一般社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
ET2023-15(2023-09)

ネットワークセキュリティ演習システムにおける受講者とChatGPTとの対話機能の開発

立岩佑一郎†

† 名古屋工業大学
名古屋市昭和区御器所町
E-mail: †tateiwa@nitech.ac.jp

あらまし　筆者はChatGPTとの対話を通して，ChatGPTがコンピュータネットワークの管理について，具体的な実装例を交えて助言できることを確認した．ここから，ネットワークセキュリティ演習において，ChatGPTが教師やTAの代わりに受講者のネットワークトラブルを解決することが期待される．筆者の担当するネットワークセキュリティ演習では，受講者は仮想マシンをノードとする仮想的なネットワークにて様々な通信実験を行う．本稿では，ネットワークの構成情報を収集し，それをカスタムYANGモデルに基づいて記述してChatGPTに伝達する方法と，受講者とChatGPTとの対話のためのユーザインタフェースについて述べる．

キーワード　ChatGPT，ネットワーク，セキュリティ，e-learning，演習，YANGモデル

Development of Dialogue Function between Students and ChatGPT in Network Security Exercise System

Yuichiro TATEIWA†

† Nagoya Institute of Technology
Gokiso-cho, Showa-ku, Nagoya-shi
E-mail: †tateiwa@nitech.ac.jp

1. はじめに

ChatGPT [1] はコンピュータサイエンスについても広範な知識を有しており，それらに関する質問に答える能力が認められる．実際に，筆者はChatGPTとの対話を通して，ChatGPTが基本的なネットワークの構築や，安全なネットワークの運用について，具体的な実装例を交えて助言できることを確認した．ここから，受講者がネットワークセキュリティ演習におけるネットワークの設計や攻撃・防衛の実施などでのトラブルに対して，ChatGPTがそれらトラブルの解決を手助けすることが期待される．

筆者の担当するネットワークセキュリティの演習授業では，受講者がLinux機器（サーバやファイアウォール）によりネットワークを構築し，そのネットワークにて攻撃や防衛を行う．この演習環境LiNeS Cloud [2] は仮想マシンをノードとする仮想的なネットワークをサーバ上に実現し，受講者はウェブブラウザを通じてそのネットワークを操作する．このため，サーバ上にて受講者のネットワーク構成を収集することは困難なことではない．また，筆者はアプリケーションとChatGPT間にてネットワーク構成情報を共有するための記述法を提案した [3].

本稿では，LiNeS Cloudで受講者が自身のネットワークについてChatGPTと対話するための機能を提案する．具体的には，ネットワークの構成情報を収集する方法を述べ，ネットワーク構成情報をChatGPTと共有するための記述法 [3] を詳解し，ChatGPTとの対話内容の構成および対話に用いるユーザインタフェースについて述べる．

2. LiNeS Cloud

User-mode Linux [4]（以降，UMLと呼ぶ）はLinux上で動作するプロセスであり，Linux計算機を仮想的に実現する．ディストリビューションイメージを引数としてUMLのプログラムを実行すると，そのディストリビューションのLinuxを実現するプロセスが起動される．UMLのホストOSで管理者権限を持っていないユーザであっても，UML内で管理者権限を持つことができ，ソフトウェアをインストールしたり，サービスを提供したりできる．UMLのネットワークインタフェースをLinux

This article is a technical report without peer review, and its polished and/or extended version may be published elsewhere.
Copyright ©2023 by IEICE

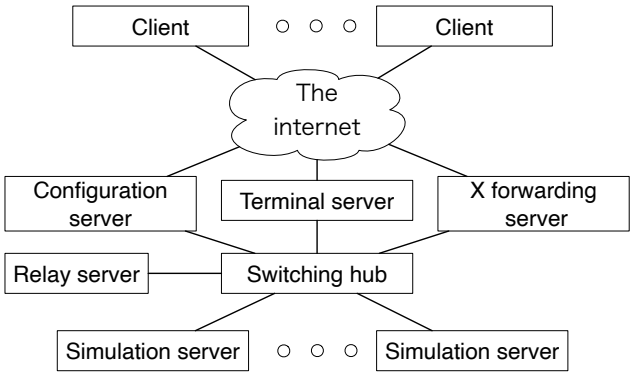

図 1　LiNeS Cloud のシステム構成
Fig. 1　System overview of LiNeS Cloud.

