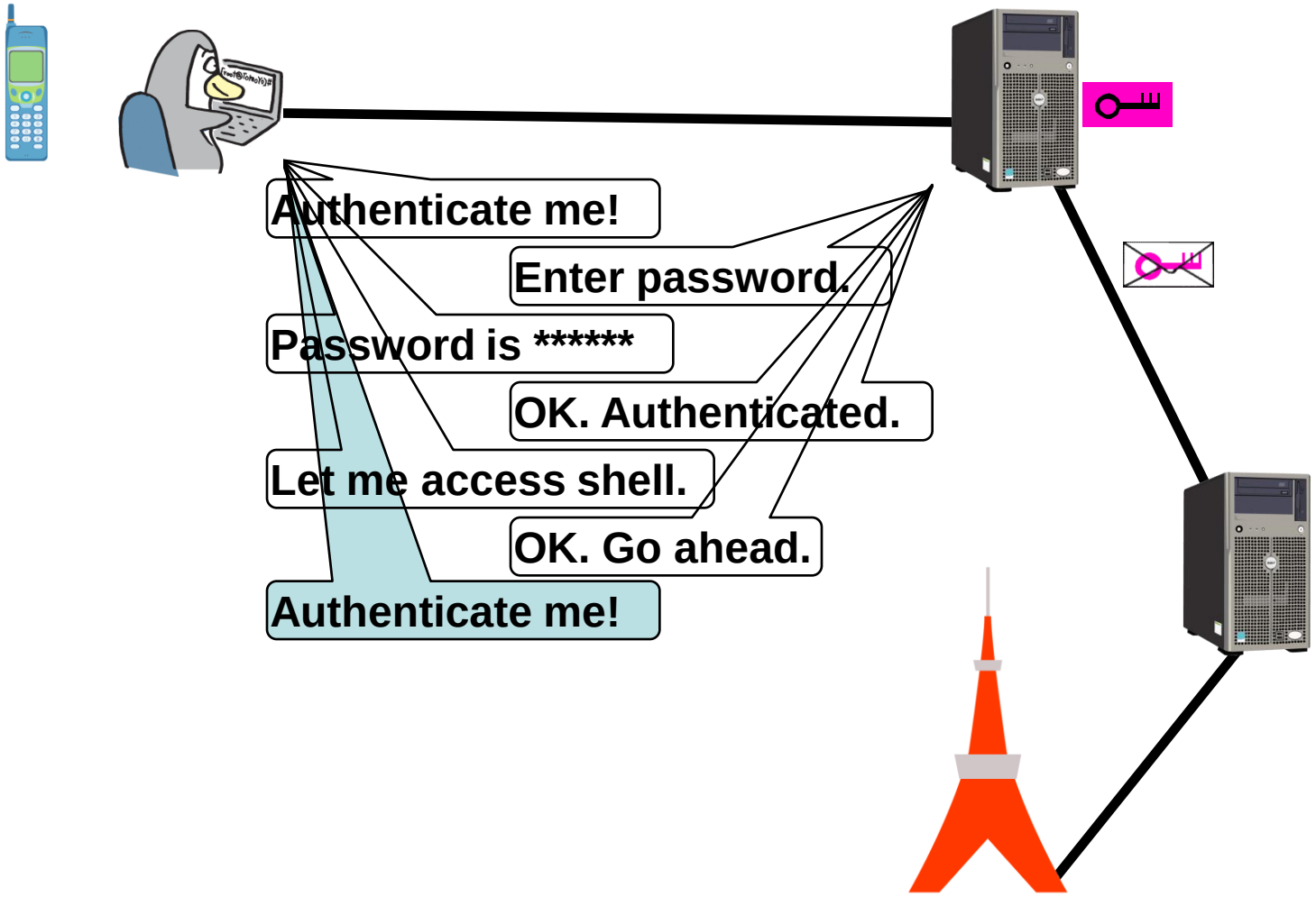
ケース2：対話型シェルセッション



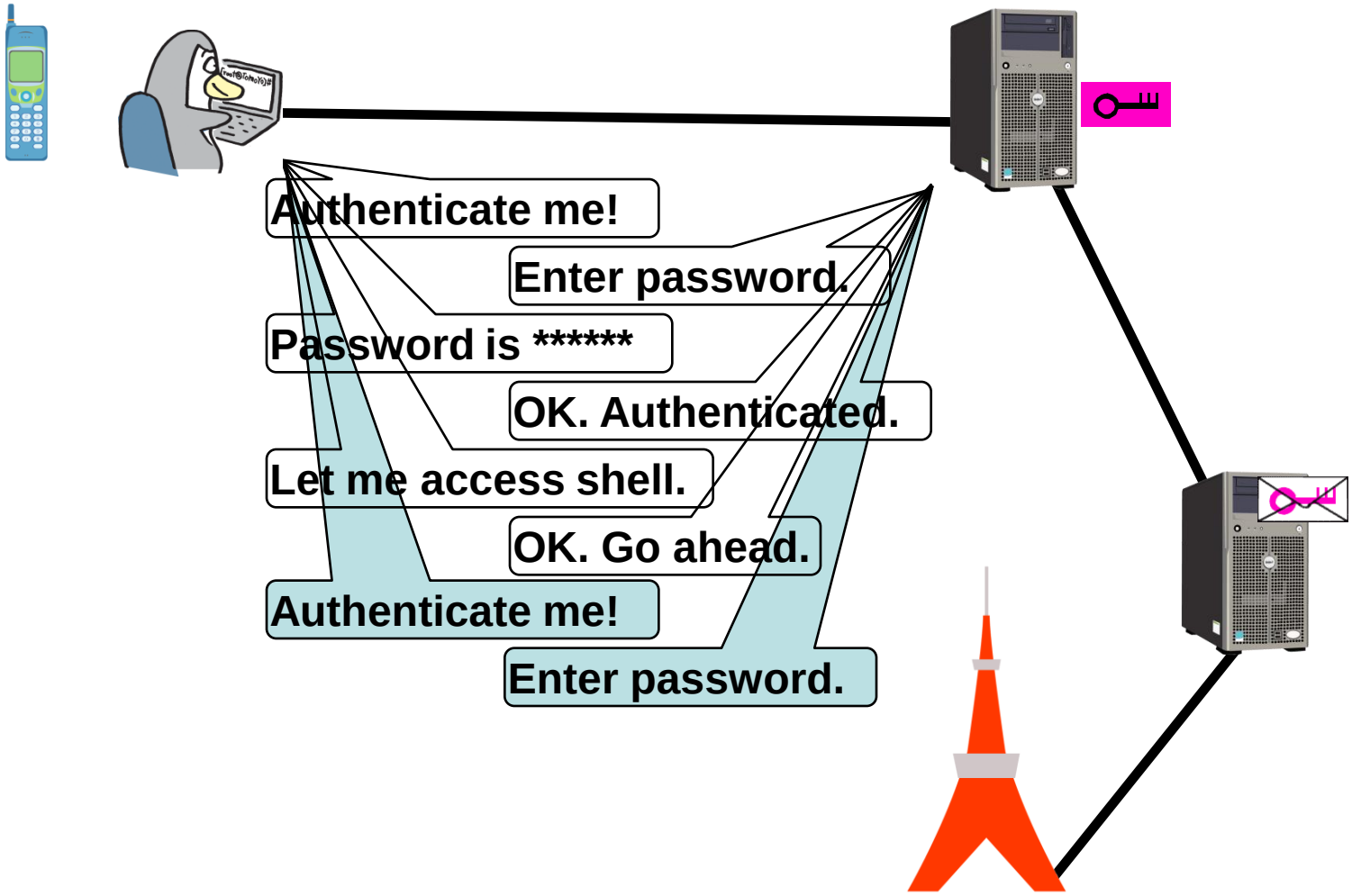
ケース2：対話型シェルセッション



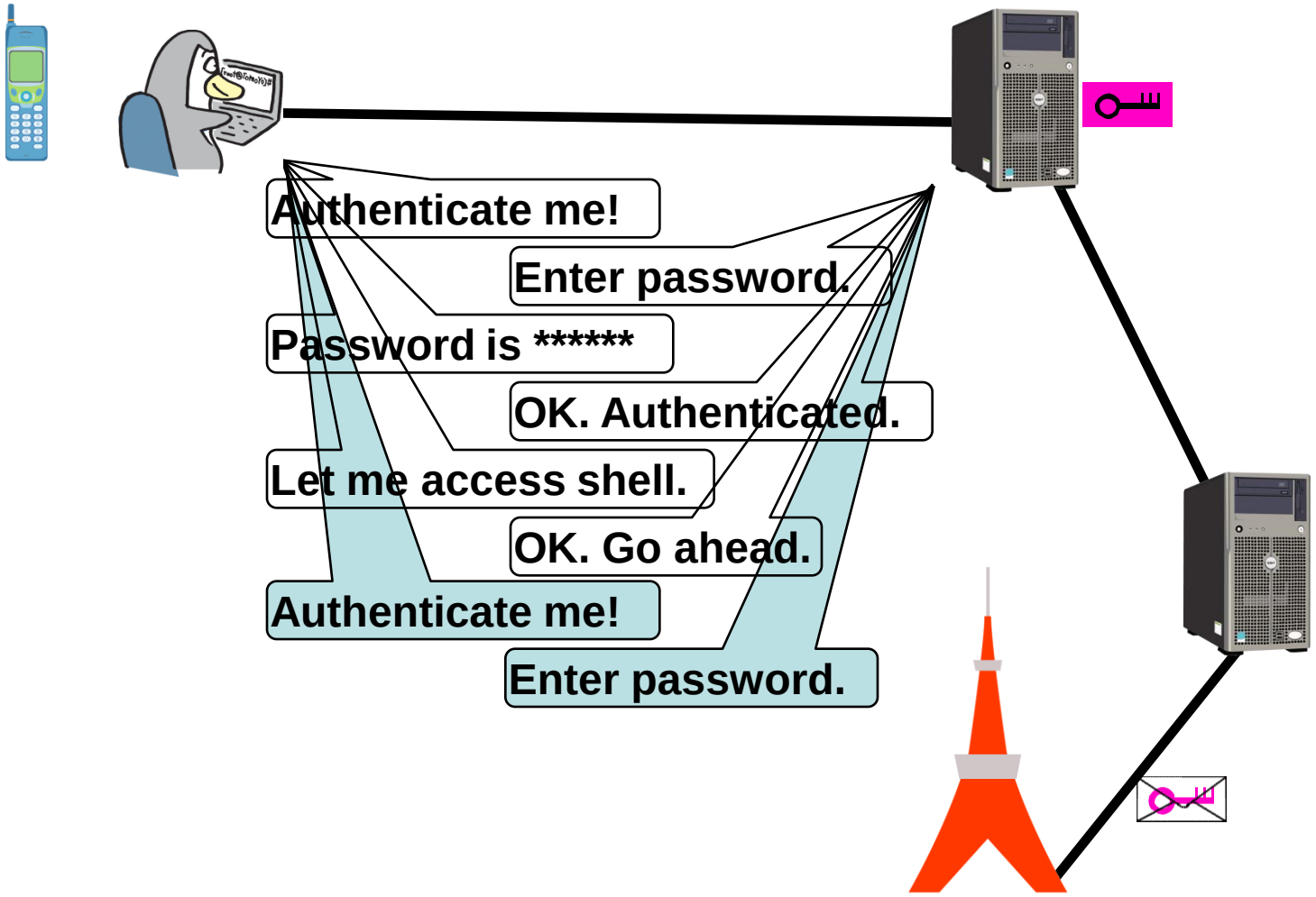
ケース2：対話型シェルセッション



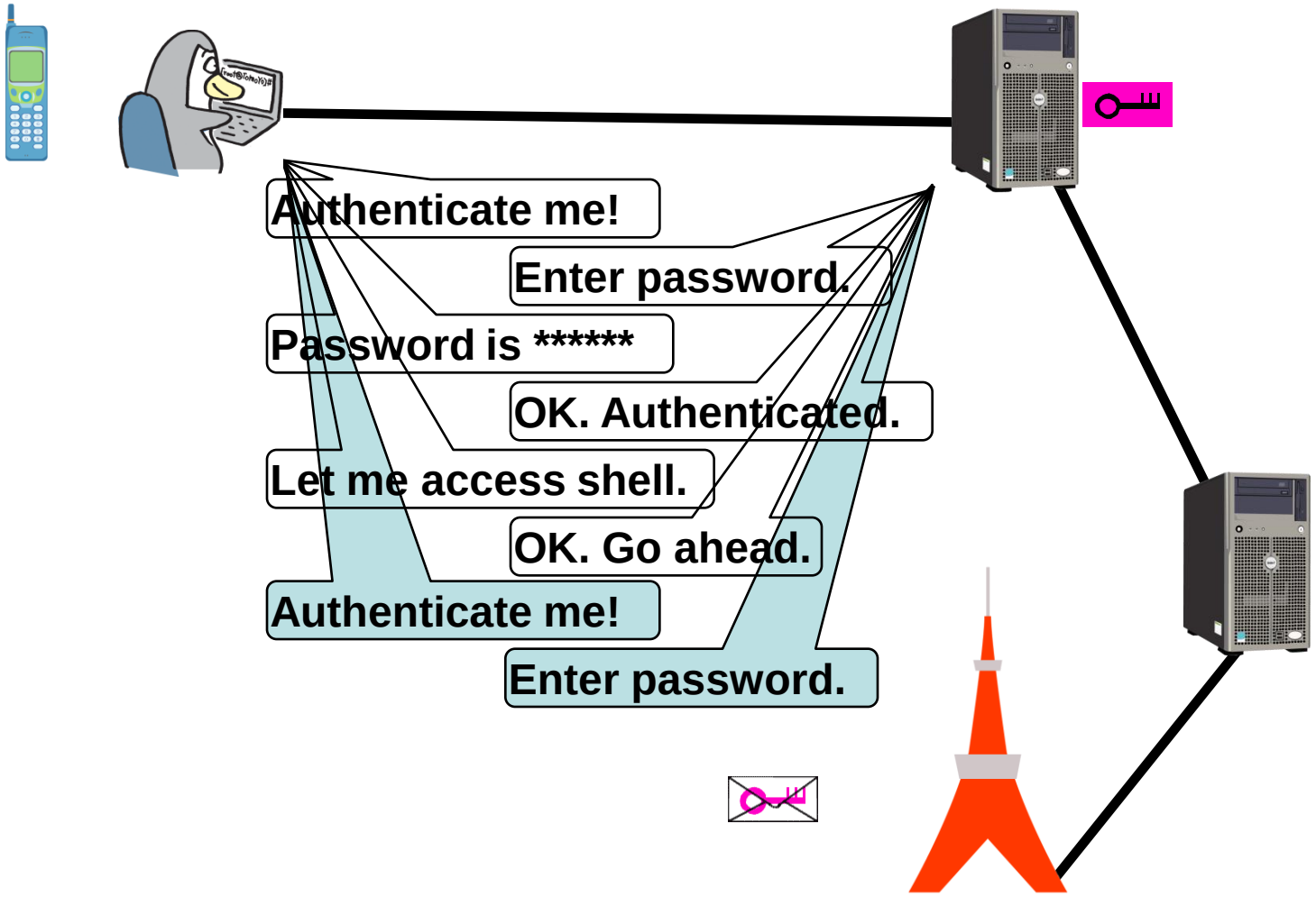
ケース2：対話型シェルセッション



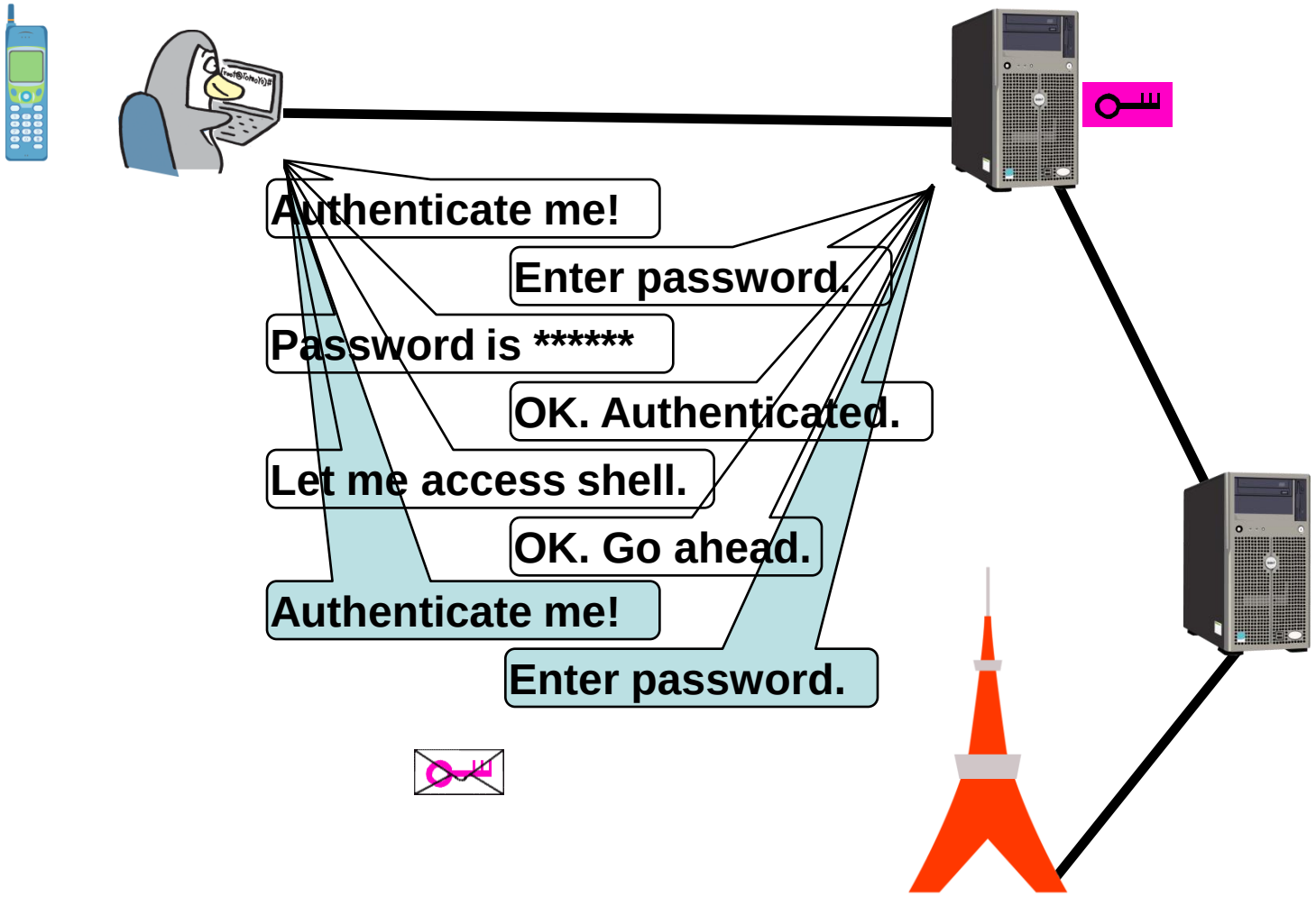
ケース2：対話型シェルセッション



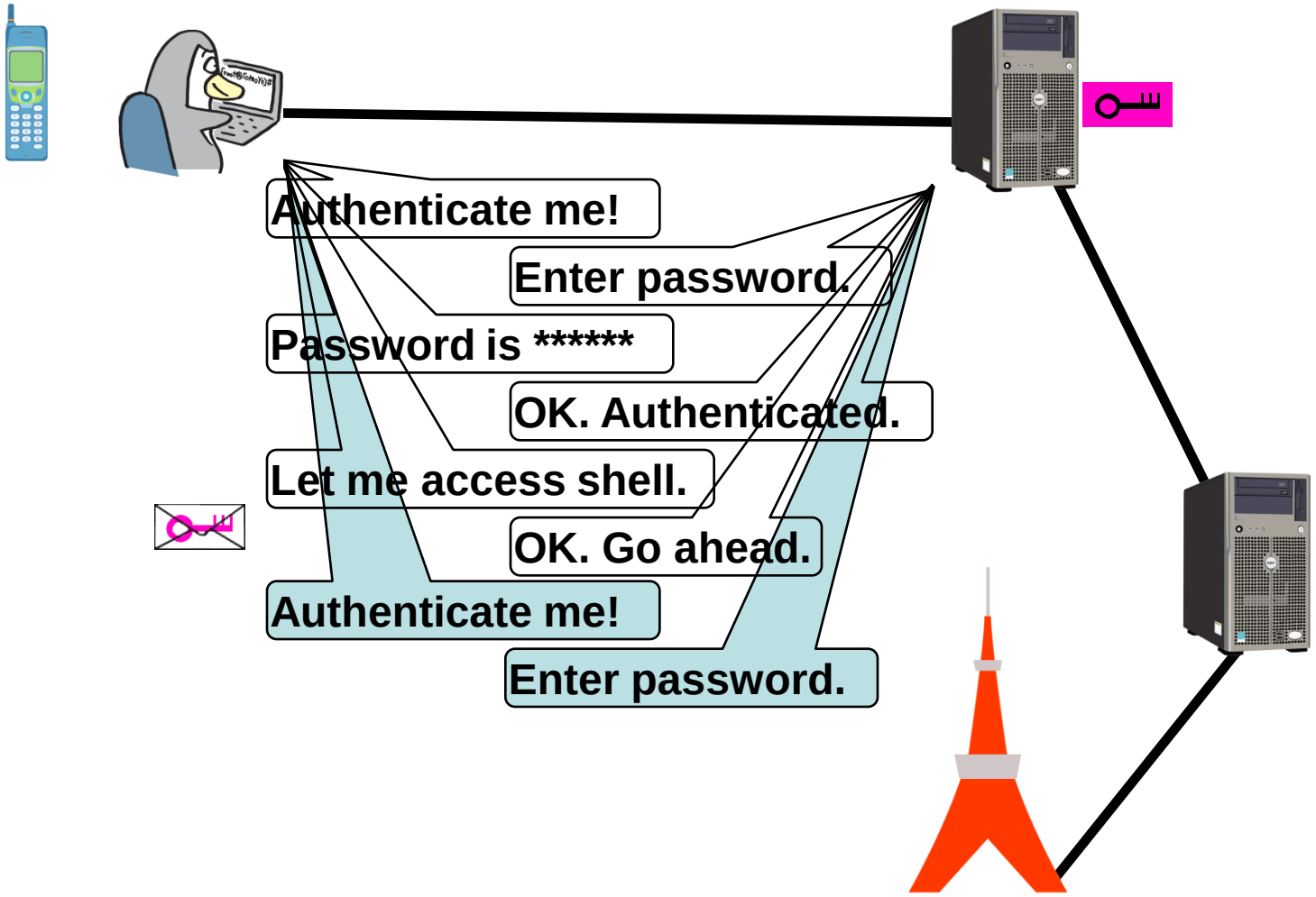
ケース2：対話型シェルセッション



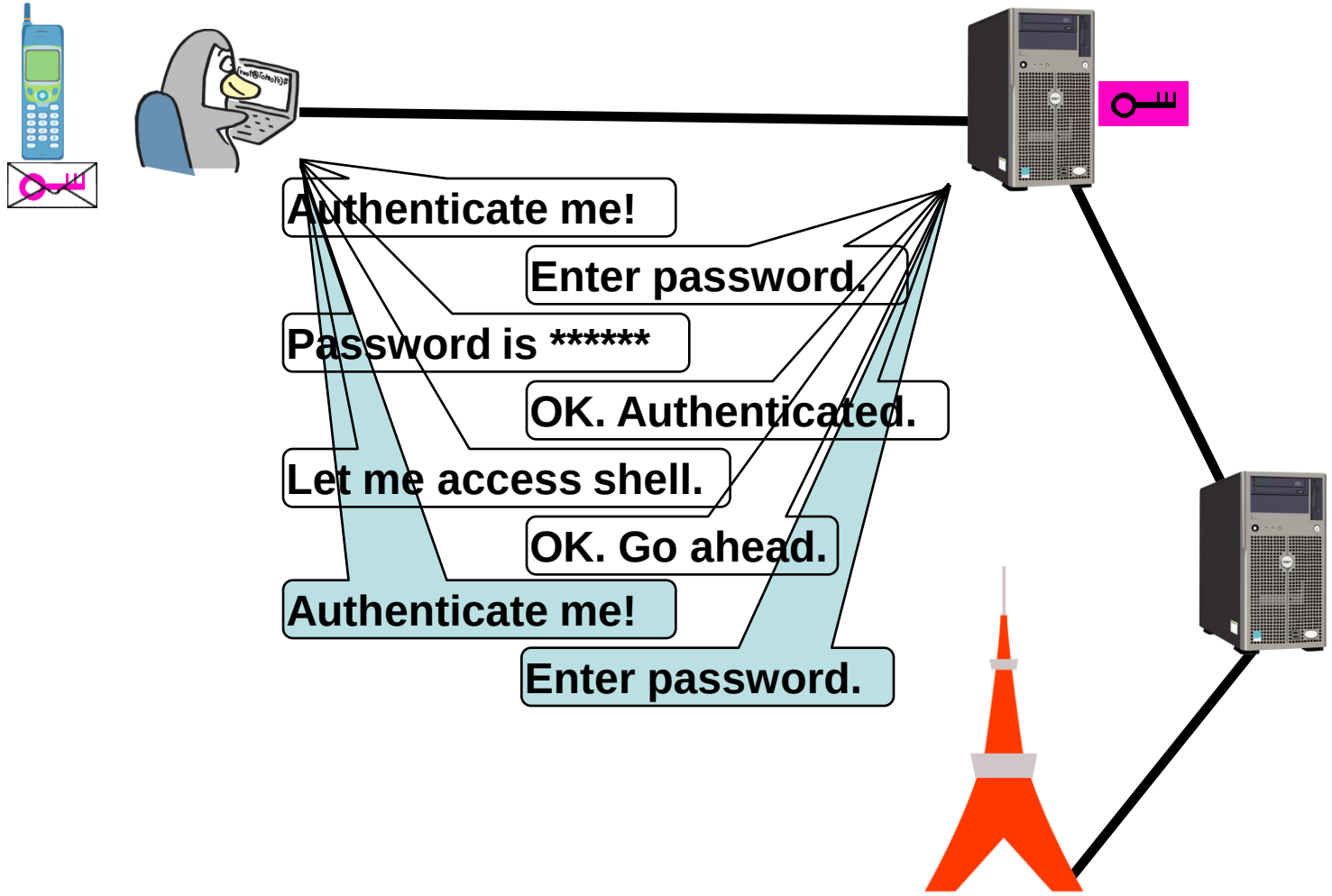
ケース2：対話型シェルセッション



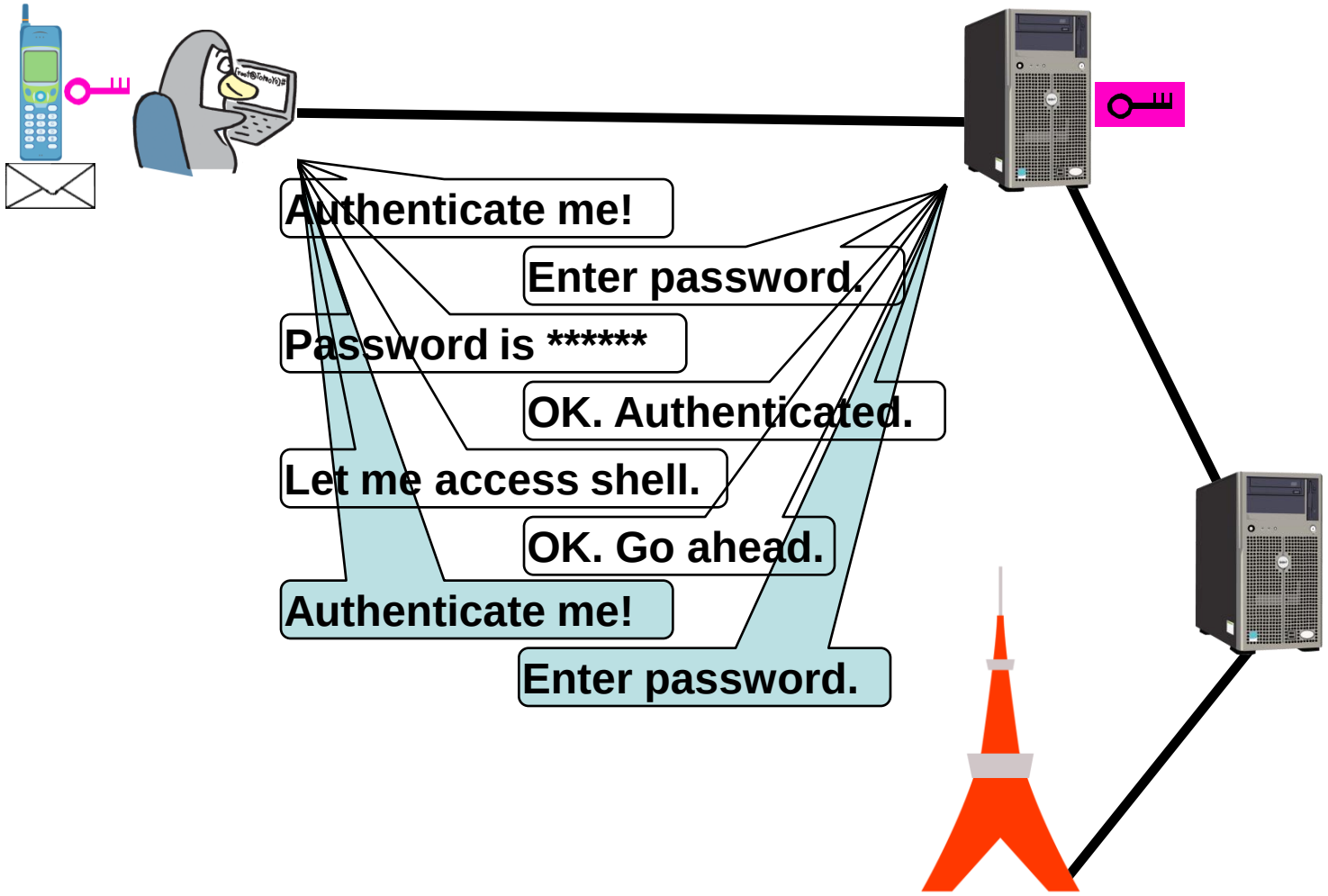
ケース2：対話型シェルセッション



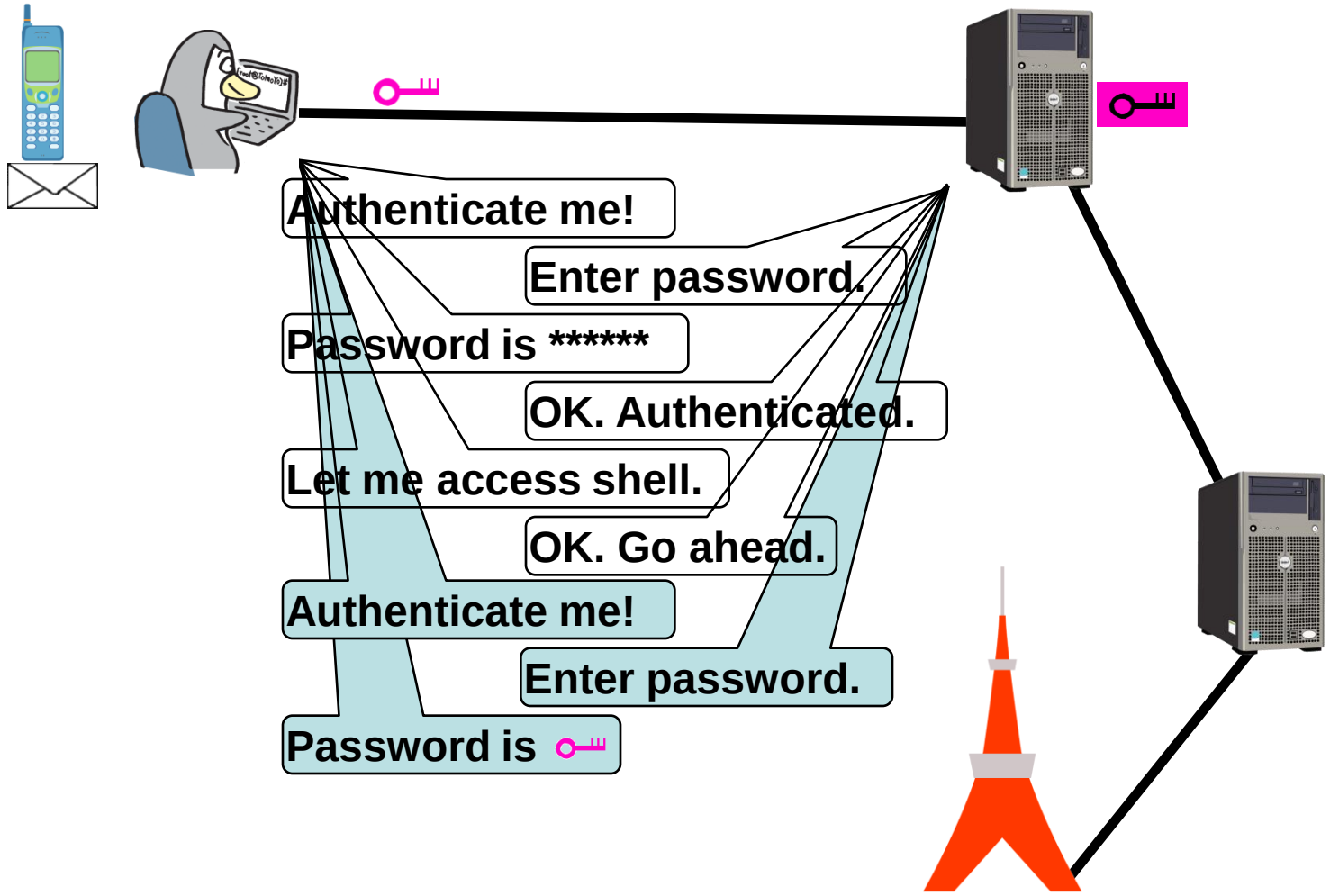
