スマートコンセントを応用したBYOD環境におけるユーザ認証

宮下　卓也*　山﨑　敦史**

User Authentication for BYOD with Smart Power Supply

MIYASHITA Takuya and YAMASAKI Atushi

BYOD PCs are student-managed and it is not authenticated by server on the school network. When a network trouble is occurred by the BYOD PC, administrators cannot investigate the reason because they don't know who is using the network. This is a serious problem about information security.To solve the above problem, a smart power supply system for the PCs was developed in this study. The system consisted of a controller computer, an IC card reader and some smart plugs. The PCs were connected to the smart plugs and they were supplied under contorting by the smart plugs with application programs for smart phones. The IC card reader could authenticate the students with their student card. The Raspberry Pi was adopted as the controller computer.

Key Words: User Authentication, Smart Plug, BYOD, Information Security, Raspberry Pi

1．緒　　言

現代社会においてはコンピュータやインターネットを活用できる能力は必要不可欠と言っても過言ではない。それゆえ、文部科学省においては「情報活用能力」を学習の基盤となる資質・能力と位置付け、小学校学習指導要領にプログラミング教育を導入するなど、教育の情報化を推進している状況にある[1)]。

津山工業高等専門学校（ 以下､津山高専と記す）は高等教育機関であるため、初等教育機関における情報化推進より前から情報教育の必要性は気づいており、随分以前から校内に教育用パソコンを備えた演習室を設けている。しかしながら最近になり、教育用パソコンの導入・管理の費用がかさむことが問題となり、多くの教育機関で教育用パソコンを導入する代わりにBYOD（Bring Your Own Device）を推進するようになってきた。津山高専でも令和 2 年度入学学生から、すべての新入生にノートPCの購入あるいは準備をすることになっている。

旧来の教育用パソコンは学校の管理下にあったため、そのパソコンが原因でネットワークトラブルが発生した際に利用状況を特定することは難しいことではなかった。しかしながらBYOD PCは学生の管理下にあり、学生が望む時に望むままに利用されることになるため、ネットワークトラブルが発生した際に利用状況を特定するのが困難である。教育機関のネットワーク管理者としては、少なくとも利用者を特定し、当該学生から利用状況を確認する必要がある。そのためにはBYOD PCのユーザ認証をする必要がある。

無線 LAN あるいは有線 LAN を利用する際に、IEEE802.1X 認証によるユーザ認証機能が利用されているならば、ネットワーク利用時にユーザを特定することが可能である。津山高専の基幹ネットワークにおいては、学生向けに公開されている無線LANではMACアドレス登録とIEEE802.1X認証が行われているので、利用者が特定できないという問題はない。ただし、構内の小規模な演習室においては、IEEE802.1X 認証が用意されていないことも少なくないと考える。その理由には、IEEE802.1X認証を構築するための費用や運用するための技術力が要求されることが考えられる。

そこで、BYOD 環境でのユーザ認証を行うシステムの開発を研究目的とした。具体的には、BYOD PCのために用意する電源コンセントに着目した。多くの学生は、電源コンセントが用意されていれば、ほぼ間違いなく電源コンセントを利用する。そこで、学生を認証してコンセントに通電するような仕組みを設ければ、利用者を特定できると考えた。また、学生を認証するための情報源には、IC カード機能を有する津山高専の学生証が活用できると

原稿受付　令和 3 年 8 月 24 日

＊総合理工学科　情報システム系

＊＊電子・情報システム工学専攻　1 年

考えた。

電源の通電あるいは遮断はリレー回路を用いれば実現できるが、商用電源は電力が決して小さくないため、安易な工作で発火などのトラブルが発生することが懸念される。そこで、本研究では電源の制御にスマートコンセントを用いることにした。スマートコンセントは、ネットワークを介して、家電製品への給電を制御できる製品である。

以上のことから、学生証を読み取り、その結果に応じてスマートコンセントによって、コンピュータへ電源を供給するシステムを実現すればよいという発想に至った。このようなスマートコンセントを応用し、ユーザ認証を行うシステムの開発をすることが、本研究の目的である。

