

Uniwersytet Technologiczno-Przyrodniczy w Bydgoszczy



**Projekt z przedmiotu „Metody ochrony cyberprzestrzeni”  
„Phishing e-mails”**

Kierunek: Informatyka stosowana

Stopień studiów: magisterskie

Semestr: trzeci

Autor: Iwona Górską

Prowadzący przedmiot: dr hab. inż. Michał Choraś

## Spis treści

<b>1. Wprowadzenie.....</b>	<b>3</b>
<b>2. Uwagi.....</b>	<b>4</b>
<b>3. Budowa i działanie aplikacji .....</b>	<b>5</b>
<b>3.1 Struktura aplikacji.....</b>	<b>5</b>
<b>3.2 Zasada działania aplikacji.....</b>	<b>5</b>
<b>4. Podsumowanie.....</b>	<b>13</b>

# 1. Wprowadzenie

Celem projektu było przygotowanie mechanizmu wykrywania zagrożenia, jakim jest atak phishing e-mail. Zadanie zostało zrealizowane w środowisku ServiceNow. ServiceNow to platforma do zarządzania procesami, usługami i infrastrukturą przedsiębiorstwa. Platforma ServiceNow składa się z szeregu modułów, funkcji i narzędzi, takich jak workflow, SLA, notyfikacje czy formularze, za pomocą których można budować dowolne aplikacje działające na wspólnym rdzeniu, jakim jest baza konfiguracji (CMDB). ServiceNow daje możliwość korzystania z gotowych aplikacji zbudowanych na platformie lub wybrania samej platformy, na której programiści mogą tworzyć dedykowane aplikacje od podstaw.

Ja stworzyłam od podstaw dedykowaną do projektu aplikację o nazwie „Phishing”. W ramach aplikacji powstał szereg procesów odpowiedzialnych za tworzenie testowego e-maila sprawdzającego czujność użytkowników, tworzenie zwykłego e-maila umożliwiającego zaprezentowanie rozwiązania w szerszym zakresie, odpowiedniej obsługi zdarzeń mających miejsce po odebraniu e-maili przez użytkowników.

## 2. Uwagi

Aplikacja została stworzona na personalnej instancji ServiceNow zamówionej przeze mnie o adresie <https://dev66772.service-now.com>. Dostęp do instancji jest możliwy tylko w przypadku znajomości loginu i hasła któregoś z użytkowników, tylko użytkownicy z odpowiednimi uprawnieniami mają prawo odczytu plików systemowych, do których należą też pliki stworzone w ramach aplikacji „Phishing”. Wersja instancji to „New York”.

Personalne instancje można zamawiać we własnym imieniu, nie ponosi się za to żadnych opłat, nie trzeba przynależeć do żadnej organizacji, firmy współpracującej z ServiceNow komercyjnie. W związku z tym instancje tego typu mają kilka ograniczeń, takich jak:

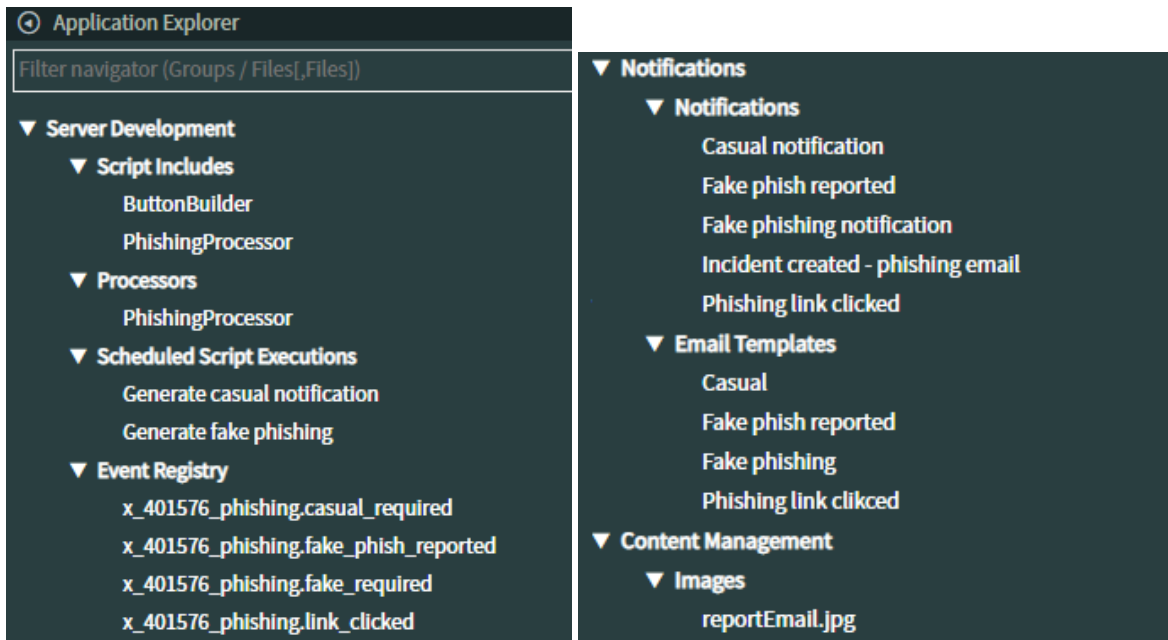
- brak możliwości publikowania aplikacji do repozytorium na Githubie itp.,
- brak możliwości wysyłania e-maili z instancji do zewnętrznych skrzynek pocztowych (na normalnych instancjach można taką opcję skonfigurować, nawet gdy instancja nie jest produkcyjna),
- ryzyko usunięcia instancji w przypadku nieaktywności przekraczającej 7 dni.