の Bridge で接続することで，仮想的なネットワークをそのホスト計算機上に実現できる．

これまでの研究 [2] で，UML を用いたネットワークセキュリティ演習のための，直感的でシームレスな操作と軽快な応答性を目指したウェブ型演習システム LiNeS Cloud を提案した．LiNes Cloud では，仮想機器であるサーバ，クライアント，ルータ，ファイアウォールを UML で，スイッチングハブを Linux の Bridge で実現している．

図 1 に示すシステム構成では，四角が計算機を意味し，実線が通信経路を意味する．模擬サーバ（Simulation server）は設定サーバ（Configuration server）からの遠隔操作を受け付け，仮想機器を稼働させてネットワークを形成する．設定サーバは，仮想機器間の接続関係を編集するためのウェブページと，UML のターミナルとの入出力のためのウェブページ，および UML の X クライアントとの入出力のためのウェブページを提供する．中継サーバ（Relay server）は，自身と模擬サーバとの間に作成されたイーサネットトンネルのトンネル口を Linux の Bridge で接続することで，二つの模擬サーバのトンネル口間でイーサネットフレームを交換できるようにする．端末サーバ（Terminal server）はターミナルページと UML のターミナルとの入出力を中継する．X 転送サーバ（X forwarding server）は，X サーバとして UML の X クライアントからの描画要求を受け付け，その出力を X ページに描画する．また，X ページからのユーザ入力を受け付け，X クライアントに送信する．クライアント（Client）上のウェブブラウザを通じて，受講者はネットワークを管理する．

3. 実　現　法

3.1　カスタム YANG モデルによるネットワークの記述

受講者が自身のネットワークについて ChatGPT と対話するさいには，対話の冒頭でそのネットワークを ChatGPT に伝えておく必要があるが，受講者自身がそれを行うのは，演習効率の低下を招くことになる．これを防ぐため，LiNeS Cloud が ChatGPT に受講者のネットワークを伝えることを考える．

筆者は文献 [3] にてアプリケーションと ChatGPT 間の共有のためのネットワーク構成情報の記述法を提案した．この提案は，アプリケーションから ChatGPT へのネットワーク構成情報の伝達だけでなく，その逆方向の伝達での利用も目的としたものである．これは，ChatGPT が自然言語処理を得意としている一方で，多くのアプリケーションは人工言語処理を得意としているためである．すなわち，ChatGPT からのネットワークに関する提案をアプリケーションが処理しやすくすることも目的としている．本研究では，LiNeS Cloud というアプリケーションが ChatGPT に受講者のネットワークを伝えるという利用のためだけでなく，受講者が ChatGPT にネットワークを設計してもらうなどの将来的な利用への拡張を踏まえて，記述法 [3] に基づいて対話機能を実現する．

記述法は YANG 言語 [5] に基づいたカスタム YANG モデルである．YANG はネットワーク構成の記述に適した人工言語であるが，その仕様は自然言語で記述され，インターネットにて広く公開されている．筆者が確認したところ，ChatGPT は初歩的なネットワークの設定を適度な精度で既存の YANG モデルに基づいて記述した．そして，RFC [6] において公開されている YANG モデルにおいて，単体で演習用ネットワークの構成のすべてを記述できるものは見当たらなかった．そこで，ietf-network [7] をその他の YANG モジュールで拡張し，その上での不足分を新規定義したカスタム YANG モジュールで拡張した．ただし，前者の拡張は YANG モデルの規則に従っていないが，GPT はその拡張を適度な精度で理解できているようであった．

