Ewelina Jasińska
Wydział Informatyki i Zarządzania
Wyższej Szkoły Informatyki w Łodzi

Promotor: dr hab. Marek Rudnicki, prof. WSInf

SYSTEMY IMMUNOLOGICZNE W PROBLEMACH BEZPIECZEŃSTWA SYSTEMÓW INFORMATYCZNYCH

Streszczenie – Celem artykułu jest omówienie budowy systemu chroniącego komputer przed zagrożeniami z wykorzystaniem algorytmu opartego o system immunologiczny człowieka. Pomysł stworzenia systemu opartego o immunologię, autorstwa prof. dr hab. Marka Rudnickiego, okazał się znakomitym rozwiązaniem. Połączenie głównej idei rozpoznawania własnych elementów, z utworzeniem „terenu zagrożenia", pozwoliło na wytworzenie programu posiadającego dużą skuteczność a jednocześnie nieobciążającego zasobów systemowych. System posiada jedynie początkową bazę danych (podobnie jak antygeny chroniące organizm dziecka, zawierają informację genetyczną pochodzącą od matki), którą w miarę funkcjonowania uzupełnia, emulując proces uczenia się. Program samodzielnie rozpoznający zagrożenia, korzystający jedynie z „inteligencji", którą dał mu programista i którą może wykorzystywać do wykrywania nowych form ataków, oraz dostosowujący się do działającego systemu jest marzeniem każdego administratora.

1 Wstęp

System immunologiczny, rozumiany, jako odporność na zachorowania, opisano po raz pierwszy już około III w. p.n.e.; Dziś odpowiednie narzędzia pozwalają na bardzo dokładne śledzenie zachowań komórek na poziomie molekularnym. Mimo zaawansowanych metod badawczych, immunologia nadal stanowi tajemnicę. Jaki jest mechanizm pamięci? Czemu komórki, które mają chronić organizm, nagle go atakują? I wreszcie, jak nasze komórki, spośród miliardów innych potrafią odróżnić pożyteczne od szkodliwych, choć po niemal 30 latach prac, nie potrafimy tego nauczyć komputerów?

Od wieków ludzie czerpali inspiracje z natury naśladując jej formy, lub korzystając z wzorców procesów, jakie zachodzą codziennie wokół nas. Nawet, jeśli uznamy, naturalną ewolucję za samoregulujący się system, z pewną liczbą zmiennych – nie będziemy mogli go naśladować.

Różnorodność biologiczna, efektywność, samokontrola, monstrualna liczba zmiennych – niemożliwa nawet do wyliczenia, powodują, że choć jesteśmy częścią naturalnego środowiska, nie potrafimy do końca zrozumieć istoty jego działania. Potrafimy rozwiązać wiele problemów w oparciu o wzorce, jakie określa naturalny bieg życia; i choć nie do końca są jasne wszystkie jego tajemnice w naszym posiadaniu jest „algorytm" przybliżający ludzkość do rozwiązania zagadki.

Stawiając samych siebie za obiekt badań, analizujemy świat w skali mikro, zagubiony w rozległych przestrzeniach historii, samotny obiekt na fali rozległego oceanu natury.

2 System immunologiczny w organizmie człowieka.

2.1 Typy odporności.

W organizmie istnieją dwa rodzaje odporności:

- **nieswoista**, – za którą odpowiadają czynniki komórkowe, są to naturalne bariery takie jak np. skóra, stąd wynika podział na: bierną – nazywaną też opornością, do której należą np. wydzieliny komórek nabłonkowych, tworzące nieprzyjazne środowisko dla bakterii; i odporność czynną, jaką stanowią reakcje oczyszczające organizm, takie jak kaszel. W odporności nieswoistej biorą udział makrofagi (komórki żerne), monocyty i granulocyty. Bariera nieswoista jest czynnikiem wyzwalającym obronę swoistą [1].
- **swoista** – właściwy system odpornościowy, na który składają się:
    - **odporność wrodzona** - szybko reagująca forma odporności nie posiadająca pamięci, istnieje od narodzin i posługuje się najprostszym sposobem rozpoznawania elementów szkodliwych: komórki odpornościowe porównują antygeny do siebie, jeśli istnieje różnica w składzie lub wyglądzie komórki to zostaje ona zniszczona;
    - **nabyta** – odporność która dojrzewa wraz z człowiekiem, posiada własną pamięć – uczy się nowych typów zagrożeń. Dzięki niej powstają specjalne komórki do likwidowania wrogich organizmów wyspecjalizowane tylko w jednym ich rodzaju [1].

Do komórek zapewniających odporność nabytą należą limfocyty typu B i T, oraz immunoglobuliny. Limfocyty B to komórki produkowane przez

szpik kostny; w ciągu swego istnienia wędrują do zwojów nerwowych i śledziony gdzie zmieniają się w plazmocyty, które następnie produkują przeciwciała zwane także immunoglobulinami. Limfocyty B po wykryciu ciała obcego, mogą zmienić się w dwie formy: limfocyty pamięci lub komórki plazmatyczne, które produkują przeciwciała dla konkretnego antygenu [1]. Limfocyty typu T, w przypadku zakażenia, wspierają produkcję przeciwciał, i jednocześnie niszczą antygeny i zakażone komórki przez bezpośredni kontakt. W podobny sposób działają także limfocyty NK – (z ang. *Natural Killers*) – cytotoksyczne, mające właściwość spontanicznego zabijania komórek docelowych (własnych – zaatakowanych przez wirusa lub rozpoznanych, jako wadliwe, oraz antygenów) [2].

2.2 Przebieg odpowiedzi immunologicznej.

W momencie wniknięcia do organizmu antygenu następuje odpowiedź immunologiczna, (reakcja organizmu na kontakt z antygenem), która może przybrać dwa typy:

- **odpowiedź typu humoralnego** – biorą w niej udział wolne przeciwciała znajdujące się w płynach tkankowych i wydzielinach surowiczo – śluzowych. Limfocyty B i T zaczynają gwałtownie wytwarzać przeciwciała, dzieląc się, aby zwielokrotnić linię obronną [2].
- **odpowiedź typu komórkowego** - określana jako „*późna*”, ponieważ następuje dopiero po upływie 24 godzin (trwa 4-5 dni, zależnie od drogi infekcji), i jest z warunkowana przez limfocyty typu T, które atakują antygeny [2].

Pojawienie się w organizmie tego samego antygenu po raz kolejny wytwarza **odpowiedź wtórną**; przeciwciała szybciej osiągają odpowiednią ilość i cechuje je większa „swoistość” (lepsze dopasowanie do antygenu). Jest to rezultat pamięci immunologicznej, która nie do końca jest mechanizmem rozpoznanym. Jedną z idei tłumaczących zachowanie układu odpornościowego jest teoria długowieczności limfocytów B; w momencie kontaktu a antygenem limfocyty B dzielą się na dwie grupy: jedną - produkującą przeciwciała i drugą, zachowującą pamięć tego wydarzenia. Obie grupy komórek powstają w trakcie pierwotnej odpowiedzi immunologicznej. Podział ten dotyczy także limfocytów typu T, choć w tym przypadku są one bardziej skuteczne, ponieważ potrafią przenikać przez tkankę łączną i narządy nie limfatyczne [2]. Przykładem sztucznie wyzwalanej reakcji wtórnej jest podanie antygenu, lub gotowych przeciwciał w postaci szczepienia [2].

Antygen po przeniknięciu barier fizycznych dociera do działających między odpornością wrodzona a nabytą, komórek dendrycznych (lub makrofagów śledziony, limfocytów B i limfocytów T), występujących na

skórze oraz w limfie i krwi. Komórki te, prezentują antygenu limfocytom, poprzez rozkładanie białek na peptydy za pomocą cząsteczek powierzchniowych [1]. Każda sygnatura antygenu jest unikatowa, podobnie jak jej powierzchnia, której badanie za pomocą „determinantów antygenowych", receptorów umieszczonych na powierzchni antyciała, pozwala na odkrycie obcego organizmu. Aby rozróżnić antygen, komórka musi stwierdzić, że jest on „obcy", czyli „**nie swój**" [1].