W związku z tymi ograniczeniami, które zaobserwowałam, zdecydowałam się na archiwizowanie mojej aplikacji za pomocą wyeksportowania jej do pliku typu „update set” – taki plik powinien być możliwy do zaimportowania na inną instancję i jest formą zabezpieczenia w przypadku utraty danych czy nawet usunięcia instancji. W kwestii niemożliwości wysyłania notyfikacji na zewnętrzne skrzynki pocztowe, moim rozwiązaniem jest odczytywanie e-maili na samej instancji, w sekcji „Outbound”.

### 3. Budowa i działanie aplikacji

#### 3.1 Struktura aplikacji

W ramach aplikacji powstało 5 notyfikacji (definicji e-maili), 4 e-mail templates, 4 definicje zdarzeń, 2 skrypty typu scheduled script execution, 2 skrypty typu script include i 1 procesor.



#### 3.2 Zasada działania aplikacji

Główny proces w prezentowanym rozwiązaniu zaczyna się od dwóch scheduled script executions. Są one zaprojektowane tak, aby wykonywać się raz dziennie. Można je także uruchomić ręcznie na żądanie.

- 1) Generate casual notification – wybiera losowego użytkownika i tworzy w systemie zdarzenie „x\_401576\_phishing.casual\_required”, dane wybranego użytkownika wysyła jako parametry tworzonego zdarzenia.
















Scheduled Script Execution  
Generate casual notification

Update

Execute Now

Delete

Conditional ☐

Run this script               

```
1 var eventName = 'x_401576_phishing.casual_required';
2 var userId;
3 var userName;
4 //1 - choose random user
5 var grUser = new GlideRecord('sys_user');
6 grUser.query();
7 var usersNr = grUser.getRowCount();
8 var randomNr = Math.floor(Math.random() * usersNr) + 1;
9 var counter = 1;
10 while(grUser.next()){
11     if(counter == randomNr){
12         userId = grUser.sys_id + '';
13         userName = grUser.name + '';
14         break;
15     }
16     counter++;
17 }
18
19 //3 - create the event
20 gs.eventQueue(eventName, current, userId, userName);
```
















- 2) Generate fake phishing - wybiera losowego użytkownika i, jeśli dany użytkownik nie otrzymał podobnej wiadomości w ciągu ostatnich 60 dni, tworzy w systemie zdarzenie „x\_401576\_phishing.fake\_required”, dane wybranego użytkownika wysyła jako parametry tworzonego zdarzenia.