図 2 は不足分を定義したカスタム YANG モジュールであり，紙面の都合上，一部を省略している．ietf-network モジュールの node コンテナに，device-properties という新しいコンテナを追加している．device-properties コンテナは，機器名を文字列で格納するリーフ device-name，機器の種類を列挙子で格納するリーフ device-type，機器の電源状態を列挙子で格納するリーフ power-state から構成される．

提案する YANG モデルを樹形図 [8] に基づいて図 3 に示す．要素の後に付随する丸括弧内の文字列は，ChatGPT へ伝えたい演習用ネットワークの構成要素を表す．①と②は ietf-network モジュールの要素である．③は図 2 に示したカスタム YANG モジュールでの拡張により追加された要素で，⑤と⑧は ietf-network-topology モジュール [7] での拡張により追加された要素である．④，⑥，⑦はそれぞれ ietf-system [9]，ietf-interfaces [10]，ietf-routing [11] に定義された要素であるが，各モジュールでの定義には ietf-network の拡張は宣言されていない．すなわち，④，⑥，⑦を ietf-network の要素として扱うことは YANG の規則に従っていない．しかし，YANG のコンセプトと ChatGPT による YANG への解釈の柔軟性から，ChatGPT がこのカスタム YANG モデルをこちらの意図通りに解釈できることが期待できる．

3.2　ChatGPT への送信用データの収集

図 3 に示したネットワーク構成情報を収集することを考える．図中の①〜⑤および⑧に必要なデータは LiNeS Cloud のデータベースにて管理されている．⑥と⑦に必要なデータは UML にて管理されており，UML のコンソール上での Linux コマンド

```
module network-devices {
  import ietf-network { prefix inet; }
  augment "/inet:networks/inet:network/inet:node" {
    container device-properties {
      leaf device-name {
        type string;
        description "The name of the device.";
      }
      leaf device-type {
        type enumeration {
          enum "L2-switch";
          enum "linux-server";
          enum "linux-client";
          enum "linux-firewall";
          enum "linux-router";
        }
        description "The type of the device.";
      }
      leaf power-state {
        type enumeration {
          enum "on";
          enum "off";
        }
        description "The power state of the device.";
      }
    }
  }
}
```

図 2　カスタム YANG モジュール

Fig. 2　Custom YANG module.

```
ietf-network:networks/network
├─network-id(ネットワークID)①
├─node*
│  ├─node-id(機器ID)②
│  ├─network-devices:device-properties③
│  │  ├─device-name(機器名)
│  │  ├─device-type(機器の種類)
│  │  └─power-state(電源状態)
│  ├─ietf-system:system-state/platform④
│  │  ├─os-name(OS名)
│  │  └─os-release(OSのリリース情報)
│  ├─ietf-network-topology:termination-point*⑤
│  │  └─tp-id(イーサネットポートID)
│  ├─ietf-interfaces:interfaces-state/interface*⑥
│  │  ├─name(イーサネットポートID)
│  │  ├─admin-status(有効状態)
│  │  ├─phys-address(MACアドレス)
│  │  └─ietf-ip:ipv4/address*
│  │     ├─ip(IPアドレス)
│  │     └─netmask(サブネットマスク)
│  └─ietf-routing:routing-state/ribs/rib/routes/route*⑦
│     ├─destination-prefix(宛先)
│     └─next-hop
│        ├─next-hop-address(ネクストホップアドレス)
│        └─outgoing-interface(送信インタフェース)
└─ietf-network-topology:link*⑧
   ├─link-id(リンクID)
   ├─source
   │  ├─source-node(機器ID)
   │  └─source-tp(イーサネットポートID)
   └─destination
      ├─dest-node(機器ID)
      └─dest-tp(イーサネットポートID)
```

図 3　カスタム YANG モデル

Fig. 3　Custom YANG model.