W momencie stwierdzenia „obcości" limfocyt typu B komunikuje się z limfocytami typu T za pomocą mediatorów (informacji chemicznej, która następnie rozpuszcza się w krwi), lub kontaktu bezpośredniego, jednocześnie rozpoczynając produkcję komórek antyciał. Początkowo produkowane przeciwciała mają niskie wartości powinowactwa (siły wiązania z antygenem), oraz niskie wartościowości (czyli małą liczbę determinantów antygenowych); w miarę postępu czasu zarówno jedna jaki i druga cecha rosną, doprowadzając do utworzenia cząsteczek nawet do 100 razy bardziej efektywnych od początkowych, np. cząsteczka z jednym wiązaniem może związać antygen lub nie, ale efektywność cząsteczki z dwoma wiązaniami zwiększa się o 10 – 100 razy [1]. Możliwe jest to poprzez proces rearanżacji genów, następujący w sposób losowy; owocuje to nadprodukcją antyciał o różnych cechach zmierzających do uodpornienia organizmu na inne antygeny, proces ten nazywany jest hipermutacją somatyczną. W trakcie infekcji produkowanych jest do miliona komórek o różnych determinantach antygenowych, ale tylko te, które związą ze sobą antygen są przekształcane na limfocyty B i T, a komórki niedopasowane umierają. Proces ten nazywa się selekcją klonalną [1]. dodatkowo uruchamiany jest proces tworzenia komórek pamięci, zawierających informacje o antygenie i typie limfocytu go zwalczającego. Limfocyty po związaniu z bakterią tworzy kompleks, który jest usuwany przez komórki żerne (makrofagi) [1].

3 Sztuczne systemy immunologiczne – Artifitial Immune Systems.

3.1 Historia badań nad systemem immunologicznym.

Teoria wykorzystania systemu immunologicznego w sieci komputerowej, opisana została w 1974 roku przez Niels`a Jerne`a i Geoffrey`a W. Hoffmann`a; jej naczelnym założeniem było uznanie, że komórki typu B systemu immunologicznego posiadają podobną sieć powiązań zmierzającą do stabilizacji organizmu, jaką charakteryzuje się struktura połączonych hostów; w podobny sposób tłumaczono zjawiska

zachodzące w sieci, jako rezultat działania selekcji klonalnej, oraz niskiej i wysokiej tolerancji systemu w odniesieniu do antygenów.

W 1984 roku Jerne otrzymał nagrodę Nobla za osiągnięcia w dziedzinie medycyny i fizjologii, jego teoria nazwana później „*I-J paradox*", dotycząca selekcji klonalnej oraz określenia roli limfocytów T i B w systemie odpornościowym człowieka została wykorzystana w tworzeniu sieci symetrycznej. Pojawiły się przypuszczenia, że ideę tę, można zastosować jako algorytm optymalizacyjny w poszukiwaniu najkrótszej drogi (graf niezorientowany) w sieci symetrycznej [3]. Geoffrey Hoffmann opracował szczegółową strukturę sieci immunologicznej opartej na symetrycznych symulatorach i inhibitorach w interakcjach między komórkami (po raz pierwszy omówił istnienie komórek zabijających – *killer cells*). Jego teoria doczekała się testów w przestrzeni matematycznej, system Hoffmana z niezwykła stabilnością reagował na nowe zagrożenia, odpowiednio przełączając się ze stanu stagnacji w stan zagrożenia patogenem; co więcej duża liczba patogenów powodowała wzrost antygenów a jaj zmniejszenie indukcję komórek B. System ten nazwano AIN – *Artificial Immune Networks*.

Kolejne wersje system AIN (AINE – Timmis i aiNet - de Castro), oraz algorytmu minimalnego drzewa rozpinającego wykorzystywano do optymalizacji i samoorganizacji obiektów w sieci przy minimalnej liczbie parametrów kontrolnych; system odznaczał się zdolnością uczenia się i odpornością na zakłócenia.

Początkiem stworzenia teoretycznych rozważań dotyczących implementacji systemu immunologicznego w świat cyfrowy, był rok 1986, gdy dwaj naukowcy: J. Doyne Farmer i Alan Perelson w artykule pt.: "*The Immune System, Adaptation, and Machine Learning*", rozważali możliwość zastosowania systemu posiadającego cechy i zachowania podobne do ludzkiego układu immunologicznego, do ochrony sieci; podobne zainteresowanie wyrazili Francisco Varela i Hugues Bersini („*Hints for adaptive problem solving gleaned from immune networks. Parallel Problem Solving from Nature, First Workshop*", 1990 r.). W 1990 roku Hugues Bersini, zbudował podstawy modelowe i algorytmiczne do implementacji systemów immunologicznych.

W 2000 roku Jon Timmis, Mark Neal, i John Hunt zarysowali model systemu samouczącego się; rok wcześniej Dipankar Dasgupta, Yuehua Cao i Congjun Yang wykorzystali system sztuczny system immunologiczny (AIN) do rozpoznawania i klasyfikacji zadań nawet przy niewielkich zasobach komputera [4]. W tym samym roku, Mark Neal zaprojektował nowy system oparty o AIN, nazwany SSAIS – *Self Stabilizing Artificial Immune System*, którego najważniejszą zaletą była decentralizacja systemu, co spowodowało próby wprowadzenia do rozwiązań algorytmu genetycznego i algorytmów roju. Takim samym zainteresowaniem cieszyły się algorytmy rozmyte zaproponowane przez

Olfę Nasraoui, wnoszące do rozważań dodatkowy element – oddziaływania w przestrzeni czasu (wieku leukocytów T i B oraz starzenia się wzorców i kontrolowania populacji komórek) [5]. Selekcja klonalna została wykorzystana w pracy Sławomira T. Wierzchonia, dotyczącej systemu samoorganizującego się przy użyciu małej ilości parametrów [3].

Jednocześnie pojawiła się druga koncepcja, równoważna selekcji klonalnej, Stephanie Forrest (1994 rok), zainspirowana mechanizmem używanym do trenowania komórek typu T – NSA – *Negative Selection Algorithm*. Algorytm negatywnej selekcji opierał się na rozpoznaniu komórek własnych organizmu od patogenu za pomocą mechanizmu „swoje – nie swoje". Forrest stworzyła aplikację której podstawą było utworzenie wzorca „swoich" komórek i znajdowanie anomalii za pomocą porównania z wzorcem detektorów. Jednocześnie określiła poziom wrażliwości jednostek wykrywających, który wzrastał wraz z liczbą przeprowadzonych dopasowań, zarówno pozytywnych jak i negatywnych (model statystyczny).

3.1.1 Model sztucznego układu immunologicznego wg Steven Hofmeyer`a i Stephanie Forrest.

Opracowany w 2000 roku system miał zastosowanie w wykrywaniu włamań, polegające na rozpoznaniu obiektów „swoje – nie swoje", opierające się na ciągu tekstowym w kodzie ASCII, zawierającego dane o połączeniach domeny [5]. „Zdrowe" połączenie TCP składało się z tripletów zawierających informacje o porcie, numerze IP stacji docelowej i wysyłającej. Detektory sprawdzały czy występuje komplet danych w każdym pakiecie, wówczas pakiet był przesyłany do kolejnych detektorów, które go rozkładały i sprawdzały jego poprawność. Detektory te były generowane losowo, jeśli nie zidentyfikowały poprawnie żadnej ramki ich działanie, po upływie pewnego czasu, było zakańczane. Mechanizmem uwalniającym sygnał alarmowy mógł być także człowiek, operujący komputerem, który potwierdzał zgłaszane nieprawidłowości jako forma ataku. To współdziałanie człowieka z systemem było elementem uczenia się systemu.

