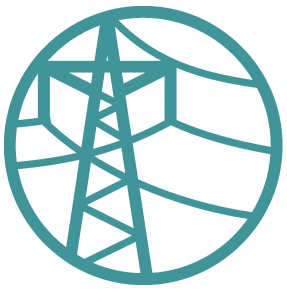


WYDZIAŁ
ELEKTROTECHNIKI
I INFORMATYKI
POLITECHNIKI RZESZOWSKIEJ

Motyka Beniamin

Bezpieczeństwo IT dla firm - opis i implementacja

Praca dyplomowa inżynierska

Opiekun pracy:
dr Michał Piętal

Rzeszów, 2021

Spis treści

1. Wprowadzenie

W dzisiejszych czasach śmiało można stwierdzić, iż Internet stał się ważną częścią ludzkiego istnienia. Niewątpliwy wpływ na ten stan rzeczy miała pandemia COVID-19 - sprawiła ona bowiem, że pewne dziedziny życia, takie jak dydaktyka czy praca wykonywana umysłowo, przeszły swoistą transformację. Miejsca, w których spotykali się studenci wraz z wykładowcami, czy pracownicy w biurze, stały się puste. Zastąpiła je komunikacja zdalna – przez Internet.

Fakt, iż ludzkość została zmuszona, by przenieść znaczną część swojego funkcjonowania w sieć, niesie ze sobą poważne konsekwencje. Szybkie – jak do tej pory – tempo rozwijania się technologii informatycznych stało się nieporównywalnie bardziej dynamiczne, a co się z tym wiąże, obecne w sieci liczne zagrożenia stały się coraz powszechniejsze i trudniejsze w identyfikacji.

[coś tu jeszcze będzie]

1.1. Cel i zakres pracy

Celem niniejszej pracy inżynierskiej jest wyeksponowanie, opis oraz implementacja najbardziej pospolitych zagrożeń i luk bezpieczeństwa w Internecie nie tylko dla zwykłych użytkowników, ale również dla małych i średnich przedsiębiorstw. Dzięki temu, że powyższa idea zostanie zrealizowana w formie aplikacji e-learningowej, istnieje realna szansa na zwiększenie świadomości społecznej, edukację oraz poprawę zabezpieczeń systemów teleinformatycznych i infrastruktury sieciowej. Zakresem pracy są takie zagadnienia jak:

- Przegląd, dokumentacja i implementacja zagrożeń i luk bezpieczeństwa w sieci.
- Stworzenie aplikacji e-learningowej przy użyciu technologii opisanych w kolejnym rozdziale. Użytkownicy będą mogli wziąć udział w interaktywnej prezentacji każdego z zagrożeń.
- Zasugerowanie potencjalnych rozwiązań na opisane cyberzagrożenia.
- Przedstawienie wniosków i implikacji płynących z powyższych.

[coś tu jeszcze będzie]

2. Technologie wykorzystane w aplikacji

2.1. Javascript

JavaScript to skryptowy język programowania umożliwiający zaimplementowanie na stronach internetowych logiki, przez co strona może nie tylko wyświetlać statyczne treści, ale również wykonywać akcję, która stoi za wciśnięciem przycisku, kontrolować multimedia, obsługę dynamicznego tworzenia treści... Jest to trzecia warstwa standardowych technologii internetowych, do których należą jeszcze HTML i CSS [2].

Aplikacja Internetowa opisana w tej pracy inżynierskiej składa się z dwóch projektów, które zostały wykonane w całości w języku JavaScript, zarówno po stronie klienta - używając biblioteki React.js, jak i po stronie serwera - przy wykorzystaniu środowiska Node.js.

[coś tu jeszcze będzie]

2.1.1. React.js

React to darmowa i otwartoźródłowa biblioteka stworzona i wspierana przez programistów firmy Facebook. Jego podstawowym celem jest rozdzielenie części interfejsu użytkownika na komponenty - samodzielne elementy kodu, które mogą być złączone w całość, aby stworzyć pełny widok interfejsu. Oprócz tego, zadaniem biblioteki jest zarządzanie stanem treści na stronie i renderowaniem wybranych elementów do drzewa DOM.