ケース2：対話型シェルセッション



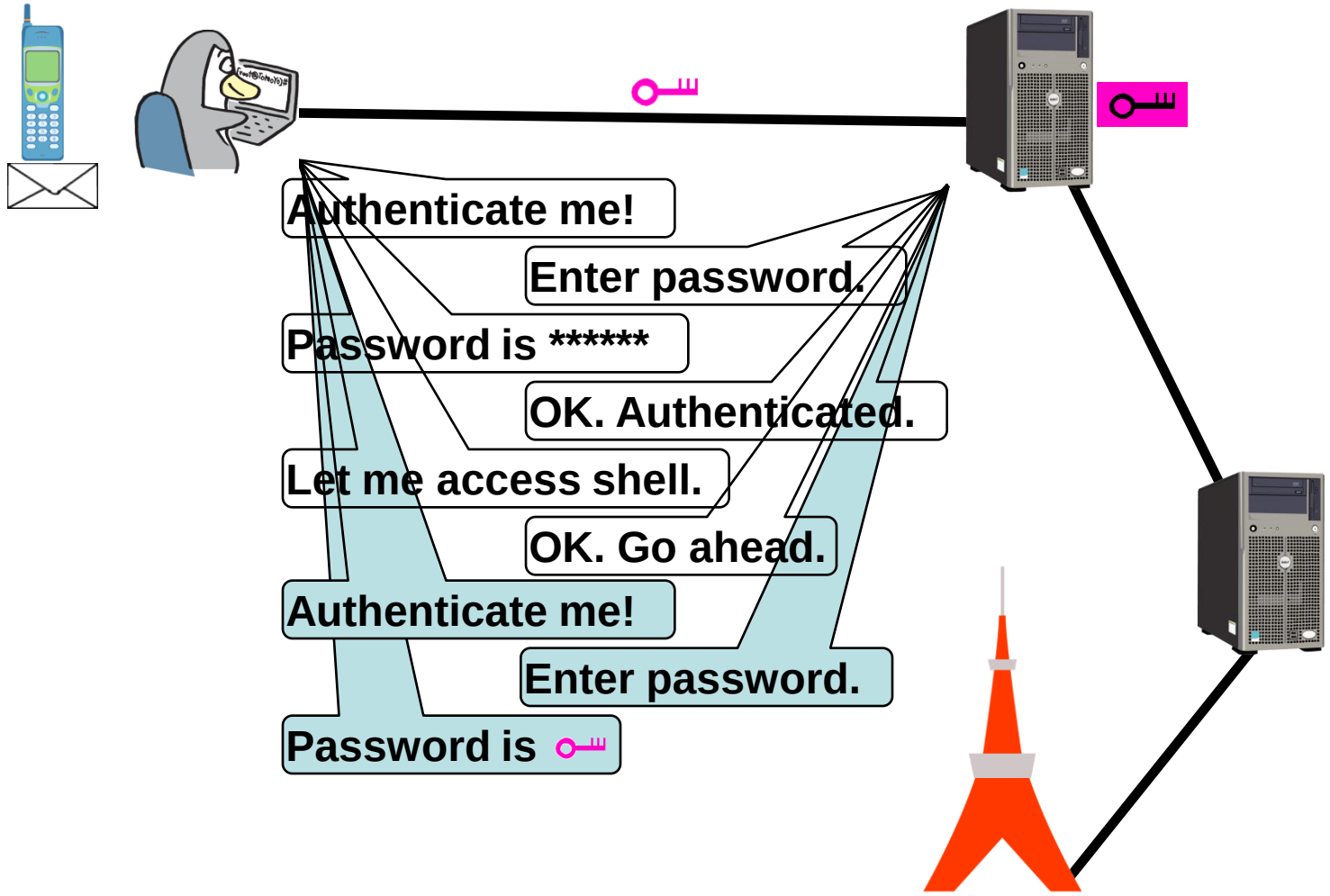
ケース2：対話型シェルセッション



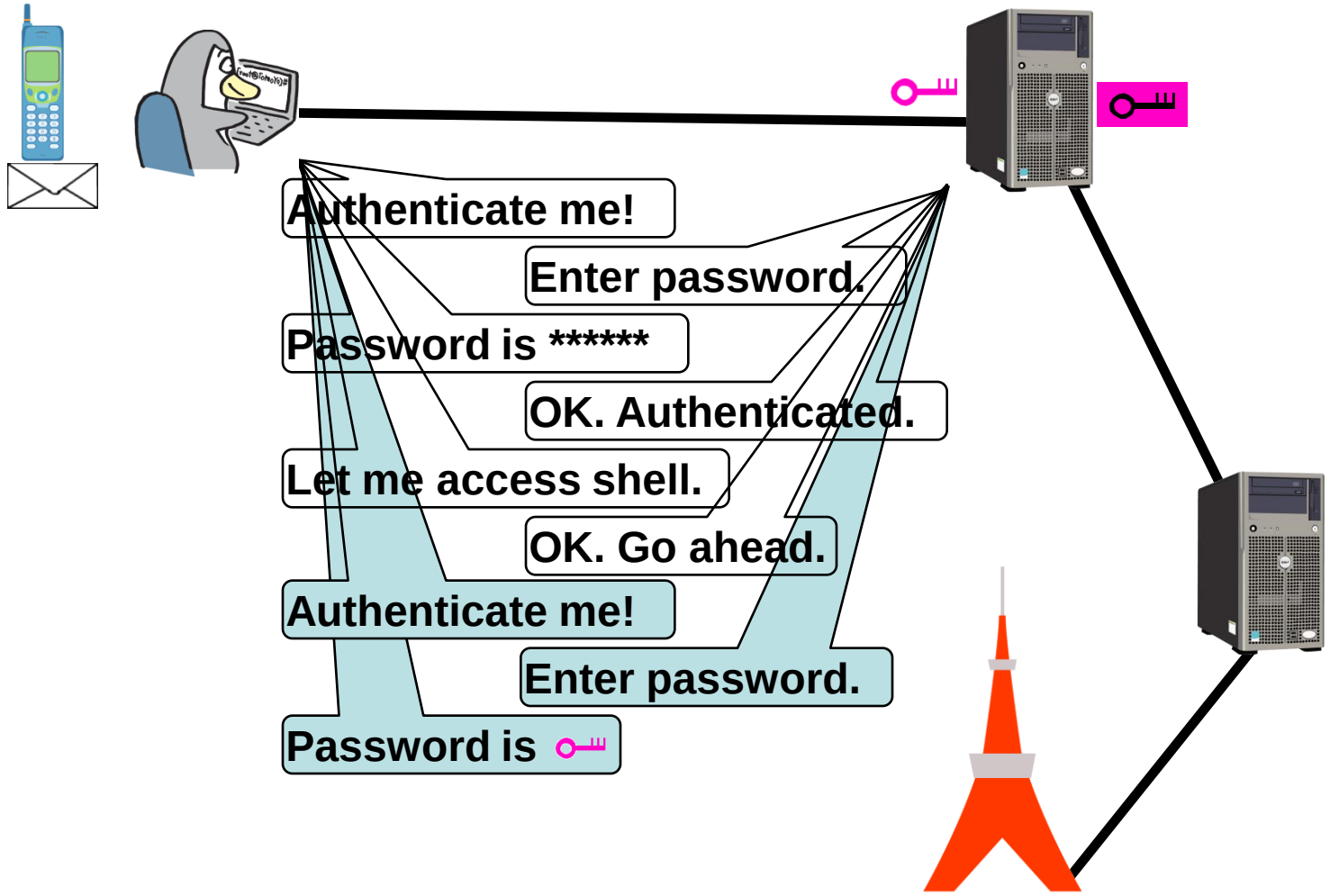
ケース2：対話型シェルセッション



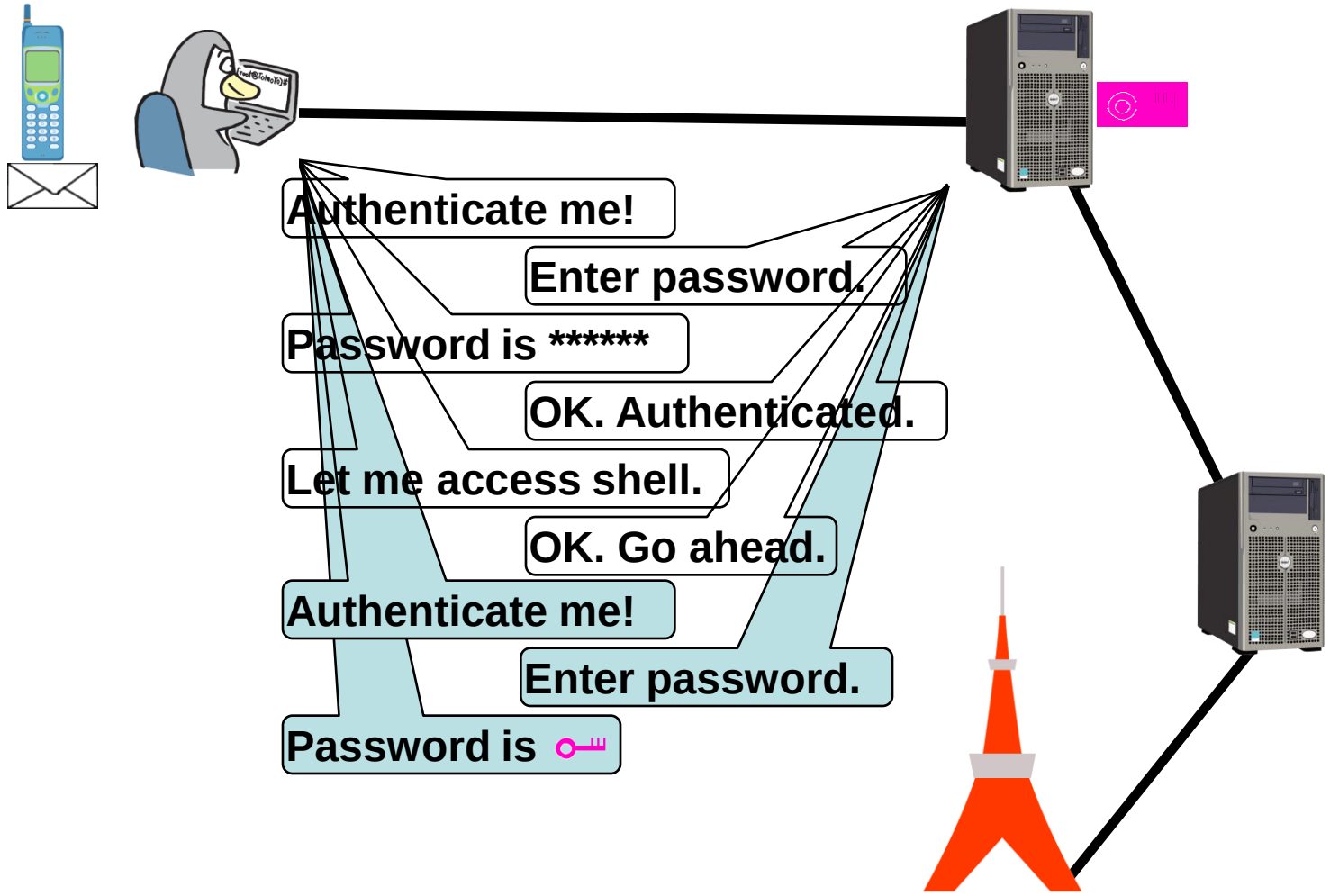
ケース2：対話型シェルセッション



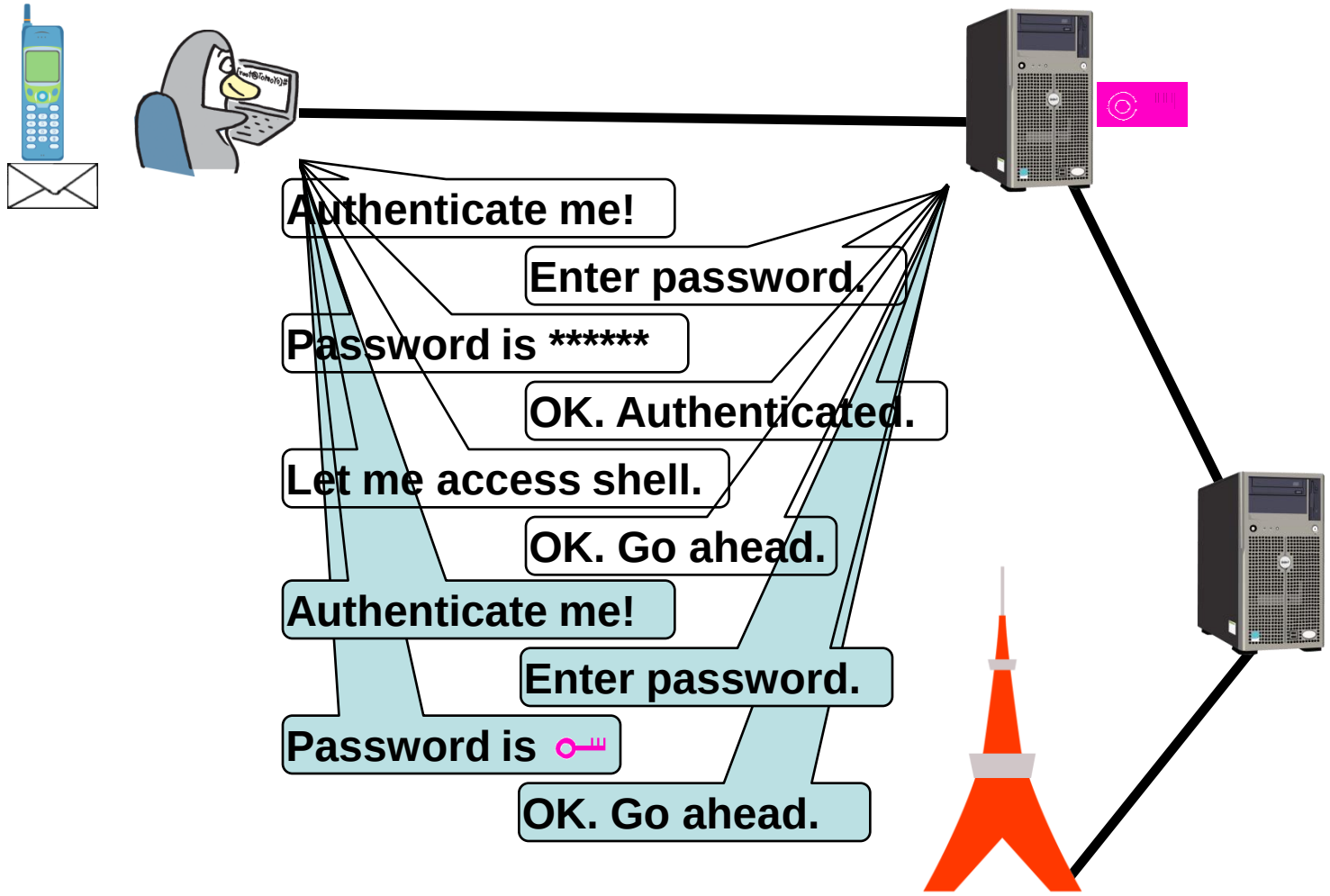
ケース2：対話型シェルセッション



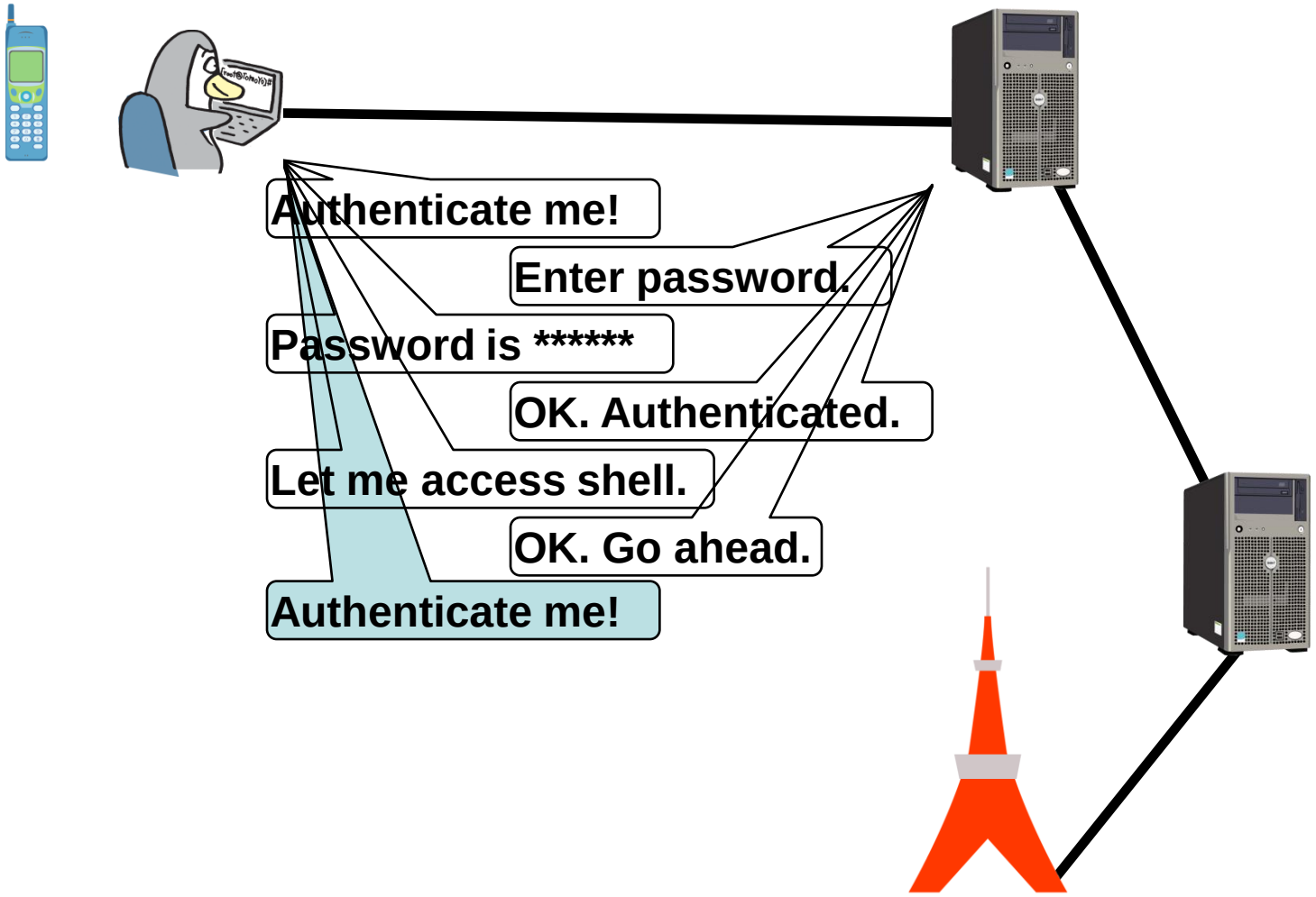
ケース2：対話型シェルセッション



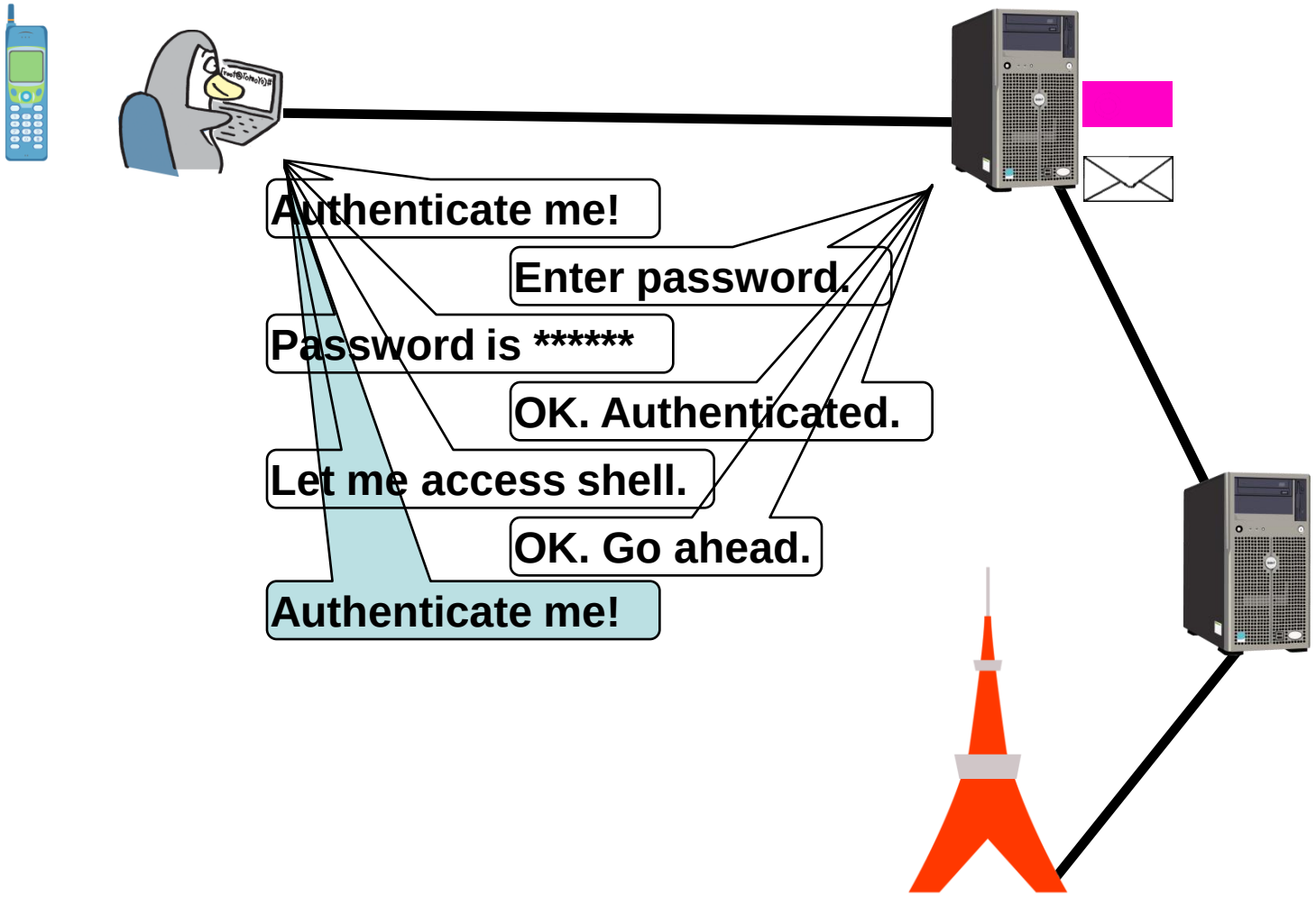
ケース2：対話型シェルセッション



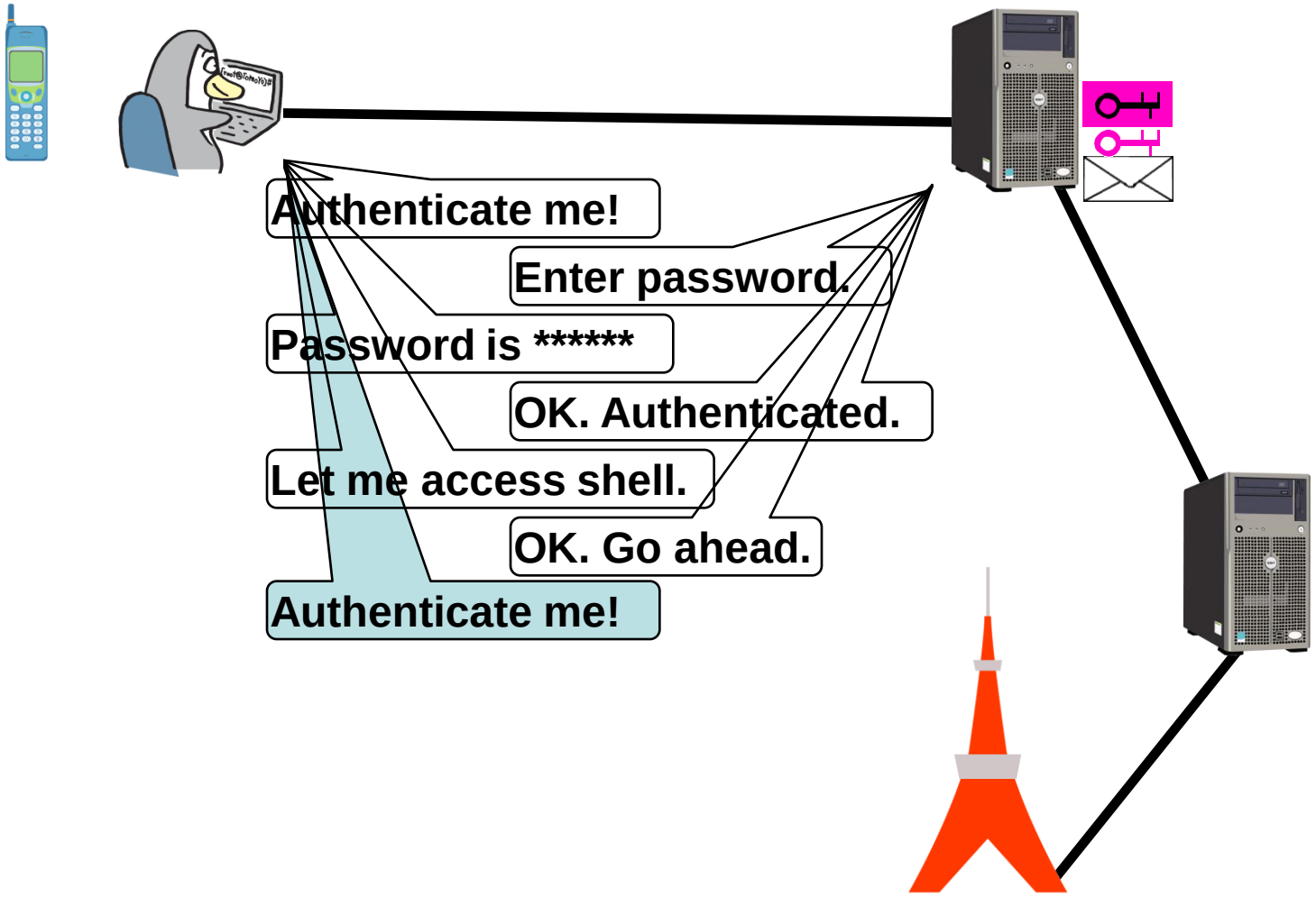
ケース2：対話型シェルセッション



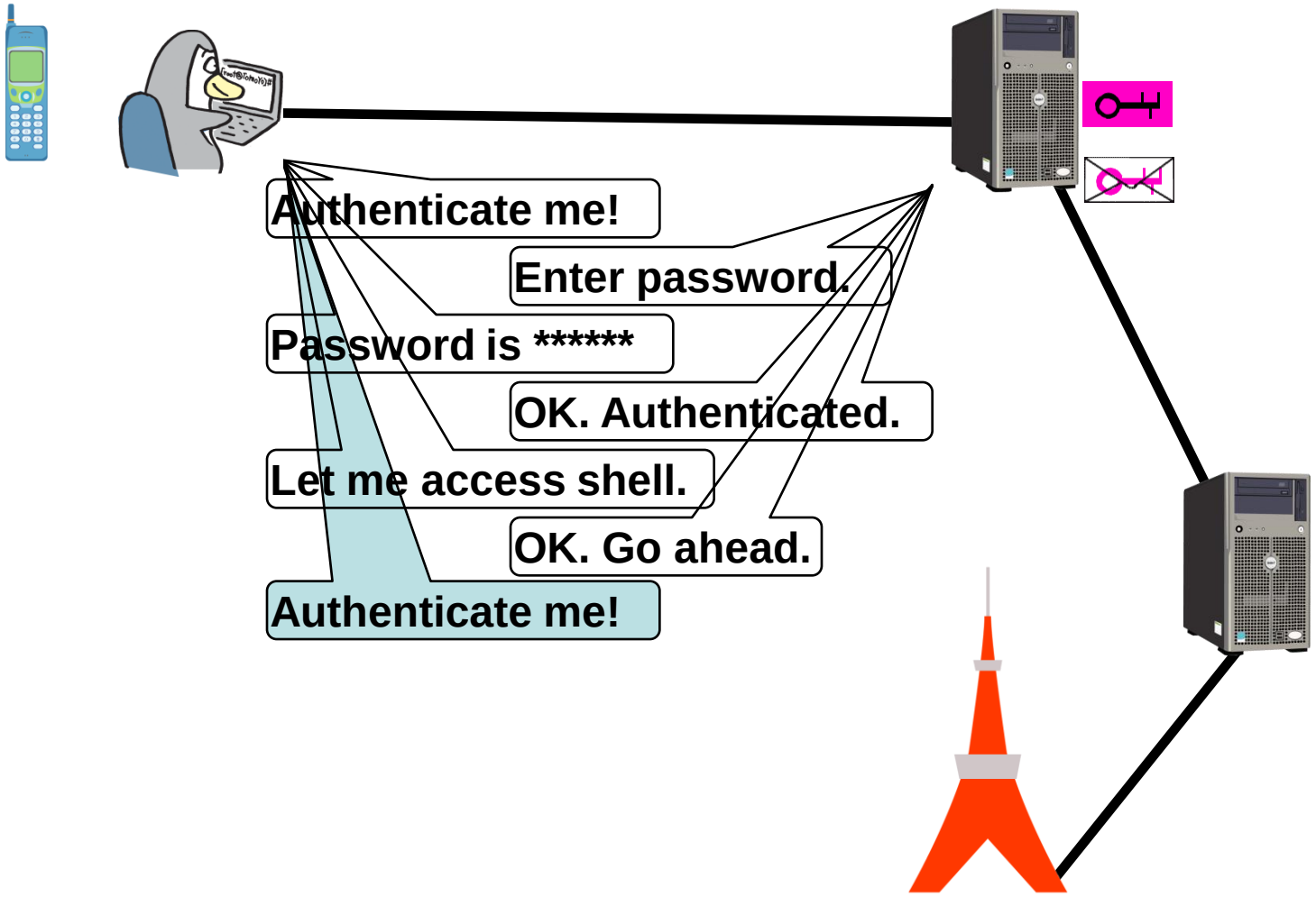
ケース2：対話型シェルセッション



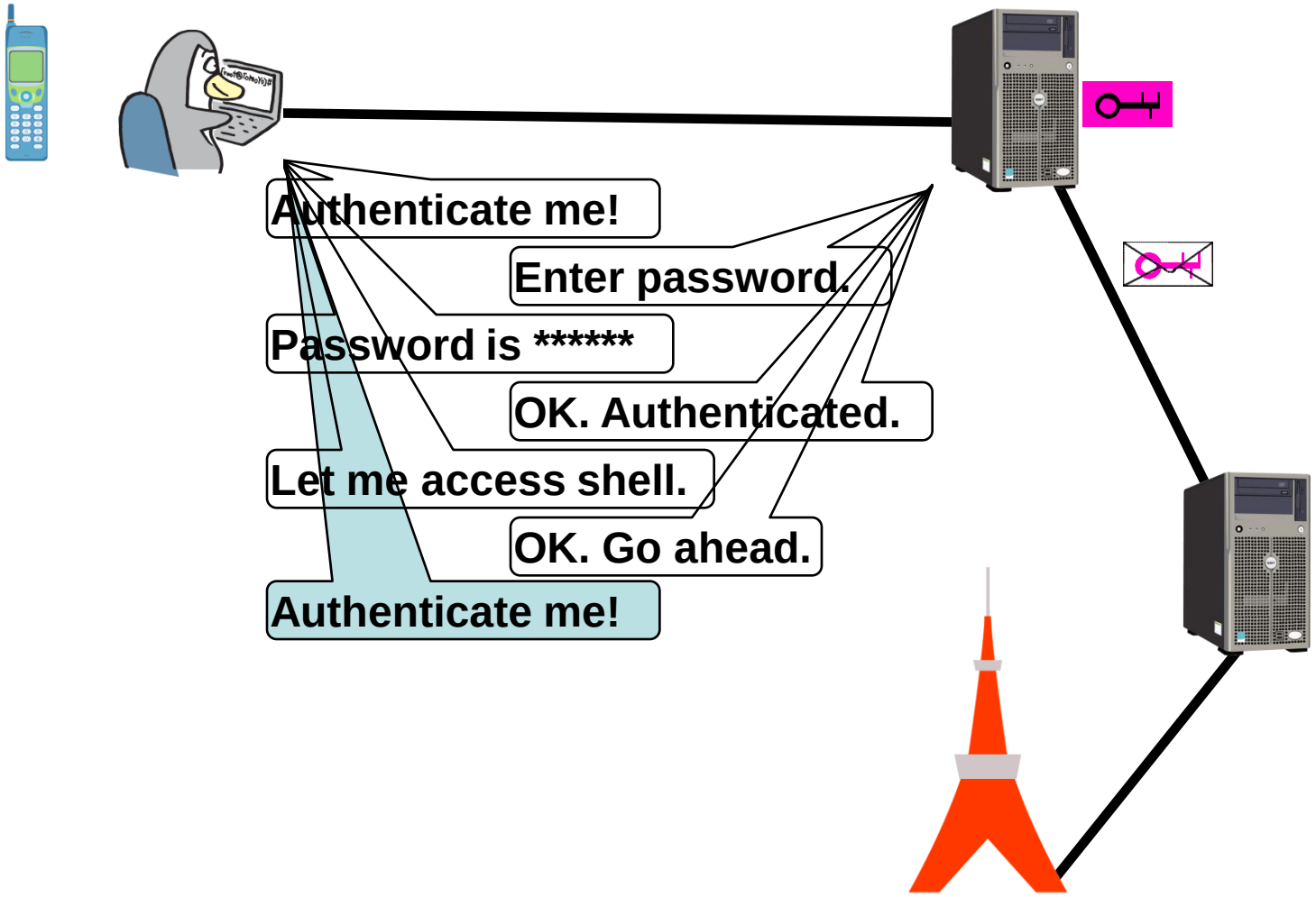