Dopasowanie sensora do antygenu polegało na tym, by nie przekroczyć określonej liczby błędów, czyli różnic między wzorcem – agentem a ciągiem; z drugiej strony ciągi nie powinny być sztywno przypisane, ponieważ ograniczyło by to możliwość uczenia się systemu. Założono, że dopasowanie oznacza posiadanie tych samych cech zarówno w detektorze jak i w pakiecie sieciowym (podobnie dzieje się w przypadku determinantów antygenów). Następnie pakiety były przeglądane według zasady zwanej **odległością Hamminga**: okno

czasowe przesuwało się o określoną ilość znaków szukając dopasowania. Początkowo sensory były tworzone losowo, jeśli nie zostały dopasowane w ciągu dwóch dni, umierały; następnie dopasowane obiekty poddawane były selekcji klonalnej, zmieniając się w komórkę pamięci. Forrest i Hoffmeyer, uznali że połączenia komputerowe muszą być deklarowane jako niechciane, co spowodowało, że po pierwszym rozpoznanym elemencie blokowano dalsze porównania. Podstawowym problemem było powstawanie tzw. „dziur", pojawiających się z dwóch powodów: po pierwsze sama odległość z jaką przesuwało się „okno porównań" było dziurą, a po drugie z powodu tylko częściowo dopasowanych wartości, co mogło prowadzić do usuwania poprawnych sensorów. Duże wymagania modelu co do ilości wątków i nadprodukcji sensorów, duża ilość alarmów oraz wolne uczenie się systemu, spowodowały że Głównie z tych powodów system nie doczekał się efektywnej implementacji.

3.1.2 Model sztucznego układu immunologicznego wg Jungwon`a Kim`a i Peter`a Bentley`a.

Osią układu była biblioteka genów, zbudowana z tablic na które składały się poszczególne wartości zapożyczone z wnętrza pakietów sieciowych; geny gromadziły:

- typy protokołów;
- zakres portów;
- liczba pakietów danego typu i czas w jakim docierały do komputera;
- czas trwania połączenia;
- dane statystyczne dotyczące natężenia ruchu sieciowego zależnie od pory dnia i stosunku pakietów z flagami SYN i FIN (by ustalić stosunek zapytań do połączeń).

Zestawy informacji służyły do tworzenia obiektów – sensorów, zawierających losowe dane z poszczególnych tablic, oprócz tego do bazy program dodawał nowe cechy mnożąc kombinacje. Na podstawie profilu poprawnego ruchu sieciowego eliminowano sensory za pomocą algorytmu negatywnej selekcji, a następnie trafiały do odpowiednich węzłów sieciowych, gdzie kolejny raz były eliminowane: jeśli uzyskały dopasowanie zostawały sensorami pamięciowymi, były klonowane a ich klony przesyłane na inne węzły, jeśli nie zostały dopasowane zanikały. Działanie to wzorowano na selekcji klonalnej i sposób tworzenia antygenów w naturalnym systemie immunologicznym. Aby wykryć atak ustalono pewien poziom możliwych „niedopasowań", w momencie przekroczenia poziomu w określonym czasie następował alarm. Dzięki

losowości system miał elastycznie dostosowywać się do możliwych ataków, lecz jego implementacja nie doszła do skutku [6].

Pojawienie się nowych problemów takich jak: gwałtowny wzrost przepustowości, nowe formy ataków, wykorzystywanie switch`y i pełny duplex łącza, zwiększa trudność organizacji pakietów i wykrywania ataków. Obecnie zainteresowanie Sztucznymi Systemami Immunologicznymi wciąż rośnie; w 2002 roku zorganizowano pierwszą konferencję pod nazwą ICARIS (*International Conference on Artificial Immune Systems*); odbywa się ona do dziś, publikując najciekawsze wystąpienia uczestników [7]. Pojawienie się w 1994 roku, „Teorii niebezpieczeństwa" Polly Matzinger stało się inspiracją do sprawdzenia jej rozwiązania w praktyce.

3.1.3 Model sztucznego układu immunologicznego w oparciu o „Teorię niebezpieczeństwa" Polly Matzinger.

W 1994 roku Polly Matzinger, w artykule „*Tolerance, Danger and the Extended Family*", zaproponowała "Teorię niebezpieczeństwa" (ang. *Danger Theory*), wyjaśniającą metodę aktywacji systemu immunologicznego [8]. Każda komórka organizmu, w czasie „negatywnej" śmierci (wynikającej z nieplanowanego uszkodzenia lub ataku antygenu), wysyła sygnał alarmowy, który przywołuje na miejsce zdarzenia komórki antyciał. Zwykle pierwsze są limfocyty typu B, które dokonują „oględzin", a po wykryciu antygenu wysyłają kolejny sygnał alarmowy wybudzający limfocyty typu T ze stanu spoczynku. Jednocześnie na terenie odkrycia antygenu ustanawiany jest „*teren zagrożenia*", na którym wszystkie komórki są w stanie gotowości [9]. Teoria niebezpieczeństwa nadal wykorzystuje rozpoznanie „swoje – nie swoje", ale nie jest ona najważniejsza, ponieważ to nie antygen powoduje wyzwolenie alarmu ale sygnał wysyłany z sensorów jest reakcją na zachowanie poszczególnych części systemu [10]. Podejmowano liczne próby osadzenia „Teorii niebezpieczeństwa" w systemie IDS, pierwszą z nich było wykorzystanie algorytmu grupującego alarmy na *false positives* i *true positives* autorstwa Herve`a Debar`a i Andreas`a Wespi [11]. Największym zainteresowaniem cieszy się tworzenie obszaru niebezpiecznego, pozwalającego na zwiększenie poszukiwań możliwych związków między wydarzeniami.

Wymienione najważniejsze koncepcje wykorzystywane do implementacji sztucznego systemu immunologicznego, są wciąż rozbudowywane o nowe postacie algorytmów, ulepszone metody przeszukiwania danych: zarówno ze źródeł lokalnych jak i zewnętrznych.

4 Podstawowe algorytmy immunologiczne

Techniki wykorzystywane do konstruowania systemu immunologicznego opierają się na następujących algorytmach:

- **algorytm selekcji klonalnej** – jego istotą jest dążenie do jak najlepszego dopasowanie komórek B do antygenów, ich namnożenia (wraz z mutacjami następującymi w celu lepszego dopasowania) a następnie zmianę najlepiej dopasowanych osobników na komórki pamięci. Algorytm ten znajduje zastosowanie w optymalizacji oraz rozpoznawaniu kształtów, a także jako element uzupełniający w algorytmach genetycznych;
- **algorytm negatywnej selekcji** – zainspirowany przez sposób „uczenia" swoistości komórek T i B; leukocyty reagujące z komórkami własnymi organizmu są usuwane, a pozostają te, które nie uszkadzają komórek organizmu. Algorytm ten znajduje zastosowanie w rozpoznawaniu wzorców i wykrywania anomalii w systemach komputerowych;
- **teoria niebezpieczeństwa** – oparta o bogaty system komunikacji między komórkami wyszukującymi antygeny, rozpoznającymi je i zabijającymi. Zarówno aktywacja komórek jak i ich działanie, śmierć czy klonowanie jest rozstrzygana poprzez zaistnienie sygnału kontrolującego w określonym czasie a często także na danym terytorium; dzięki temu komórka może tworzyć system pamięci lub zostać wydalona z organizmu;
- **algorytmy komórek dendrycznych** – oparty o model komórek prezentujących antygen limfocytom, stanowią połączenie między systemem wrodzonym a nabytym, jednocześnie są koordynatorami komórek typu T. Idea algorytmów tego typu opiera się na uniwersalności komórek – potrafią one „wybierać" wiele antygenów i generalizować ich cechy szczególne, informacje o nich przekazywane są za pomocą złożonego sygnału chemicznego do komórek typu B i T. Algorytm ten wykorzystywany jest do przetwarzania informacji ale także do wykrywania anomalii w systemach bezpieczeństwa [12];
- **algorytm sieci immunologicznej** – tzn. sieci idiotypowych, nazwa pochodzi od determinantu antygenu, a konkretnie charakterystycznej jego części, po której jest rozpoznawany i wiązany przez przeciwciało. Antygen stymuluje powstanie grupy przeciwciał – protein, które tworzą sieć idiotopową, istniejącą także wtedy gdy antygen zostanie usunięty z organizmu. Sieć immunologiczna jest stosowana jako model pamięci immunologicznej i pozwala na szybką produkcję pamiętających antygen komórek [13];

- **algorytmy hybrydowe** – łączące w jeden kilka lub więcej algorytmów dla skuteczniejszego działania.

