Łukasz Krzywiecki

Department of Computer Science, FFPT, Wrocław University of Science and Technology

Contact Teaching Research Activity

Timetable

Monday

09:15-11:00

Net. Tech. (lab), 317.2 (D-1)

11:00-13:00

Consultations, 210 (D-1)

Tuesday

07:30-09:00

Net. Tech. (lab), 317.3 (D-1)

09:00-11:00

Consultations, 210 (D-1)

11:00-13:00

Sys. Sec. I (lec), 202 (C-7)

13:00-15:00

Sys. Sec. I (exe), 303 (C-7)

Wednesday

13:15-15:00

Net. Tech. (lec TP), 1.30 (C-13)

15:15-17:00

ML Sec. (semi TP), 214a (D-1)

Thursday

11:15-13:00

Sec. Project, 210 (D-1)

Friday

11:15-13:00

Sys. Sec. I, (lab), 305 (D-20)

13:15-15:00

Sys. Sec. I, (lab), 305 (D-20)

15:00-17:00

Consultations, 210 (D-1)

Students activity

Research papers with students

Łukasz Krzywiecki, Mirosław Kutyłowski, Rafał Rothenberger. **Hierarchical Ring Signatures Immune to Randomness Injection Attacks.** <u>Paper under submission: 2020</u>

Łukasz Krzywiecki, Adam Połubek. **Multi-Signature Scheme Resistant to Randomness Injection Attacks - Bitcoin Case.** <u>Paper under submission:</u> 2020.

Łukasz Krzywiecki, Mirosław Kutyłowski, Jakub Pezda, Marcin Słowik. **Anonymous Deniable Identification in Ephemeral Setup and Leakage Scenarios** (Brief Announcement). <u>CSCML 2019: 320-323.</u>

Łukasz Krzywiecki, Marta Słowik, Michał Szala. Identity-Based Signature Scheme Secure in Ephemeral Setup and Leakage Scenarios. ISPEC 2019: 310-324.

Patryk Kozieł, Łukasz Krzywiecki, Damian Stygar. Identity-based Conditional Privacy-Preserving Authentication Scheme Resistant to Malicious Subliminal Setting of Ephemeral Secret. ICETE (2) SECRYPT 2019: 492-497.

Łukasz Krzywiecki, Tomasz Wlisłocki. **Deniable key establishment resistant against eKCI attacks.** <u>Security and Communication Networks, vol. 2017, Article ID 7810352, 13 pages, 2017. Wiley|Hindawi.</u>

Students cryptography projects

Proof-of concept implementation of the domain-specyfic pseudonymous signatures scheme (doi: 10.1007/978-981-13-1483-4 8). Demo

Rust binding to the MCL library. Crate repository

Node.js (express.js) server, using MCL library, for testing cryptographic schemes. Github repository

Python binding to the MCL library (ver 1). Github repository

Python binding to the MCL library (ver 2). Github repository

Courses

TS Sys. Sec. I Sys. Sec. II Cloud Sec. Diploma Seminar

Technologie Sieciowe: Listy zadań



Lista nr 1 - Wstęp

- 1. Przetestuj działanie programów:
 - a) Ping: Sprawdź za jego pomocą ile jest węzłów na trasie do (i od) wybranego, odległego geograficznie, serwera. Uwaga: trasy tam i z powrotem mogą być różne. Zbadaj jaki wpływ ma na to wielkość pakietu. Zbadaj jak wielkość pakietu wpływa na obserwowane czasy propagacji. Zbadaj jaki wpływ na powyższe ma konieczność fragmentacji pakietów. Jaki największy niefragmentowany pakiet uda się przesłac. Przeanalizuj te same zagadnienia dla krótkich tras (do serwerów bliskich geograficznie). Określ "średnicę" internetu (najdłuższą sćieżkę którą uda sie wyszukać). Czy potraficz wyszukać trasy przebiegające przez sieci wirtualne (zdalne platformy "cloud computing"). Ile węzłów mają scieżki w tym przypadku.
 - b) Traceroute,
 - c) WireShark.
- Napisz sprawozdanie zawierające: opis programów, wywołania dla powyższych zagadnień z analizą wyników, wnioski dotyczące przydatności tych programów.

Lista nr 2 - Model

- Rozważmy model sieci S = <G,H>. Przez N=[n(i,j)] będziemy oznaczać macierz natężeń strumienia pakietów, gdzie element n(i,j) jest liczbą pakietów przesyłanych (wprowadzanych do sieci) w ciągu sekundy od źródła v(i) do ujścia v(j).
 - Zaproponuj topologię grafu G ale tak aby żaden wierzchołek nie był izolowany oraz aby: |V|=20, |E|<30.
 Zaproponuj N oraz następujące funkcje krawędzi ze zbioru H: funkcję przepustowości 'c' (rozumianą jako maksymalną liczbę bitów, którą można wprowadzić do kanału komunikacyjnego w ciągu sekundy), oraz funkcję

przepływu 'a' (rozumianą jako faktyczną liczbę pakietów, które wprowadza się do kanału komunikacyjego w ciągu sekundy). Pamiętaj aby funkcja przeplywu realizowała macierz N oraz aby dla każdego kanału 'e' zachodziło: c(e) > a(e).