ケース2：対話型シェルセッション



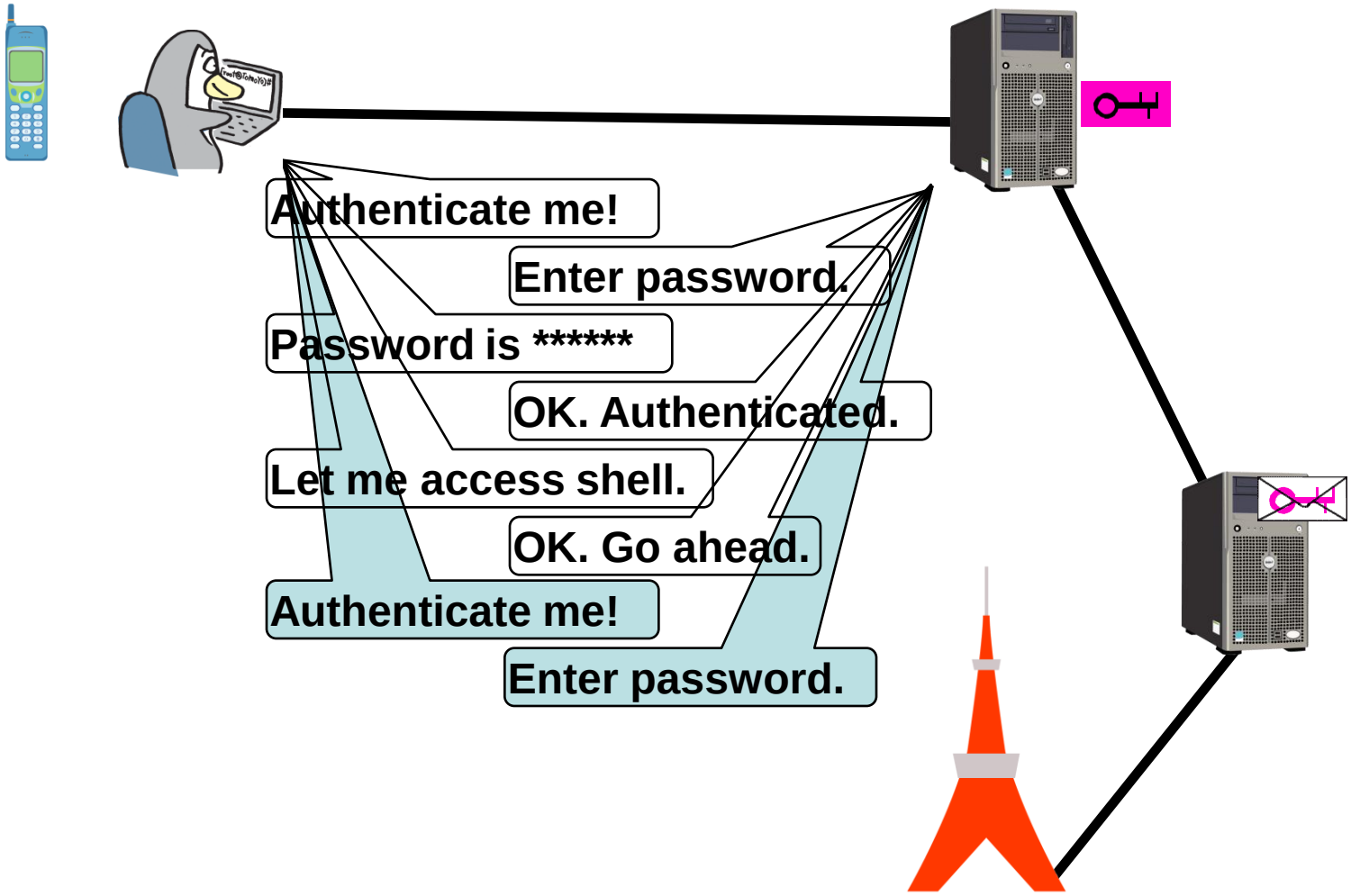
ケース2：対話型シェルセッション



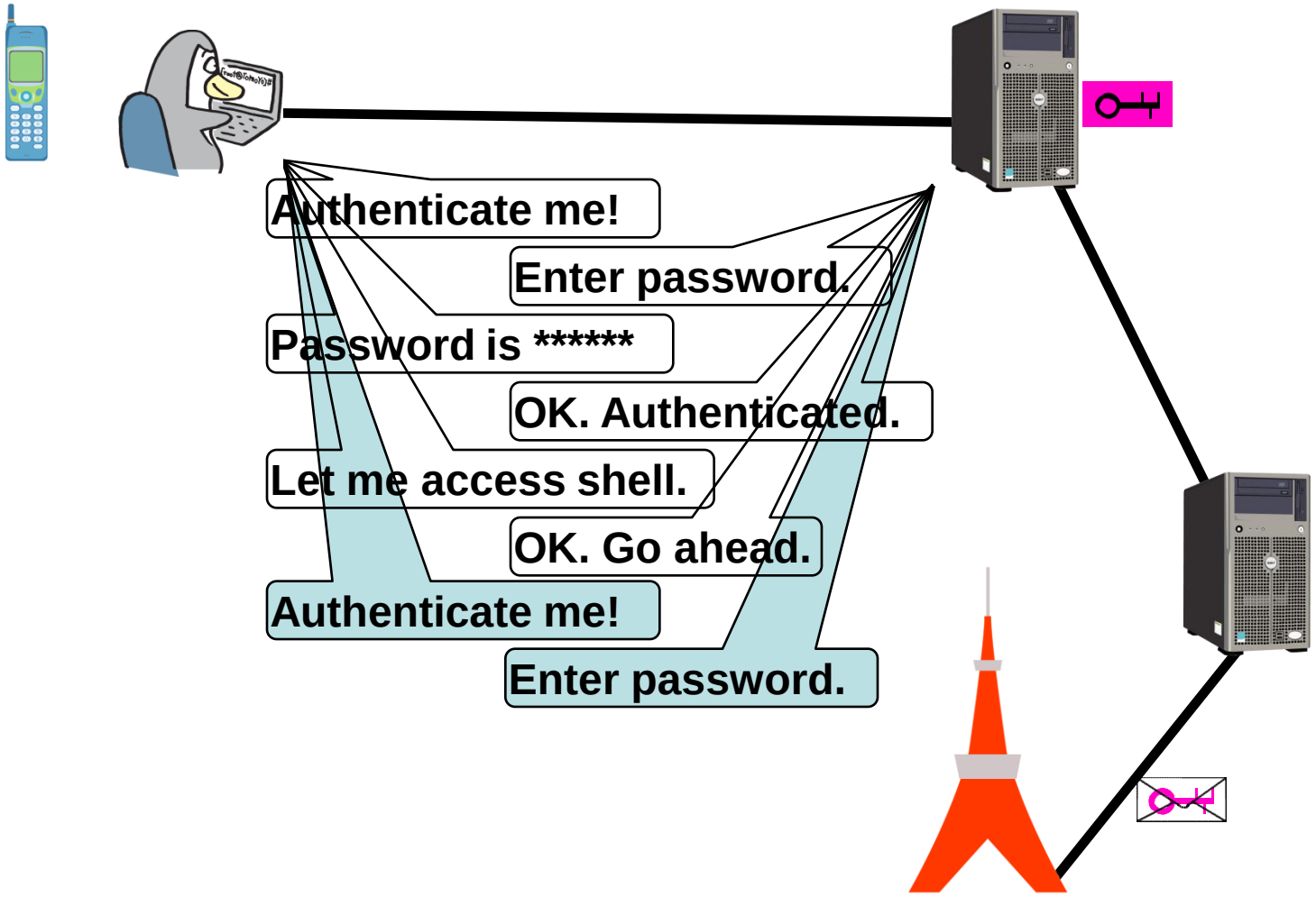
ケース2：対話型シェルセッション



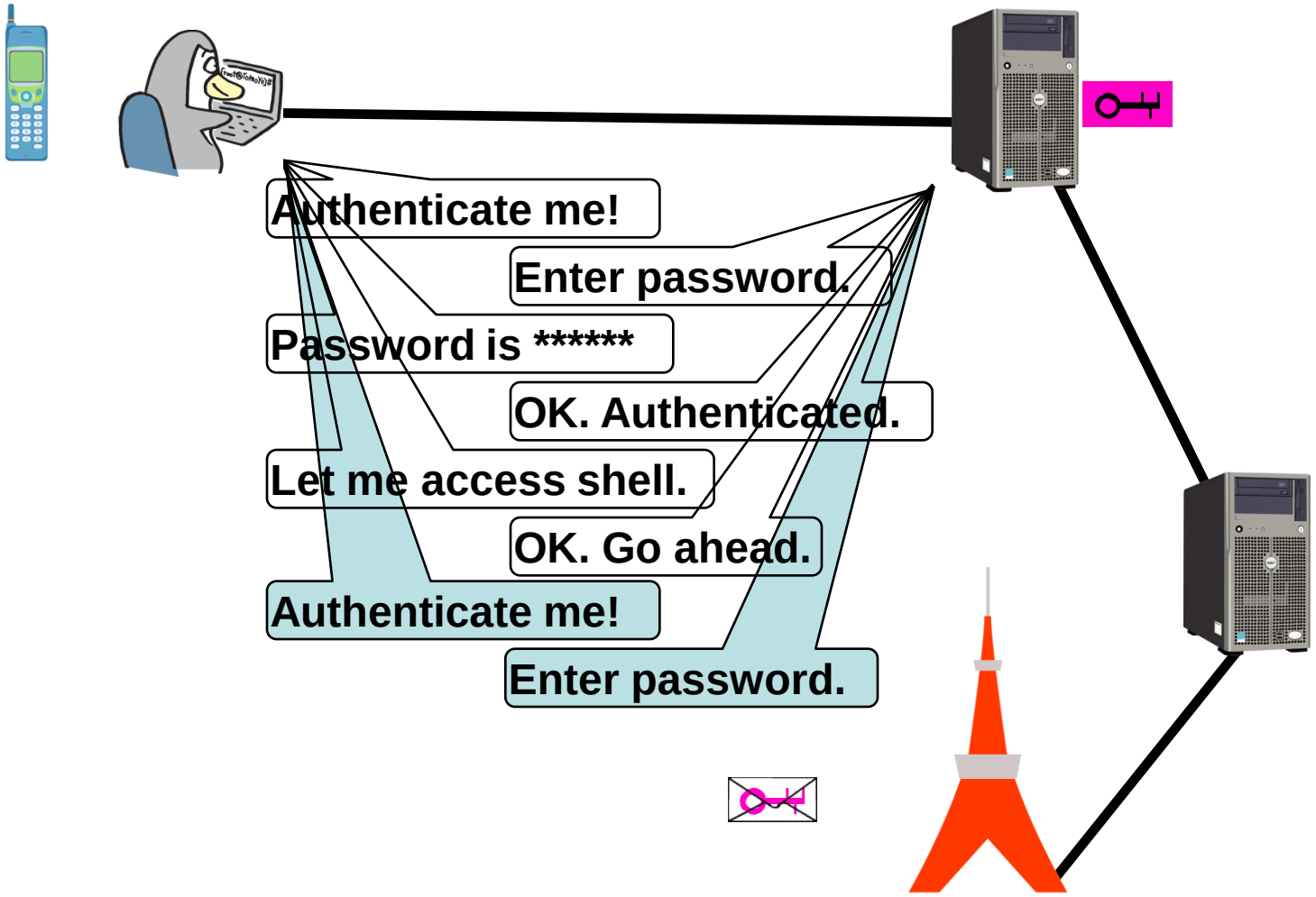
ケース2：対話型シェルセッション



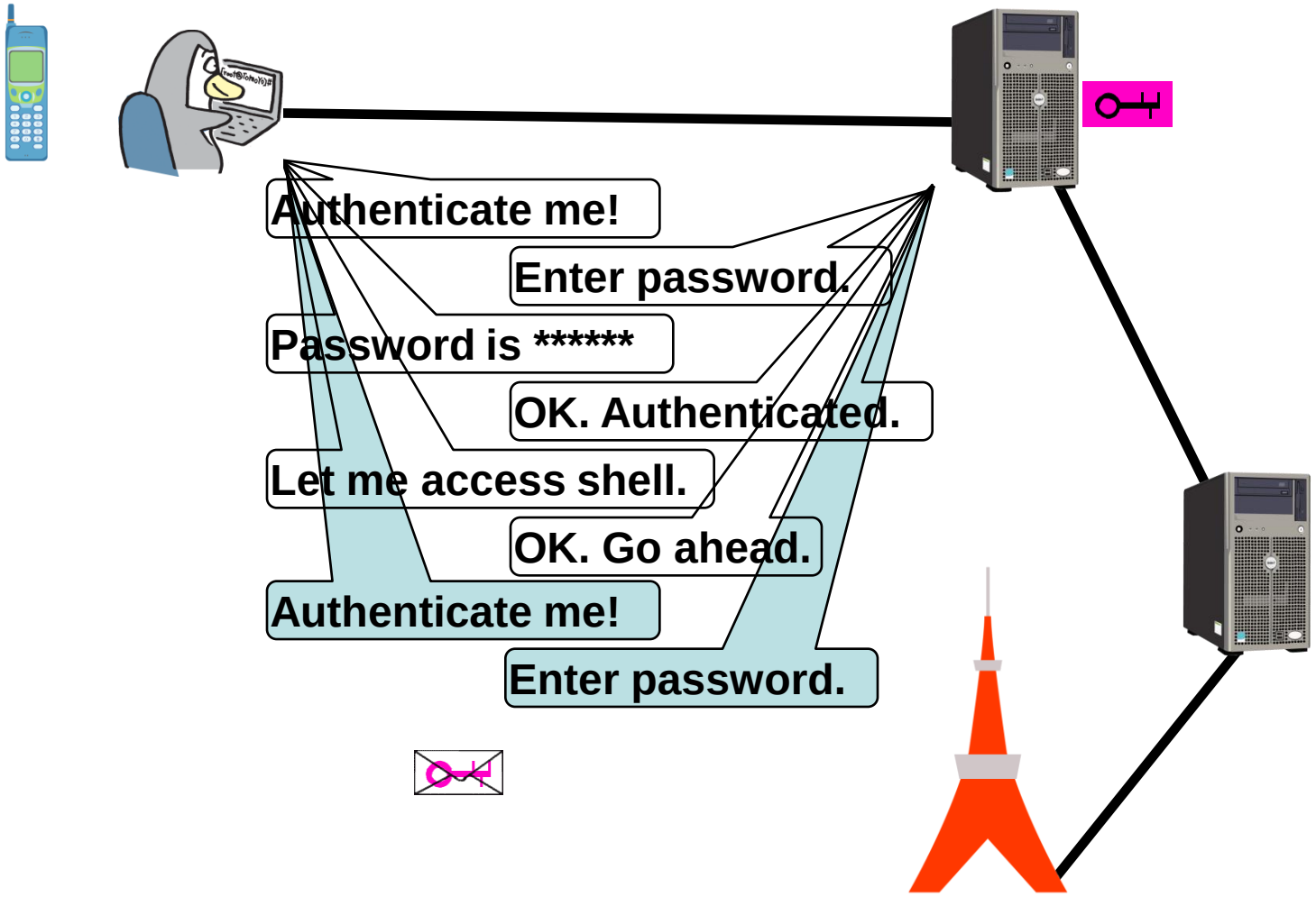
ケース2：対話型シェルセッション



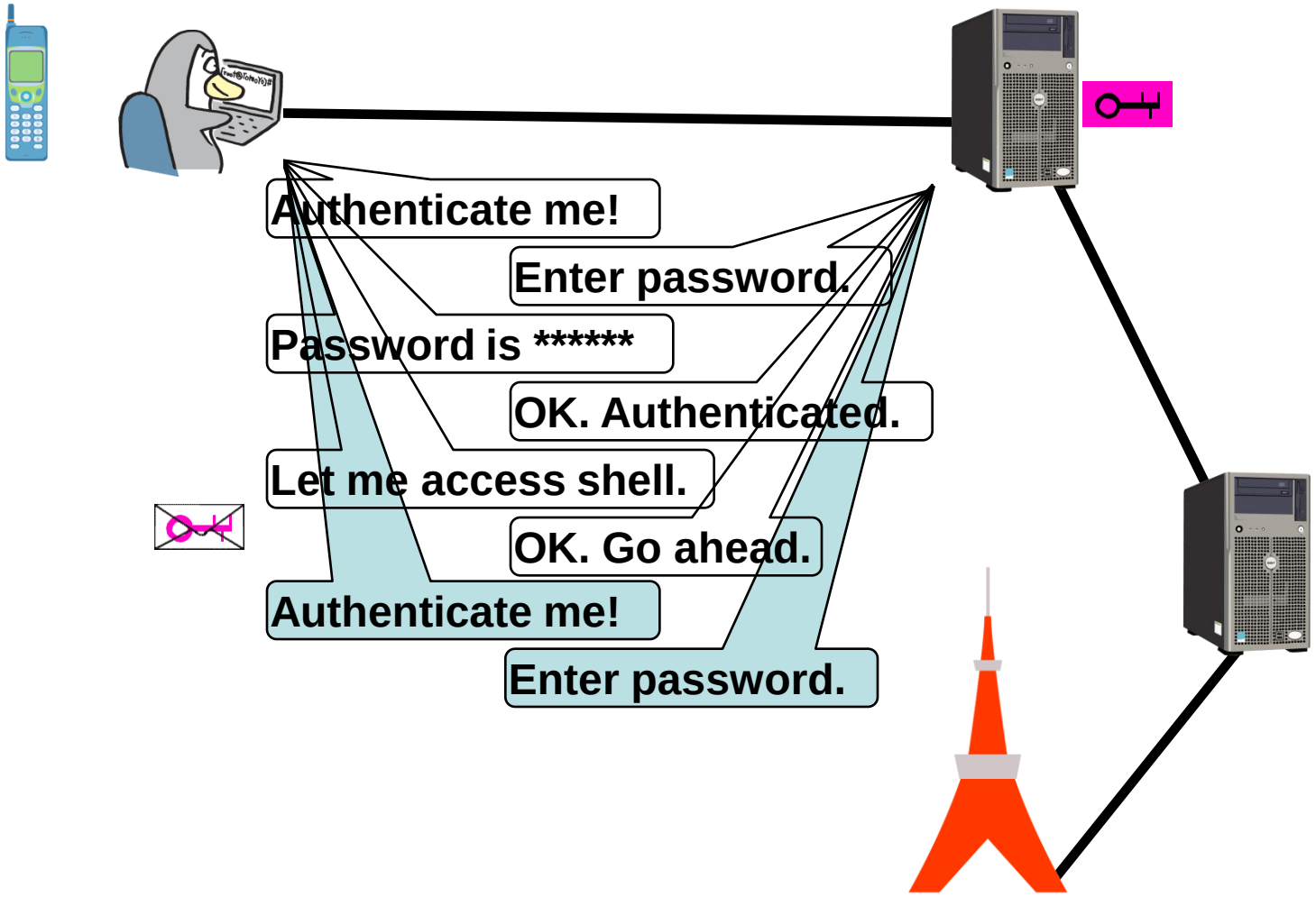
ケース2：対話型シェルセッション



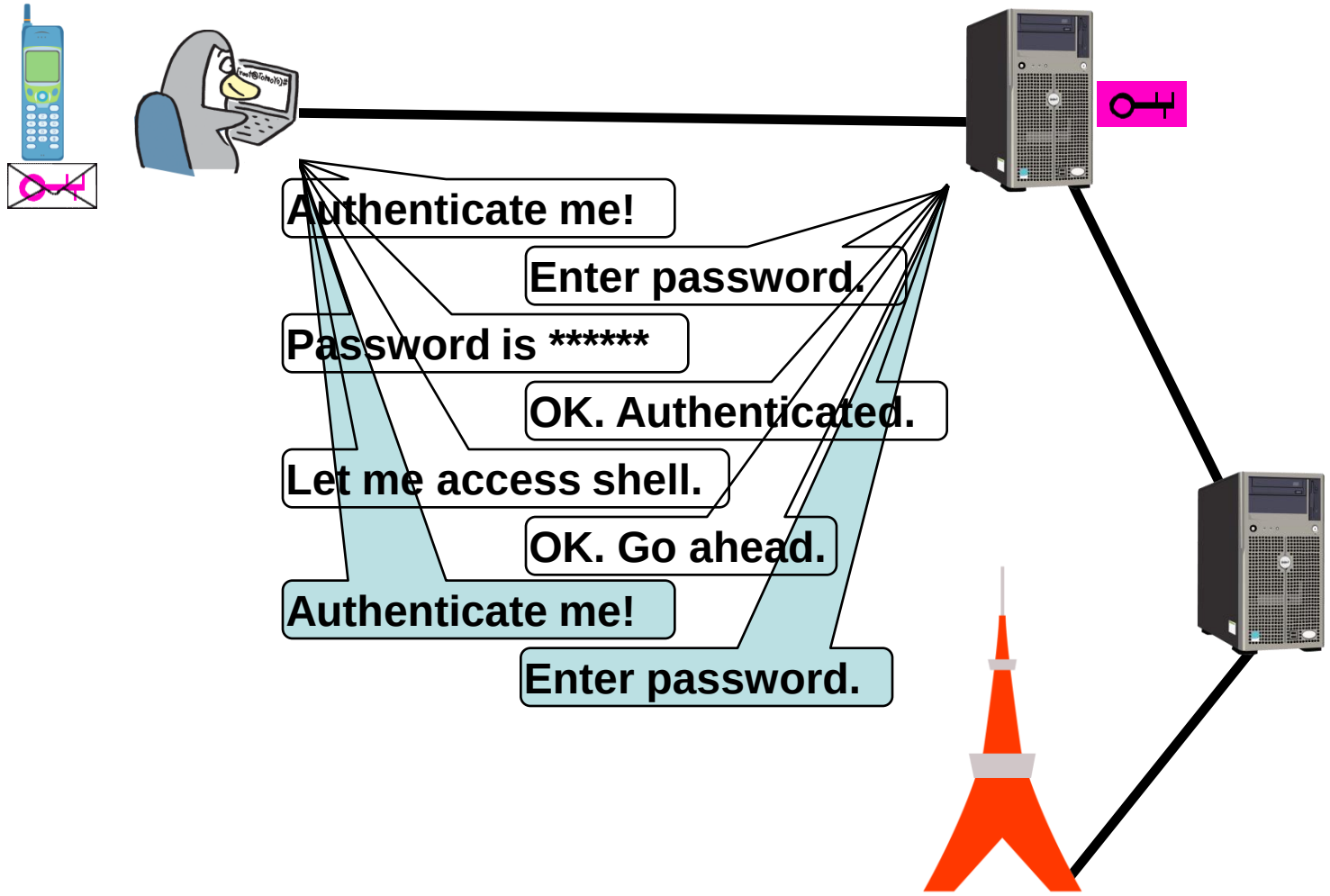
ケース2：対話型シェルセッション



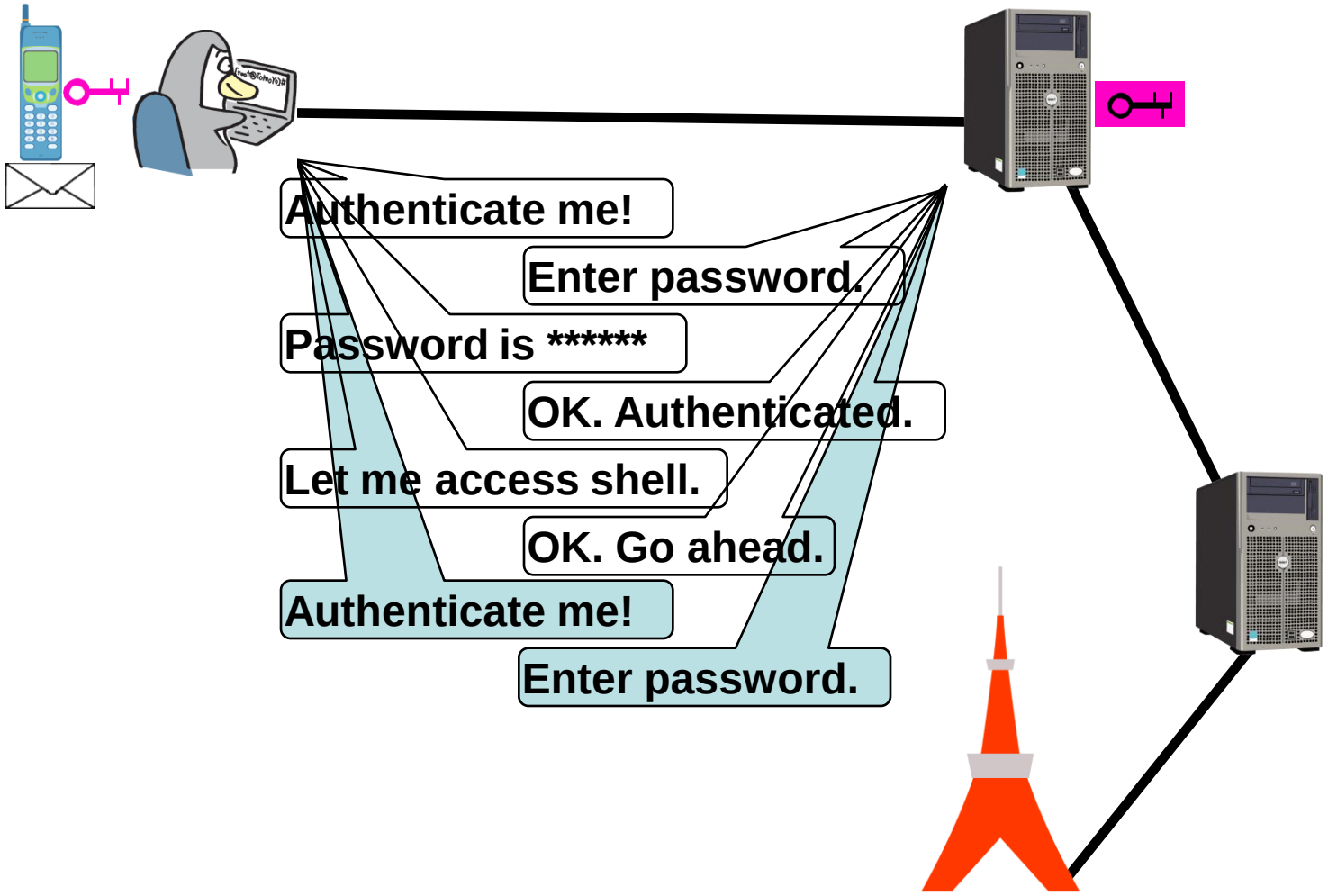
ケース2：対話型シェルセッション



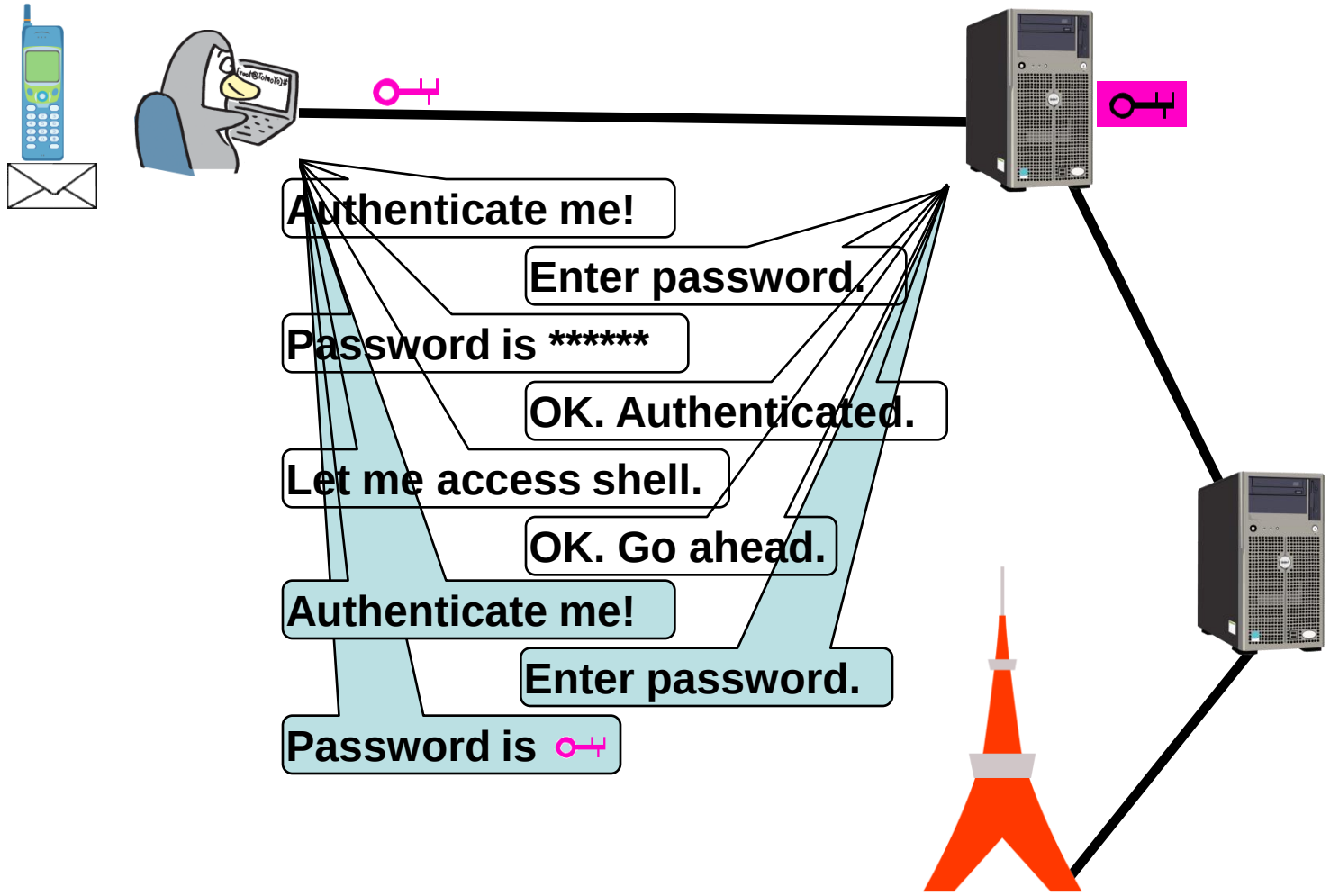
ケース2：対話型シェルセッション