Choć głównym założeniem sztucznego systemu immunologicznego jest uczenie się, to każdy program wyposażony jest w reguły rozróżnienia „swoje/ nie- swoje"; w analogii do natury stanowią one podstawę odporności jaką dziecko otrzymuje od rodziców, a następnie pobiera razem z mlekiem matki (przykładem może być opracowany w 2006 roku, przez Qiao i Jianping`a system AINIDS, oparty o wyszukiwanie na podstawie reguł). Problemem jest dostosowanie sensorów do działającego systemu, w którym określenie „swoje" bez przerwy podlega ewolucji, co przy słabej zdolności uczenia się może doprowadzić do autodestrukcji. Wielu badaczy wybiera jedynie jeden typ komórek do naśladowania np. leukocyty typu B (np. w systemie AIS Neal`a) [14]. W przypadku NIDS przy tworzeniu reguł są brane pod uwagę następujące dane: IP źródła i celu oraz ich numery portów, ilość połączeń, typ protokołu, stan i czas trwania połączenia oraz ilość wysłanych bitów. Algorytm selekcji klonalnej i negatywnej selekcji na obecnym etapie rozwoju technologicznego stanowi problem. Ponieważ obecne technologie uniemożliwiają płynną pracę z kilkuset procesów w sposób dość skomplikowany (porównujące ciągi znaków lub liczb np. w systemie FKM opracowanym w 2005 roku przez Xian`a, najbardziej obciążające pamięć systemu), zwykle programiści dążą do ograniczenia ilości sensorów. Większość pomysłów związanych z multiplikowaniem wielomilionowych populacji przeciwciał istnieje obecnie tylko w formie papierowej. Dla ludzkiego organizmu produkcja białych ciałek określana jest w milionach dla jednego tylko antygenu, człowiek osiąga zysk nawet gdy część z nich osiągnie dopasowanie z zdrowymi komórkami i razem z nimi zostanie usunięta z organizmu. W przypadku systemu komputerowego nie można pozwolić sobie na wyprodukowanie nawet tysiąca obiektów, które będą do czasu zakończenia treningu działać w programie.

Reasumując obecnie najważniejsze cele do osiągnięcia dla programisty to:

1. **Rozdzielność**: w naturalnym systemie immunologicznym polega na różnych sposobach rozpoznawania zagrożenia i produkuje antyciała dla każdego antygenu, brak centralnego ośrodka pozwala na walkę z infekcjami w różnych miejscach na raz. Podobnie w systemie komputerowym, jednoczesne monitorowanie najważniejszych obszarów jest w przypadku zagrożeń sieciowych bardzo ważne.
2. **Samoorganizacja**: naturalny system immunologiczny ma różne metody rozpoznawania ataku, dzięki bazie genetycznej otrzymanej od matki, negatywnej selekcji usuwającej wadliwie działające antyciała i selekcji klonalnej faworyzującej klonowanie

antyciał, które potrafią wykryć i związać się z antygenem. W przypadku komputera zastąpić ją może globalna baza, co ułatwi analizę i adaptację danych. Ponadto poszczególny agent systemu musi zachowywać się jak zorganizowana osoba, posiadać własną inteligencję – uczyć się, współpracować z innymi agentami – być uzależnionym od zebranych informacji i budować wspólną wykorzystywaną przez wszystkich bazę.

3. **Wydajność**: organizm jest zawsze przygotowany na zagrożenie dzięki zapasowej liczbie potomnych antyciał, są one wydajniejsze ponieważ mają pamięć ataków poprzednich i są wspierane przez bazę danych zapisaną w genach. Komputer powinien pracować dokładnie tak jak wzór immunologiczny – rozwiązywać małe zadania ale osiągać maksymalne korzyści [6].

System IDPS dla hosta, używany jest wówczas, gdy nie można monitorować aktywności komputera za pomocą programów sieciowych (np. sieciowe analizatory nie potrafią deszyfrować ruch tunelowanego). Systemy działające na hoście wspomagają następujące metody monitorowania:

- monitorowanie działań użytkownika – pozwala na powiązanie zdarzeń między logowaniem do zasobów a alarmami za pomocą: określenia czasu zdarzenia lub typu alarmu, określenie ważności, zależności, priorytetu; oprócz tego wypisane są dodatkowe informacje w postaci IP i portu źródłowego komputera, oraz decyzji podjętych przez użytkownika w momencie alarmu;
- typy wykrytych zdarzeń – niektóre IDPS monitorują integralność plików systemowych, zwykle wykorzystywane jest badanie zachowania kodu poprzez, analizę i zablokowanie możliwości wykonywania kodu wymagając uruchomienia autoryzowanych usług. Pozwala to także na wykrywanie ataku *buffer overfow*, poprzez wyszukiwanie konkretnych charakterystyk i sekwencji instrukcji dążących do przejęcia pamięci wykorzystywanej przez inne procesy. Śledzenie powiązań między aplikacjami pozwala przewidzieć, jakie procesy pojawią się po uruchomieniu konkretnej aplikacji, biblioteki lub usługi. Systemowej bazie może towarzyszyć analiza statystyk udostępnianych przez system operacyjny, które można wykorzystać zwiększając trafność wykrywania włamań;
- analiza ruchu sieciowego – przegląd pakietów, zwłaszcza warstwy sieciowej, aplikacji i transportowej, wymaga działania programów, które mają dostęp do Internetu np. klienta email, klienta *peer- to – peer*; ich monitorowanie

i obserwacja zachowań pozwala ustalić zakres bezpieczeństwa. IDPS`y działające na hoście, połączone są z firewallem chroniącym przed nieautoryzowanym dostępem; firewall może stworzyć listę hostów komunikujących się z Internetem (wspominana już tzw. czarna i biała lista) wraz z listą numerów docelowych;

- monitorowanie systemu plików – jedna technika monitowania nie jest pewna, ponieważ większość robaków korzysta kilku metod podszywania się pod nazwy znanych plików systemowych, zmian ustawień dostępu itp.. Wykorzystywane sposoby filtrowania systemu plików to:
    - o sprawdzanie integralności plików – cykliczne zbieranie informacji o plikach, lub badanie sumy kontrolnej plików kompresowanych, porównywanie danych o wielkości plików. Niestety pozwala to jedynie stwierdzić, że plik już został zmieniony;
    - o sprawdzanie atrybutów plików - periodyczne sprawdzanie najważniejszych plików (podobnie jak wyżej wymienione) potrafi wykryć już zastosowane zmiany ale nie zapobiegać im;
    - o badanie dostępu do pliku – próby dostępu do danego pliku wymagające uwierzytelnienia lub specjalnie chronionych przed zmianami, podobnie jak zapis, odczyt, wykonanie, są zmieniane przez robaki lub konie trojańskie;
    - o analiza logów – monitorowanie logów systemowych i dzienników aby określić sposób działania szkodliwych programów, mogą one zawierać informację o wykonywanych operacjach np. wyłączenie systemu, próby zalogowania czy uzyskania dostępu do procesów bądź zmiany w konfiguracji .
- **„hartowanie" systemu operacyjnego** – za pomocą wprowadzania rekonfiguracji np. w firewallu czy rejestrze, doprowadza samoczynnie do obrony hosta przed atakiem;

Większość systemów IDPS, powoduje dużą ilość alarmów zarówno *false negatives* jak i *false positives*, jednak trafność wykrywania jest po stronie aplikacji pracujących na hostach ze względu na możliwość wykorzystania wielu technik wyszukiwania. Pozwala to na odnalezienie wątku, akcji i związku między nimi którego nie odkrył by program działający jedynie w sieci. Działanie systemu zabezpieczeń, na pojedynczej stacji posiada liczne ograniczenia; należą do nich:

- **opóźnienia alarmów** – generowanie w czasie rzeczywistym alarmów nie gwarantuje ich wystąpienia przed zdarzeniem, niektóre z nich mogą nastąpić nawet kilka dni po zdarzeniu np. zmiana zawartości pliku;
- **opóźnienie raportów** - niekiedy alarmowanie może być ustawione na cykliczne odświeżanie a nie na czas rzeczywisty co sprawia że co 16-60 minut jest opóźnione powiadomienie, spowodowane jest to zebraniem najważniejszych elementów raportowania do jednego pliku;
- **zużycie zasobów komputera** – system działa jednocześnie z wieloma usługami systemowymi co może powodować ograniczenie wydajności systemu;
- **konflikty z kontrolą dostępu** – agent badający np. integralność lub zawartość plików nie może działać w sposób niezgodny z systemem bądź plikami (zapisywanymi właśnie). Problemy te można zredukować za pomocą synchronizacji czasowej wątków lub dostępu do repozytorium;
- **restartowanie komputera** – niekiedy aplikacje wymagają ponownego uruchomienia komputera co może spowodować problemy z systemem agentów, mogą nie zezwolić na takie zdarzenie lub po restarcie utracić części bazy.