Jednym z kluczowych elementów tej biblioteki jest JSX - JavaScript XML. Rozszerza to język programowania JavaScript o elementy HTML, dzięki czemu do zmiennych można przypisywać fragmenty kodu:

```
1  const pageHeader = <h2>Header of a page</h2>;
```

Listing 1: Nagłówek strony w wykonany w składni JSX

Zmienne te mogą być odpowiednio renderowane i stylowane, co zapewnia zorganizowaną strukturę aplikacji.

Oprócz zastosowania w implementacji interfejsów użytkownika dla aplikacji Internetowych, biblioteki tej można również używać w celu tworzenia w pełni funkcjonujących aplikacji mobilnych (React Native).

2.1.2. Node.js

Z racji rosnącego zainteresowania JavaScript, zaczęto się zastanawiać nad możliwością użycia tego języka programowania poza przeglądarką Internetową. W 2008 roku narodziła się idea, dzięki której stworzona została platforma uruchomieniowa na bazie silnika V8 przeglądarki Google Chrome - Node.js. Pozwoliło to na używanie JavaScript poza przeglądarką, do budowania aplikacji po stronie serwera, wykorzystując tym samym wszystkie zalety tego języka.

2.2. GraphQL

GraphQL to język zapytań, który udostępnia wspólny interfejs pomiędzy klientem a serwerem do pobierania i zarządzania danymi. Stworzony został w 2012 roku przez inżynierów Facebooka, w 2015 roku stał się otwartoźródłowy. Pozwala stronie klienta i serwera na dostęp do danych z wykorzystaniem mniejszej ilości zasobów niż w tradycyjnym REST API. Istotnym jego elementem jest silne typowanie, dzięki któremu definiowane są typy danych wymienialnych pomiędzy klientem a serwerem. GraphQL skupia się bardziej na pobieraniu elementów, podczas gdy REST używany jest do nadania struktury usługom sieciowym.

W odróżnieniu do API wykonanego w architekturze REST, GraphQL używa tylko jednej metody HTTP do otrzymywania, wysyłania, aktualizowania czy usuwania danych - metody POST. W tej metodzie, wysyłanej na jeden, wspólny dla wszystkich zapytań adres (zwyczajowo nazywany /graphql) w ciele żądania przesyłane jest 'query expression', czyli wyrażenie zapytania. Tylko i wyłącznie z ciała żądania można wywnioskować, jakiego rodzaju akcja musi zostać podjęta, oraz na jakie zasoby. Zawsze wybierany jest rodzaj operacji, jaka ma być przetworzona. Dwie najczęściej wykonywane operacje to:

- query – jest podstawową operacją, znajdującą się praktycznie w każdej aplikacji. Informuje API, że to zapytanie oczekuje jedynie na odpowiedź, nie wprowadza zmian w bazie. W rezultacie, zwracane są dane znajdujące się w żądaniu.

- mutation – operacja rodzaju **CUD** (**C**reate, **U**pdate, **D**elete), oczekuje specyficznych danych, aby wykonać wcześniej zadeklarowaną funkcję.

Zapytania GraphQL nie są w żaden sposób szybsze niż zapytania REST – cechuje je jednak brak redundancji, gdyż można z nich wybrać konkretne pola, które będą

zwrócone. W związku z tym, żądania GraphQL są zawsze mniejsze i bardziej wydajne niż zapytania REST, w których to często zwracane są redundantne dane.