実行により収集できる．

ここで，⑥と⑦に必要なデータを UML から収集するに当たり，以下の課題が挙げられる．

課題 1　受講者のコンソールを使わない：受講者の端末クライアントは Terminal server を介して UML のコンソールに接続されている．収集するにあたり，端末クライアントと Terminal server の接続を解除し，収集用アプリケーションが Terminal server を介して UML のコンソールを利用する方法が考えられる．しかし，この方法ではアプリケーションが収集している間に受講者が UML を利用できず，演習の中断を招く恐れがある．

課題 2　シェルのコマンド履歴を汚さない：演習中に受講者は自身の過去のコマンドを参照することがありうる．このときに，収集に利用したコマンドが履歴に残っていると，受講者の作業効率を低下させたり混乱を招いたりする恐れがある．

課題 1 の解決のために，UML の別のコンソール（受講者が使用していない）に擬似端末を接続して，収集用アプリケーションから擬似端末経由でそのコンソールにて収集用のコマンドを実行する．このために，まず，Terminal server は起動オプションに「con1=pty」を追加して UML を起動し，さらに，UML に tty1 を制御端末としたシェルプログラムを実行させる．そして，UML のブートログから擬似端末のスレーブ側のデバイスファイルを抽出し，それを引数として screen [13] のセッションを作成する．最後に，アプリケーションはこのセッションに接続することで，UML にて linux コマンドを実行し，必要なデータ（図 3 の⑥と⑦）を取得する．

Bash [14] は，多数の Linux ディストリビューションにおいて，デフォルトのシェルとして頻繁に選択されてきた．Bash でのコマンド実行にあたり，環境変数 HISTCONTROL に ignorespace を設定した上で，コマンドの先頭に空白を付加すると，そのコマンドは実行履歴に記録されない．UML でのシェルを Bash にして，前文のように利用することで課題 2 を解決する．

図 4 は受講者が ChatGPT と対話するときに利用される主なデータの流れを示す．受講者がウェブブラウザ（Web browser）に ChatGPT への質問や LiNeS Cloud への命令などを入力すると，ウェブブラウザはそれら（Request）を Node.js [15] に送信する．Request の内容に応じて Node.js の動作が変わり，受講者のネットワークを ChatGPT に伝達するとき，Node.js はデータベースから図 3 の①〜⑤と⑧のデータを取得し，UML から⑥と⑦のデータを収集する．その後，①〜⑧のデータをプロンプトに変換して ChatGPT に送信し，その結果をウェブブラウザに送信する．受講者のメッセージを ChatGPT に伝達するとき，

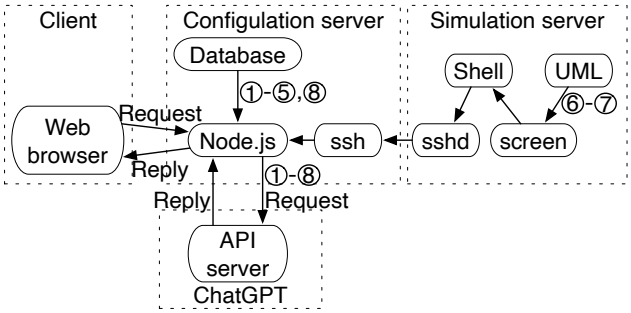

図 4　対話時のデータフロー

Fig. 4　Data flow during dialogue.

あなたは以下の「カスタムYANGモデル」に基づいた「YANGインスタンス」を記憶し、私からの質問があるまで待機します。

カスタムYANGモデル: ⓐ

YANGインスタンス: ⓑ

図 5　ChatGPT へのプロンプト

Fig. 5　Prompt to ChatGPT.