Wybór systemu operacyjnego na jakim działa system IDPS jest podstawową decyzją, gdyż zarówno sterowniki jak i architektura programu musi wspierać działania systemu. System operacyjny nie może wstrzymywać działania agentów, czy powodować dodatkowych konfliktów. Podobnie ma się sprawa z IDPS: system musi oszczędnie gospodarować wykorzystaniem zasobów systemowych tak, aby można było nadal prowadzić podstawowe czynności na stacji. Znajomość sieci komputerowej jaką zarządzamy, powinna znaleźć odbicie w zaletach wybranego systemu służącego zabezpieczeniom, innymi słowy: nie jest potrzebny użytkownikowi serwer szyfrujący połączenia (w większości wypadków), tylko narzędzie optymalizujące ruch sieciowy, lub pozwalające na bezpieczne połączenie z siecią. Niezbędny jest także zapis danych uzyskanych w procesie uczenia się oraz w trakcie monitorowania, tworzący bazę z której system będzie korzystał. Organizacja danych powinna ułatwiać szybki przegląd, być możliwa do przeniesienia, zabezpieczona przed awarią tak, by istniała możliwość odbudowy i – na początek – niewielka (aby program mógł dostosować się do danej sieci), ale jednocześnie posiadająca najważniejsze wyjątki działania. Ciekawym zagadnieniem może być komunikacja między poszczególnymi częściami systemu, każdy „agent” musi mieć dostęp do

danych, które będą dla nich uniwersalne i jednocześnie możliwe do edycji.

5 System FF_IDS - budowa.

Program FF_HIDS jest próbą zaimplementowania w języku programowania, elementów sztucznego systemu immunologicznego, w celach ochrony tego systemu przed zagrożeniami. Jest to system działający na hoście, analizujący sieć metodą *Network Behavior Analysis*, działający jako pasywna sonda. Techniki wykrywania opierają się na analizie heurystycznej, bazie sygnatur, wykrywaniu anomalii i analizowaniu norm statystycznych – a więc jest to system hybrydowy. Informuje użytkownika w sposób pasywny – o wydarzeniach, które miały miejsce na stacji roboczej, ale jednocześnie posiada cechy systemów uczących się. Domyślnie miejscem działania programu jest komputer posiadający połączenie z siecią LAN lub WiFi, z systemem operacyjnym Windows XP z SP2 (procesor Pentium lub AMD 2GHZ, 2 GB RAM).

Aplikacja została napisana w języku obiektowym C# .NET, zaprojektowanym dla Microsoftu, który przedstawił rozszerzenia dla tego języka w środowisku Visual Studio 2005. Obecnie C Sharp, jest jednym z popularniejszych języków programowania obiektowego korzystającym z technologii ASP.NET czy LINQ. Język umożliwia programowanie z wykorzystaniem baz danych ADO.NET, języka znaczników - XML, a ostatnio także tworzenia treści dynamicznych (Silverlight). Środowisko .NET, jest wydaną w 2002 roku platformą programistyczną wraz z bibliotekami klas oraz środowiskiem CLR umożliwiającym wykorzystanie wielu języków (nawet w jednym programie), np. C#, Visual Basic, J#, C++, F#. Program powstał w oparciu o środowisko udostępnione w 2010 roku. Dołączona do programu biblioteka ShapPcap 3.7.0 w wersji dostosowanej do języka C# i platformy .NET, zarówno w wersji 32, jak i 64- bitowej, została wykorzystana jako interfejs do przechwytywania pakietów sieciowych i zapisywania ich w postaci binarnej. Postać plików umożliwia wyróżnienie źródła pakietów, inkapsulacji nagłówka, sprawdzenia sumy kontrolnej oraz zaawansowanego wyświetlania [15]. SharpPcap, powstał na podstawie programu WinPcap (Tamir Gal) w 2004 roku, w cztery lata później projekt przejął Chris Morgan, obecnie jest udostępniona w licencji GNU. Z uwagi na możliwość wykorzystania biblioteki w wielu systemach, stała się ona częścią wielu programów, zarówno dla MacOS, jak i Linux`a.

5.1 Założenia architektury systemu FF_IDS.

FF_IDS wykorzystuje następujące warstwy systemu operacyjnego:

- **warstwa sprzętowa**- ogląd zużycia zasobów tj. procesor, pamięć, powierzchnia dysków twardych, zmienne systemowe;
- **warstwa systemowa** – ogląd systemu plików części składowych systemu (usług), działania urządzeń sieciowych, przegląd pakietów sieciowych, przeglądanie dzienników systemu;
- **programy narzędziowe i użytkowe** – monitorowanie działania programów, badanie wybranych ścieżek przez użytkownika, badanie statystyk sieciowych (procesy i rejestr systemu);

System FF_HIDS do poprawnego działania wymaga uruchomienia na koncie administratora aby uzyskać dostęp do możliwości przeglądania procesów, usług i rejestru systemowego Windows, oraz zainstalowania programu obsługującego bibliotekę WinPcap (np. WireShark lub Ethereal) lub samej biblioteki, do przeglądu zapisanych pakietów i umożliwienia ich przechwytywania.

Program działa wykorzystując modele systemu immunologicznego. Pierwszy z nich opiera się o rozpoznanie „swoich/nie swoich" elementów systemu. W organizmie ludzkim limfocyty wykrywające istnienie antygenów czerpią wiedzę z bazy genetycznej, jaką dziedziczą po rodzicach. Na początku życia człowieka, taka baza jest najważniejszym zbiorem informacji pozwalającym na obronę organizmu. W programie FF, rozróżnienie obiektów dokonywane jest przez formy „limfocytów" zwanych agentami. Agenci wpierają się wiedzą zapisaną w bazach dwóch typów: uzyskane w czasie działania systemu, oraz edytowanych przez użytkownika. Pierwszy typ danych zawiera najbardziej podstawowe informacje o elementach szkodliwych systemu operacyjnego (np. lista usług lub portów). Drugim typem jest baza utworzona w czasie działania systemu, zapisana w czasie poprzedniej aktywności użytkownika. Obie bazy, w trakcie działania, mogą być uzupełniane przez użytkownika, co pozwala na dostosowanie programu do specyfiki danego systemu operacyjnego rodzaju lub aktywności.

Rozwiązaniem jakie wykorzystano w systemie FF_IDS, jest „teoria niebezpieczeństwa". Polly Metzenger założyła, że nieplanowana śmierć komórki wywołuje powstanie terenu niebezpieczeństwa i bardziej restrykcyjne badanie komórek systemu. Podobnie w tym wypadku niespodziewane pojawienie się nowego obiektu, zmiany lub zamknięcia procesu, powoduje utworzenie konkretnego stanu, który zależy od ilości wyjątków. Następują po sobie:

- stan normalny – komunikaty są rzadkie, priorytety należą do stanu mieszania i nie grupują się;
- stan zagrożenia – komunikaty oznaczone priorytetem 2;
- stan niebezpieczeństwa – priorytet 1;

- stan ataku – priorytet 0.