2.3. MongoDB

MongoDB to nierelacyjna baza danych zorientowana na obiekty zwane dokumentami. W przeciwieństwie do baz relacyjnych, nie używa ona tabel, rzędów czy kolumn do składowania i odczytu danych. W zamian wykorzystuje kolekcje - w nich znajdują się zestawy dokumentów i funkcji, które odpowiadają tabelom w relacyjnych bazach danych. Dokumenty składają się z par klucz-wartość. Są to podstawowe jednostki danych używane w MongoDB.

Nierelacyjne bazy danych - w przeciwieństwie do relacyjnych - cechują się rozszerzalnością horyzontalną, czyli możliwością rozbudowy bazy o dodatkową pamięć masową.

[coś tu jeszcze będzie, wiem że to nie ma sensu narazie]

Skalowalność wertykalna z kolei skupia się na zwiększeniu mocy obliczeniowej serwera bazy danych, poprzez ulepszenie procesora czy zwiększenie pamięci operacyjnej co generuje znacznie większe koszty, szczególnie przy bazach danych składających się z milionów rekordów.

3. Zagrożenia bezpieczeństwa

tekst

3.1. Phishing - opis

tekst

3.2. Phishing - przykład implementacji

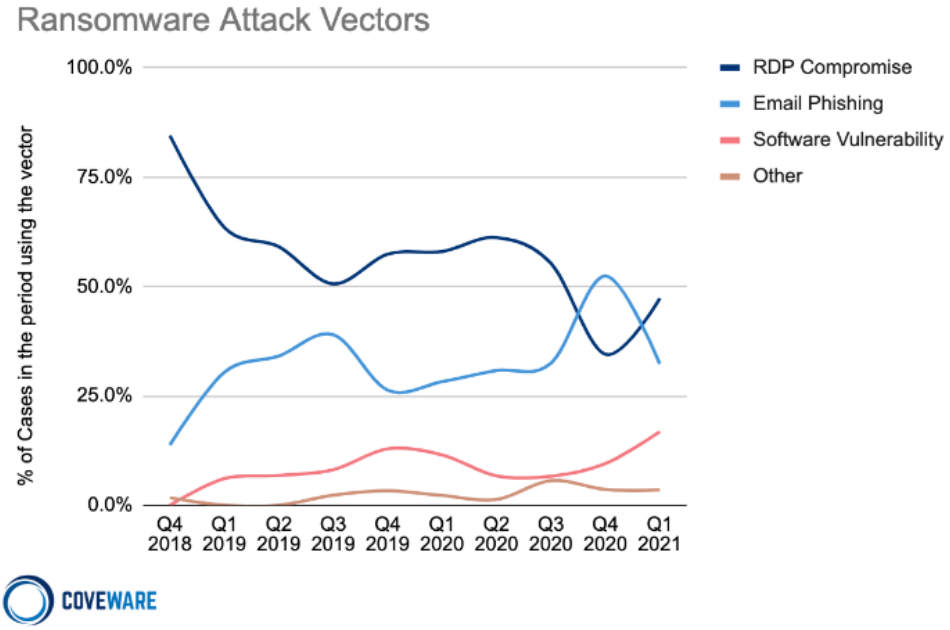
tekst

3.3. Ransomware - opis

Ransomware to typ złośliwego oprogramowania, którego celem jest zablokowanie dostępu do komputera osobistego poprzez zaszyfrowanie wszystkich możliwych plików. Czas trwania tej procedury zależy między innymi od wybranego algorytmu szyfrowania lub danych na urządzeniu, jednak może wykonać się w czasie zaledwie pięciu godzin [8]. Zazwyczaj ten czas jest znacznie dłuższy, gdyż przed rozpoczęciem procesu wymagany jest dokładny rekonesans systemu, w którym ransomware się znajduje. Ostatecznie rozpoczynany jest proces szyfrowania asymetrycznego. Tym sposobem użytkownik końcowy traci możliwość odczytu danych na swoim urządzeniu, a do odszyfrowania plików, wymagany jest klucz posiadany przez atakującego.