2．電源制御システムの概要

本研究では開発するシステムを、電源制御システムと記す。このシステムのイメージを図 1 に示す。

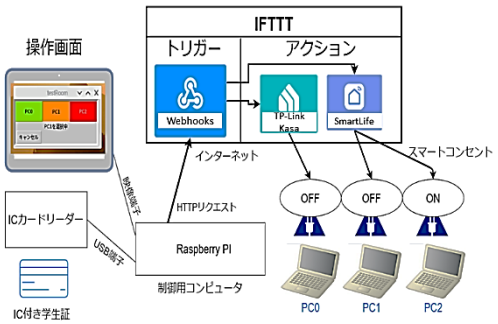

図 1　電源制御システムの構成図

電源制御システムは、学生証（IC カード）を読み取る IC カードリーダ、スマートコンセント、およびそれらの制御用コンピュータによって構成される。このシステムの制御には、種々の機能との連携を考慮し、Python[2]によるプログラムを用いる。スマートコンセントは専用アプリである「TP-Link KASA」または「Smart Life」により電源の ON あるいは OFF を切り替える。このスマートコンセントの制御には、様々なサービスとサービスの連携を可能とする API を提供するクラウドサービス IFTTT[3] を経由して行う。

図中の各 PC は、以下の条件を満たしたときに、スマートコンセントから電源が供給される。

条件 1　PC の利用前に学生証を IC カードリーダに読み取らせている

条件 2　IC カードの読み取り結果が、認証用データベースに登録されている学生証の情報と一致する

条件 3　読み取られた IC の ID がすでに別の PC で使用されていない

学生 A が PC1 用コンセントを使用する際の手順と、電源制御システムの動作は以下のようになる。

(1) 学生 A がシステム操作画面（GUI）で、例えば PC1 をクリックする
(2) 学生 A が自分の学生証を IC カードリーダに接触させる
(3) 制御用コンピュータが学生証の ID 部分を抽出し、認証用データベースと照合する
(4) 照合結果が正常であれば制御用コンピュータが IFTTT 経由で PC1 用のスマートコンセントのスイッチを入れる
(5) その結果、スマートコンセント経由で PC1 にのみ電源が供給される
(6) 学生 A は PC1 の電源ボタンを押し、PC の利用を開始する

電源制御システムによって各 PC の電源が上記のように制御される結果、各 PC の利用者を学生証に基づいて特定することが可能となる。

なお、PC の利用が終わった際に上記とほぼ同じ手続きで終了通知をしてもらうべきであるが、面倒に感じる学生は終了手続きをしないことが考えられる。そこで、スマートコンセントの消費電力をモニターし、利用がなくなったら自動的にスマートコンセントをオフにするようにするか、タイマー機能によって電源を遮断するなどの対応をすることが考えられるが、現時点では未対応である。

3．ハードウェア構成要素のセットアップ

3．1　制御用コンピュータと IC カードリーダ

制御用コンピュータとして次の条件を満たすコンピュータを採用する。

・ネットワークに接続可能
・カギ付き BOX に収納可能な小型筐体
・OS およびプログラミング環境の存在

スマートコンセントは無線 LAN 経由で制御されるため、制御用コンピュータはスマートコンセントと同じ LAN に接続されている必要がある。また電源制御システムを直接学生が触ることができる

と、電源を別の場所からとったり、スマートコンセントを強制的にONにしたりすることができるようになる。そのため、電源制御システムの根幹部分はカギ付きBOXなどに収納する必要がある。最後の条件については、本システムの機能を実現するために当然必要な条件である。

以上のような条件を満たすコンピュータとして、本研究ではRaspberry Pi3 Model B（以下ラズパイ）を採用した。ラズパイを利用するためには、まず最初にOSをインストールする必要がある。本研究では、RaspbianをOSとして採用した。

ICカードである津山高専の学生証を読み取るために、ラズパイにICカードリーダをUSBポート経由で接続する。今回は「非接触ICカードリーダ/ライター　Pasori（SONY社製RC-S380）」（以下カードリーダ）を使用した。ラズパイでカードリーダを使用するためには、「nfcpy」[4]の導入が必要である。nfcpyはPythonでICカードリーダを使用可能にするモジュールであり、ラズパイのターミナルで図2のコマンドによってインストールする。

```
$ sudo pip3 install –U nf cpy
```