Node.js は受け取ったメッセージ（Request）を ChatGPT に送信し，得られた応答（Reply）をウェブブラウザに送信する．

3.3　ChatGPT へのプロンプト

図 5 は，図 4 において Node.js が API server に①〜⑧を送信する際に用いるプロンプトである．ⓐにカスタム YANG モデル（図 3）を，ⓑに①〜⑧に基づいた YANG インスタンスを記述する．

4.　プロトタイプシステム

図 6 はクライアント上で起動したウェブブラウザにおける対話機能の実行例を示す．受講者は領域Ⓐにてネットワークトポロジーを作成し，別のウィンドウにて各々の機器に設定を施す．受講者が ChatGPT と現在作成中のネットワークについて対話するには，まず，ボタンⒷを押下して，システムにこのネットワークを ChatGPT に伝達させる．その後，テキストボックスⒹにメッセージを入力し，ボタンⒺを押下することで，システムにそのメッセージを ChatGPT に伝達させる．最後に，システムが ChatGPT からの応答を受け取ると，その内容が領域Ⓒに追記される．現在の領域Ⓒでは，機器の電源状態や IP アドレスを問合せたり，ping コマンドを生成してもらったりしている．そして，ChatGPT はそれらの依頼を適度な精度で達成している．

5.　おわりに

本稿では，受講者が ChatGPT と対話する機能のために，カスタム YANG モデルによるネットワーク構成情報の共有方法，受講者の作業に影響しないように UML から内部データを収集する手法，および ChatGPT へのプロンプトの構成を提案した．また，プロトタイプシステムを実装し，受講者のネットワークが ChatGPT へ送信されていること，そのネットワークに対する質問への ChatGPT からの回答を提示できることを確認した．

図 6　対話機能の実行例

Fig. 6　Execution example of dialogue function.

今後の課題として，対話機能の性能評価が挙げられる．ChatGPT はモデルによって応答速度と回答精度が反比例し，例えば，GPT-3.5 と GPT-4 では前者の方が応答速度が速いが回答精度が悪く，後者の方はその反対となる傾向がある．まず，ネットワーク構成情報の収集にかかった時間と ChatGPT の回答にかかった時間を計測する．その後，実際の演習の指導書に基づいて，様々なネットワークの状況やトラブルを想定した対話にて，ChatGPT がどの程度の精度で質問を理解し，回答を生成できるかを評価する．

謝辞　本研究は JSPS 科研費 20K12108 の助成を受けたものです．

文　献

[1] Introduction - OpenAI API, https://platform.openai.com/docs/introduction, (2023.07.18 取得).
[2] Yuichiro Tateiwa, “LiNeS Cloud: A Web-Based Hands-On System for Network Security Classes with Intuitive and Seamless Operability and Light-Weight Responsiveness,” IEICE Transactions on Information and Systems, Vol. E105.D, No. 9, pp.1557–1567, 2022.
[3] Yuichiro Tateiwa, “アプリケーションと ChatGPT 間においてネットワーク構成情報を共有するためのカスタム YANG モデルの提案”, 電子情報通信学会, ソサエティ大会予稿集, 印刷中.
[4] The User-mode Linux Kernel Home Page, http://user-mode-linux.sourceforge.net/, (2023.07.18 取得).
[5] M. Bjorklund, Ed., “The YANG 1.1 Data Modeling Language,” RFC 7950, 2016.
[6] RFC INDEX, https://www.rfc-editor.org/rfc-index.html, (2023.07.18 取得).
[7] A. Clemm et al., “A YANG Data Model for Network Topologies,” RFC 8345, 2018.
[8] M. Bjorklund et al, “YANG Tree Diagrams,” RFC 8340, 2018.

[9] A. Bierman, M. Bjorklund, “A YANG Data Model for System Management,” RFC 7317, 2014.
[10] M. Bjorklund, “A YANG Data Model for Interface Management,” RFC 8343, 2018.
[11] L. Lhotka et al., “A YANG Data Model for Routing Management (NMDA Version),” RFC 8349, 2018.
[12] M. Bjorklund, “A YANG Data Model for IP Management,” RFC 8344, 2018.
[13] Screen - GNU Project - Free Software Foundation, https://www.gnu.org/software/screen/, (2023.7.18 取得).
[14] Bash - GNU Project - Free Software Foundation, https://www.gnu.org/software/bash/, (2023.7.18 取得).
[15] Node.js, https://nodejs.org/en/, (2023.7.18 取得).