Scheduled Script Execution  
Generate fake phishing

Update

Execute Now

Delete

Run this script               

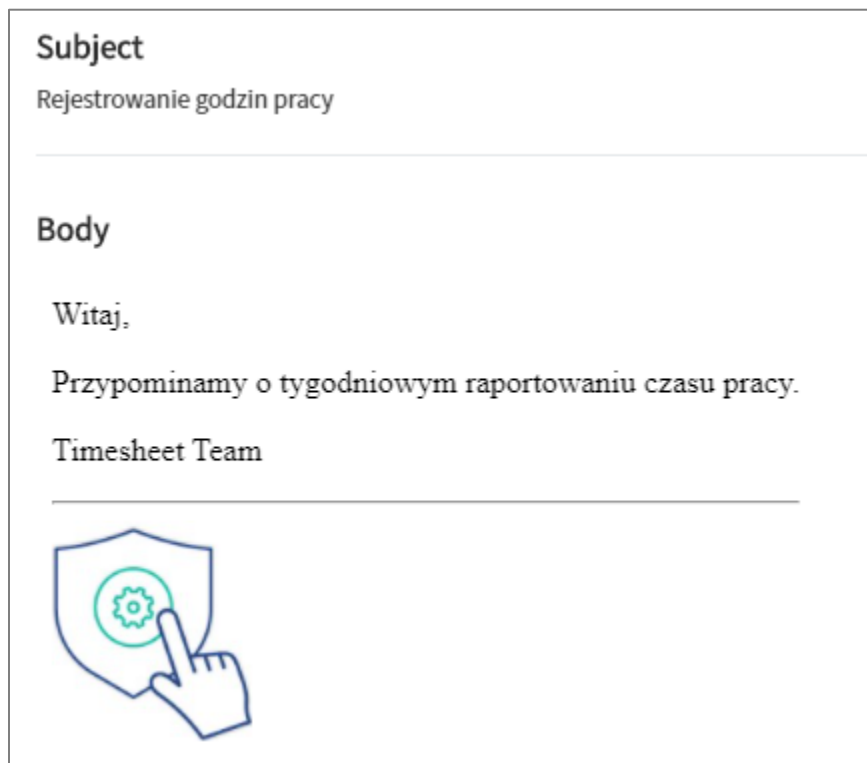
```
1 var eventName = 'x_401576_phishing.fake_required';
2 var userId;
3 var userName;
4 //1 - choose random user
5 var grUser = new GlideRecord('sys_user');
6 grUser.query();
7 var usersNr = grUser.getRowCount();
8 var randomNr = Math.floor(Math.random() * usersNr) + 1;
9 var counter = 1;
10 while(grUser.next()){
11     if(counter == randomNr){
12         userId = grUser.sys_id + '';
13         userName = grUser.name + '';
14         break;
15     }
16     counter++;
17 }
18
19 //2 - check if this event has been created for that user during last 2 months
20 var wasCreated = false;
21 var grEvent = new GlideRecord('sys_event');
22 grEvent.addQuery('sys_created_onONLast 60
days@javascript:gs.beginningOfLast60Days()@javascript:gs.endOfLast60Days()');
23 grEvent.addQuery('name', eventName);
24 grEvent.addQuery('parm1', userId);
```

```
Scheduled Script Execution
Generate fake phishing

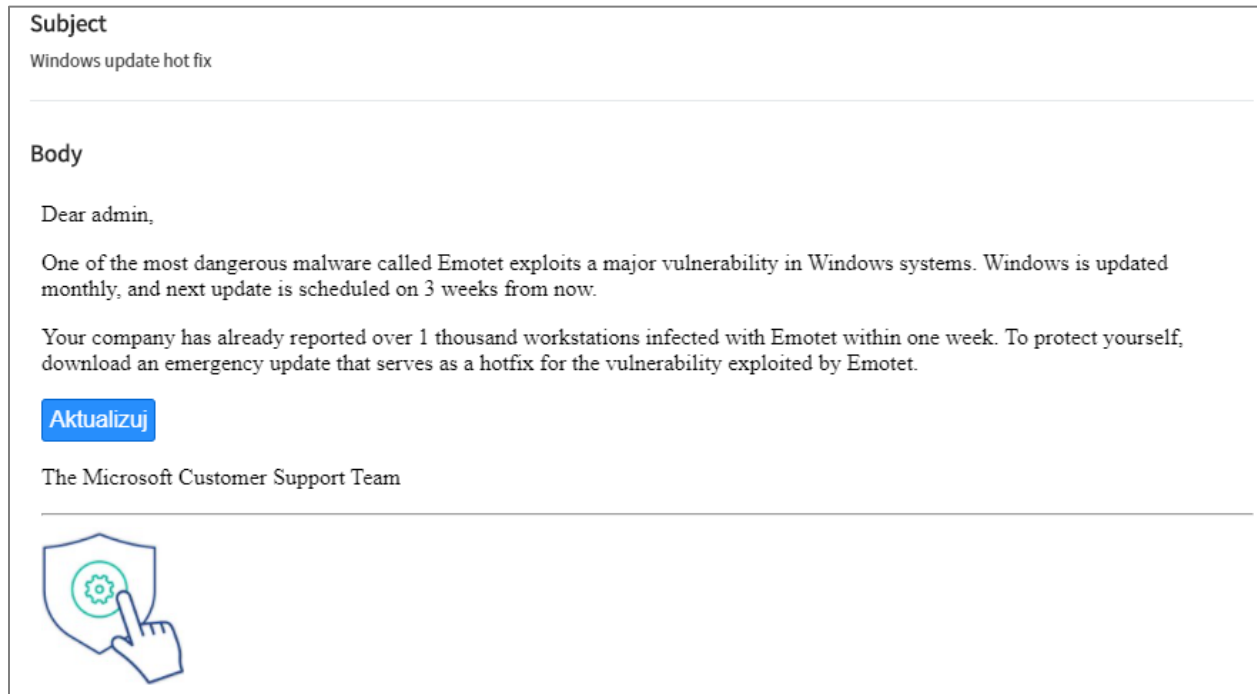
16     counter++;
17 }
18
19 //2 - check if this event has been created for that user during last 2 months
20 var wasCreated = false;
21 var grEvent = new GlideRecord('sysevent');
22 grEvent.addQuery('sys_created_onONLast 60
days@javascript:gs.beginningOfLast60Days()@javascript:gs.endOfLast60Days()');
23 grEvent.addQuery('name', eventName);
24 grEvent.addQuery('parml', userId);
25 grEvent.query();
26 if(grEvent.next()){
27     wasCreated = true;
28 }
29 gs.info("wasCreated " + wasCreated);
30 /*
31 Security restricted: Read operation against 'sysevent' from scope 'x_401576_phishing' has been
refused due to the table's cross-scope access policy
32 x_401576_phishing: 0
33 - this error forced me to check "Can read" in "Applications access" for table 'sysevent'
34 */
35
36 //3 - if the event was not created for them in that period, create it now
37 if(!wasCreated)
38     gs.eventQueue(eventName, current, userId, userName);
```