図2　nfcpyインストールコマンド

続いて、プログラム実行時にsudoを不要にするために、USBデバイスに関する設定を変更した。そのコマンドを図3に示す。なお、紙面の都合上で複数行表示としているが、実際には各先頭が$であり、合計2行のコマンドである。

以上の作業を行い、ICカードの読み取り試験を行ったところ、問題なく動作していることが確認された。

３．２　スマートコンセントとの接続

本システムでは制御用コンピュータから、サービスパイプ機能と言えるWebサービスのIFTTT宛に、電源を制御する命令を与える。制御命令を受け取ったIFTTTは、スマートコンセントに対して制御信号を送る。このような処理を実現するための設定手順を説明する。

本研究では、IFTTTを利用してネットワーク経由で制御できるスマートコンセントして、Gosund（型番Gs-wp6）とtp-link（型番HS105）の2種類を用いた。

スマートコンセントは、本来はスマートフォンから電源のON/OFFをする電源コンセントである。その制御には、各メーカから提供されているスマートフォンアプリを用いる。具体的には、GosundはSmart Life - Smart Living[5]、tp-linkはKasa Smart[6]というアプリを使用する。この2つのスマートコンセント制御アプリはIFTTTと連携可能なものである。

まず最初に、スマートコンセントの制御に関する設定作業を行った。各アプリを起動し、画面表示に従って作業を行ったところ、スマートフォンとスマートコンセントを接続することができた。

IFTTTではIFTTTに対応しているサービス同士を連携させることが可能である。IFTTTは「IF This Then That」の頭文字であり、「もし、これをしたら、あれをする」という意味である。サービスを連携するときは「これをしたら」にあたる「トリガー」と「あれをする」にあたる「アクション」を設定する。本システムにおいては、トリガーは「スマートコンセントのスイッチを入れるための条件が満たされた」であり、アクションは「スマートコンセントのスイッチを入れる」となる。

先述のように、本研究で使用している2つのスマートコンセントの制御アプリは、IFTTTに対応している。そのため、IFTTTのアクションに指定すれば、IFTTTを介してスマートコンセントの制御が可能となる。一方、トリガーは制御用コンピュータからの命令送信が該当することになる。トリガーの実現のために、IFTTTに対応しているWebhooks[7]を使用した。

Webhooksとは、Webアプリケーションでイベントが実行された際に外部サービスへHTTPで通知する仕組みである。IFTTTの公式サイトでトリガーサービスとしてWebhooksを選定すると、IFTTTアカウントに対してユニークな暗号キーが割り振られる。制御用コンピュータではPythonで作成した

```
$sudo sh −c ’ echo SUBSYSTEM==¥” usb ¥ ” , ACTION==¥” add ¥ ” , ATTRS{ idVendor }==¥”054 c ¥ ” ,
   ATTRS{ i d P r o d u c t }==¥”06 c3 ¥ ” ,
   GROUP=¥” plugdev ¥” >> / e t c / udev / r u l e s . d / nf c d e v . r u l e s ’
$ sudo udevadm c o n t r o l −R # t h e n re −a t t a c h d e v i c e
```

図3　USBデバイスの設定変更コマンド

制御プログラム（詳細は次節で述べる）が動作している。スマートコンセントを制御する場合には、制御プログラムが HTTP 通信で IFTTT の Webhooks トリガーへ暗号キーを含めて POST する。その結果、IFTTT で Webhook を受けて、スマートプラグが操作されることになる。

4．電源制御システム用プログラムの開発

4．1　利用者向けの操作画面とマルチスレッド処理

2 節で説明したように、電源制御システムによって管理しているコンセントを利用する際は、まず最初に利用者がシステム操作画面（GUI）で使用したいコンセントを選択する必要がある。このための操作画面について説明する。