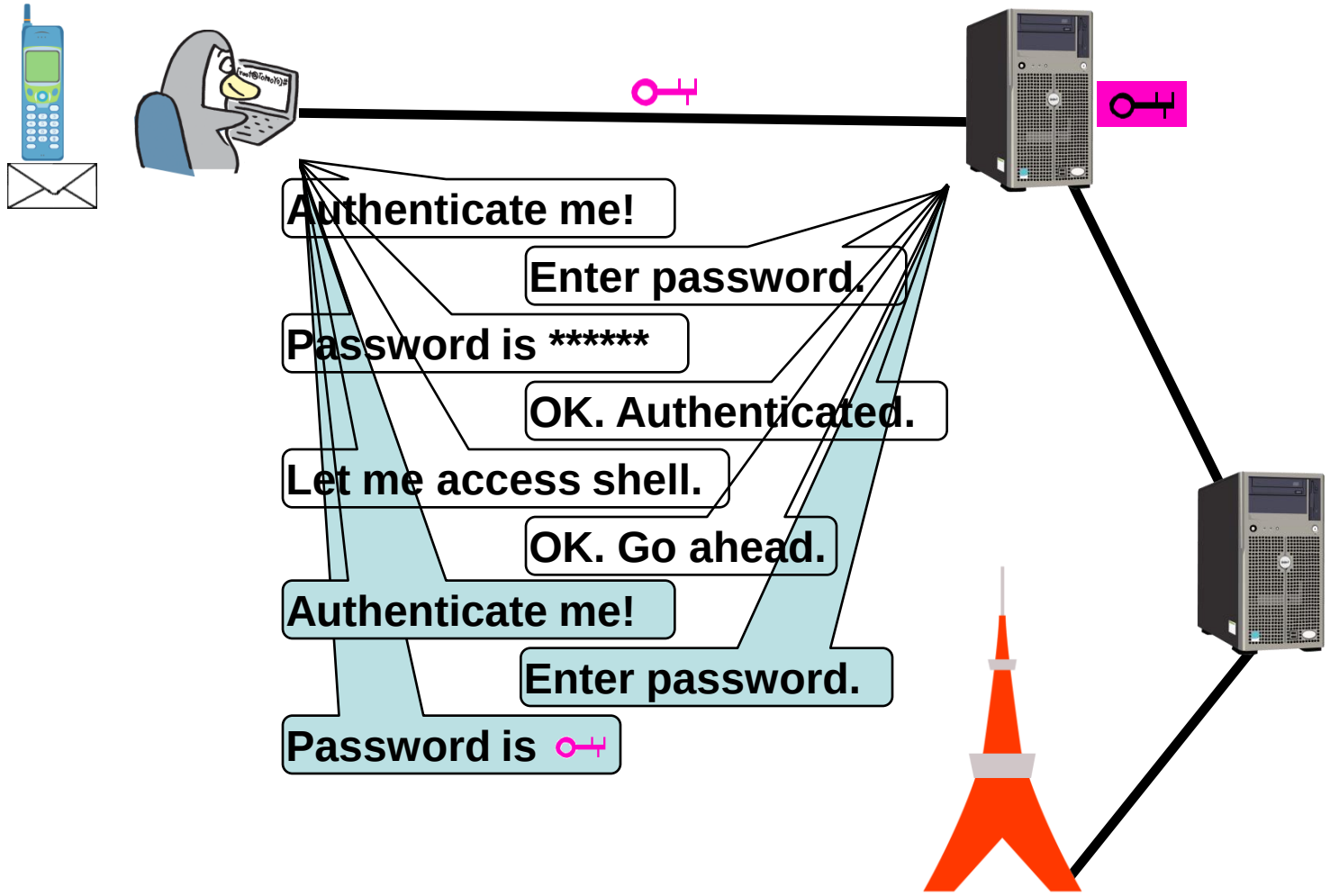
ケース2：対話型シェルセッション



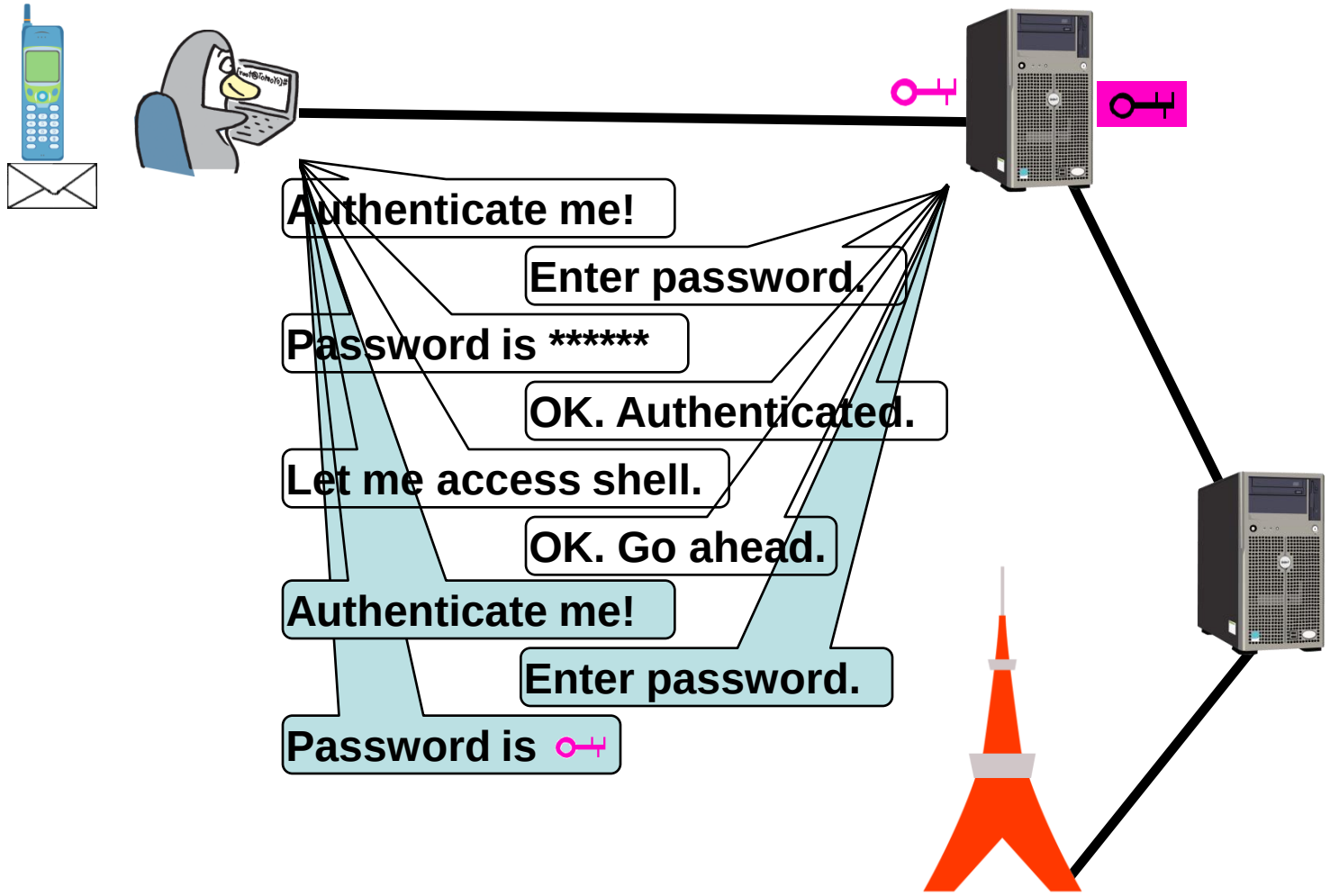
ケース2：対話型シェルセッション



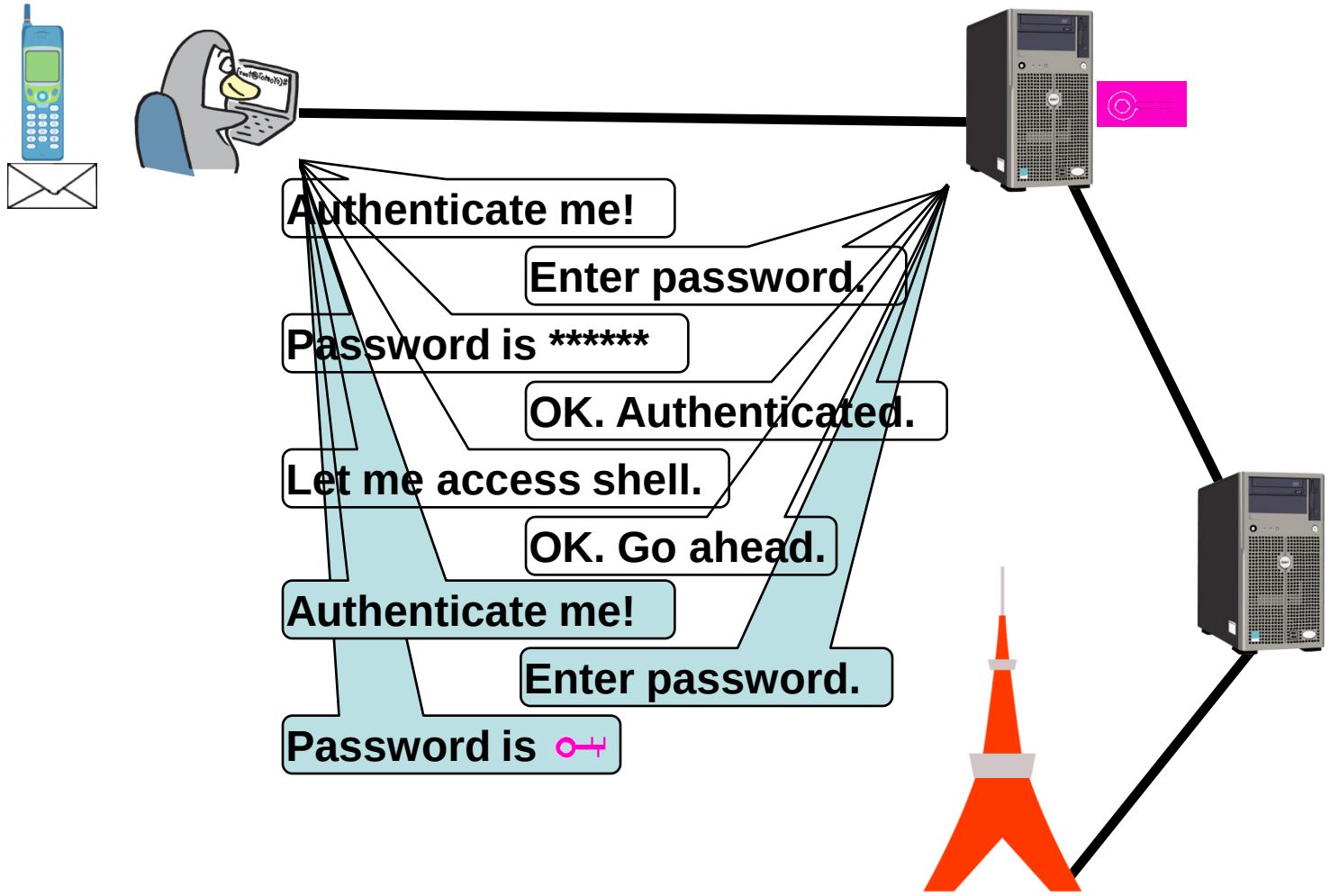
ケース2：対話型シェルセッション



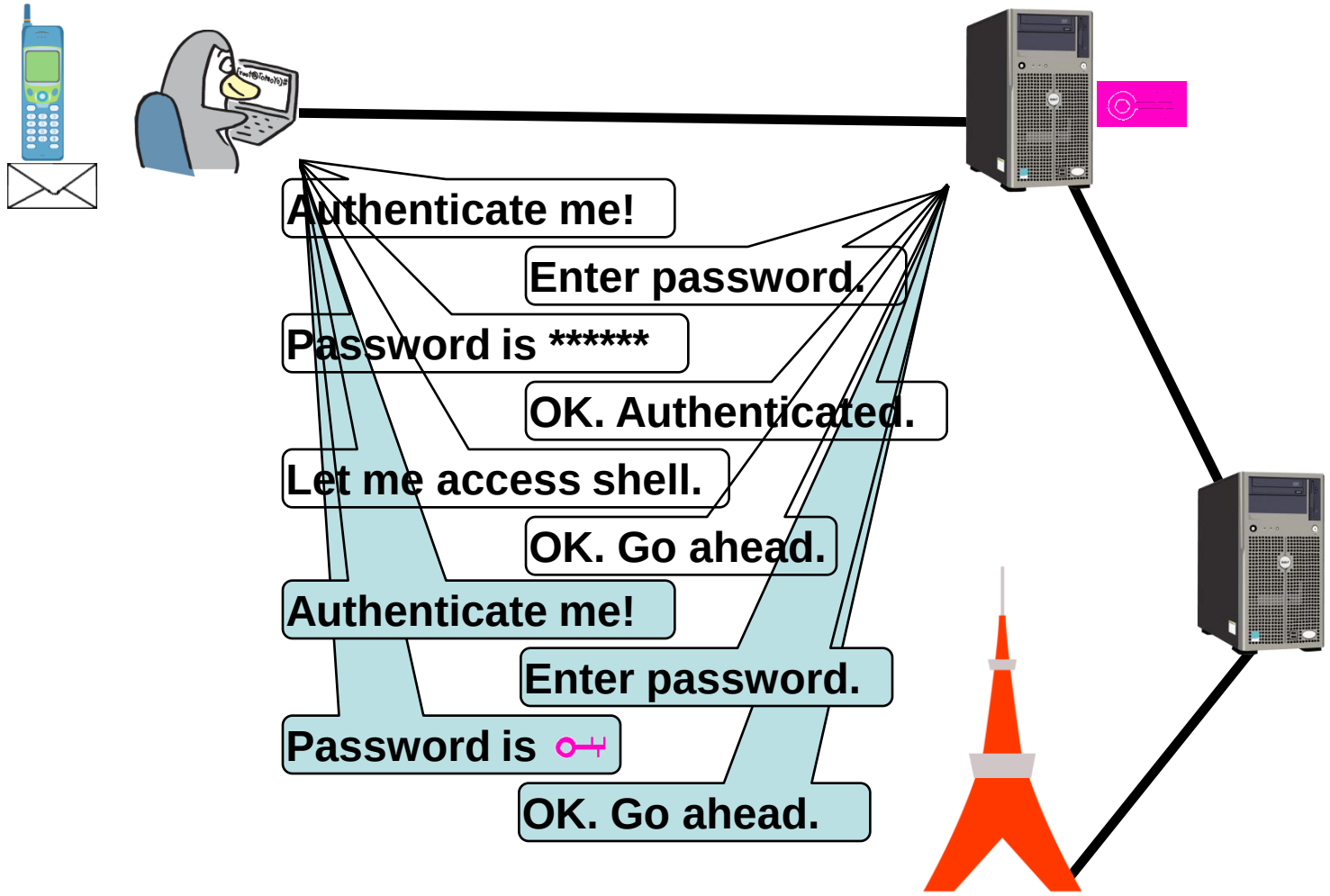
ケース2：対話型シェルセッション



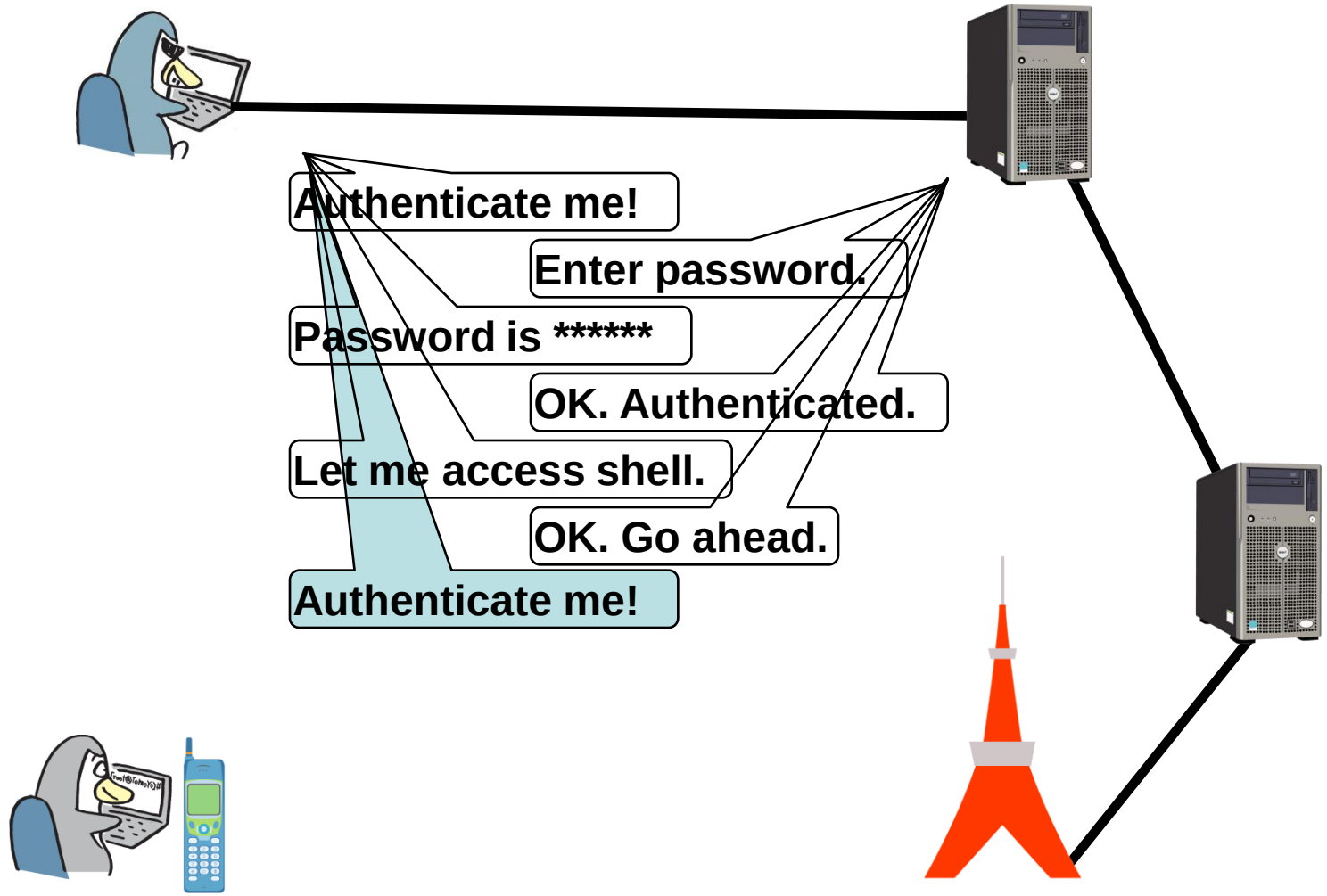
ケース2：対話型シェルセッション



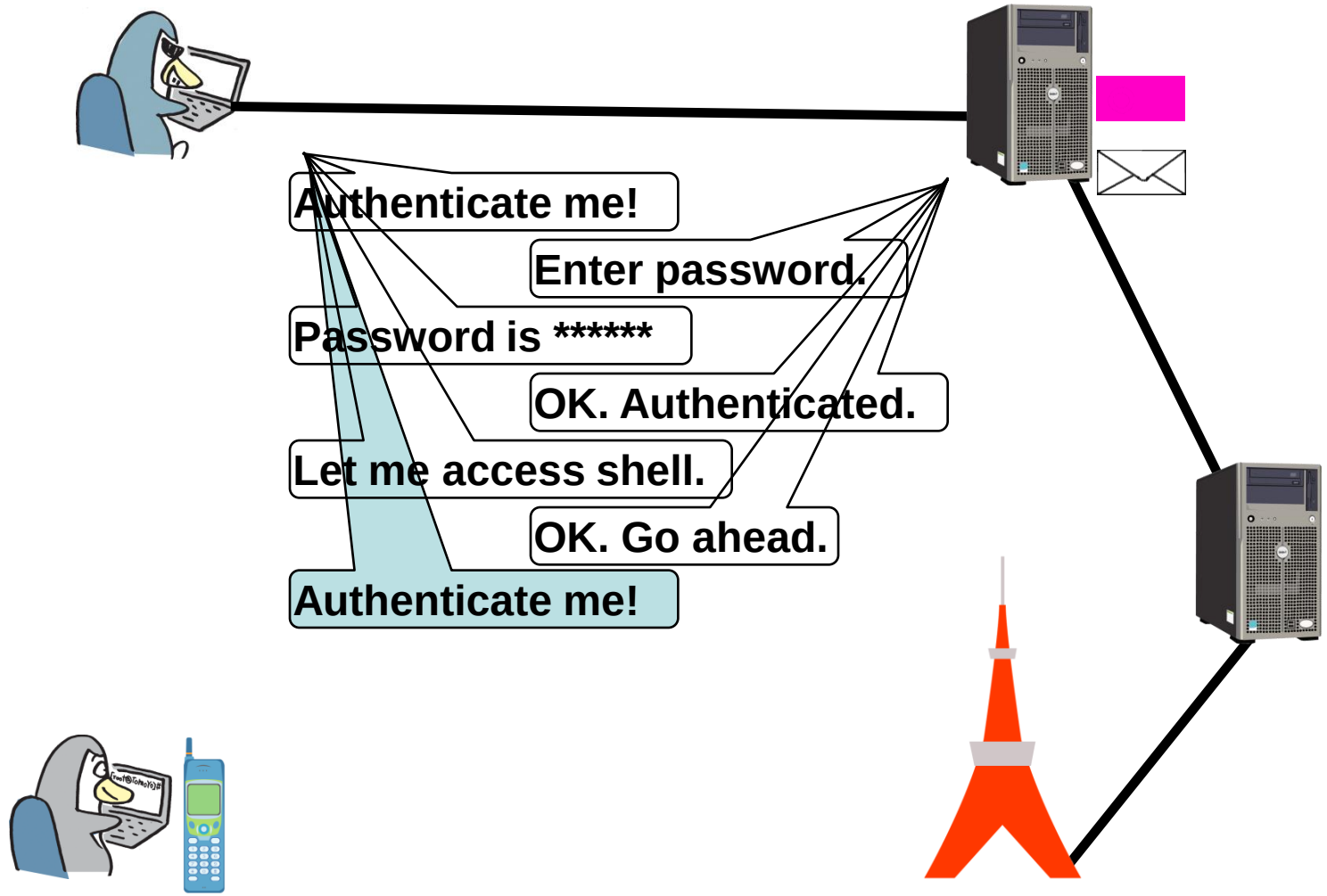
ケース2：対話型シェルセッション



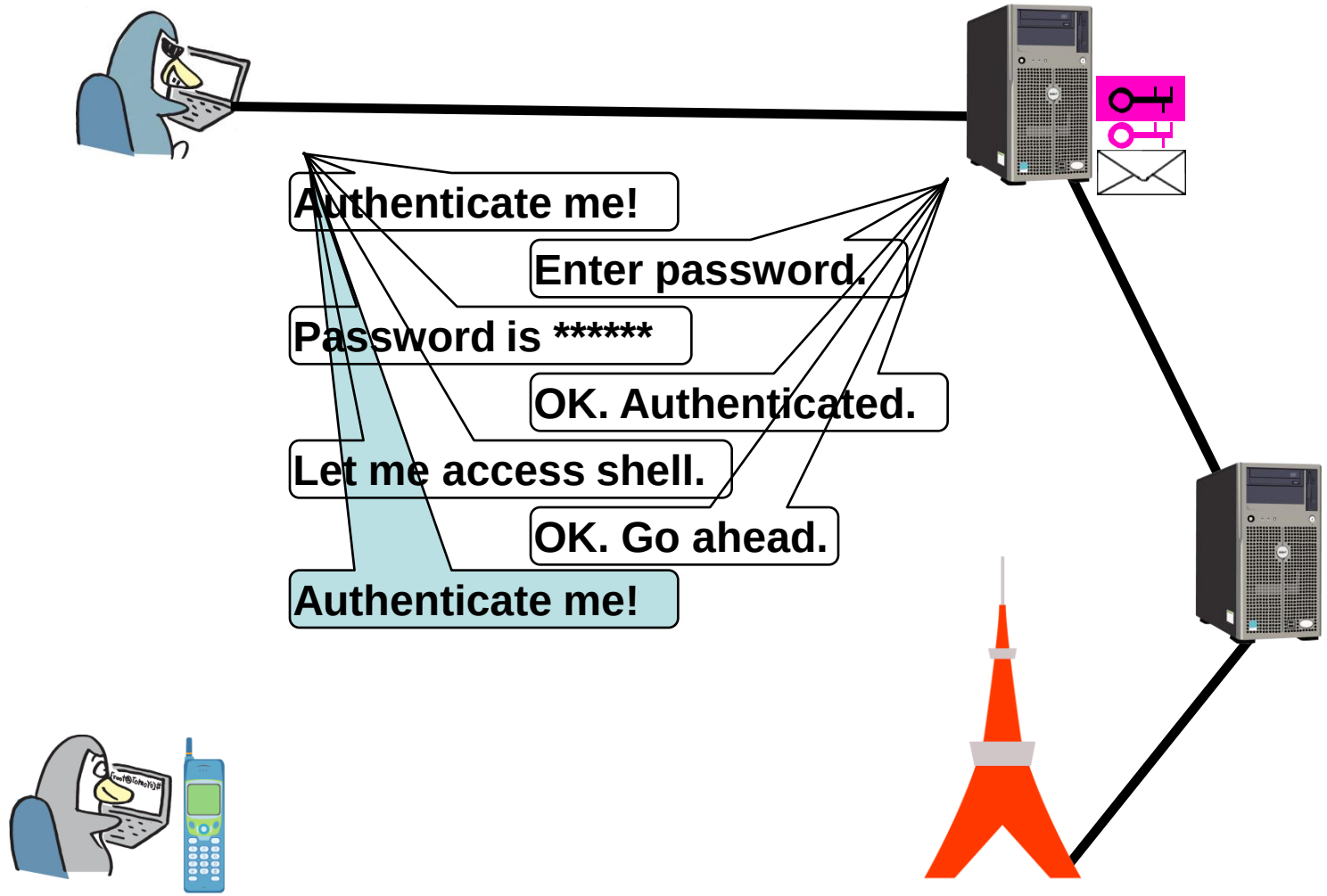
ケース2：対話型シェルセッション



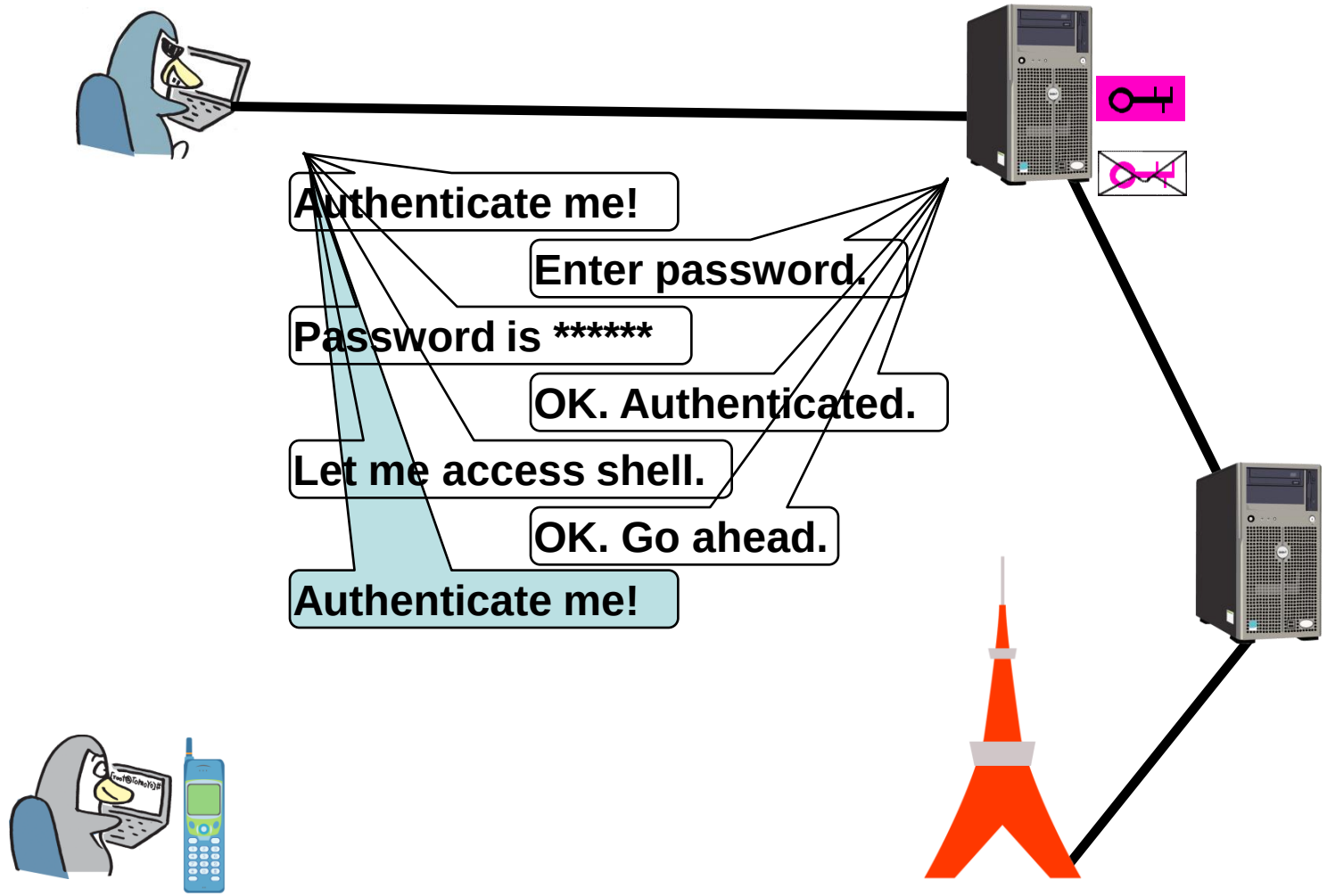
ケース2：対話型シェルセッション



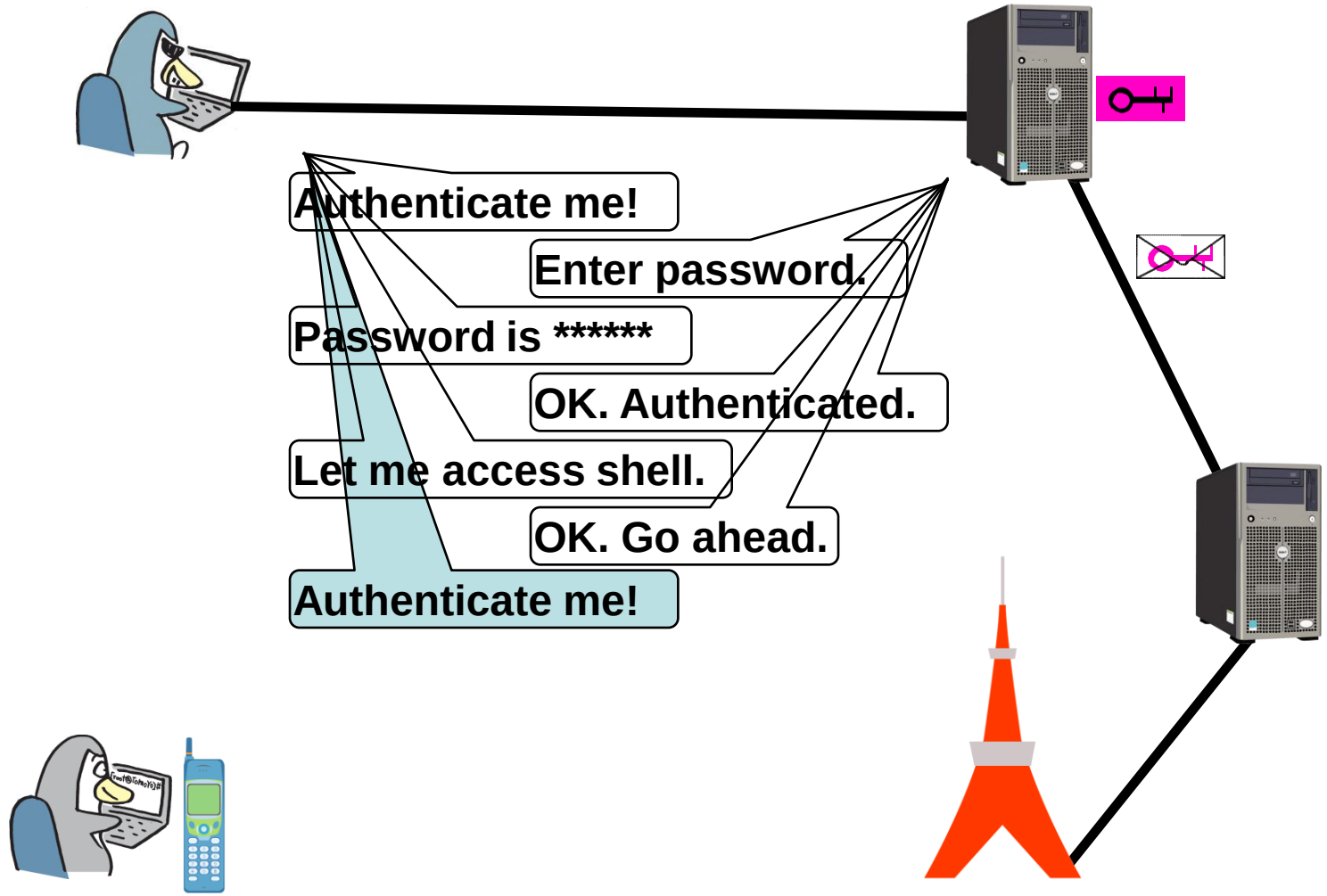