W przypadku każdego z wygenerowanych zdarzeń, triggerowana jest odpowiednia notyfikacja:

- 1) Casual notification - triggerowana na event „x\_401576\_phishing.casual\_required”, odbiorca e-maila jest odczytywany z parametru tego eventu, stworzony dla niej e-mail template to: „Casual”.



- 2) Fake phishing notification - triggerowana na event „x\_401576\_phishing.fake\_required”, odbiorca e-maila jest odczytywany z parametru tego eventu, stworzony dla niej e-mail template to: „Fake phishing”.



Jak widać na ilustracjach, pod każdym mailem znajduje się przycisk umożliwiający zgłoszenie e-maila jako niepokojącej wiadomości do celu zbadania pod kątem phishing e-mail przez organizację. Ponadto w mailu będącym testem czujności odbiorcy znajduje się przycisk „Aktualizuj”. Oba przyciski zbudowane są za pomocą połączenia odpowiedniego mail scriptu ze script includem:

ReportButtonScript + ButtonBuilder

PhishButtonScript + ButtonBuilder

Wewnątrz ButtonBuilder generowane są linki do procesora z wymaganymi parametrami:

- typ akcji – report/phishClicked,
- identyfikator notyfikacji – w celu dalszego rozpoznania, czy e-mail był testowy czy inny,



- Script Include  
ButtonBuilder

UpdateDelete

```
1 var ButtonBuilder = Class.create();  
2 ButtonBuilder.prototype = {  
3   initialize: function(notifId, recipientId) {  
4     this.notifId = notifId;  
5     this.recipientId = recipientId;  
6   },  
7  
8   getLinkPhishButton: function(){  
9     var link = this._generateLink('phishClicked');  
10    var backgroundColor = 'background-color: #278efc;';  
11    var border = 'border: 1px solid #0368d4;';  
12    var color = 'color: #ffffff;';  
13    var fontSize = 'font-size: 16px;';  
14    var fontFamily = 'font-family: Helvetica, Arial, sans-serif;';  
15    var textDecoration = 'text-decoration: none; border-radius: 3px;';  
16    var webkitBorder = '-webkit-border-radius: 3px;';  
17    var mozBorder = '-moz-border-radius: 3px;';  
18    var display = 'display: inline-block;';  
19    var padding = 'padding: 5px;';  
20  
21    var result = '<a href=' + link + ' ' ;  
22    result += 'style="' + backgroundColor + border + color + fontSize + fontFamily +  
23    textDecoration + webkitBorder + mozBorder + display + padding;  
24    result += '">;  
25    result += 'Aktualizuj';  
26    result += '</a>';  
27  
28    return result;  
29  },  
30  
31  getReportButton: function(){  
32    var link = this._generateLink('report');  
33    //var link = '';  
34    var image = '';  
35    var result = '<a href=' + link + ' >' + image + '</a>';  
36    return result;  
37    //return 'test';  
38  },  
39  
40  /**  
41   * @return {String}  
42   */  
43  _generateLink: function(actionType){  
44    var url = "";  
45    url = "phishing_processor.do?action_type=" + actionType + "&notif_id=" +  
46    this.notifId + "&recipient_id=" + this.recipientId;  
47    return url;  
48  },  
49  
50  type: 'ButtonBuilder'  
51 }
```
- Script Include  
ButtonBuilder

