

SYSTEM MONITORUJĄCY AKTYWNOŚĆ UŻYTKOWNIKÓW W LAN

DOKUMENTACJA PROJEKTU

TEMAT: DOKUMENTACJA KOŃCOWA
AUTORZY: SEBASTIAN ŁUCZAK, CEZARY ZAWADKA
DATA EDYCJI: 26.01.2010
STATUS: FINALNY

SPIS TREŚCI

Temat: Dokumentacja końcowa	1
Wstęp	2
Opis kluczowych funkcji i metod	7
Podsumowanie	10

WSTĘP

Poniższy projekt jest implementacją monitora aktywności użytkowników w sieci LAN. W tym rozdziale opisana zostanie ogólna koncepcja jego budowy i pracy, zakres funkcjonalności i instrukcje związane z jego prawidłową kompilacją.

Następne rozdziały przedstawiają opis kluczowych klas metod i funkcji.

Ostatni rozdział stanowi podsumowanie implementacji projektu, a także zawiera przemyślenia co do możliwości jego usprawnień.

MONITOR SIECI LAN

GŁÓWNA KONSEPCJA	Wykorzystanie protokołu ARP i analizy jego ramki do celu monitorowania aktywności użytkowników w sieci LAN.
ZAKRES FUNKCJONALNOŚCI	<ul style="list-style-type: none"> • Monitorowanie stanu sieci w której uruchomiona jest aplikacja • Monitorowanie stanu sieci w których działają klienci z poziomu zarządcy • Przechowywanie informacji o aktywności hostów w bazie danych • Wizualizacja stanu sieci monitorowanej przez zarządcę i klienty • Dostarczenie informacji o historii aktywności użytkownika w danych ramach czasowych w obszarze wszystkich dołączonych podsieci • Wyświetlenie archiwalnego stanu danej sieci
BUDOWA	<p>Aplikacja została podzielona na kilka modułów (klas) realizujących oddzielne zadania:</p> <p>Obsługa interfejsu sieciowego wysyłanie i odbieranie pakietów protokołu ARP (klasa CNetworkAdapter)</p> <p>Przechowywanie danych obsługa informacji o hostach z sieci lokalnej i z sieci klientów, łączność z bazą danych (klasa CDataBaseWrapper)</p> <p>Łączność łączność z klientami, pobieranie listy aktywnych hostów z sieci klientów (klasa CConnectionMgr)</p>

	<p>Wizualizacja Graficzne przedstawienie stanu sieci (klasa CGViewer)</p> <p>Interfejs użytkownika linia komend (klasa CMainLoop)</p> <p><u>Wielowątkowość</u> Ponadto w celu zapewnienia sprawnej obsługi interfejsu użytkownika oraz niezależności działań wykonywanych przez poszczególne moduły aplikacja została podzielona na kilka niezależnych wątków.</p> <p><u>Założenie symetrii</u> Połączenia z instancjami aplikacji w innych podsieciach są typu peer-to-peer, każda aplikacja jest równorzędna, może jednocześnie zbierać informacje z kilku podsieci i wysyłać informacje o swojej podsieci.</p>
--	---

PRZYGOTOWANIE DO URUCHOMIENIA

INSTRUKCJE CO DO KOMPILACJI	<p>Do poprawnej kompilacji i uruchomienia niezbędne jest posiadanie pakietów bibliotek wymienionych poniżej i rozwiązanie konfliktów między biblioteką WinPcap i plikami dostarczonymi z systemem operacyjnym (tylko na niektórych maszynach roboczych), a także posiadanie zainstalowanego środowiska IDE Microsoft Visual Studio 2008.</p> <p>W projekcie Visual Studio, w Project->Properties, w zakładce Debugging należy wprowadzić /dlls jako Working Directory. Pozostałe ustawienia są przenośne i zostały dostarczone.</p> <p>*stwierdzono występowanie asercji z biblioteki sqlite3 przy zamykaniu aplikacji, jest to od nas niezależne (boost::timed_mutex)</p>
BIBLIOTEKI (OPIS OGÓLNY)	<ul style="list-style-type: none"> • SQLite3 – integracja z bazą danych SQLite • WinPcap 4.1.1 – obsługa zróżnicowanych interfejsów sieciowych, rozsyłanie i przechwytywanie pakietów • BOOST – zapewnienie wielowątkowości i nowoczesnych mechanizmów programistycznych • SDL – biblioteka międzyplatformowa zapewniająca programowanie grafiki z pominięciem systemu operacyjnego • SDL_NET – rozszerzenie SDL dostarczające obsługę gniazdek sieciowych • SDL_IMAGE – biblioteka zapewniająca łatwą obsługę plików graficznych o różnych popularnych rozszerzeniach • Visual Leak Detector - wykrywanie wycieków pamięci (nieobowiązkowe przy kompilacji)