Ten rodzaj szkodliwego oprogramowania jest szczególnie niebezpieczny dla przedsiębiorstw, gdyż utrata ważnych dokumentów czy danych finansowych, może się wiązać z poważnymi konsekwencjami. Celem ransomware nie jest usunięcie lub kradzież plików, ale zablokowane ich, i oczekiwanie na ewentualną spłatę okupu ze strony ofiary.

Sposoby, poprzez które ransomware dostaje się do przedsiębiorstw, przedstawione są na poniższym rysunku, przygotowanym przez organizację CoveWare [7], która specjalizuje się w przeciwdziałaniu tego typu oprogramowaniu:



Rysunek 3.1: Wektory ataku ransomware

Jak można zauważyć, wiodącym źródłem pozyskania tego złośliwego oprogramowania jest Remote Desktop Protocol (RDP) [6] - protokół umożliwiający połączenie się z urządzeniem, oraz przejęcie nad nim pełnej kontroli, używany w firmach jako narzędzie do konfiguracji urządzeń. Zaraz za nim znajdują się maile phishingowe, opisane w poprzednim rozdziale, a następnie różnego rodzaju luki w oprogramowaniu.

Najlepszą metodą przeciwko tego typu złośliwemu oprogramowaniu jest systematyczne wykonywanie archiwizacji danych. W tym wypadku, jeśli ransomware trafi do komputera i zaszyfruje wszystkie pliki, pozostaje jedynie odtworzyć kopię zapasową.

[coś tu jeszcze będzie]

3.4. Ransomware - przykład implementacji

tekst

3.5. Keylogger - opis

Keyloggery występują zazwyczaj w formie złośliwego, ukrytego oprogramowania. Nie są widoczne na pierwszy rzut oka dla użytkownika i w zdecydowanej większości przypadków działają w tle, często podszywając się pod inną aplikację, tym samym maskując swoją obecność.

Podstawowe właściwości tego typu programu można opisać jako przejmowanie kontroli nad procedurami związanymi z obsługą klawiatury systemu operacyjnego, na którym się znajduje.

Głównym celem tego oprogramowania jest zbieranie danych o tym, jakie klawisze na klawiaturze zostały naciśnięte przez użytkownika, a następnie okresowe wysyłanie zebranych informacji do atakującego. Posiadając wiedzę na temat tego, co zostało wpisane na urządzeniu, można bez problemu uzyskać dostęp do wrażliwych informacji takich jak prywatna korespondencja czy poufne dane. Do bardziej zaawansowanych funkcji należy między innymi przesyłanie zrzutów ekranu, rejestrowanie historii otwieranych programów i przekazywanie tych informacji dalej.

Keyloggery oprócz formy programowej istnieją również jako osobne urządzenia, które podłączane są do jednostki zazwyczaj poprzez interfejs USB. Mogą także występować jako urządzenie pośredniczące pomiędzy klawiaturą a złączem USB komputera.

Sposobem na unikanie tego typu oprogramowania jest przede wszystkim systematyczne sprawdzanie uruchomionych procesów, ale także używanie odpowiedniego

antywirusa.

3.6. Keylogger - przykład implementacji

tekst

3.7. Wstrzyknięcie SQL - opis

tekst

3.8. Wstrzyknięcie SQL - przykład implementacji

tekst

3.9. DOS - opis

Celem denial-of-service są zazwyczaj serwisy Internetowe małych i średnich przedsiębiorstw. Atak ten polega na wykonaniu tak wielu żądań do serwera, aby ten przestał odpowiadać. Rodzaj DOS może różnić się w zależności od warstwy modelu OSI na której działa:

- Warstwa sieci - ICMP Flood, Smurf Attack
- Warstwa transportowa - SYN Flood, UDP Flood
- Warstwa aplikacji - HTTP Flood

Blokada danego adresu IP, z którego przychodzi wiele żądań w krótkim okresie czasu, nie stanowi większego problemu dla firewalli, dlatego też coraz powszechniejszymi stają się ataki DDOS - distributed denial-of-service. Różnica polega na tym, że żądania wysyłane są z wielu lokacji jednocześnie, co znacznie bardziej utrudnia identyfikację i zablokowanie nagłego ruchu przez firewall.