Python には tkinter[8] という標準のグラフィックユーザーインターフェース（以下 GUI）フレームワークがある。本研究では、システム操作画面を実現するにあたって、tkinter を拡張した tkinter.ttk モジュール [9]をインポートして使用した。作成したシステム操作画面を図 4 に示す。

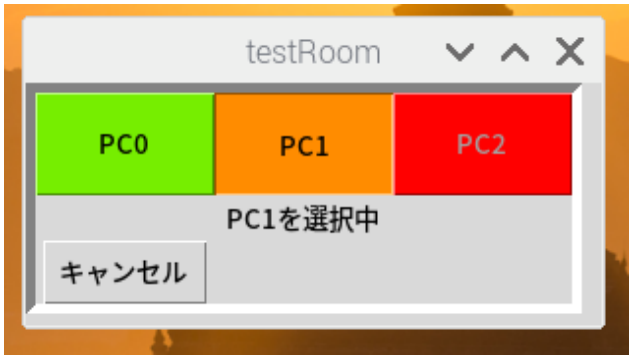

図 4　システム操作画面

図 4 中のボタンの色は各コンセントの使用状況を表示している。緑色であれば（その時点では）誰も利用していないことを意味しており、赤色であれば誰かが利用していること意味している。利用者が PC を利用しようとして、ある空きコンセントを選択した場合にボタンは緑色からオレンジ色に変化する。

システム操作画面の GUI クラスは次の処理を実装した。

・操作画面の設定
・操作画面のループ表示
・操作から処理を実行
・IC カードの読み取り管理
・プラグ起動関数の呼び出し

GUI クラスは操作画面を表示するために、根幹的な部分である root.mainloop( )関数を使用する必要がある。この関数はプログラムが停止するまで、その動作を繰り返すようになる。その結果、画面表示のために関数が実行された場合、繰り返し処理から抜け出すことができなくなり、他の処理を実行することができない。

この繰り返し処理の問題を解決するためにマルチスレッド処理を実装した。制御プログラムにおいて root.mainloop( )関数を実行する前に他の処理をマルチスレッドで実行するように実装し、図 5 に示すように並行して処理を実行するようにした。

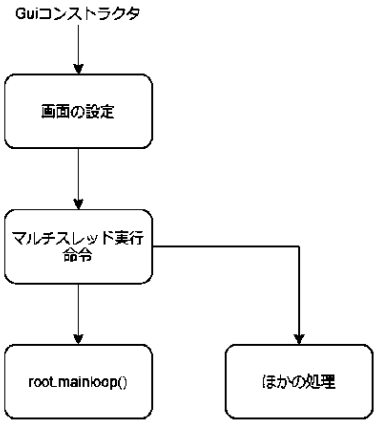

図 5　GUI のマルチスレッド処理

4．2　学生証の読み取りプログラムとマルチスレッド処理

先に述べた IC カードである学生証を読み取るための Python モジュールである nfc.py は、プログラム中で利用するとカードを読み取るまで待機状態となる。その結果、他の処理に移行することができない。もし利用者があるコンセントを選択し、学生証を読み取る時点で利用をとりやめたいと考えた場合は、すでに IC カード読み取りの待機状態となっているため、何らかの中断処理が必要となる。IC カードを読み取る前に動作を中断する方法はプログラムを停止させる以外に見つけることができなかった。そこで、別の選択キャンセルの方法を実現した。読み取り動作を中断せずに選択のキャンセルを行う方法を次ページの図 6 に示す。

図 6 に示すように、制御プログラム中から操作画面を表示する前に IC カードの読み取り動作がマルチスレッドかつ無限ループで実行される。そのスレッドではカードの読み取りプログラムが IC カードを読み取るまで待機状態となり、常に読み取り後に読み取り動作を繰り返す。操作画面で利用希望 PC の未選択状態で IC カードを読み取った

場合、不正な読み取りとして読み取り待機状態に戻る。ここで、利用希望の選択状態を判別するため変数 isSelect を実装した。isSelect の値は操作画面の利用希望 PC が選択されているときには True、選択されていないときには False が代入されている。