METODA DZIAŁANIA	
PODSTAWY	<p>Działanie aplikacji oparte jest o analizę ramek protokołu ARP. Ramki te, rozsyłane w sposób cykliczny na wszystkie adresy należące do danej podsieci, generują odpowiedzi konkretnych hostów przesłane na interfejs sieciowy obsługiwany przez aplikację. Ramki odpowiedzi ARP są przechwytywane i na ich podstawie budowana jest lokalna baza danych o hostach należących do danej podsieci.</p> <p>Rekord pojedynczego hosta (klasa ActiveHost) składa się z:</p> <ul style="list-style-type: none"> • Sprzętowego adresu MAC • Sieciowego adresu IP • Adresu IP podsieci do której należy host • Czasu pierwszego odnotowania obecności hosta w sieci • Czasu ostatniego odnotowania obecności hosta w sieci • Abstrakcyjnego czasu życia hosta, który tworzony jest na potrzeby aplikacji <p>Częstotliwość rozsyłania ramek żądań ARP i wartości początkowe czasu życia hosta pobierane są z pliku konfiguracyjnego i możliwe do modyfikacji z poziomu konsoli w trakcie działania programu.</p>
MECHANIKA	<p>Przez czas działania aplikacji, hosty przechowywane są w kontenerze asocjacyjnym, którego kluczem jest adres MAC hosta. Ciągłość odpowiedzi danej stacji, lub jej brak, ma wpływ na czas jej życia TTL. Po każdej odpowiedzi hosta na zapytanie ARP parametr TTL odnawiany jest do wartości maksymalnej. Przez cały czas cyklicznie wywoływane jest zmniejszanie TTL. Gdy wartość ta wynosi '0', zakłada się, że stacja jest wyłączona z sieci i jej rekord trafia do bazy SQLite z odnotowanym czasem odłączenia. Dzięki parametrowi TTL można decydować po jakim czasie (lub po ilu wysłanych pakietach ARP, na które nie przyszła odpowiedź) host zostaje uznany za nieaktywny.</p>
ZASADA WSPÓŁPRACY ZARZĄDCA-KLIENT	<p>Aplikacja dopuszcza pracę sieciową opartą o hierarchię zarządcy ->klienty. Każda instancja aplikacji jest symetryczna i może pełnić zarówno rolę zarządcy jak i klienta.</p> <p>Zarządca przechowuje spis klientów, który może być dowolnie rozszerzany i redukowany w trakcie jej działania. Na jego podstawie zarządca dokonuje cyklicznych odpytań klientów. Otrzymane rekordy przechowuje w oddzielnym kontenerze i analogicznie do rozwiązania lokalnego, dokonuje wpisów do bazy SQLite.</p> <p>Zamknięcie aplikacji powoduje przekazanie całej zawartości kontenerów do bazy danych.</p>