- Niech miarą niezawodności sieci jest prawdopodobieństwo tego, że w dowolnym przedziale czasowym, nierozspójniona sieć zachowuje T < T_max, gdzie: T = 1/G * SUM_e(a(e)/(c(e)/m a(e))), jest średnim opóźnieniem pakietu w sieci, SUM_e oznacza sumowanie po wszystkich krawędziach 'e' ze zbioru E, 'G' jest sumą wszystkich elementów macierzy natężeń, a 'm' jest średnią wielkością pakietu w bitach. Napisz program szacujący niezawodność takiej sieci przyjmując, że prawdopodobieństwo nieuszkodzenia każdej krawędzi w dowolnym interwale jest równe 'p'. Uwaga: 'N', 'p', 'T_max' oraz topologia wyjsciowa sieci są parametrami.</p>
- Przy ustalonej strukturze topologicznej sieci i dobranych przepustowościach stopniowo zwiększaj wartości w macierzy natężeń. Jak będzie zmieniać się niezawodność zdefiniowana tak jak punkcie poprzednim (Pr[T < T_max]).
- Przy ustalonej macierzy natężeń i strukturze topologicznej stopniowo zwiększaj przepustowości. Jak będzie zmieniać się niezawodność zdefiniowana tak jak punkcie poprzednim (Pr[T < T_max]).

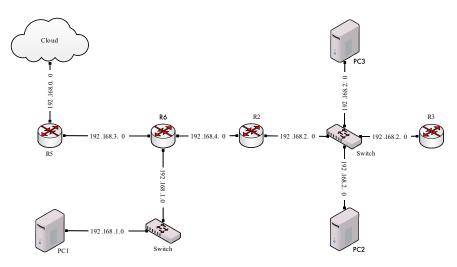
Napisz sprawozdanie zawierające opis zrealizowanych programów, komentarz do przeprowadzonych badań oraz wnioski.

Lista nr 3 - Ramkowanie

- 1. Napisz program ramkujący zgodnie z zasadą "rozpychania bitów" (podaną na wykładzie), oraz weryfikujacy poprawność ramki metodą CRC . Program ma odczytywać pewien źródłowy plik tekstowy 'Z' zawierający dowolny ciąg złożony ze znaków '0' i '1' (symulujacy strumień bitów) i zapisywać ramkami odpowiednio sformatowany ciąg do inngo pliku tekstowego 'W'. Program powinien obliczać i wstawiać do ramki pola kontrolne CRC formatowane za pomocą ciągów złożonych ze znaków '0' i '1'. Napisz program, realizujacy procedure odwrotną, tzn. który odzczytuje plik wynikowy 'W' i dla poprawnych danych CRC przepisuje jego zawartość tak, aby otrzymać kopię oryginalnego pliku źródłowego 'Z'.
- 2. Napisz program (grupę programów) do symulowania ethernetowej metody dostepu do medium transmisyjnego (CSMA/CD). Wspólne łącze realizowane jest za pomocą tablicy: propagacja sygnału symulowana jest za pomoca propagacji wartości do sąsiednich komórek. Zrealizuj ćwiczenie tak, aby symulacje można było w łatwy sposób testować i aby otrzymane wyniki były łatwe w interpretacji.

Lista nr 4 - Konfiguracja sieci IP

W symulatorze GNS3 skonfiguruj wirtualną sieć o podanej topologii, tak aby:



- 1. Wirtualna sieć była połączona z zewnętrzną ('fizyczną') siecią 'Cloud'.
- 2. Ruter R5 uzyskiwał dynamiczny adres IP z sieci 'Cloud'.
- 3. Pozostałe urządzenia posiadały statyczne adresy w swoich sieciach.
- 4. Możliwe było wysyłanie komunikatów "ping" pomiędzy dowolna parą urządzeń sieci wirtualnej.
- 5. Możliwe było wysyłanie komunikatów "ping" z dowolnego urządzenia w sieci wirtualnej na zewnętrzny adres, np. 'google.com'.
- 2. Ustaw przechwytywanie komunikatów w sieciach: 192.168.0.0, 192.168.2.0, 192.168.3.0.
- 3. Przeanalizuj przechwycone komunikaty dla zapytania wysłanego z komputera PC2: 'ping google.com'.

Lista nr 5 - HTTP

- 1. Plik <u>server3.pl</u> zawiera przykładowy program serwera protokołu HTTP.
 - 1. Uruchom ten skrypt, przetestuj, zastanów się jak działa.
 - 2. Nawiąż połączenie za pomocą przeglądarki internetowej.
 - 3. Zmień skrypt (lub napisz własny serwer w dowolnym języku programowania) tak aby wysyłał do klienta nagłówek jego żądania.
 - 4. Zmień skrypt (lub napisz własny serwer w dowolnym języku programowania) tak aby obsugiwał żądania klienta do prostego tekstowego serwisu WWW (kilka statycznych ston z wzajemnymi odwołaniami) zapisanego w pewnym katalogu dysku lokalnego komputera na którym uruchomiony jest skrypt serwera.
 - 5. Przechwyć komunikaty do/od serwera za pomocą analizatora sieciowego przeanalizuj ich konstrukcję.

Requirements

Students should solve, deliver and explain personally the solutions for the list of excercises to the tutor/superviser of the lab. A list can be passed only if all its exercises are solved. Please do not email the solutions. A student will pass the course only if all the lists are solved and delivered before the end of the semester.

© 2020 Łukasz Krzywiecki. Coded with W3.CSS