ケース2：対話型シェルセッション



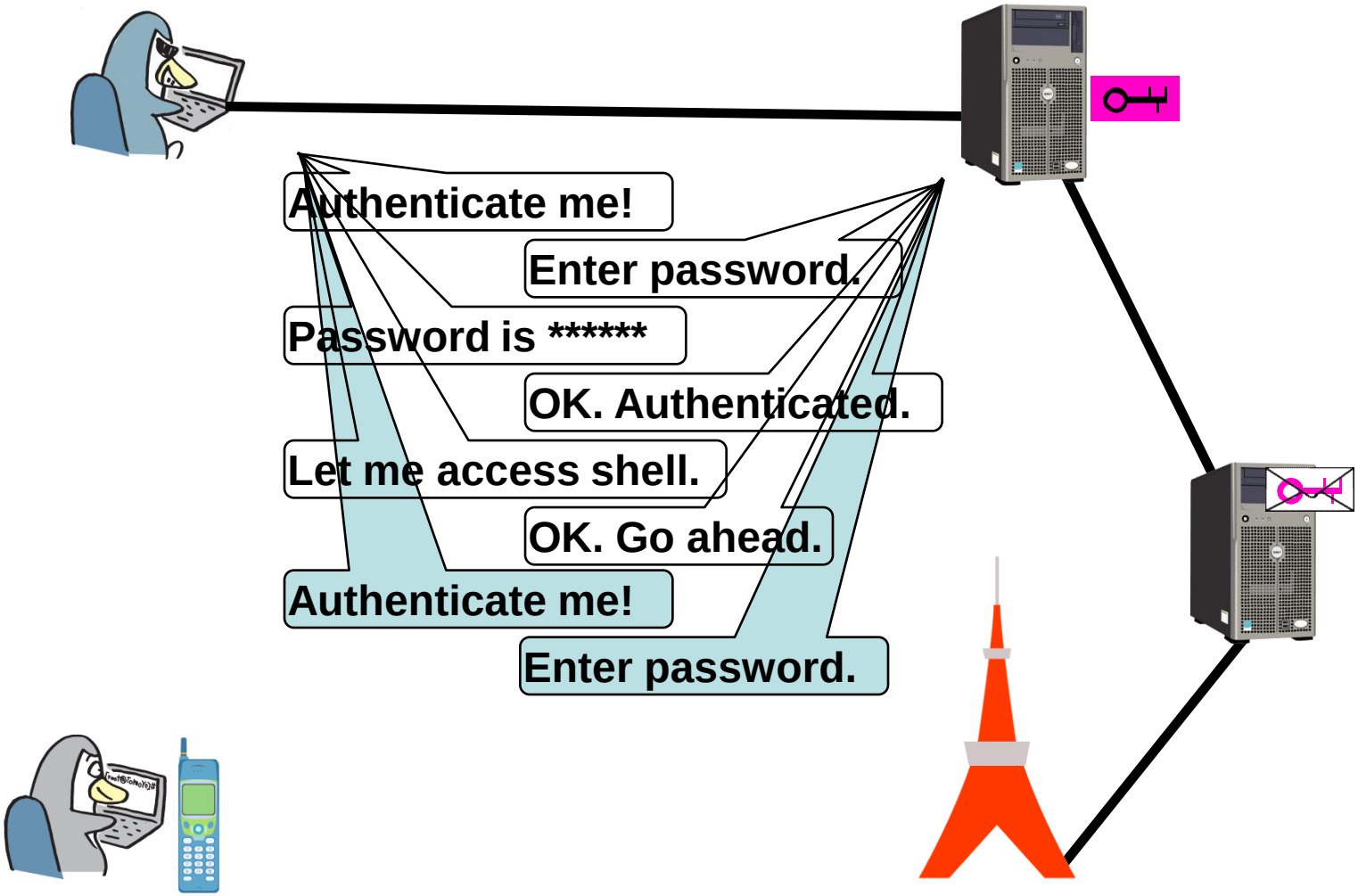
ケース2：対話型シェルセッション



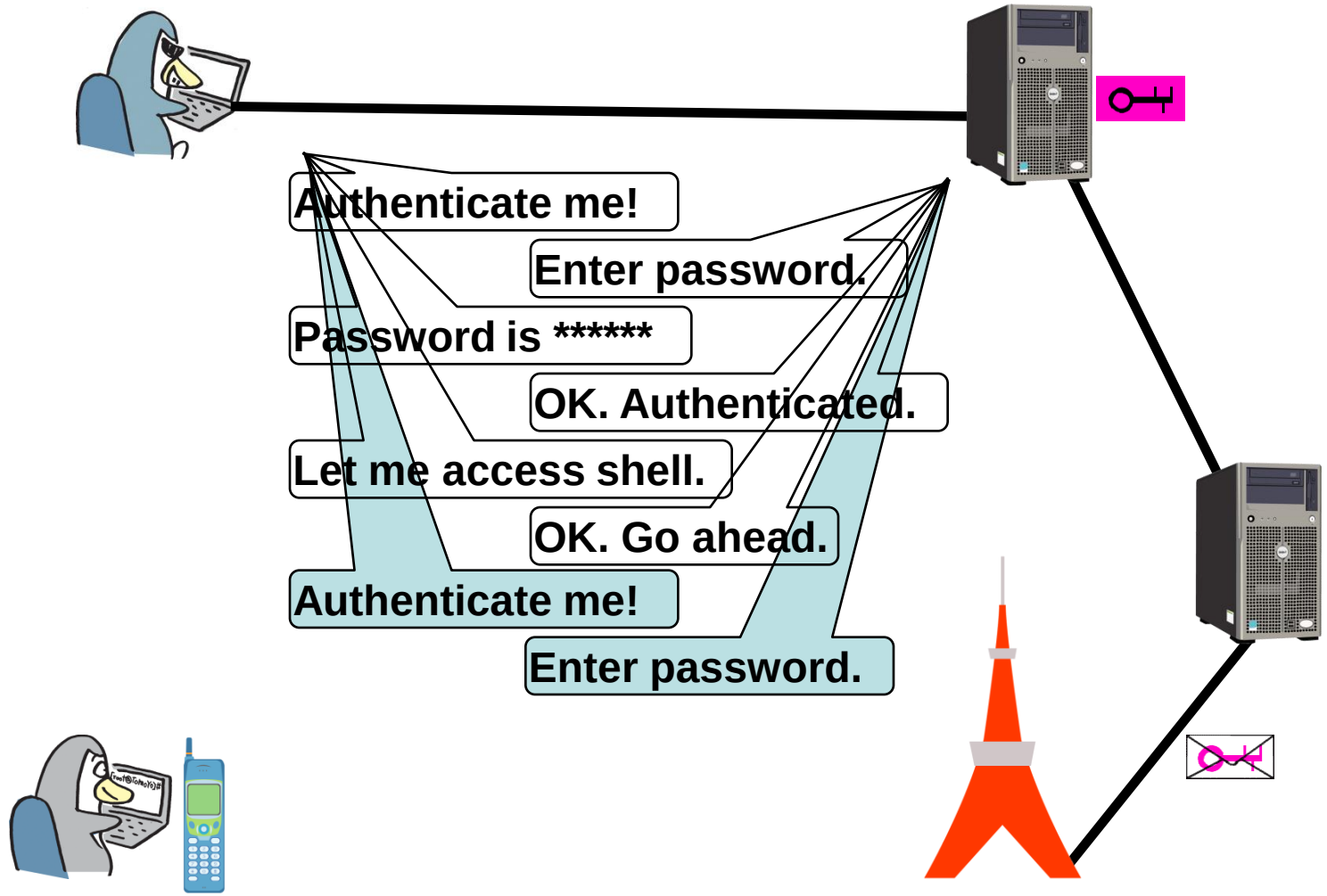
ケース2：対話型シェルセッション



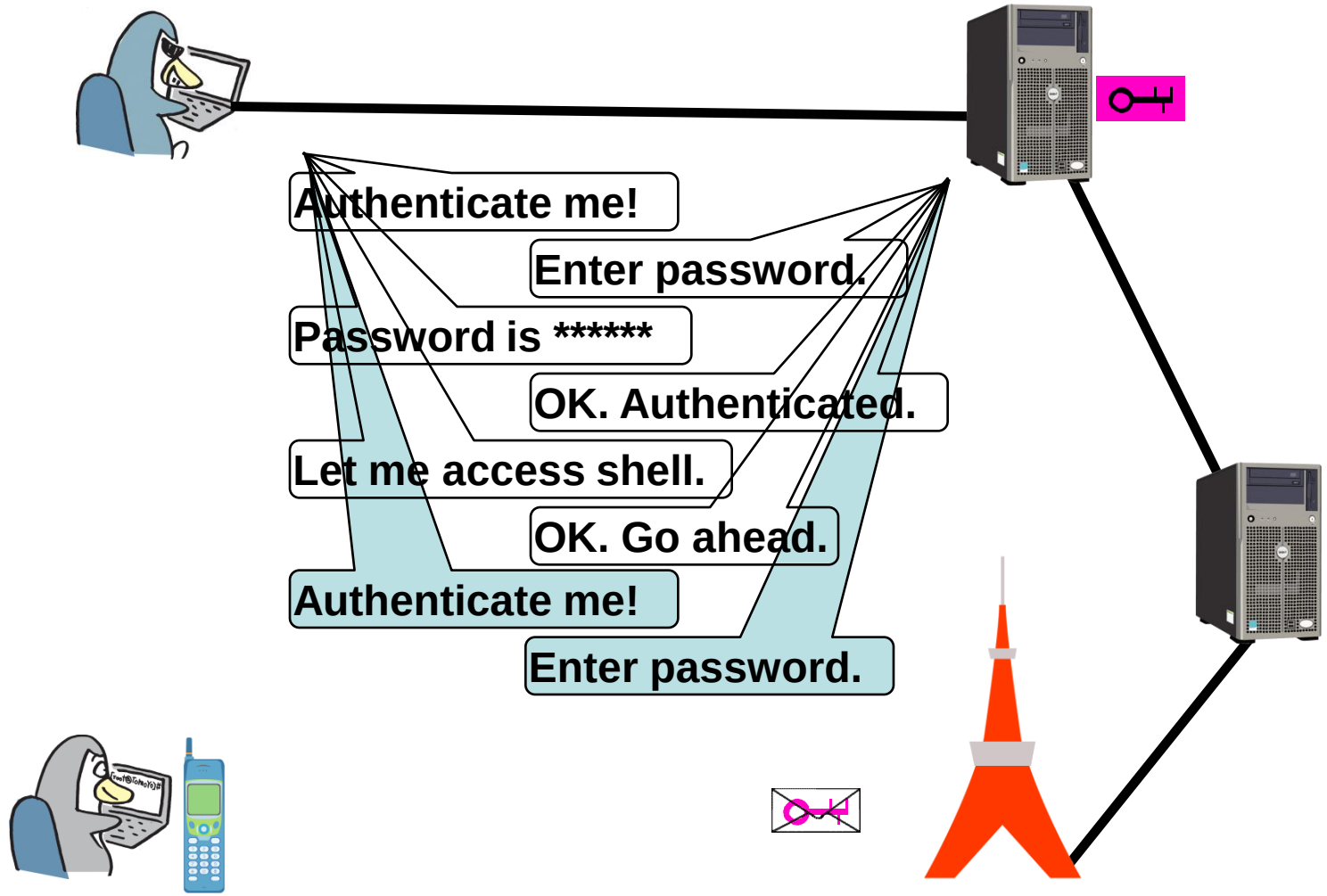
ケース2：対話型シェルセッション



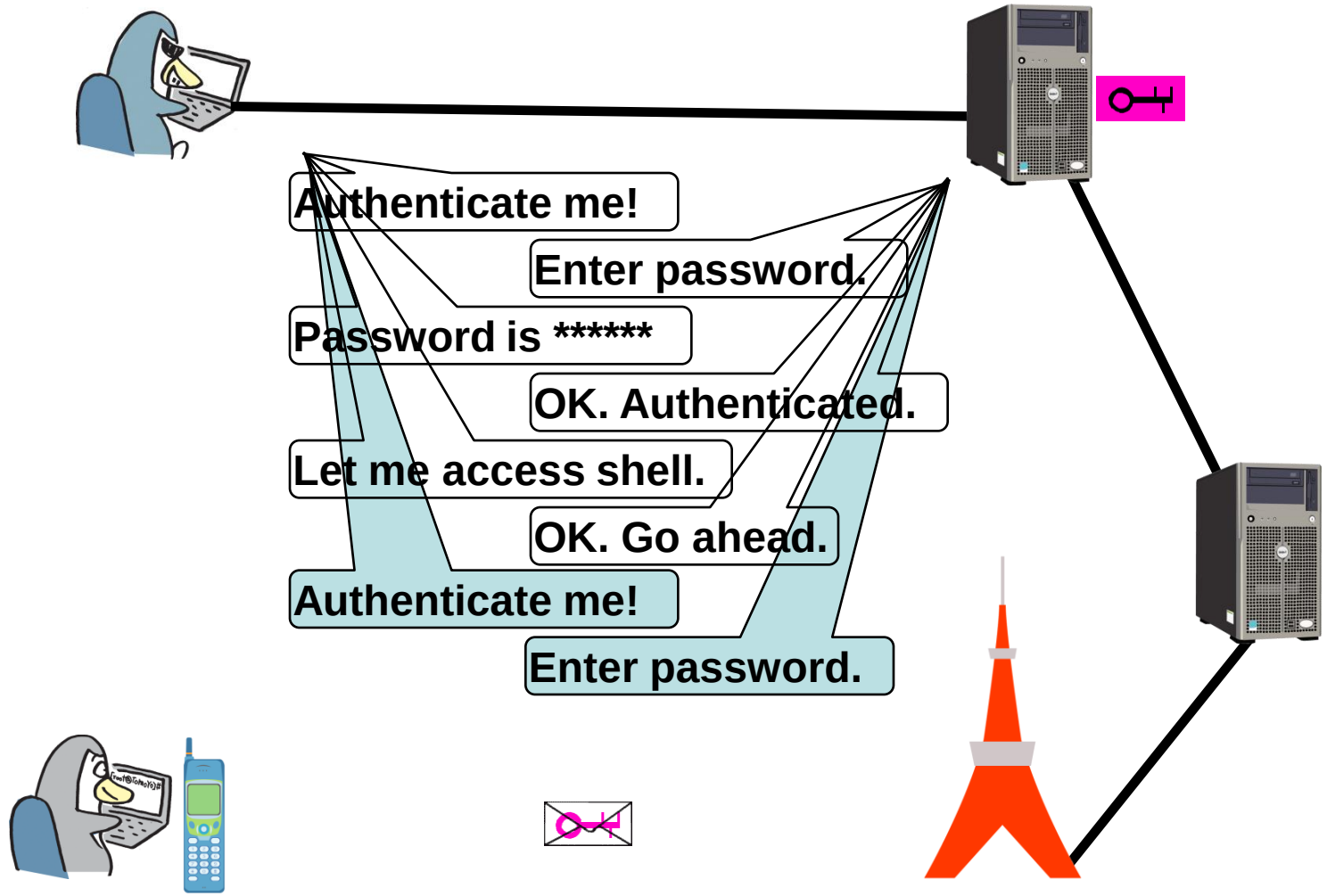
ケース2：対話型シェルセッション



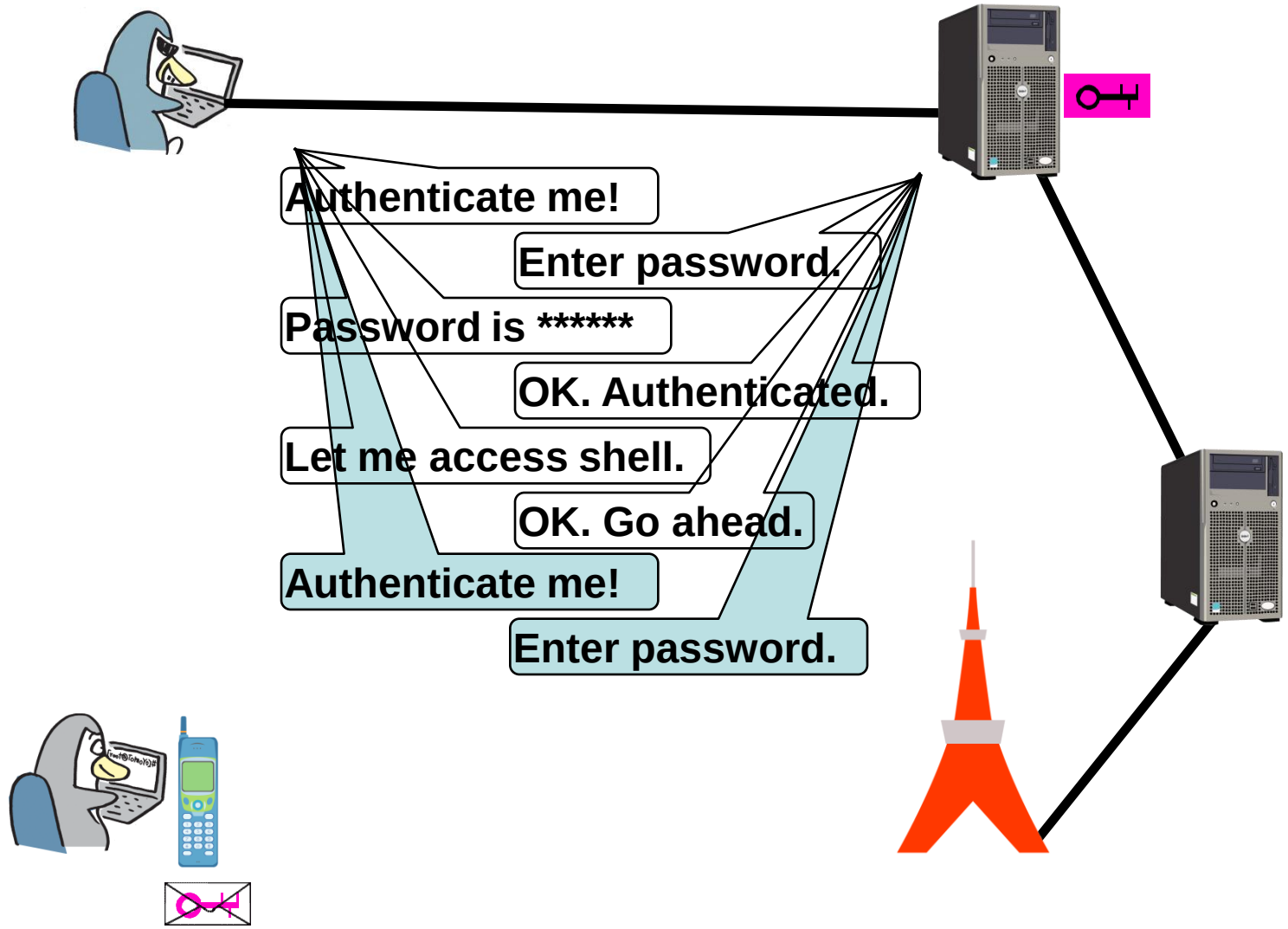
ケース2：対話型シェルセッション



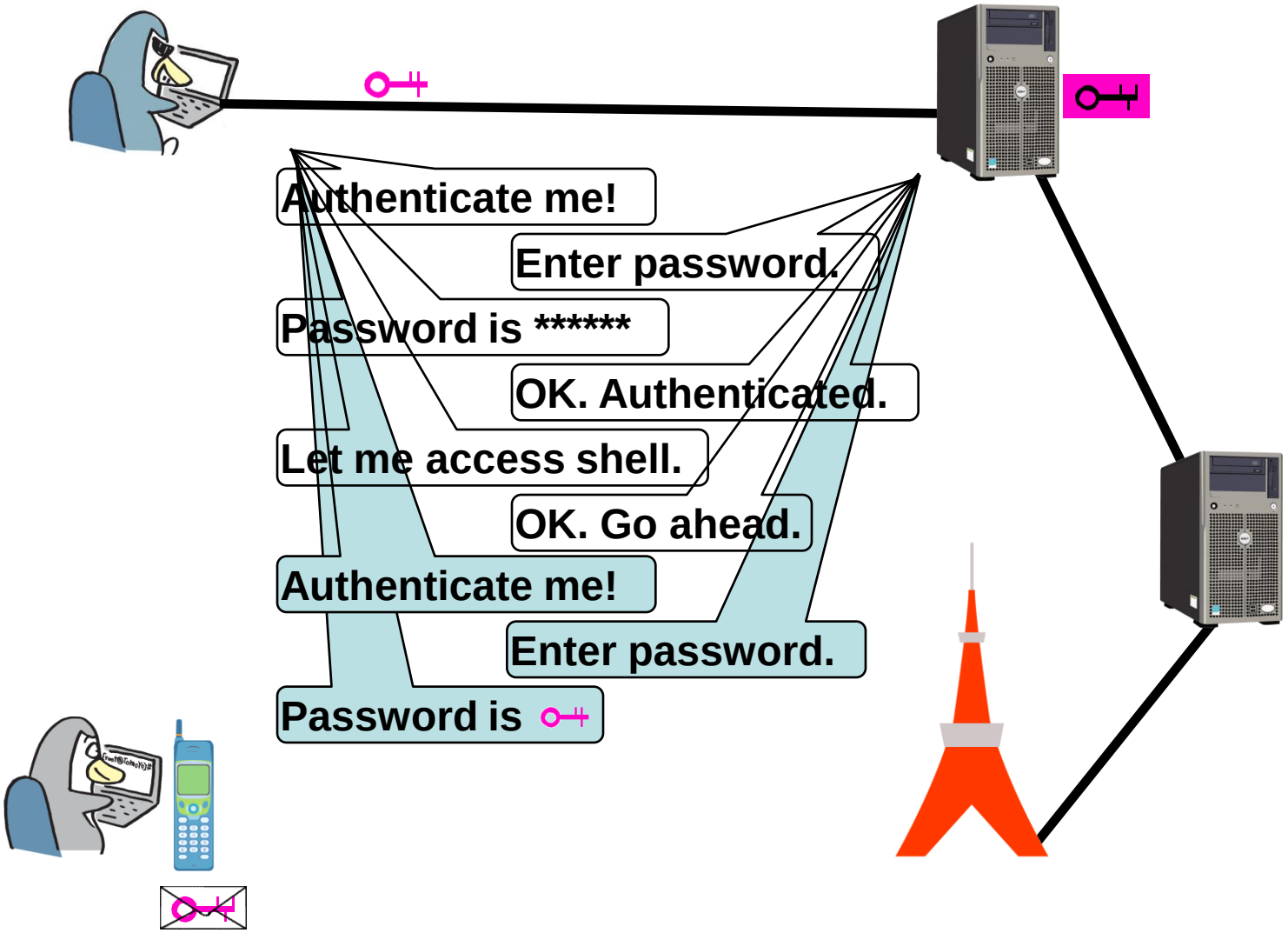
ケース2：対話型シェルセッション



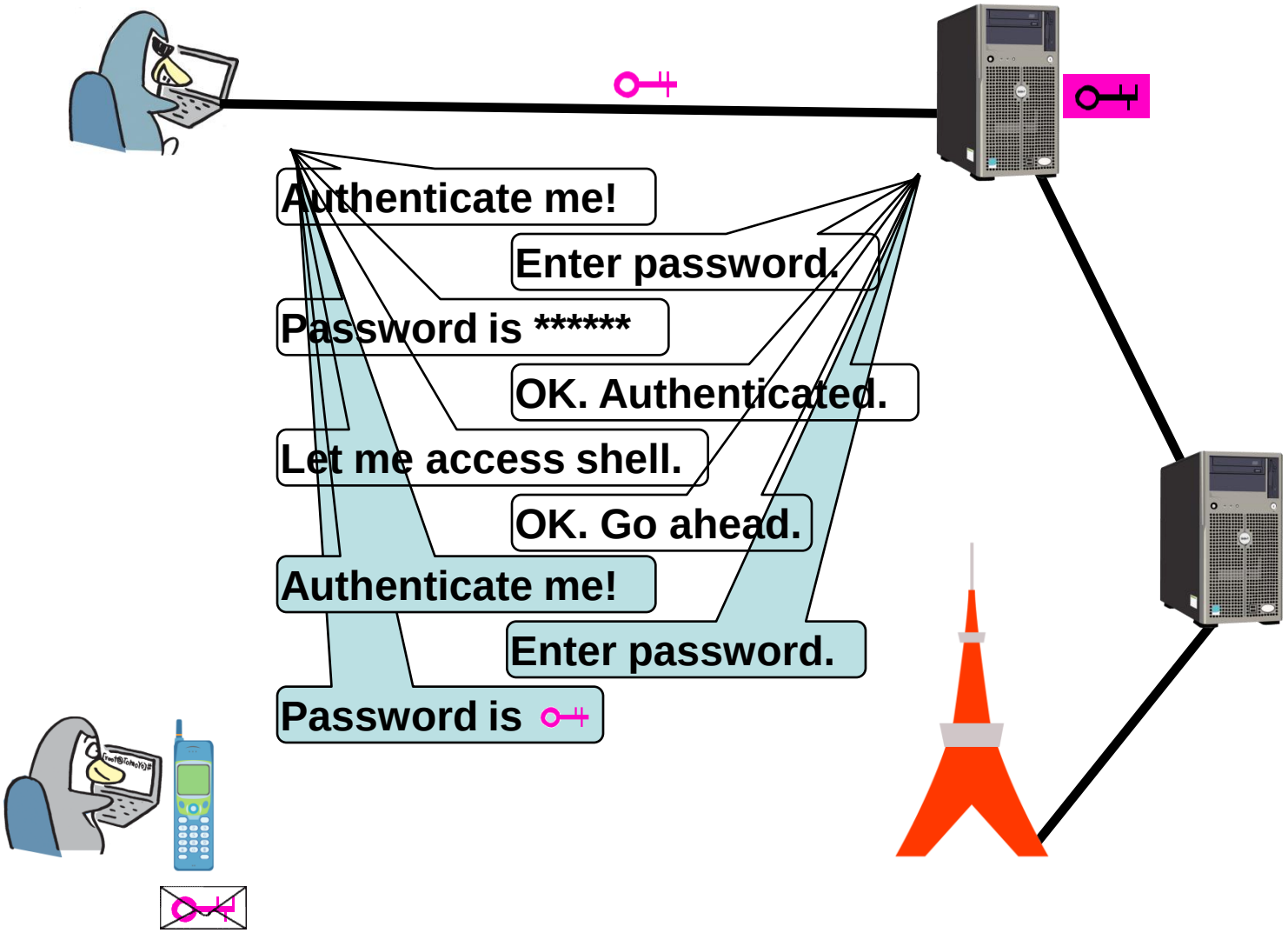
ケース2：対話型シェルセッション



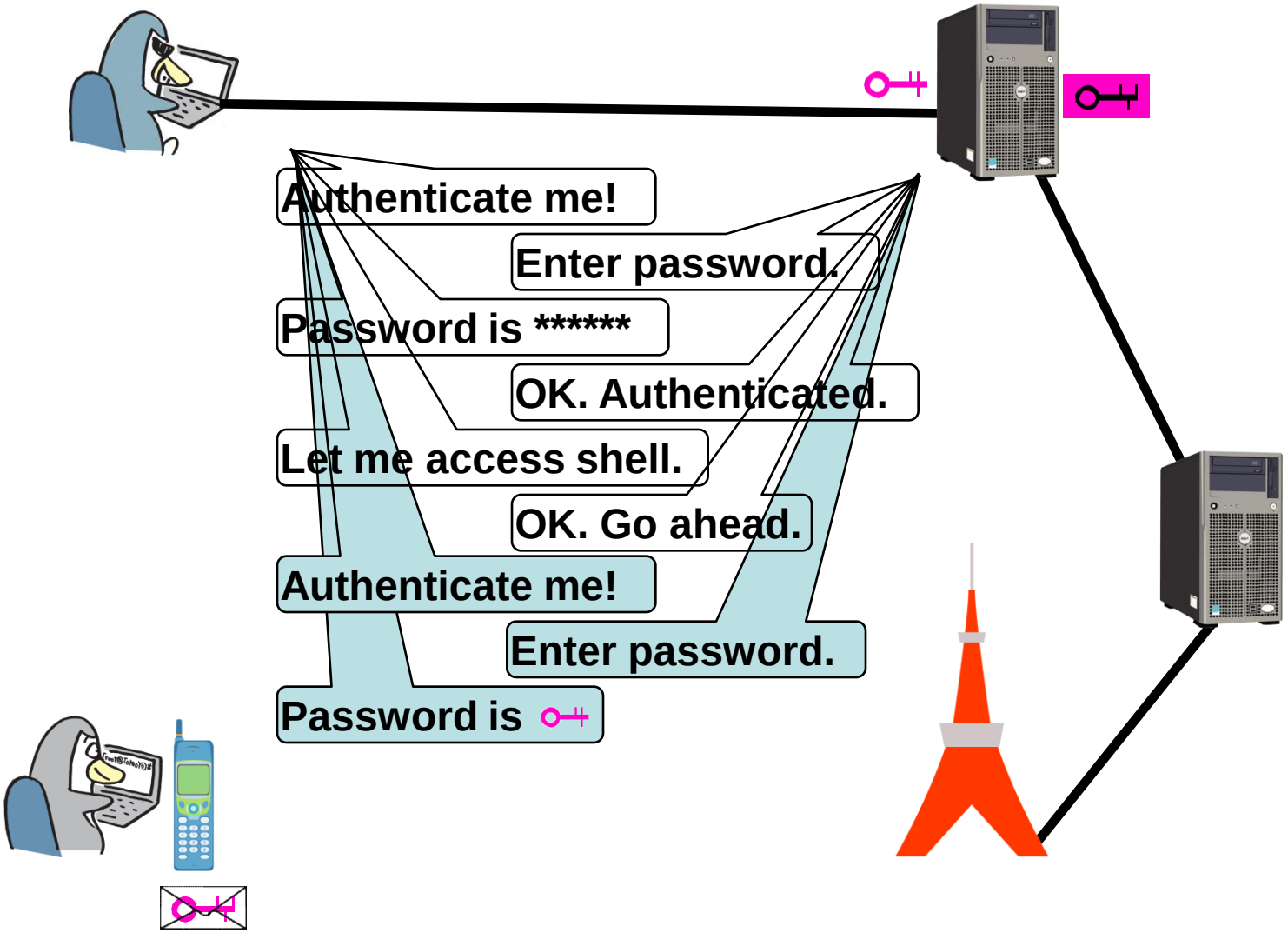
ケース2：対話型シェルセッション



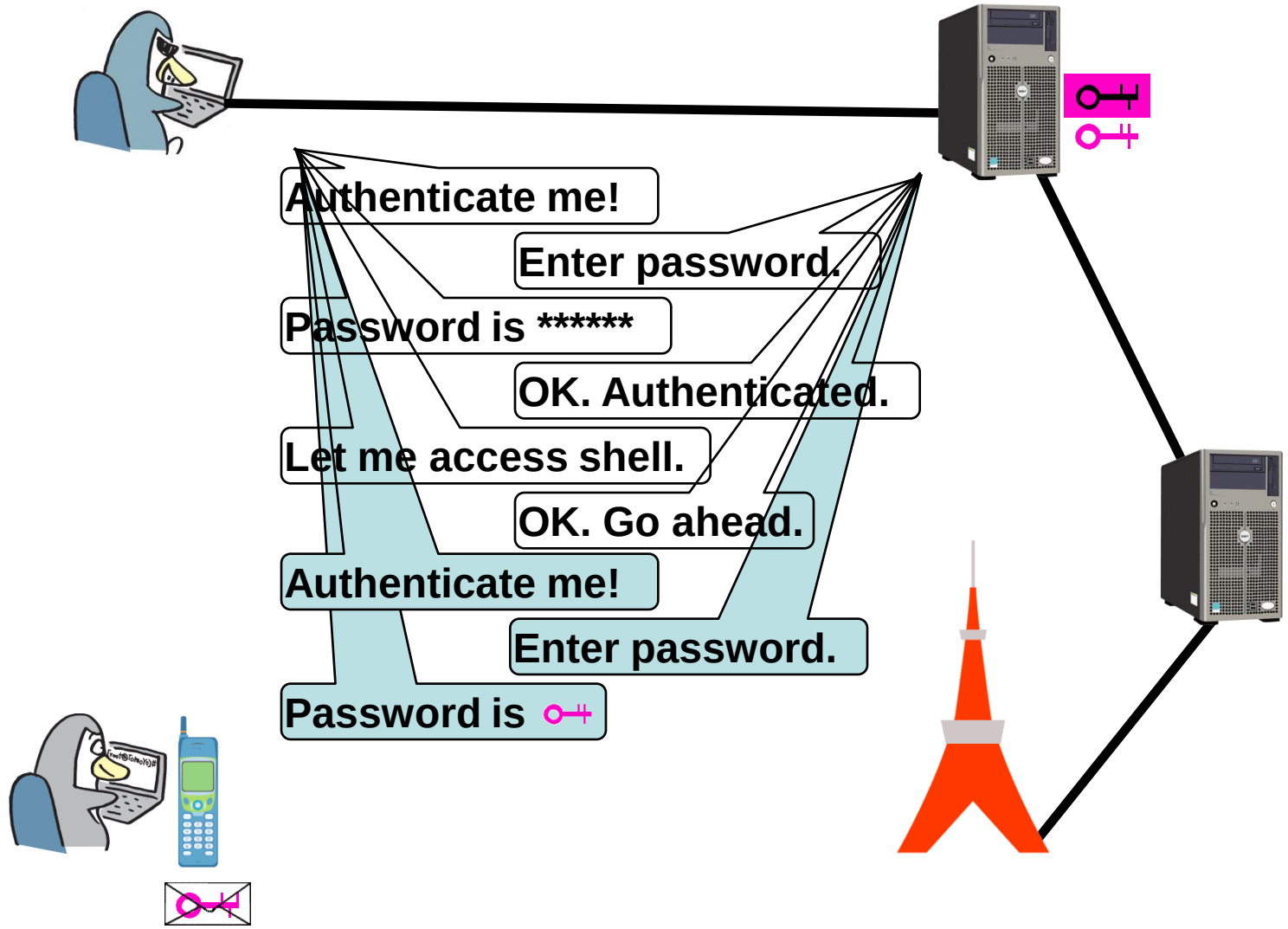