<p>PODSTAWOWE DZIAŁANIA NA BAZIE DANYCH SQLITE</p>	<p>Baza danych przechowuje archiwalne informacje o stanie sieci w której pracuje aplikacja i wszystkich podsieci dostarczonych poprzez klienty. Za pośrednictwem aplikacji możliwe jest pobranie do kontenera archiwalnego ostatniego stanu danej podsieci, oraz odpytanie o historię aktywności konkretnej stacji. W tym ostatnim przypadku możliwe jest zawężenie rozpatrywanych ram czasowych. Historia hosta, odpowiednio sformatowana, wyświetlana jest na konsoli.</p> <p>Wszelkie inne odpytania bazy możliwe są poprzez dostępne narzędzia, np. SQLite Manager (plugin do przeglądarki internetowej Firefox)</p>  <p>Prezentacja różnych zapytań o historię host'a</p>
<p>WIZUALIZACJA STANU SIECI</p>	<p>Monitor sieci LAN pozwala obserwować stan sieci w sposób graficzny. Dopuszcza trzy tryby przepatrywania:</p>

- obserwację hostów z sieci w której pracuje aplikacja
- obserwację hostów z sieci dostarczonych przez klienty
- obserwację zapisanego w bazie danych archiwalnego stanu danej podsieci

Dwa pierwsze tryby ukazują obecny stan każdej stacji (aktywna/nieaktywna) oraz ich adresy sieciowe i sprzętowe.

Ostatni tryb jest wizualizacją offline, na podstawie której można zaobserwować jakie hosty przebywały w danej sieci. Informacja ta jest przydatna przy odpytywaniach bazy.



Prezentacja użycia wizualizatora graficznego sieci

INTERFEJS UŻYTKOWNIKA

Podejmowanie decyzji w aplikacji odbywa się poprzez CLI zbudowane na niezależnym wątku w celu zapewnienia sprawnej obsługi.

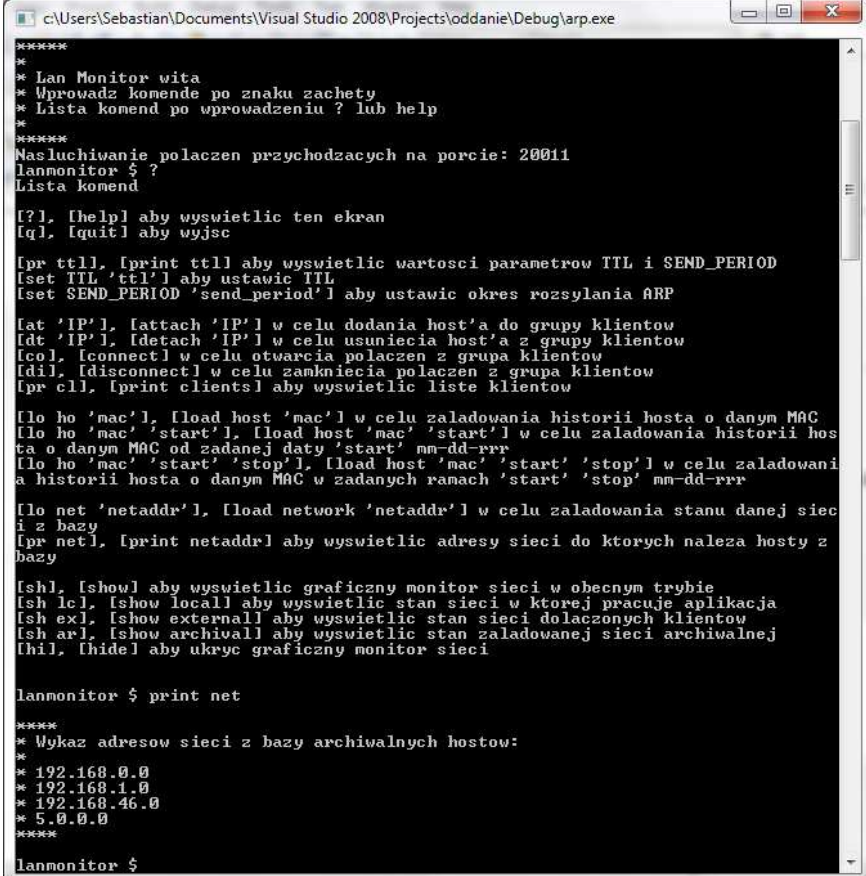
Wszystkie komendy są dokładnie opisane w pomocy zawartej w linii komend i możliwej do wywołania po wpisaniu 'help' lub '?'.
Część poleceń posiada swój własny system odpowiedzi, aby zapewnić płynną obsługę.