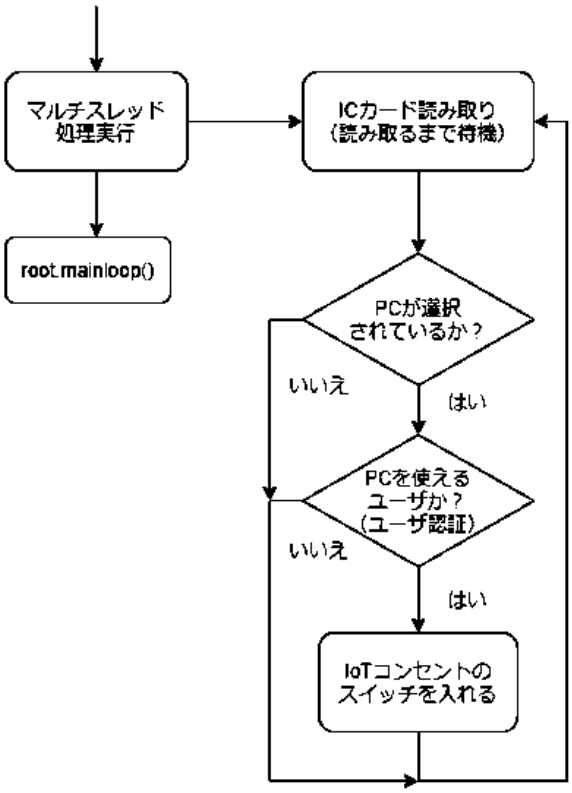

図6　カードの読み取り処理フローチャート

キャンセルボタンが押下されキャンセルされる処理が発生すると、isSelect に False が代入され未選択状態となる。先述したとおり未選択状態で IC カードを読み取った場合、不正な読み取りとなるためキャンセル処理が実装されている。

４．３　ユーザ認証用データベースの実装

電源制御システムにおいて、学生証を IC カードで読み取った後、IC カード情報に基づいてユーザ認証が行われる。認証を行うための身元情報はデータベースで管理するように実装した。電源制御システムにおけるデータベースは図 7 に示す ER 図のような構造で構築した。なお、現段階ではプロトタイプ作成を行っており、データベースの正規化は深く行っていない。

データベースは3つのテーブルに分かれている。1つめは学生の名前や学籍番号および IC カード ID などの情報をレコードとして持つテーブルである。2 つめはコンセント使用記録を保存するテーブルである。3つめはその時点での利用状況を保存するテーブルである。

Python には簡易的にデータベースを作成可能な SQLite3[10]モジュールがある。本研究では、オブジェクト（DAO）を生成するクラス Dao.py を作成した。本システムの DAO は次のような機能を持つ。

- 学生管理テーブルの作成
- 使用ログ記録テーブルの作成
- 利用状況テーブルの作成
- 学生管理テーブルに学生の情報を追加
- 学生管理テーブルから学生の情報を取得
- 使用ログ記録テーブルに使用ログを追加
- 使用ログ記録テーブルから使用ログを取得
- 利用状況テーブルに利用状況を更新
- 利用状況テーブルから利用状況を取得
- データベース情報をファイルに保存

電源制御システムでは、上記の SQL 関係の機能を利用して、ユーザの認証をしたり、使用ログを記録したりしている。

４．４　不正利用の通知機構の実装

電源制御システムでは、条件を満たされた場合にのみコンセントが利用可能となる。逆に、条件を満たさなかった場合は、何らかの不正利用状態となっている。そこで、この不正利用時にシステ