[coś tu jeszcze będzie]

Potencjalnym rozwiązaniem na tego typu ataki może być użycie modułu równoważenia obciążenia - load balancera oraz odpowiednio skonfigurowanego serwera pośredniczącego pomiędzy klientem a serwerem - reverse proxy.

3.10. DOS - przykład implementacji

tekst

3.11. Ataki XSS - opis

Cross-site scripting (XSS) jest jedną z najbardziej powszechnych podatności współczesnych aplikacji webowych.

Opisując XSS nie sposób nie wspomnieć o Regule tego samego pochodzenia (Same-origin policy) [3]. Jest to jeden z wielu fundamentalnych mechanizmów bezpieczeństwa, zaimplementowany w każdej przeglądarce internetowej. Nie pozwala on żadnej stronie na podjęcie akcji lub odczytania zawartości innej strony, na przykład w dwóch kartach przeglądarki. W związku z tym, wszystko, co się dzieje na stronie internetowej otwartej przez użytkownika, jest izolowane, i nie będzie miało wpływu na pozostałe otwarte strony.

Cała struktura strony internetowej zakodowana językiem HTML może być zmieniana poprzez JavaScript, używając DOM API [4]. Jako rezultat, prosty skrypt może całkowicie zmienić zawartość, wygląd, a przede wszystkim funkcjonalność strony internetowej.

Ciasteczkami [5] nazywa się niewielkie informacje wysyłane przez serwer do przeglądarki internetowej urządzenia końcowego. Służą temu, by zapisać obiekty danych w przeglądarce, które przy ponownym odwiedzeniu strony, mogą być przesłane do tego samego serwera, z którego przyszły. W związku z tym, przy odwiedzaniu różnorakich stron wymagających autoryzacji, użytkownik nie musi się za każdym razem logować, gdyż wszystkie potrzebne dane są zawarte w ciasteczkach, które przesyłane są razem z żądaniem.

XSS opiera się głównie na wstrzyknięciu do strony internetowej złośliwego skryptu, który, dla przykładu, może odczytać ciasteczka użytkownika lub inne poufne informacje, które przechowuje przeglądarka, wysłać je do atakującego, aby ten – używając zapisanych w ciasteczkach danych – mógł zalogować się na konto użytkownika, który nieświadomie uruchomił dany skrypt. Pomimo wielu zabezpieczeń, które są wbudowane w przeglądarki, te nie są w stanie odróżnić czy dany skrypt jest złośliwy, czy nie - dlatego też odporność aplikacji internetowej na tego typu atak stoi przede wszystkim po stronie programistów.

Najczęstszym miejscem, w którym można spotkać tę podatność, są formularze, do których użytkownik wpisuje jakąś treść, która następnie jest wyświetlana. Jeśli treść którą przesłał użytkownik nie zostanie odpowiednio okrojona, może dojść do sytuacji,

w której to użytkownik wstrzyknie złośliwy skrypt.

[coś tu jeszcze będzie]

3.12. Ataki XSS - przykład implementacji

tekst

3.13. Path Traversal - opis

"Path Traversal"jest to nazwa dla powszechnej luki bezpieczeństwa aplikacji Internetowych, której niedopatrzenie, w procesie tworzenia oprogramowania, może skutkować w wycieku wrażliwych danych z serwera, na którym umieszczona jest aplikacja.

Dzięki językom programowania działających po stronie serwera takim jak PHP zewnętrzne skrypty i pliki mogą być dołączane do aplikacji w sposób dynamiczny. Krytycznym elementem w tego typu funkcjonalnościach jest odpowiednio zaimplementowana logika dołączania plików oraz walidacja danych wejściowych, gdyż w przeciwnym wypadku, atakujący może odczytywać zawartość plików lokalnych, jak i zdalnych. Celem tej podatności jest zlokalizowanie i odpowiednie wykorzystanie parametrów przekazywanych do aplikacji, poprzez które dynamicznie dołączane są różne skrypty.