W czasie trwania wyżej wymienionych stanów, agenci zmieniają także czas skanowania systemu na częstszy. Poziom priorytetu zależy od następujących zdarzeń:

- pojawienie się wyjątku o podobnym priorytecie od tego samego lub innego agenta – zmniejsza priorytet o jeden i przesuwa na wyższy poziom;
- ważność odkrytych wydarzeń, które są porównywane z bazą poszczególnych agentów;

Mechanizm ten wykorzystują IDPS-y oparte na wykrywaniu anomalii: działania nienormalne są skutkiem istnienia w systemie aplikacji, użytkowników, lub plików powodujących działanie systemu odbiegające od normy, np. pojawienie się nowych procesów, usług, połączeń. Monitorowanie warstwy aplikacji pozwala na odnalezienie zbieżności w wyjątkach, ich rosnąca liczba natomiast, podkreśla brak przypadkowości. Podobne zachowanie w organizmie ludzkim alarmuje możliwość pojawienia się stanu chorobowego, np. gorączka może być spowodowana wieloma przyczynami, ale jeśli jest poparta bólem w określonym miejscu i krwawieniem, jest objawem poważnego zagrożenia (np. zakażenia). Ten uproszczony mechanizm naśladujący ludzki organizm, nie jest idealny a prawdopodobieństwo pojawienia się alarmów niezwiązanych ze szkodliwym program czy atakiem z zewnątrz, wciąż istnieje; jednak możliwość konfrontacji „objawów" z administratorem pozwala na lepsze dopasowanie systemu do potrzeb użytkownika.

Nad wszystkimi elementami działa Zarządca Agentów, śledzący priorytety komunikatów i zarządzający uruchamianiem agentów. Zarządca dokonuje także zmiany priorytetów i ustawia ich kolejkę wyświetlania, zmuszając do oczekiwania komunikaty o niskiej wartości priorytetów, bądź przyspiesza wyświetlanie ważnych. Powrót do „stanu normalnego" jest możliwy o ile komunikaty o zagrożeniu są przejmowane przez użytkownika, lub nagromadzenie wyjątków wygaśnie. Jeśli w trakcie działania systemu agent nie odbiera wyjątków, wówczas przełącza się w stan spoczynku, polegającego na rzadszym skanowaniu systemu; wyjątek ponownie wzbudza agenta powodując częstsze przeszukiwanie systemu.

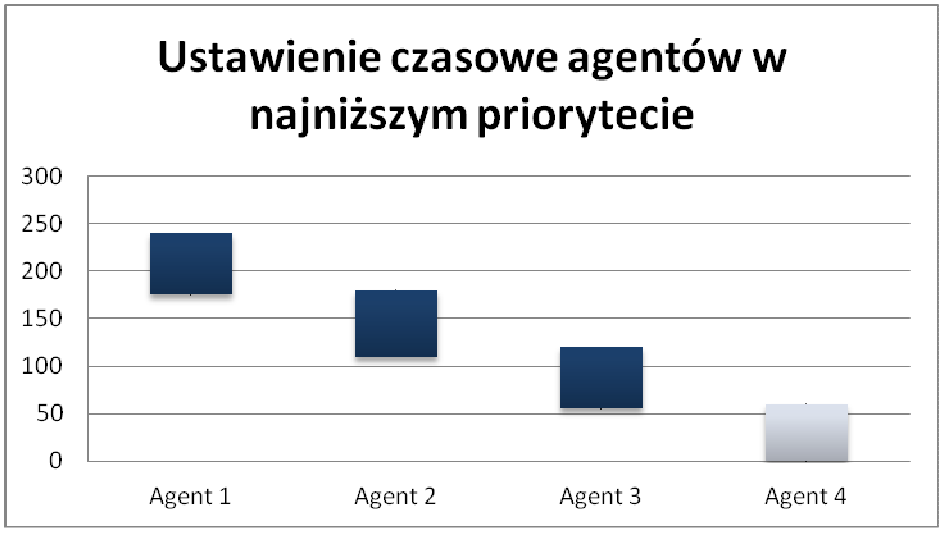
Określenie czasu skanowania systemu i pakietów oparte zostało na pomyśle Kai Hwang`a, Hua Liu i Ting`a Chen`a, opisanego w artykule opisującym system Cooperative AIDS, o bardzo szerokim zakresie badania protokołów sieciowych za pomocą wzorców. Analizując działanie systemów IDS, autorzy powiązali częstotliwość występowania zdarzeń negatywnych w określonym ciągu czasowym, co pozwoliło im podzielić czas pracy systemu na okna czasowe (podobne w rozkładzie do okna Hanninga). Przy częstości skanowania pakietów co 2 sekundy, otrzymywali oni 80% szansę na odkrycie włamania. Podobnie w progra-

mie FF_IDS, skanowanie przez aktywnych agentów określonych przedziałów systemu następuje w kolejce czasowej. Agenci zajmujący się podobnymi częściami systemu startują kolejno, pozwala to na uzyskanie jednolitej czasowej linii np. dla wydarzeń związanych z działającymi procesami i usługami. Czas działania agenta może zostać zmieniony zależnie od nagromadzenia wyjątków w programie: mniejsza liczba wyjątków zwiększa czas wysyłania komunikatów do użytkownika. Szybkość ta, jest sterowana za pomocą „opóźnienia", które jest dodawane do czasu uruchomienia: najkrótszy czas skanowania wynosi co 60 sekund, najdłuższy 120 sekund. W momencie zgrupowania co najmniej trzech komunikatów w jednym czasie, występuje stan „gorączki" co zmienia priorytet wszystkich agentów. Ponieważ liczba komunikatów do osiągnięcia kolejnych stanów jest coraz większa, zapobiega to przypadkowemu nagromadzaniu się wyjątków w pamięci programu i obciążaniu systemu. Wszyscy agenci pracują w sześciu podstawowych priorytetach; trzy z nich należą do tzw. strefy mieszania, a pozostałe trzy ustalają zakres niebezpieczeństwa:

- Mieszanie:
  - Priorytet 5 - pojawienie się co najmniej czterech komunikatów o tym samym priorytecie, zmienia ich ważność na poziom 4; jeśli liczba komunikatów nie przekracza dozwolonej liczby, przedłuża czas spoczynku agenta poprzez dodanie do normalnego czasu 10 sekund;
  - Priorytet 4 – jeżeli liczba wyjątków nie przekroczy czterech, oprócz dodatkowego opóźnienia, zmienia się priorytet komunikatu na niższy; w przypadku wystąpienia trzech wyjątków czas nie ulega zmianie, ale zmienia się priorytet zdarzania;
  - Priorytet 3 – podobnie jak wyżej;
- Zakres niebezpieczeństwa:
  - Priorytet 2 – „Zagrożenie" - do uruchomienia stanu potrzebne są cztery komunikaty o tym samym priorytecie; jednocześnie użytkownik powiadamiany jest o zaistnieniu takiego stanu; jeśli odpowiedź jego nie potwierdza przyczyny powstania stanu zagrożenia to priorytet wzrasta, w przeciwnym wypadku maleje o jeden i dodawane jest opóźnienie 10 sekund;
  - Priorytet 1 – „Niebezpieczeństwo"- do przejścia na wyższy poziom potrzebnych jest pięć komunikatów o ważkości priorytetu równego jeden; informacja o zaistnieniu stanu jest

kierowana do użytkownika, który decyduje o zmianie priorytetu;

- Priorytet 0 – „Atak” – występuje w momencie pojawienia się pięciu wyjątków o tym samym priorytecie, agenci uruchamiają się co 60 sekund, a użytkownik poinformowany jest o wystąpieniu stanu ataku i dodany zostaje wpis do dziennika systemowego.



Rys. 1. Wizualizacja uruchamiania agentów FF_IDS.

W wyższych priorytetach czas uruchamiania agentów może doprowadzić do nakładania się czasów skanowania, zwiększa to prawdopodobieństwo wystąpienia określonej liczby komunikatów i w miarę wzrostu priorytetów wymagana jest ich większa liczba. Jedynie zróżnicowanie priorytetu komunikatów powoduje, że jedynie w przypadku kryzysu pojawia się komunikat o ataku.