UpdateDelete
- ```
24 result += 'AKTUALIZUJ';  
25 result += '</a>';  
26  
27 return result;  
28  
29  
30 getReportButton: function(){  
31   var link = this._generateLink('report');  
32   //var link = '';  
33   var image = '';  
34   var result = '<a href=' + link + ' >' + image + '</a>';  
35   return result;  
36   //return 'test';  
37 }  
38  
39 /**  
40  * @return {String}  
41  */  
42 _generateLink: function(actionType){  
43   var url = "";  
44   url = "phishing_processor.do?action_type=" + actionType + "&notif_id=" +  
45   this.notifId + "&recipient_id=" + this.recipientId;  
46   return url;  
47 }  
48  
49 type: 'ButtonBuilder'  
50  
51 }
```

- 1) Przycisk zgłoszenia phishing e-mail – wywołanie procesora PhishingEmail z akcją typu „report”.
- 2) Przycisk „Aktualizuj” - wywołanie procesora PhishingEmail z akcją typu „phishClicked”.

Kolejna część procesu jest wykonywana przez procesor PhishingEmail. Na podstawie danych zebranych z parametrów procesor decyduje się na jedną z trzech możliwych ścieżek:

- 1) Testowy link został kliknięty – generowane jest zdarzenie „x\_401576\_phishing.link\_clicked”. Na to zdarzenie triggerowana jest notyfikacja „Phishing link clicked”. Z parametru zdarzenia odczytywany jest odbiorca e-maila.



- 2) E-mail został zgłoszony i pochodzi z testowej definicji notyfikacji – generowane jest zdarzenie „x\_401576\_phishing.fake\_phish\_reported”. Na to zdarzenie triggerowana jest notyfikacja „Fake phish reported”. Z parametru zdarzenia odczytywany jest odbiorca e-maila.

**Subject**

Test phishing email - wynik

---

**Body**

Dzień dobry,

Dziękujemy za zgłoszenie phishing e-mail. To był test przygotowany przez nasz zespół. Gratulujemy czujności!

Pozdrawiamy,  
Security Team

---



- 3) E-mail został zgłoszony i nie pochodzi z testowej definicji notyfikacji – procesor tworzy w systemie incydent i wysyła e-maila z informacją o nim do użytkownika. Incydent zostanie przypisany do odpowiedniego zespołu i pracownicy organizacji zajmą się analizą zagrożenia. Użytkownik, widząc maila z numerem incydu, wie, że jego wątpliwości zostały podjęte przez organizację i jest w stanie śledzić postępy sprawy – posiada numer incydu, a nawet bezpośredni link w e-mailu.

**Subject**

Incydent INC0000060 stworzony - phishing email

---

**Body**

Dzień dobry,

Incydent INC0000060 został utworzony w Twoim imieniu w związku ze zgłoszeniem phishing email. Aby śledzić status prac nad incydem, użyj poniższego linku:

[INC0000060](#)

Pozdrawiamy,  
Security Team






---



<

☰

Incident  
INC0010008



Follow

▼

Update

Resolve

Delete

Number

INC0010008

Contact type

-- None --

▼

\* Caller

Michael Hoefer

🔍

📄

ℹ️

State

New

▼

Category

Inquiry / Help

▼

Impact

3 - Low

▼

Subcategory

-- None --

▼

Urgency

3 - Low

▼

Service

🔍

Priority

5 - Planning

Configuration item

🔍

Assignment group

🔍

Assigned to

🔍

\* Short description

Phishing email

🔍

Description

Phishing email has been reported. Sys id of the email is e618c99cdbf82010542c8a18489619a2

## **4. Podsumowanie**

Aplikacja jest rozwiązaniem „zaszytym” niejako w platformie ServiceNow, której popularność rośnie z roku na rok i jest pewnym POC mogącym zostać rozwiniętym w przyszłości i służyć rzeczywistym użytkownikom. Rozwiązanie jest zaprojektowane i zaimplementowane w sposób umożliwiający prezentację pomysłu. Maile i zdarzenia generowane są umownie, według założeń na potrzeby projektu.