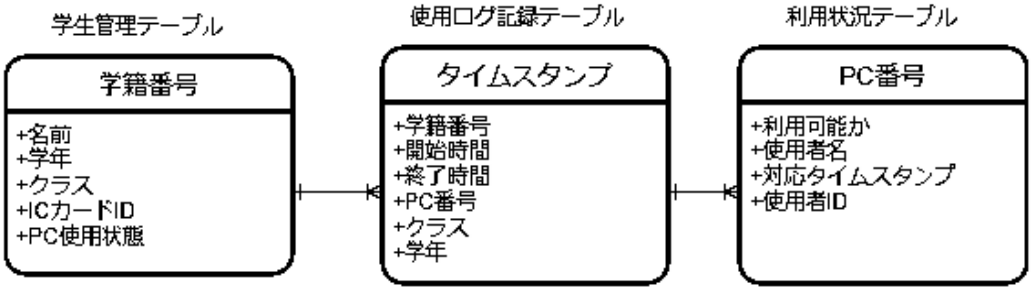

図7　システムデータベース ER 図

ムがビープ音を鳴らし、利用者に注意喚起をするようにした。

注意喚起のビープ音は以下のいずれの条件で鳴る。その際は、スマートコンセントは ON にならないため、電源として利用することはできない。

- 操作画面においてコンセントが選択されていないまま学生証を読み取ったとき
- データベースに登録されていない学生証 ID を読み取ったとき
- すでにコンセントを使用している学生証 ID を読み取ったとき

５．動作確認

試作したシステムについて、3 人の正規利用者と 1 人の非正規利用者を想定して、個別あるいは同時の利用状況を想定し、動作確認を行った。その概要と結果を以下に示す。なお、ユーザ A、B、C が正規利用者、ユーザ D は非正規（未登録）利用者とした。

試験 1：ユーザ A による複数コンセントの利用
結果 1：2 つめ以降は利用不可だった

試験 2：ユーザ A、B、C が各 1 コンセント同時利用
結果 2：複数利用可能だった

試験 3：ユーザ D による利用
結果 3:認証不可のためコンセント利用不可だった

以上の動作確認の結果、システムが正常に動作していることが確認できた。

６．まとめ

本研究では、学生管理の BYOD PC に対して、スマートコンセントを応用してユーザ認証を行うシステムを開発した。

動作試験の結果、想定した利用に対しては、システムが正常に動作していることが確認された。ただし、現段階は試作中のため、制御用コンピュータや IC カードリーダおよびスマートコンセントがむき出しの状態であり、カギ付き BOX には収容していない。また、現時点では問題ではないが、本システムで採用している IFTTT や Webhook の仕様が変わったり、あるいは利用料金の増加がなされるなどの利用制限が行われたりすると、根本からシステムの処理方法の変更が必要になる。さらにはインターネット経由でのクラッキングに対しての検討もまだできていない。これらの問題の解決は、今後の課題である。

また、そもそもの問題として、学生が電源コンセントを利用しない場合は、本システムではユーザ認証を行うことができない。ただし、津山高専では無線ネットワーク利用時にユーザ認証が行われているため、現実に利用者が特定できない状況はネットワークも電源も利用しないスタンドアローンな状況のみである。この場合は、そもそも構内ネットワークに対してのトラブルの原因となる可能性はかなり低いと考える。

参　考　文　献

1) 文部科学省　教育の情報化の推進: https://www.mext.go.jp/a_menu/shotou/zyouhou/index.htm（参照 2021-08-14).
2) python: https://www.python.org/（参照 2021-08-15).
3) IFTTT: https://ifttt.com/（参照 2021-08-15).
4) raspberryPi に NFC リーダーをつなぐ: https://qiita.com/irutack/items/61a783eb9d5c78d5a3f6（参照 2021-08-15).
5) SmartLife - Smart Living: https://play.google.com/store/apps/details?id=com.tuya.smartlife&hl=ja&gl=US（参照 2021-08-15).
6) KasaSmart: https://play.google.com/store/apps/details?id=com.tplink.kasa_android&hl=ja&gl=US（参照 2021-08-15).
7) Webhooks: https://ifttt.com/maker_webhooks（参照 2021-08-15).
8) Tkinter --- Tcl/Tk の Python インタフェース: https://docs.python.org/ja/3/library/tkinter.html（参照 2021-08-15).
9) tkinter.ttk --- Tk のテーマ付きウィジェット: https://docs.python.org/ja/3/library/tkinter.ttk.html（参照 2021-08-15).
10) SQLite3 --- SQLite データベースに対する DB-API 2.0 インタフェース: https://docs.python.org/ja/3/library/sqlite3.html（参照 2021-08-15).