ケース2：対話型シェルセッション



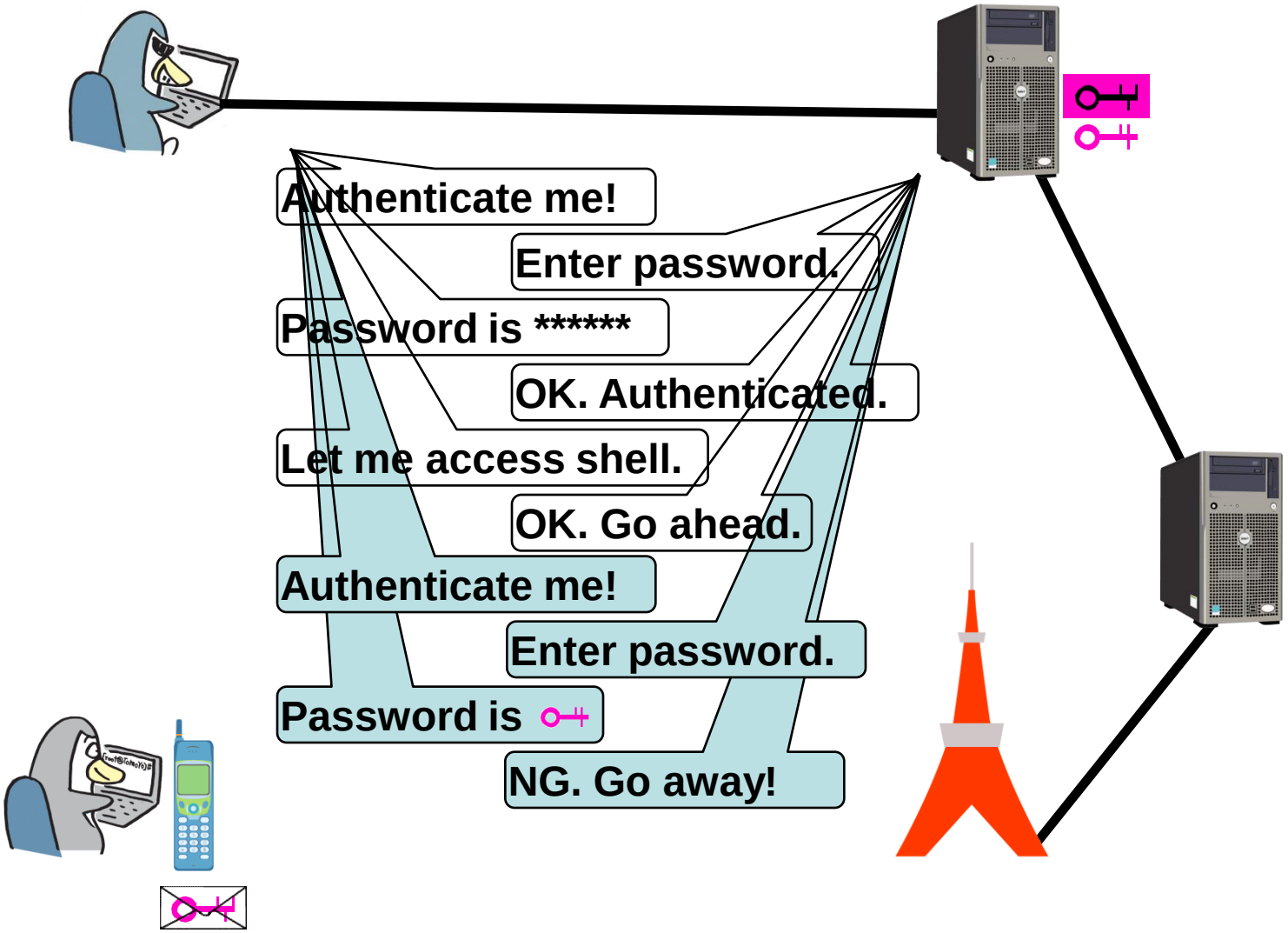
ケース2：対話型シェルセッション



ケース2：対話型シェルセッション



ケース2：対話型シェルセッション



ケース2：対話型シェルセッション

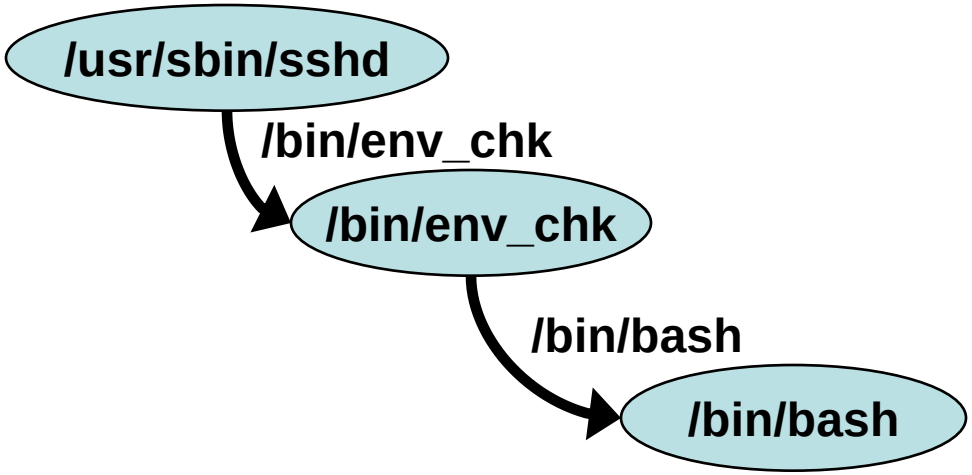
- 利点
  - ワンタイムパスワードを生成したプロセス自身が検証も行う
    - 時刻やカウンタなどの同期機構が不要
    - プロセスの消滅と同時にワンタイムパスワードも失効
  - ワンタイムパスワードが侵入者以外に漏洩しても問題にならない
    - プロセスを生成したユーザ以外には何の価値もない

ケース2：対話型シェルセッション

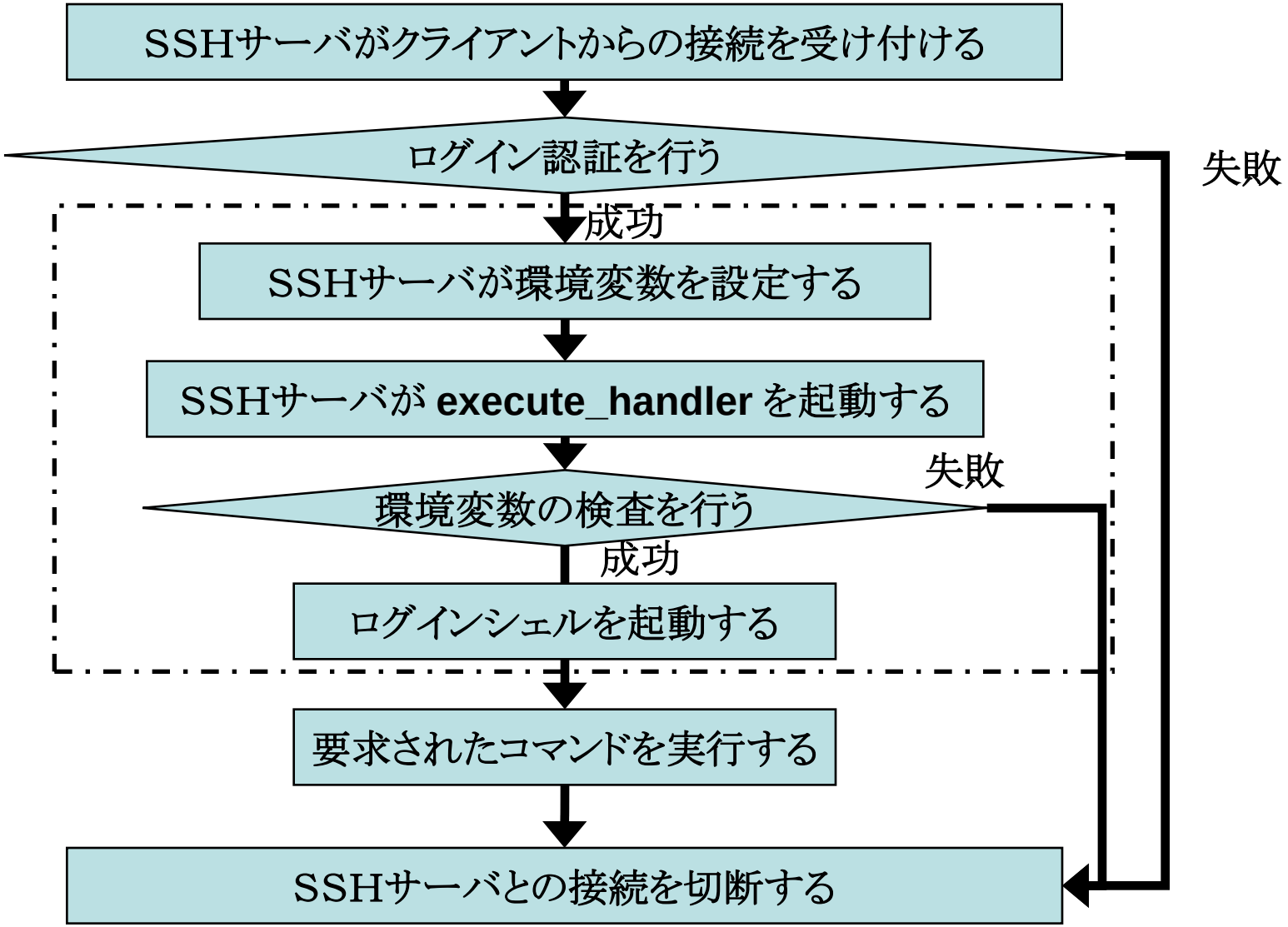
- 難点
  - メールを受信できる必要がある
    - 携帯電話のようなメール受信端末を所持していること
  - メールを送信できる必要がある
    - SMTPサーバやWebサーバのメール送信CGIなどが利用可能なこと