Agenci mogą występować w dwóch formach:

- aktywna porównująca obiekty do posiadanych reguł, przekazuje listę informacji do wyświetlenia do agenta biernego a w razie zagrożenia informuje moduł zarządzający, który wysyła informację do użytkownika w postaci komunikatu MessageBox, lub chmurki.
- pasywna – działają skanując system cyklicznie, informują użytkownika o działaniach systemu; jeśli to możliwe, pozwalają na dodawanie wyjątków i zarządzanie interfej-

sem danego agenta; dane wyświetlane są w poszczególnych formatkach.

Każdy agent posiada własny sposób porównywania komunikatów, ale wszystkie przekazywane są do zarządcy agentów w momencie pojawienia się wyjątku. Następnie klasa ZarzadcaAgentow, decyduje o pojawieniu się „terenu zagrożenia", poprzez badanie liczby komunikatów występujących w danym parytecie. Czas uruchamiania agentów tworzy kolejkę, gwarantującą przełączanie komunikaty na wyższe poziomy pod warunkiem ich narastania w określonym czasie. Pomysł ten opiera się o badanie całościowe: atak wywołuje nie jedna, lecz kilka zmian w obszarze systemu, stąd wymaganie namnożenia komunikatów pozwalają na uniknięcie fałszywych alarmów.

Uruchomienie programu powoduje wywołanie okna powitalnego, umożliwiającego dostęp do poszczególnych agentów. Panel główny pozwala na uruchomienie programu w stanie pasywnym – informującym, aktywnym – wyzwalającym start agentów przeszukujących system i analizujący statystyki, oraz - po wybraniu interfejsu sieciowego - monitoring pakietów sieciowych. Agenci aktywni pozwalają na ogląd elementów systemu należących do następujących działów:

- procesy systemowe – analizuje działające w systemie procesy, program odnotowuje ich zanikanie i pojawianie się porównując z normą jaką ustala w kontakcie z administratorem;
- pliki systemowe – agent analizuje trybuty, wielkość czas powtania i modyfikacji najważniejszych plików systemowych; użytkownik może dodać także swoje pliki do śledzenia. Różnica porównania jest formowana w postać komunikatu, a następnie przekazywana, wraz ze wskazanym priorytetem, do Zarządcy agentów, który dokonuje ich kolejkowania; w zależności od liczby komunikatów i priorytetu wątek jest usypiany lub przeniesiony na wyższy poziom. Dodatkowo dodawany jest wpis do dzienników systemu operacyjnego.
- porty sieciowe - agent ten, zbiera informacje o typach protokołów i numerach potów jakie wykorzystuje stacja, w przypadku odnalezienie połączenia zbieżnego z bazą wyjątków informuje użytkownika i dokonuje wpisu do dziennika systemowych zdarzeń. Ponieważ oczekiwanie na zapełnienie listy skanowanych portów komputera trwa kilka minut, a agent sprawdza ograniczoną ilość danych jego komunikaty mają najniższy priorytet. Przeszukiwanie listy portów ograniczono do najczęściej podsłuchiwanych i narażonych interfejsów komputera, wykorzystywanch przez włamywaczy.
- pakiety sieciowe - system informuje o następującym ruchu sieciowym:

- Wzmożony ruch pakietów ICMP – z zasady ruch ten nie powinien się pojawiać w sieci, chyba że do celów diagnostycznych;
- Zbyt duże pakiety ICPM, IP, TCP, UDP przekraczające normę wielkości, których zalew prowadzi do odmowy usługi;
- Pakiety IP o zbyt krótkim czasie życia, zgodnie z normą TTL nie powinien być mniejszy niż 30 hopów;
- Liczne pakiety TCP o braku flag (atak zero), lub ich nadmiarze (atak choinki);
- Zbyt duża kumulacja pakietów o co najmniej dwóch flagach lub nagromadzenie datagtamów o jednej fladze w określonym odstępie czasowym;
- Pojawienie się pakietów o flagach niespotykanych, np. SYN i FIN jednocześnie;
- Zbyt duża ilość pakietów UDP w określonym odstępie czasu;
- Ruch UDP, na portach powyżej 5000 (ponieważ część programów p2p z nich korzysta komunikat ma niski priorytet);

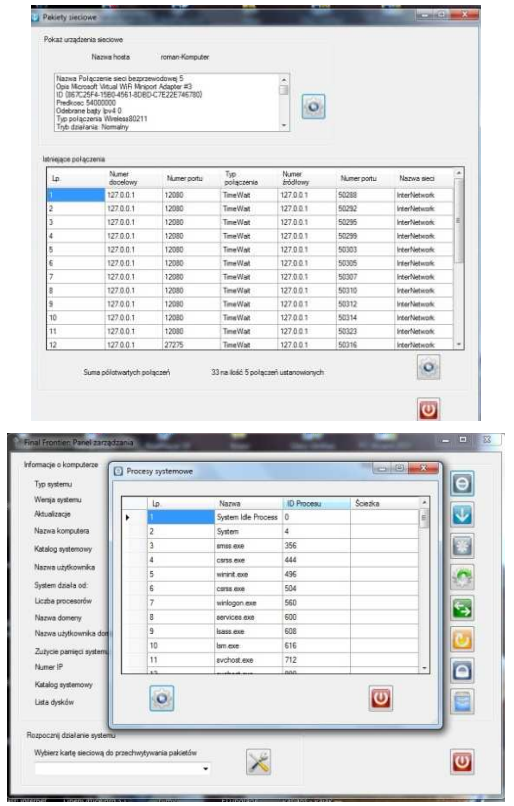
Priorytety komunikatów Agenta pakietów, ustawiają się w kolejkę razem z priorytetami Agenta statystyk sieciowych, co powoduje nagromadzenie wyjątków i komunikat o stanie zagrożenia. Wyjątkiem jest sytuacja, w której stacja otrzymuje ewidentnie niestandardowe pakiety (np. przekraczające normę wielkość), wówczas komunikaty osiągają od razu priorytet „Stanu niebezpieczeństwa" i jeśli jest ich więcej wyświetlają komunikat o ataku na stację.

- statystyki sieciowe – analizuje informacje wytwarzane przez system operacyjny, w zakresie: ilość nieudanych prób połączeń TCP, ilość resetowanych połączeń TCP, odebranych błędów połączeń TCP, dopuszczalnej liczby pakietów UDP uszkodzonych; liczby pakietów TCP i UDP retransmitowanych; liczby pakietów UDP odrzuconych. Przyczyną tych wyjątków może być zarówno błędna konfiguracja sieci, lecz w większości przypadków zwiększona liczba pakietów odrzuconych jest skutkiem skanowania portów, ataku DoS, lub rekonesansu włamywacza. Przykładem metody wykrywania włamań za pomocą badań statystyki pakietów docierających do komputera, jest „*One-class Support Vector Machine*" (SVM), zastosowany np. w DARPA (1999 rok), SNORT`cie, NIDES czy ADAM`ie [16]. Badanie systemu poddanego działaniu aplikacji szkodliwych powodowało wzrost liczby pakietów odrzuconych, uszkodzonych, bądź wymagających retransmisji; np. wykorzystanie programy Neptun, zalewającego stacje pakietami TCP z ustawionymi flagami SYN i RET, powodowało anormalny wzrost tych pakietów w stosunku do całości ruchu sieciowego.

- usługi systemowe – podobnie jak agent procesów analizuje działanie usług w warstwie systemu, zaznaczając ich pojawianie się i wyłączanie; w przypadku braku zbieżności z własną bazą danych, oznacza zdarzenia odpowiednim wpisem do dziennika systemowego.

6 Testowanie programu.

Systemy typu „*Intrusion Detection”*, służą do komunikowania zagrożenia i zbierania informacji o działaniu systemu operacyjnego. Program został stworzony do działania w Windows XP z SP2. Windows Server 2000 i 2003 jest analogicznym systemem dla Windows XP z SP2; występują tam te same gałęzie rejestru i podobne obiekty składowe systemu, stąd można go stosować także w przypadku tych systemów operacyjnych a także starszych produktów Microsoft takich jak Windows 98/2000. Podobnie w przypadku Windows Vista i Windows 7, program działał aktywnie skanując połączenia sieciowe.