Konsola posiada podstawowe zabezpieczenia przed błędnymi

Konsola posiada podstawowe zabezpieczenia przed błędnymi

wywołaniami poleceń. W obecnej wersji nie ma już możliwości podania błędnego identyfikatora interfejsu.

Za jej pośrednictwem dokonuje się podstawowych odpytań bazy danych, edycji listy klientów, inicjalizacji i rozwiązywania połączeń zarządca->klient, edycji wartości początkowej TTL hostów i częstotliwości rozsyłania żądań ARP i wyświetlania oraz ukrywania monitora graficznego w odpowiednim trybie.



```

c:\Users\Sebastian\Documents\Visual Studio 2008\Projects\oddanie\Debug\arp.exe
*****
* Lan Monitor wita
* Wprowadz komende po znaku zachety
* Lista komend po wprowadzeniu ? lub help
*****
Nasluchiwanie polaczen przychodzacych na porcie: 20011
lanmonitor $ ?
Lista komend

[?], [help] aby wyswietlic ten ekran
[q], [quit] aby wyjsc

[pr ttl], [print ttl] aby wyswietlic wartosci parametrow TTL i SEND_PERIOD
[set TTL 'ttl'] aby ustawic TTL
[set SEND_PERIOD 'send_period'] aby ustawic okres rozsyłania ARP

[at 'IP'], [attach 'IP'] w celu dodania host'a do grupy klientow
[dt 'IP'], [detach 'IP'] w celu usuniecia host'a z grupy klientow
[col], [connect] w celu otwarcia polaczen z grupa klientow
[dil], [disconnect] w celu zamknienia polaczen z grupa klientow
[pr cl], [print clients] aby wyswietlic liste klientow

[lo ho 'mac'], [load host 'mac'] w celu zaladowania historii hosta o danym MAC
[lo ho 'mac' 'start'], [load host 'mac' 'start'] w celu zaladowania historii hos
ta o danym MAC od zadanej daty 'start' mm-dd-rrr
[lo ho 'mac' 'start' 'stop'], [load host 'mac' 'start' 'stop'] w celu zaladowani
a historii hosta o danym MAC w zadanych ramach 'start' 'stop' mm-dd-rrr

[lo net 'netaddr'], [load network 'netaddr'] w celu zaladowania stanu danej siec
i z bazy
[pr net], [print netaddr] aby wyswietlic adresy sieci do ktorych naleza hosty z
bazy

[shl], [show] aby wyswietlic graficzny monitor sieci w obecnym trybie
[sh lc], [show local] aby wyswietlic stan sieci w ktorej pracuje aplikacja
[sh ex], [show external] aby wyswietlic stan sieci dolaczonych klientow
[sh ar], [show archival] aby wyswietlic stan zaladowanej sieci archiwalnej
[hil], [hide] aby ukryc graficzny monitor sieci

lanmonitor $ print net

*****
* Wykaz adresow sieci z bazy archiwalnych hostow:
*
* 192.168.0.0
* 192.168.1.0
* 192.168.46.0
* 5.0.0.0
*****
lanmonitor $