ケース3：非対話的シェルセッション

- 環境変数の有無やその内容を利用します。
  - 利用するもの
    - SSH サーバの AcceptEnv ディレクティブ
    - SSH クライアントの SendEnv ディレクティブ
    - 自作プログラム /bin/env_check
    - TOMOYO Linux の execute_handler ディレクティブ

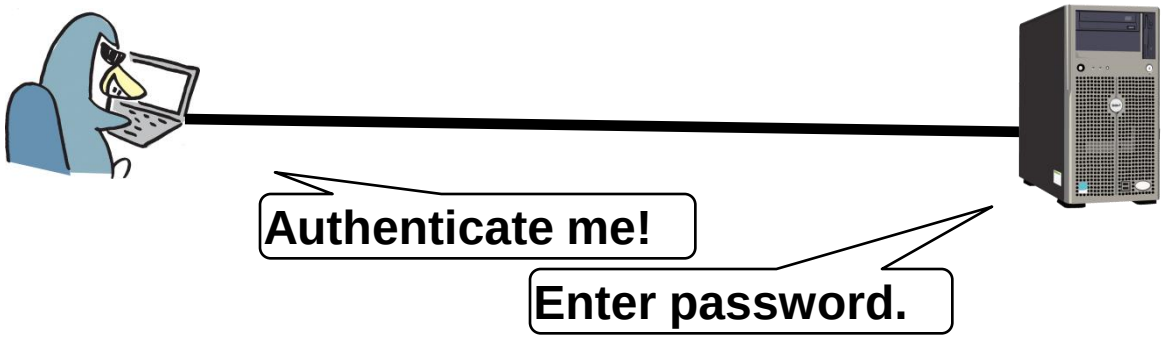


ケース3：非対話的シェルセッション

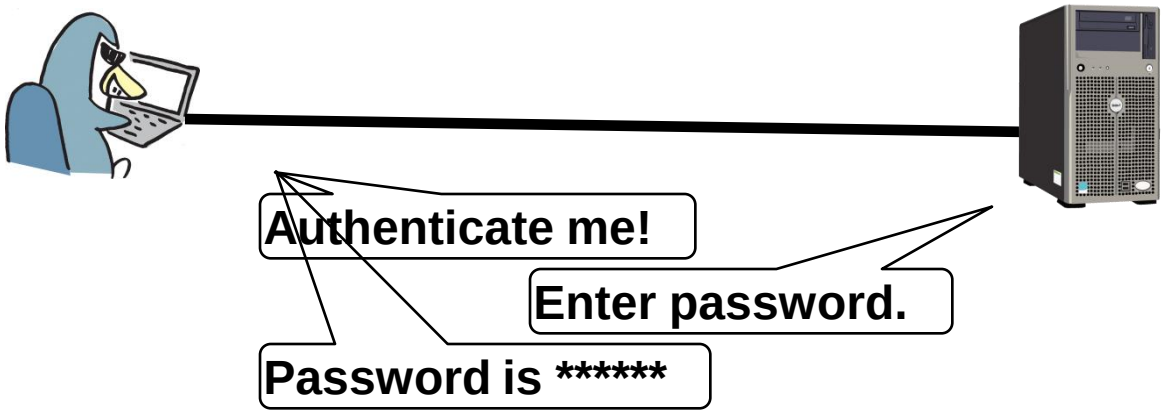


ケース3：非対話的シェルセッション

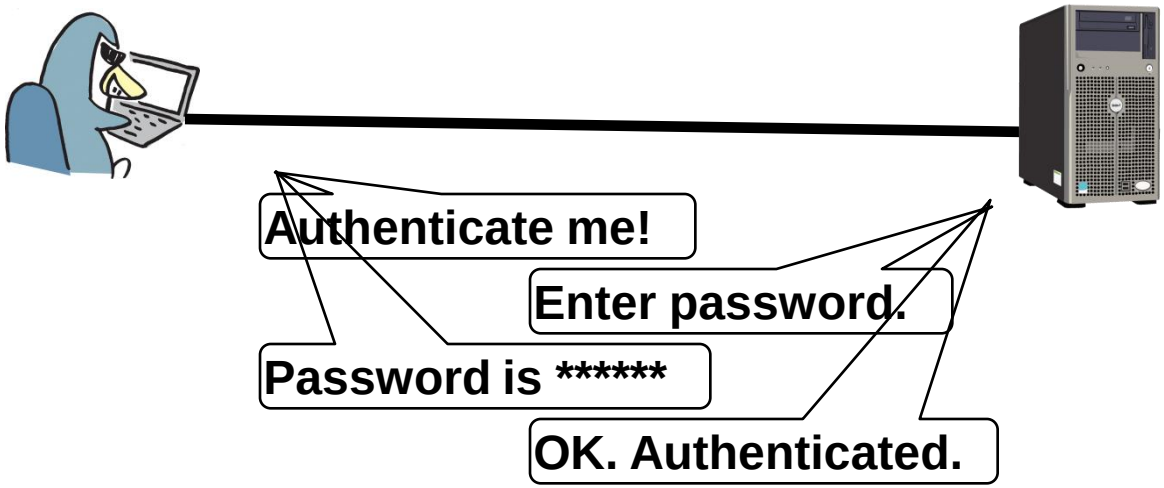
ケース3：非対話的シェルセッション



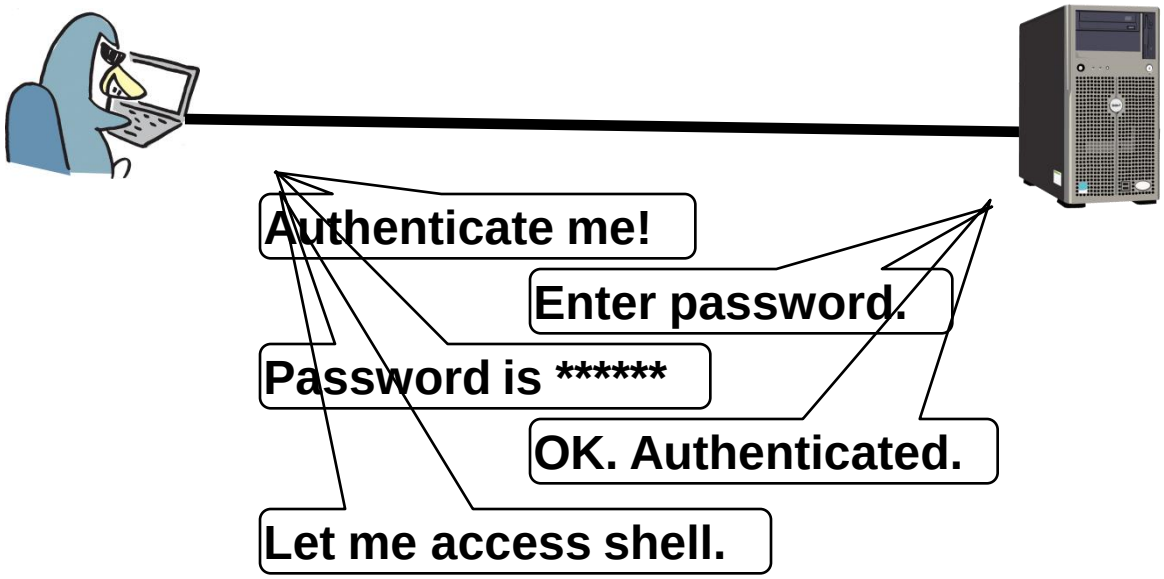
ケース3：非対話的シェルセッション



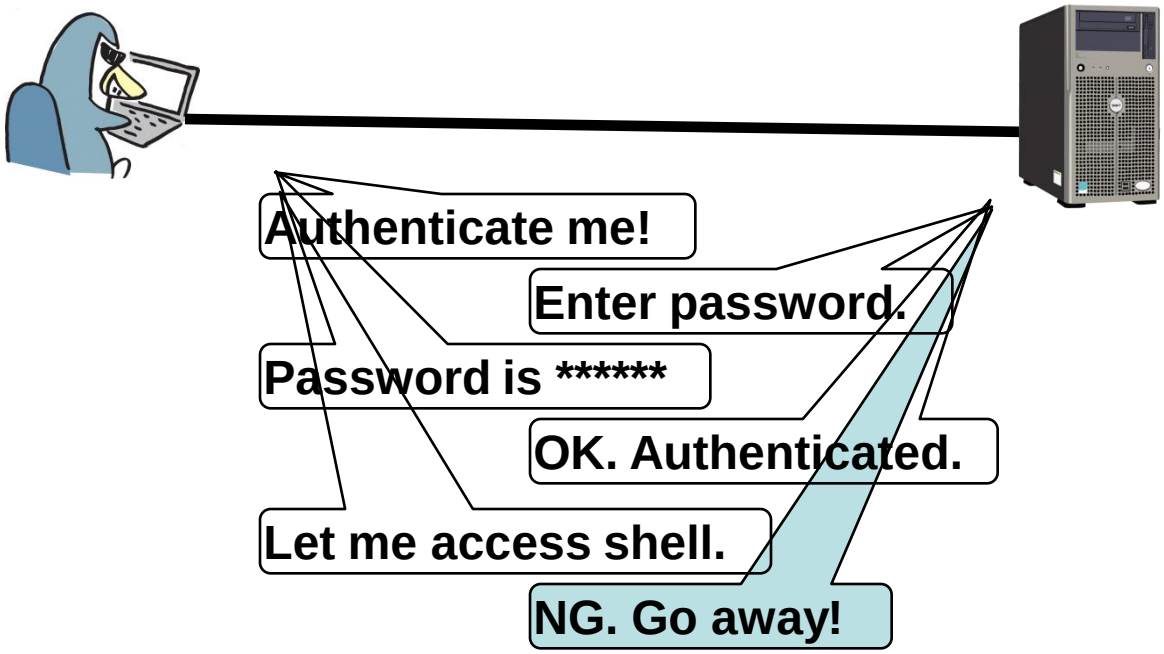
ケース3：非対話的シェルセッション



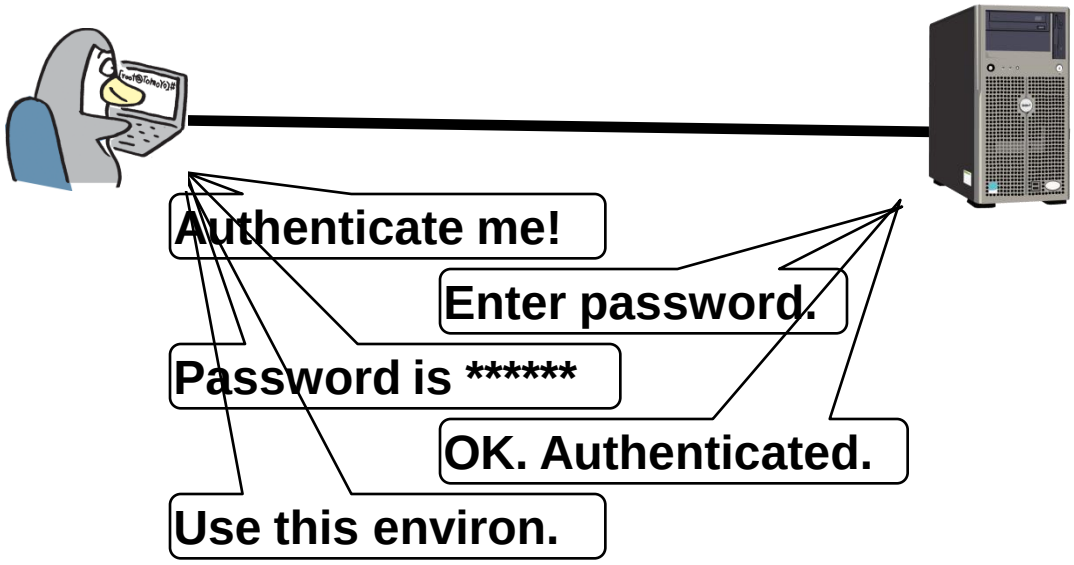
ケース3：非対話的シェルセッション



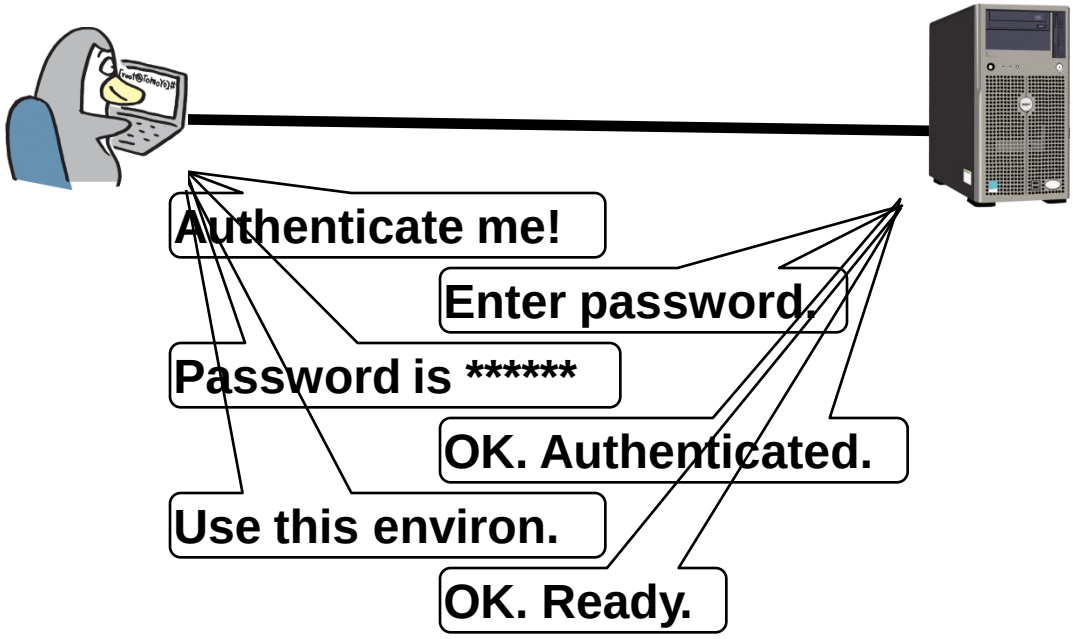
ケース3：非対話的シェルセッション



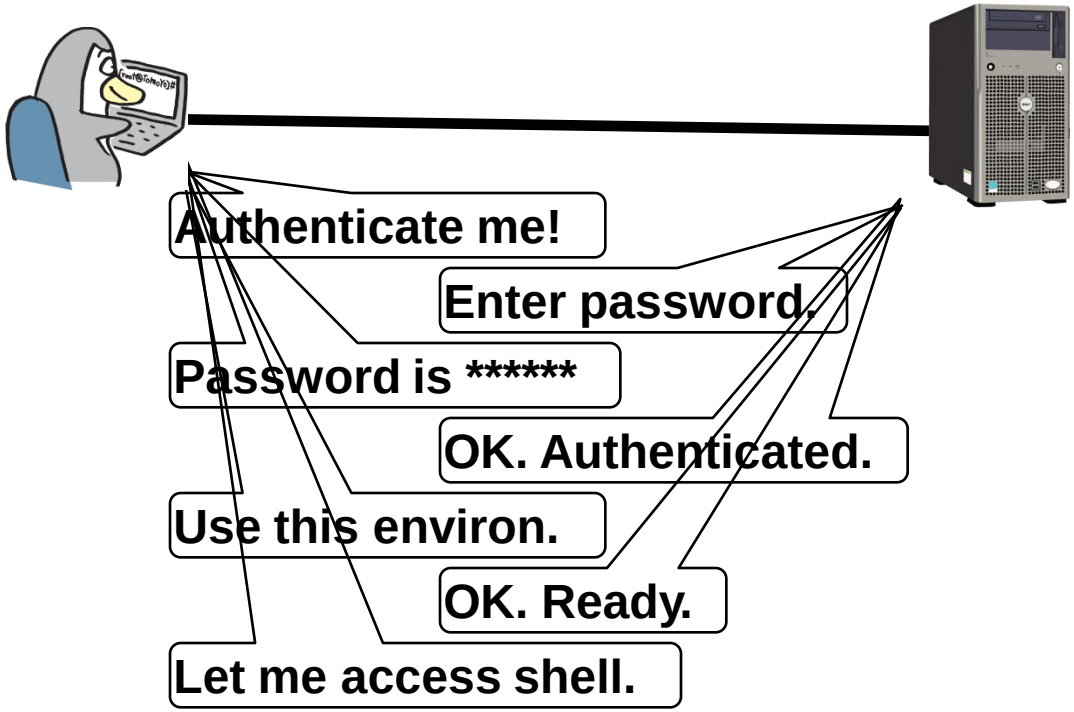
ケース3：非対話的シェルセッション



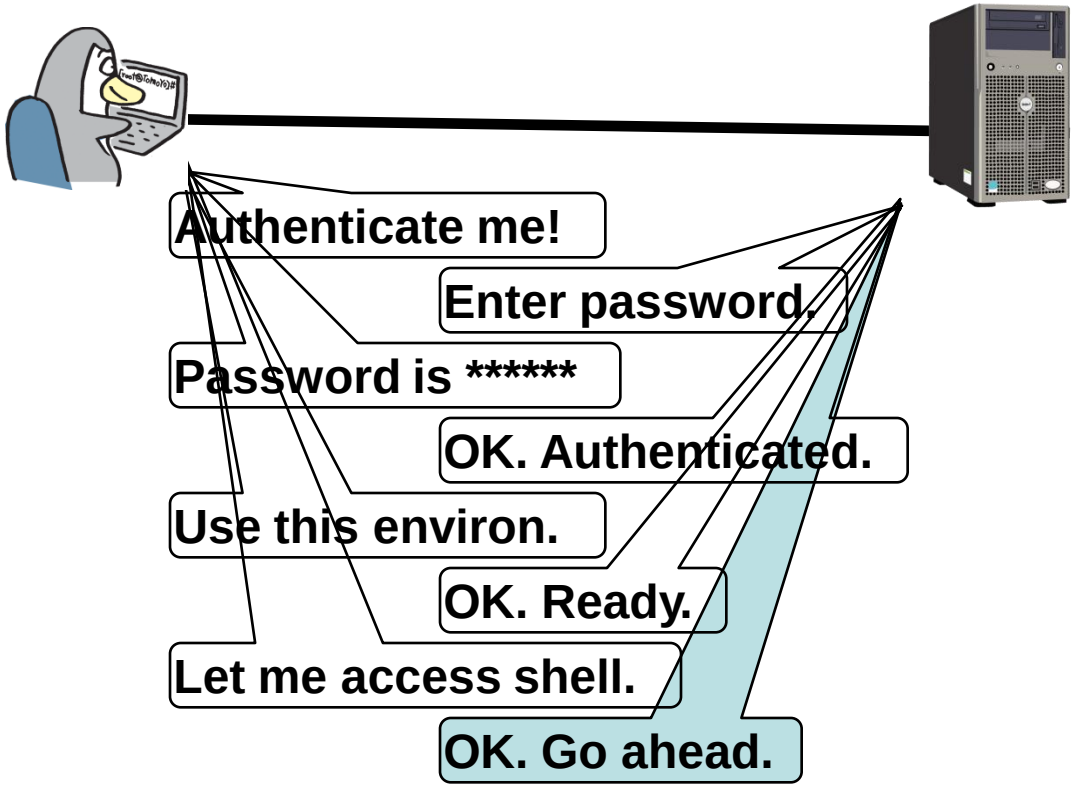
ケース3：非対話的シェルセッション



ケース3：非対話的シェルセッション



ケース3：非対話的シェルセッション



ケース3：非対話的シェルセッション

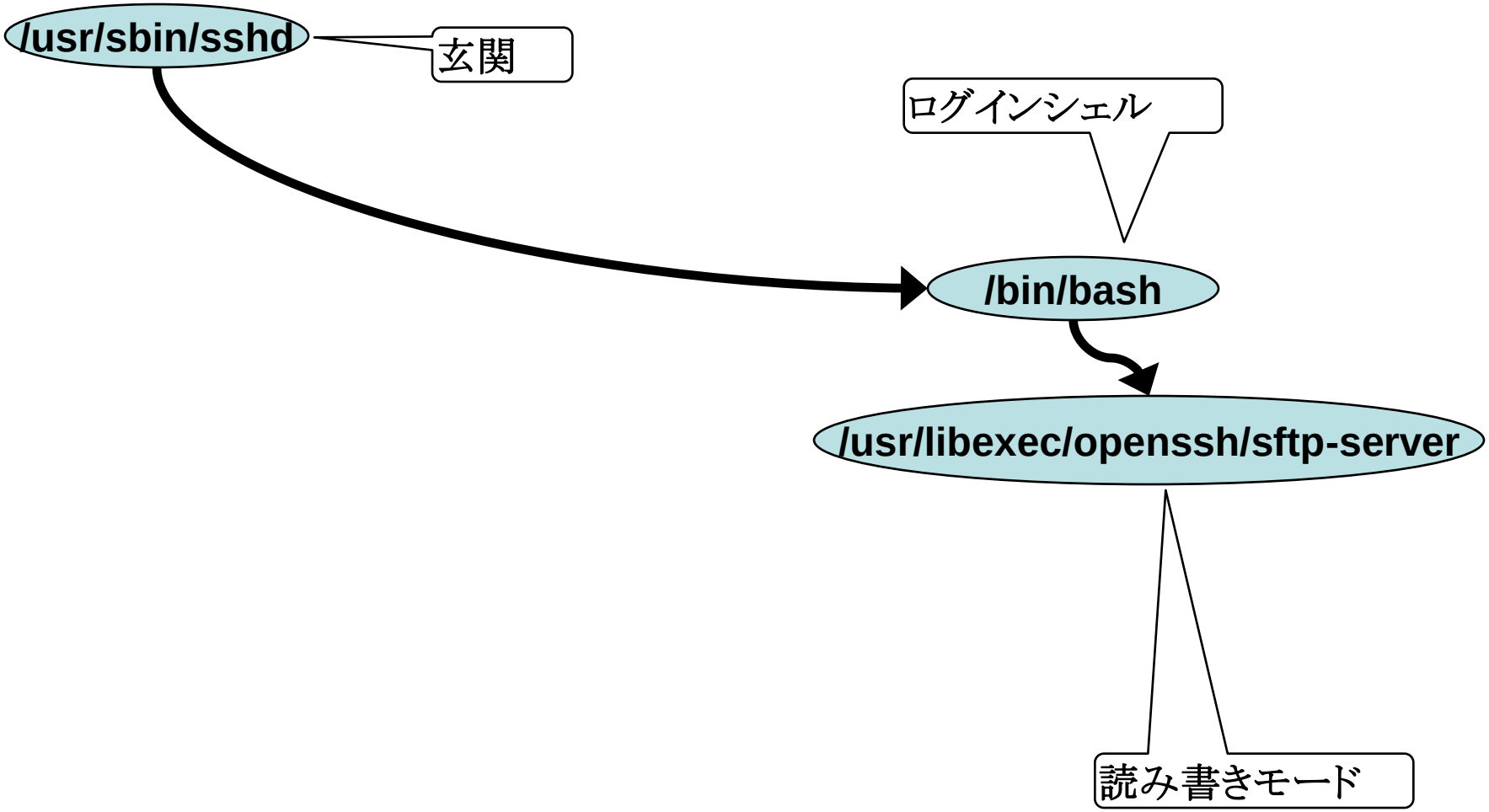
- 利点
  - クライアントからは透過に見える
    - コマンドラインでの操作が不要
    - 標準入出力の扱いが不変
  - 環境変数をパスワードの代わりに使える
    - 環境変数名を秘密にできる
    - 環境変数の内容に応じて権限を分割することも可能
  - 対話的シェルセッションを開始する前に適用することも可能

ケース3：非対話的シェルセッション

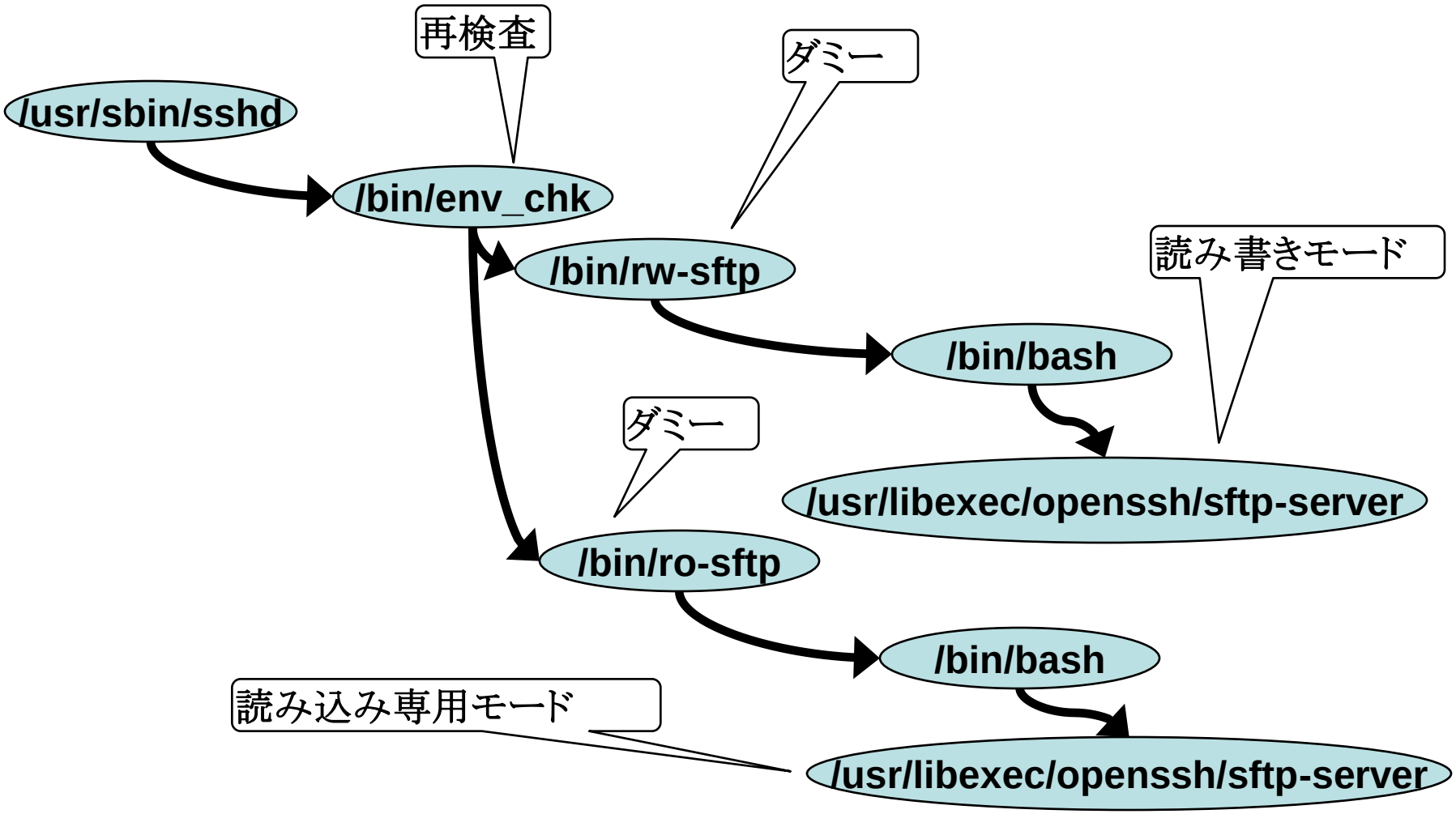
- 難点
  - TOMOYO Linux 専用
    - execute_handler を備えているのは TOMOYO Linux だけ
  - SSHクライアントが環境変数を送信する機能（SendEnv）をサポートしていない可能性