Rys. 2. FF_IDS w systemie Windows 7

Podczas pierwszego uruchomienia aktywnego przeszukiwania systemu, pojawią się komunikaty produkowane przez aktywnych agentów, zmierzające do dostosowania bazy pamięci do komputera użytkownika. „Proces uczenia" się trwa od 4 do 6 minut i polega na odpowiadaniu na zapytania generowane przez agentów. Liczba komunikatów zależy od ilości procesów i usług jakie nie są znane przez program a działają na komputerze, użytkownik musi zdecydować, które nadal monitorować, a które dodać do zaufanej bazy. Po tym czasie, następuje dostosowanie do systemu użytkownika i rozpoczyna się pełne funkcjonowanie agentów.

Działanie systemu poddano badaniu skanowania portów i wysyłania pakietów przekraczających normę, opóźnienie rozróżnienia ataku zależało od aktualnego obciążenia sieci i wahało się od 1 do 3 sekund. Program jednak wykrył zarówno anormalną liczbę pakietów ICMP czy też TCP, oraz odnotował to w wyjściu konsoli, ale jednocześnie skanował system w poszukiwaniu zmian. Agent skanowania pakietów rozpoczyna są działalność jako szósty w kolejności, zatem muszą upłynąć jeszcze dwie minuty do jego uruchomienia. Aby dokładnie zbadać działanie programu, a zwłaszcza rozpoznawanie ataków i odpowiednie generowanie komunikatów, skierowano ich wyświetlanie na konsolę, aby porównać szybkość reakcji i jej adekwatność. Jednocześnie podprogramy skanujące system zostały narażone na działanie kilku najnowszych programów z grupy malware (np. *Security Shield Virus*, *Police Trojan*), których działanie było zgłaszane administratorowi.

7 Podsumowanie i wnioski

Podstawowym celem budowy systemu FF_IDS, było wykrycie nieprawidłowości w działaniu komputera, których źródłem mogły być zagrożenia płynące z sieci. Naczelną inspiracją był system immunologiczny człowieka oraz jego mechanizmy ochronne; a także teorie z nim związane. Rozpoznanie „swoich-nie swoich" obiektów systemu pozwala ustalić zaledwie część mechanizmu ochrony; jednak najważniejszym elementem jest „Teren niebezpieczeństwa", który pozwala grupować komunikaty i określać ich rangę. Ten mechanizm pozwala na znaczne ograniczenie fałszywych komunikatów i podnieść wydajność systemu.

Choć program odpowiada tylko na podstawowe typy zagrożeń sieciowych, jego połączenie z rozpoznawaniem anomalii systemowych i badaniem statystyk okazało się obiecujące. Podczas testowania program ujawnił, nie tylko zdolność do reakcji na ataki sieciowe, ale także na zdarzenia z nią niezwiązane (np. istnienie nowych usług i procesów, które nie zostały wcześniej zainstalowane), co prowadzi do

konkluzji, że wykorzystane podstawy teoretyczne pozwalają na określenie nie tylko tych wydarzeń które są znane ale także nowych typów zagrożeń. Zaskoczeniem jest także minimalne wykorzystanie zasobów komputera pod względem wydajnościowym, choć naturalnie większa lista reguł mogła by spowolnić działanie programu. Jest to dowodem na to, że prostota systemu jest jego podstawową zaletą, choć z pewnością jego działanie można poprawić.

Rozbudowa systemu, polegała by na zapisywaniu określonych sytuacji alarmowych wywołanych przez agentów, które były by zapisywane. Wówczas podczas ponownie występującej sekwencji alarmów, program informował by szybciej, co stworzyło by odpowiednik wtórnej odpowiedzi immunologicznej i znacznie zwiększyło wydajność. Praca ta, wymagała by określonych eksperymentów i testowania programu znacznie dłużej; samo badanie odpowiedzi agentów na podstawowej wersji programu, zajęło około 60% czasu przeznaczonego na ukończenie pracy. Największym problemem, jaki napotkano podczas tworzenia programu było jego dostrojenie do systemu operacyjnego, polegające na wybraniu takich wartości aby program nie produkował fałszywych alarmów, będąc jednocześnie na tyle wrażliwym by informować o występujących anomaliach. Budowa złożonych systemów np. SNORT (skupionych jedynie na warstwie sieciowej), jest okupiona miesiącami pracy nie jednego programisty, lecz całego ich sztabu, uzupełnianego wiedzą użytkowników programu piszących dodatkowe reguły i testujących go na różnych konfiguracjach komputerów.

Zwiększenie interakcyjności programu, dotyczące generowania alarmów, mogło by polegać na modyfikacji przez użytkownika, którego konfiguracje zapisywane były zależnie od wartości i określały poziom ochrony komputera. Jednak udostępnienie takich udogodnień, oznaczało by zmianę jego interfejsu w zestaw tabel do wpisywania, co odebrało by sens użytkowania systemu także przez zwykłego użytkownika.

Literatura

[1] Lydyard P. M., Whelan A., Fanger M. W., *Immunologia. Krótkie wykłady*. Wydawnictwo Naukowe PWN, Warszawa, 2001,

[2] *Immunologia*, Wydawnictwo PWN, Warszawa, 1998,

[3] Dagsputa D., Ji Z., Gonzales F., *Artificial Immune System (AIS). Research in the Last Five Years*, [w:] Proceedings of Congress on Evolutionary Computation (CEC), 2003,

[4] Ranang M. T., *An Artificial Immune System Approach to Preserving Security In Computer Networks*, NORGES Teknisk-

Naturvitenskapelinge Universitet, Hovedoppgave, Trondheim 2002,

[5] Nasraoui O., Dagsputa D., Gonzalez F., *An Artificial Immune System Approach to Robust Data Mining*, Genetic and Evolutionary Computation Conference (GECCO Late Breaking Papers), July 2002, New York,

[6] Kim J., Bentley P., *An Artificial Immune Model for Network Intrusion Detection,* 7th European Congress on Intelligent Techniques and Soft Computing (EUFIT'99),

[7] http://www.artificial-immune-systems.org/icaris.shtml

[8] http://en.wikipedia.org/wiki/Polly_Matzinger

[9] Le Boundes J.-Y., Sarafijanović S., *An Artificial Immune System Approach to Misbehavior Detection In Mobile Ad-Hoc Networks* [w:] *Technical Reports IC*, 2003

[10] Zhu Y., Tan Y., *A Danger Theory Inspired Lerning Model Its Application to Spam Detection*, Berlin 2011 [w:] ICSI 2011, Part 1, LNSC 6728,

[11] Debar H., Wespi A., *Aggregation and Correlation of Intrusion Detection Alerts*, Recent Advances in Intrusion Denetcion 4th

[12] Greensmith J., Aickelin U., Cayzer S., *Introducing Dendric Cell as Novel Immune-Inspired Algorithm for Anomaly Detection*, 2005, ICARIS 4th,

[13] Wierzchoń S., Ciesielski K., Kłopotek M., *Algorytmy immunologiczne*, ACADEMIA: Panorama Systemy wyszukiwania informacji, 2008, nr 2

[14] Timmis J., Neal M., Hunt J., An *artifitial immune system for data analysis*. Biosystem 2000, nr 55

[15] http://sourceforge.net/apps/mediawiki/sharppcap/index.php?title=Main_Page

[16] Quang-Anh T., Duan H., Li X., *One-class Support Vector Machine for Anomaly Network Traffic Detection* [w:] Advances in Neural Networks. Second International Symposium on Neural Networks, Chongqing, 2005,

IMMUNE SYSTEMS IN THE PROBLEMS OF SECURITY OF INFORMATION SYSTEMS

Summary – The purpose of the article is to discuss the construction of a system to protect your computer from threats using an algorithm based on the human immune system. The idea of creating a system based on the immunologię, introduced by prof. Mark Rudnicki proved to be an excellent solution. The main idea of the own elements of the resolution, connection with the creation of "land risks", allowed the creation of a high efficiency. The system has only the initial database (like protecting the child's body, the antigens contain genetic information derived from the mother), which, as far as the functioning of the supplements, which emulates a learning process. The program can sense danger, using only the "intelligence", which gave him the programmer and you can use to detect new hack attacks.