Poniższy przykład prezentuje dołączenie pliku 'pl.php' jako parametr, celem ustawienia tekstu na stronie w języku polskim:

`http://samplepage.com/index.php?lang=pl.php`

Może to świadczyć o tym, że przyłączenie pliku do w kodzie źródłowym aplikacji wykonywane jest w następujący, przykładowy sposób:

```
1  include($_GET['lang']);
```

Listing 2: Listing programu PHP

Mając na uwadze fakt, że do aplikacji dynamicznie dołączany jest skrypt przekazywany w parametrze 'lang', atakujący może dokonać próby otworzenia zupełnie innego pliku, niż było to zakładane:

`http://samplepage.com/index.php?lang=../../../etc/passwd`

W rezultacie, zamiast wypisania tekstu w języku polskim, przekazana zostanie zawartość pliku '/etc/passwd', w którym to znajdują się informacje o wszystkich użytkownikach w systemie.

[coś tu jeszcze będzie]

Aby zapobiec tej luce, zamiast wykluczać lub usuwać dane ciągi znaków należy odpowiednio sprawdzać dane wejściowe i zezwalać jedynie na wybrane znaki, lub ciągi znaków, na przykład z wykorzystaniem wyrażeń regularnych.

3.14. Path Traversal - przykład implementacji

tekst

3.15. Spoofing - opis

tekst

3.16. Spoofing - przykład implementacji

tekst

4. Podsumowanie i wnioski końcowe

Literatura

[1] Michał Bentkowski, Gynvael Coldwind , Artur Czyż, Rafał Janicki, Jarosław Kamiński, Adrian Michalczyk, Mateusz Niezabitowski, Marcin Piosek, Michał Sajdak, Grzegorz Trawiński, Bohdan Widła. Bezpieczeństwo aplikacji webowych //tak, ta książka ma tylu autorów. jak to zapisać?

[2] https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScri

[3] https://developer.mozilla.org/en-US/docs/Web/Security/Same-origin_policy

[4] https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model.

[5] https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies.

[6] https://docs.microsoft.com/en-us/troubleshoot/windows-server/remote/understanding-remote-desktop-protocol

[7] https://www.coveware.com/blog/ransomware-attack-vectors-shift-as-new-software-vulnerability-exploits-abound

[8] https://thedfirreport.com/2020/10/18/ryuk-in-5-hours/

[9] https://www.cloudflare.com/learning/ddos/what-is-a-ddos-attack/

[10] https://www.educative.io/blog/what-is-nodejs

POLITECHNIKA RZESZOWSKA im. I. Łukasiewicza Rzeszów, 2021
Wydział Elektrotechniki i Informatyki

STRESZCZENIE PRACY DYPLOMOWEJ INŻYNIERSKIEJ

BEZPIECZEŃSTWO IT DLA FIRM - OPIS I IMPLEMENTACJA

Autor: Motyka Beniamin, nr albumu: EF-160780
Opiekun: dr Michał Piętal
Słowa kluczowe: (max. 5 słów kluczowych w 2 wierszach, oddzielanych przecinkami)

Treść streszczenia po polsku

RZESZOW UNIVERSITY OF TECHNOLOGY Rzeszow, 2021
Faculty of Electrical and Computer Engineering

BSC THESIS ABSTRACT

TEMAT PRACY PO ANGIELSKU

Author: Motyka Beniamin, nr albumu: EF-160780
Supervisor: (academic degree) Imię i nazwisko opiekuna
Key words: (max. 5 słów kluczowych w 2 wierszach, oddzielanych przecinkami)

Treść streszczenia po angielsku