ケース3:非対話的シェルセッション

・ 応用例:環境変数による権限分割



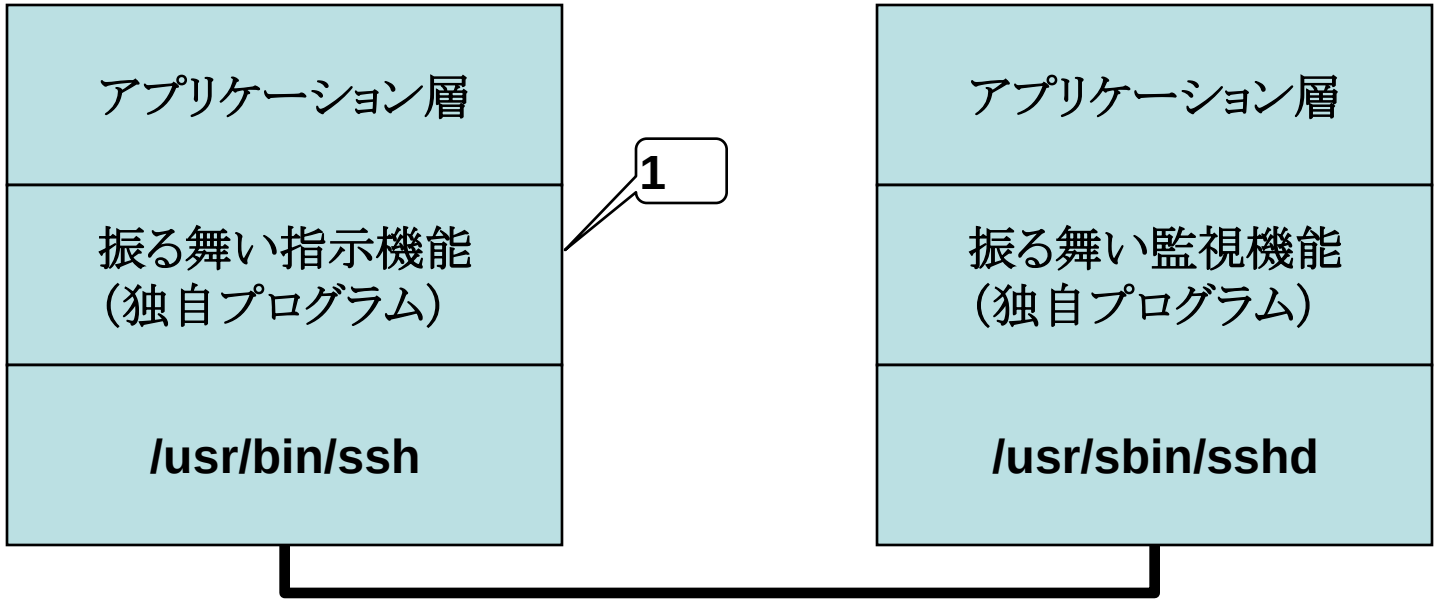
ケース3：非対話的シェルセッション



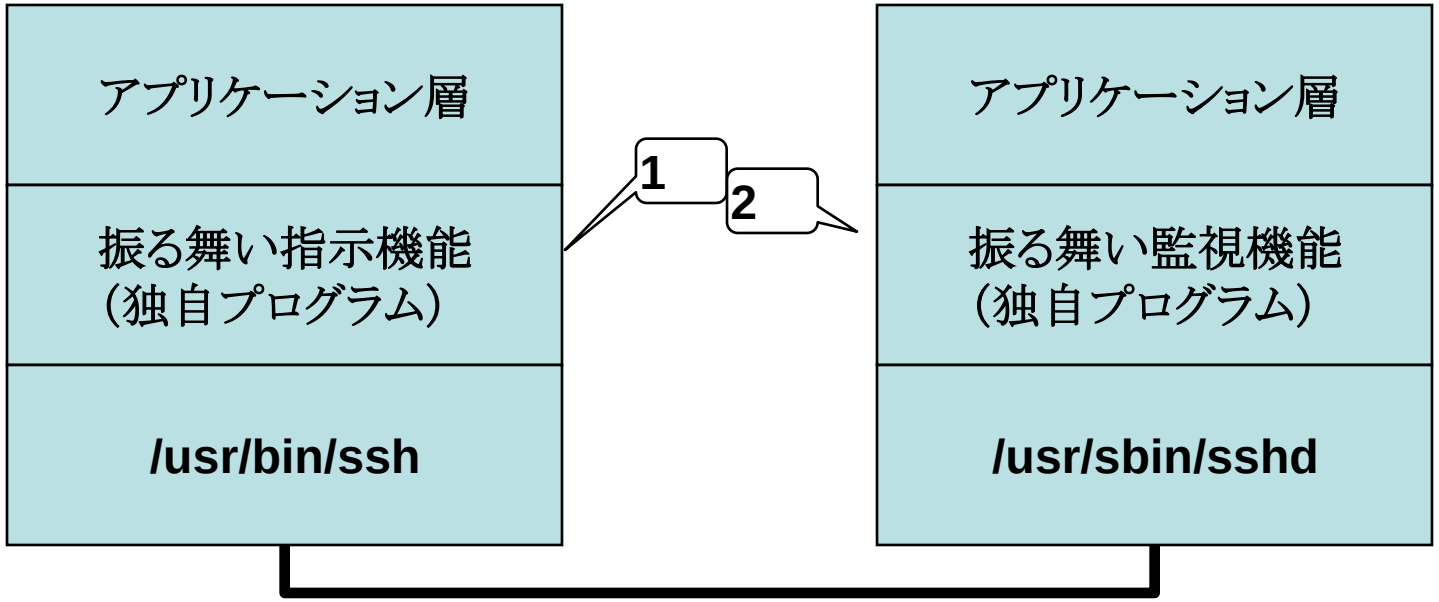
ケース4：非対話的シェルセッション

- 独自のレイヤーを構築します。
  - 利用するもの
    - scp や sftp コマンドの -S オプション
    - 振る舞いを監視するためのサーバ側プログラム
    - 振る舞いを指定するためのクライアント側プログラム
    - TOMOYO Linux の execute_handler ディレクティブ

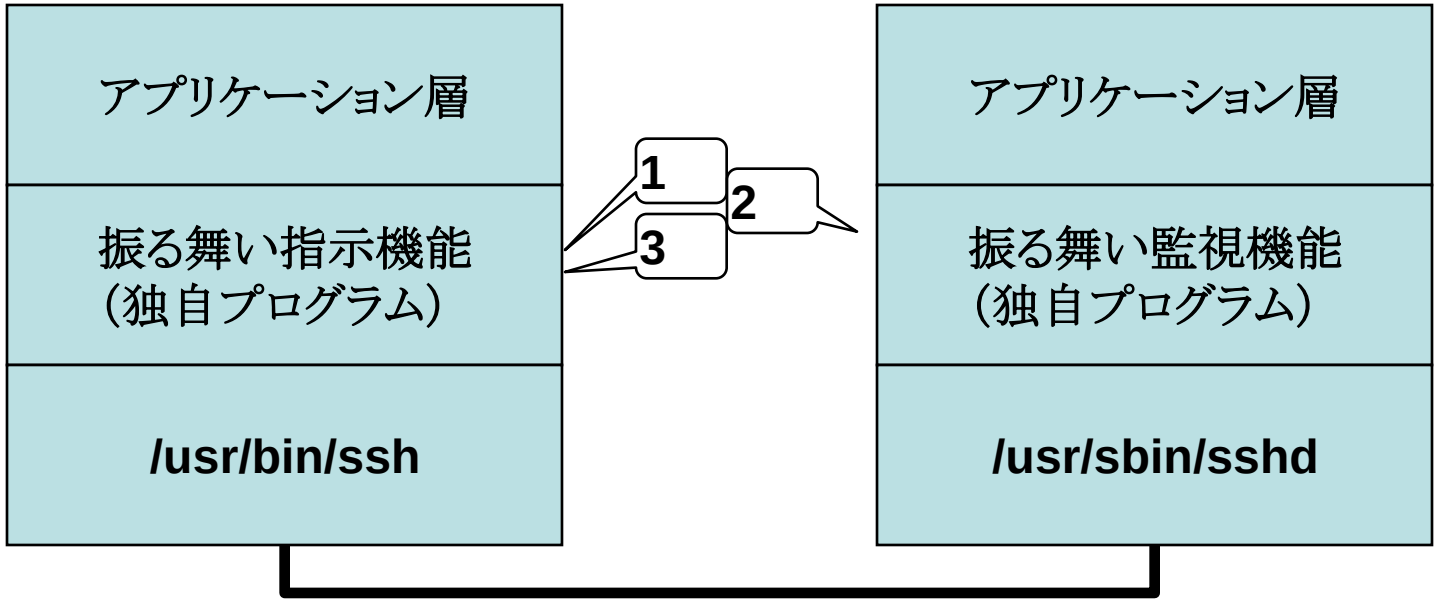
ケース4：非対話的シェルセッション



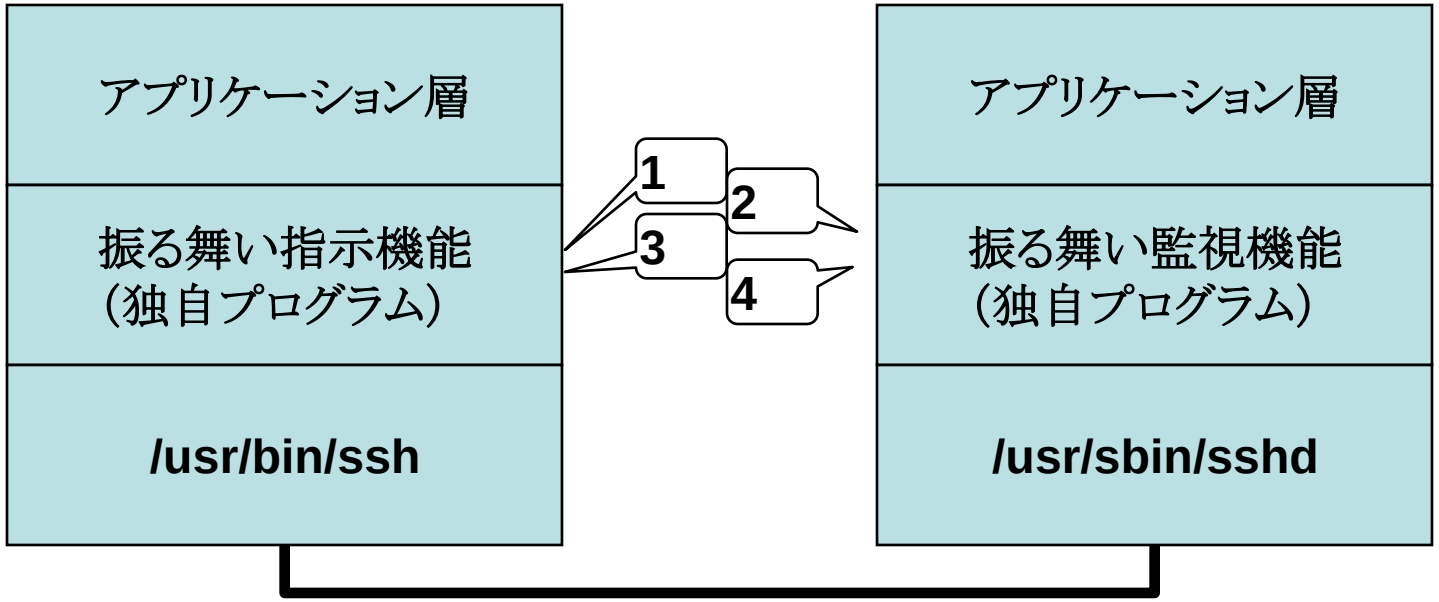
ケース4：非対話的シェルセッション



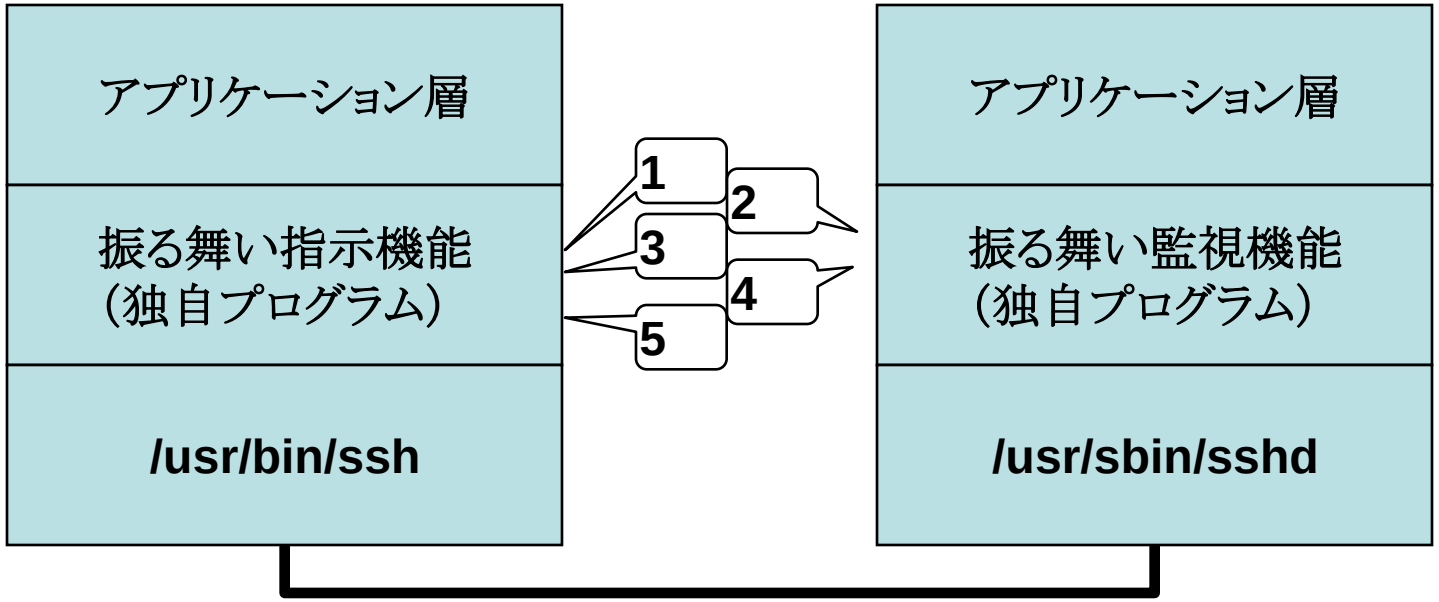
ケース4：非対話的シェルセッション



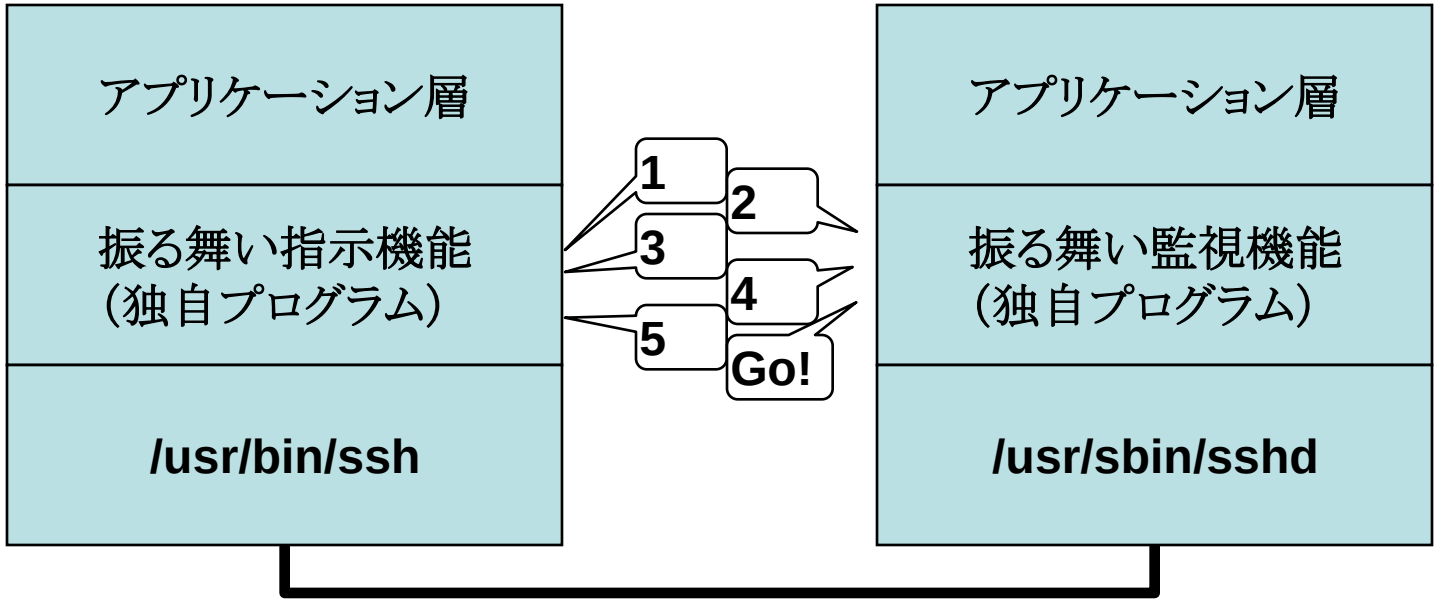
ケース4：非対話的シェルセッション



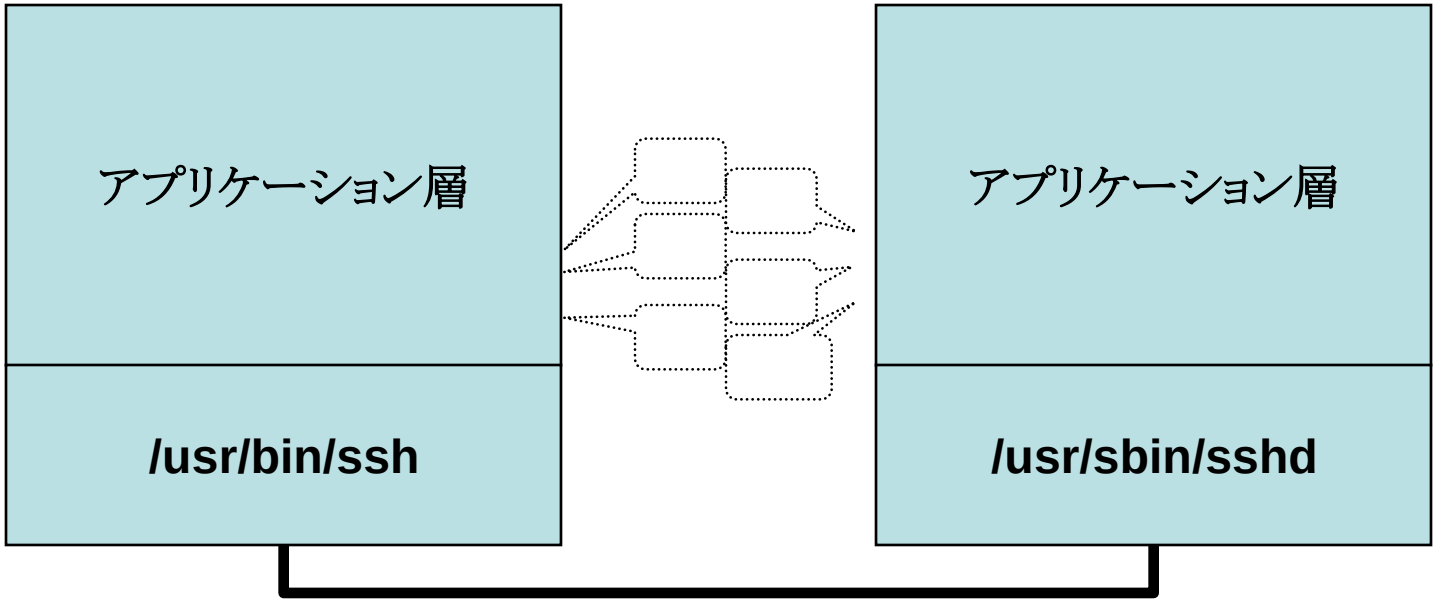
ケース4：非対話的シェルセッション



ケース4：非対話的シェルセッション



ケース4：非対話的シェルセッション



ケース4：非対話的シェルセッション

- 利点
  - 標準の機能で利用できない要素（標準入出力やコマンドラインパラメータなど）を使える
  - 環境変数と組み合わせて利用できる
    - 独自認証対応の可否に応じた権限分割ができる

ケース4：非対話的シェルセッション

- 難点
  - クライアント側にもプログラムを用意する必要がある。

PAMではできないの？

- 自由度・難易度が違います。
  - 標準入出力やパラメータなどを独占できるので、他のPAMモジュールとの干渉が起こりません。
- 誰にでも作ることができます。
  - RFCなどの標準に縛られません。

PAMではできないの？

- クライアント側の対応が不要です。
  - PAMの場合はクライアントが対応していないと使えません。
  - PAMを通過後に実行される処理（シェルなど）に対応していないクライアントはありえません。

PAMではできないの？

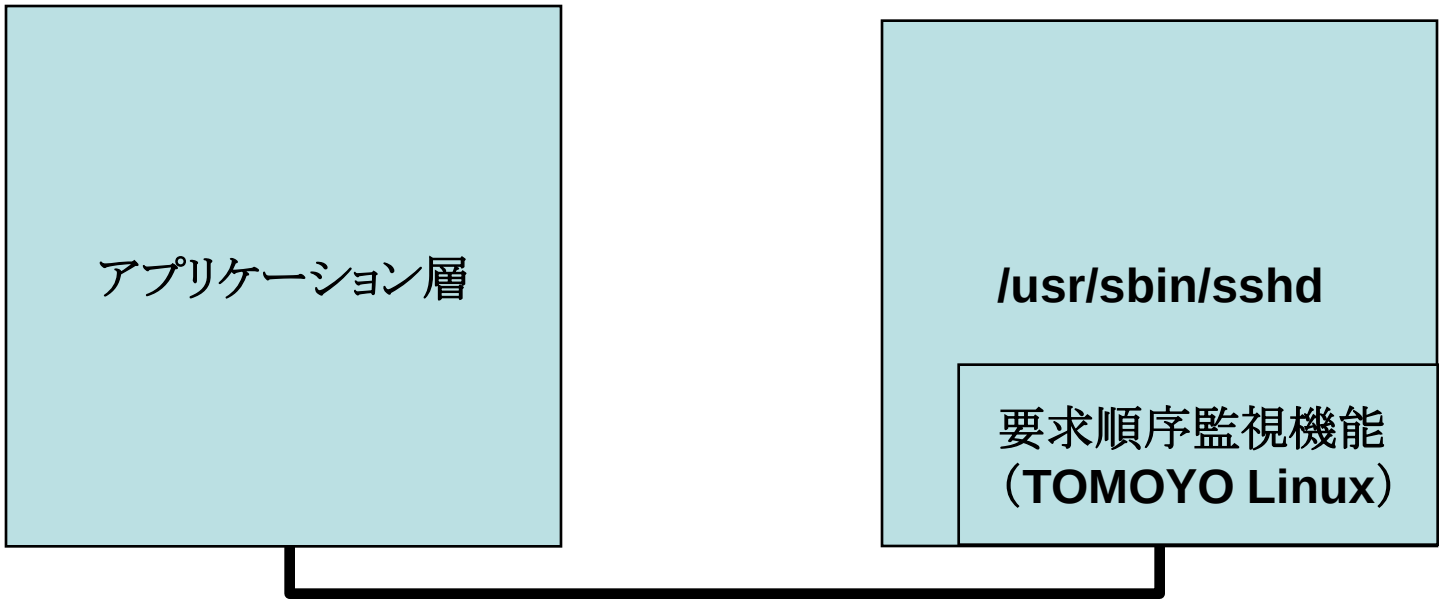
- 強制することができます。
  - 他のPAMモジュールの設定や実行結果により省略される心配がありません。
  - 生じうる状態遷移がMACにより規定されているので、想定外の抜け道（バッファオーバーフローやコマンドインジェクション）の心配がありません。
  - 外部プログラムの助けを借りるのが容易です。

まとめ

- ホスト内での認証なので独自プロトコルを利用可能です。
  - アイデアの勝負です。
- 利用可能な要素は無限にあります。
  - 正しい振る舞い(状態遷移)を知られない限りブルートフォースは不可能です。
- 安価に構築でき、負担の少ない方法を選べます。

ケース5：非シェルセッション

- クライアントプログラムをカスタマイズします。
  - 利用する機能
    - 独自のSSHクライアントプログラム（例えば JSCH）
    - TOMOYO Linux の task.state キーワード



ケース5：非シェルセッション

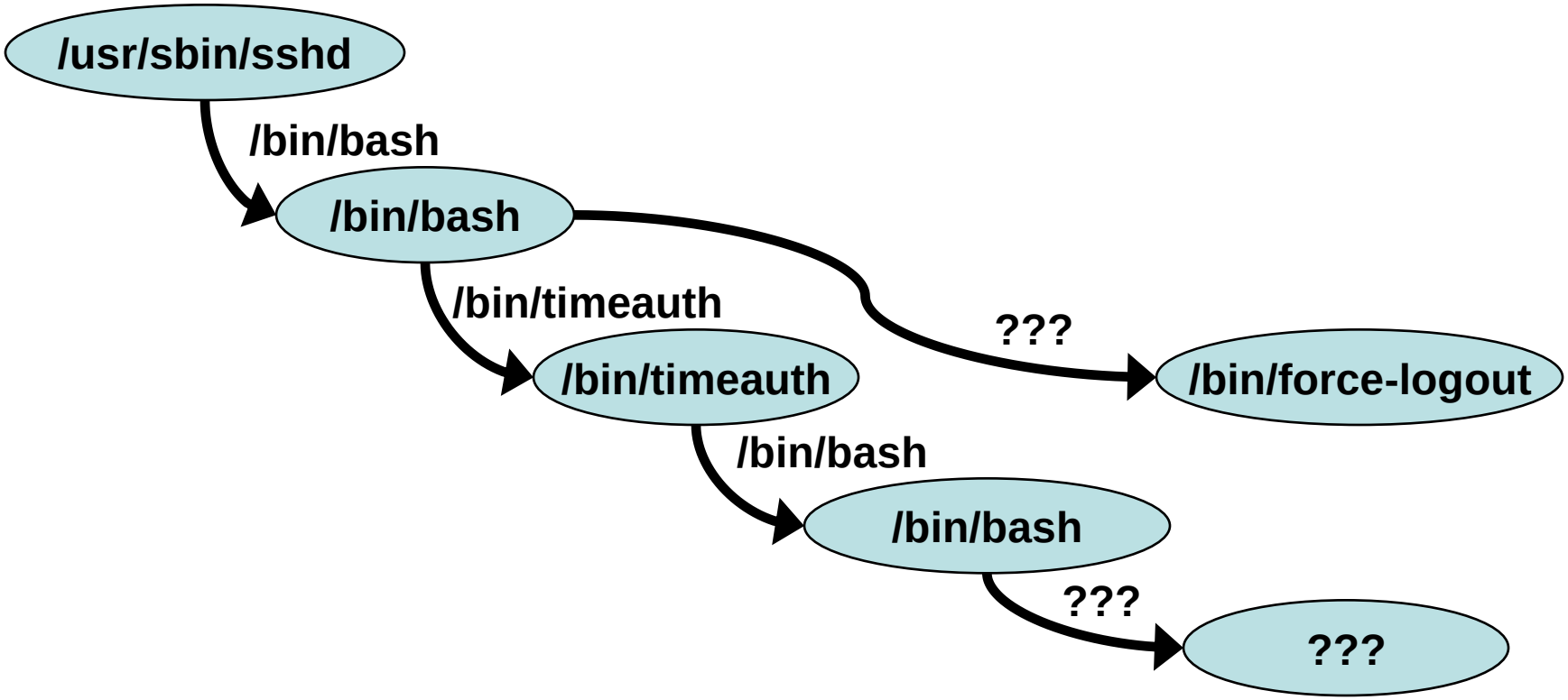
- 利点
  - プログラムの実行を伴わずに権限を切り替えできる
    - アクセス要求の順序をパスワードの代わりに使用
  - 対話的・非対話的シェルセッションを開始する前に適用することも可能

ケース5：非シェルセッション

- 難点
  - 使える要素が少ない
    - プログラムの実行を伴わないで実現する必要がある
  - おそらく TOMOYO Linux 専用
    - SSHサーバプログラム(/usr/sbin/sshd)を改造することなくプロセスの状態変数(task.state)を操作しているため
  - クライアントを自作する必要性

ケース6：即席ハニーポットの構築

・ 侵入者をハニーポットへリダイレクトできます。
  – もちろん、強制ログアウトもできます。



論文はこちらです。

- セキュリティ強化OSによるログイン認証の強化手法
  - http://sourceforge.jp/projects/tomoyo/docs/winf2005.pdf
  - http://sourceforge.jp/projects/tomoyo/docs/winf05-slides.pdf
  - 3年前に書かれた内容ですので内容が古いです。
  - しかし、考え方は現在でも役に立ちますし、実際に適用することができます。