```

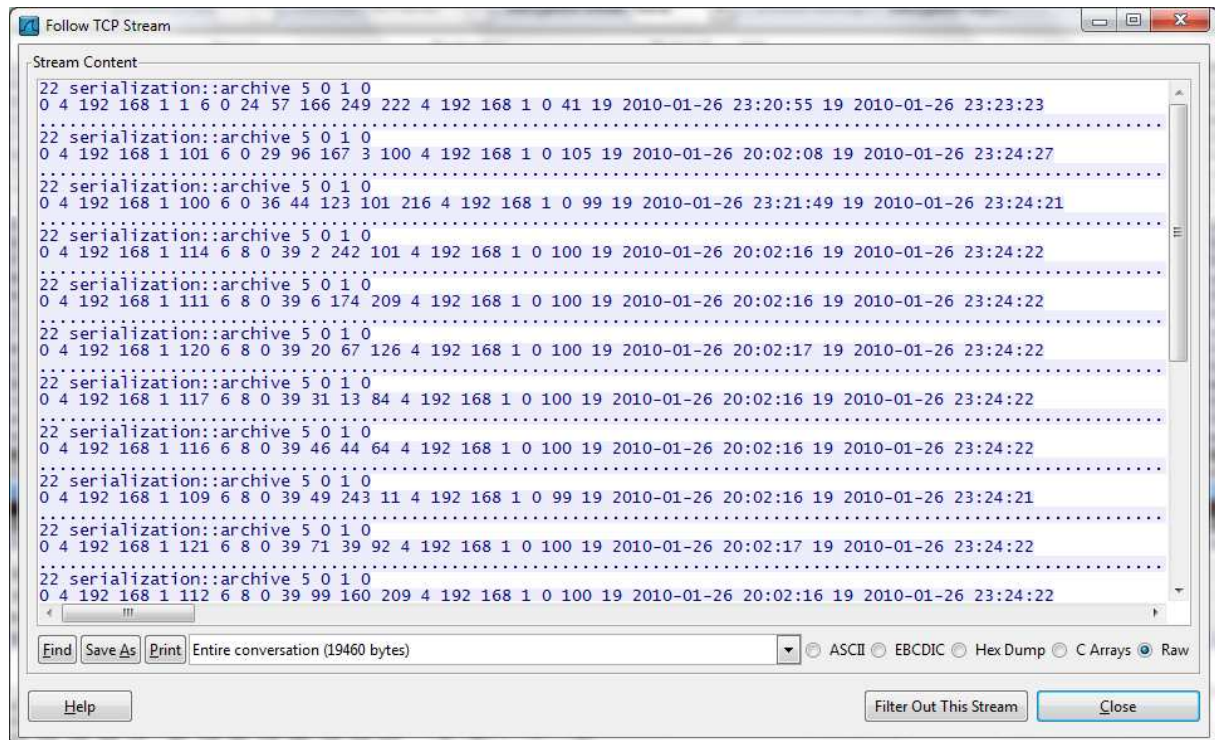
CLI – przykład użycia

OPIS KLUCZOWYCH FUNKCJI I METOD

Serializacja obiektów

ROLA	Serializacja służy to zamiany obiektu na ciąg bajtów, tak, aby można było go przesłać przez sieć, a następnie odtworzyć go po <i>drugiej stronie</i> .
ZASTOSOWANIE	W naszej aplikacji serializowane są obiekty ActiveHost z kontenera hostów sieci lokalnej, a następnie wysyłane są połączeniem TCP do

	<p>aplikacji która tego żądała. Wykorzystano bibliotekę boost::serialization która umożliwia serializowanie danych typów prostych, tablic a także obiektów i szablonów z biblioteki standardowej C++. Klasa ActiveHost implementuje metody save oraz load służące odpowiednio serializacji i deserializacji.</p>
--	--



Zserializowane ActiveHosty w strumieniu TCP w sniffer'ze Wireshark

WinPcap - Obsługa interfejsów sieciowych, wysyłania i przechwytywania pakietów

ROLA	<p>CNetworkAdapter zbudowany w oparciu o funkcje WinPcap stanowi podstawę aplikacji zapewniając dostęp do interfejsów sieciowych, rozsyłanie i przechwytywanie pakietów.</p>
ZASTOSOWANIE	<p>Przy użyciu funkcji:</p> <pre>pcap_findalldevs_ex(source, NULL, &alldevs, errbuf);</pre> <p>Uzyskujemy informacje na temat interfejsów sieciowych. Uzyskujemy komplet adresów sieciowych i masek dla każdego adaptera sieciowego. Zależnie od późniejszej decyzji, będziemy korzystać z jednego z nich.</p> <pre>pcap_sendpacket(fp_, arp_req, utils::ARP_REQ_SIZE);</pre> <p>Pozwala przesłać dowolnie skomponowany pakiet. Przy użyciu tej metody nasze ramki ARP są rozsyłane po danej podsieci.</p>

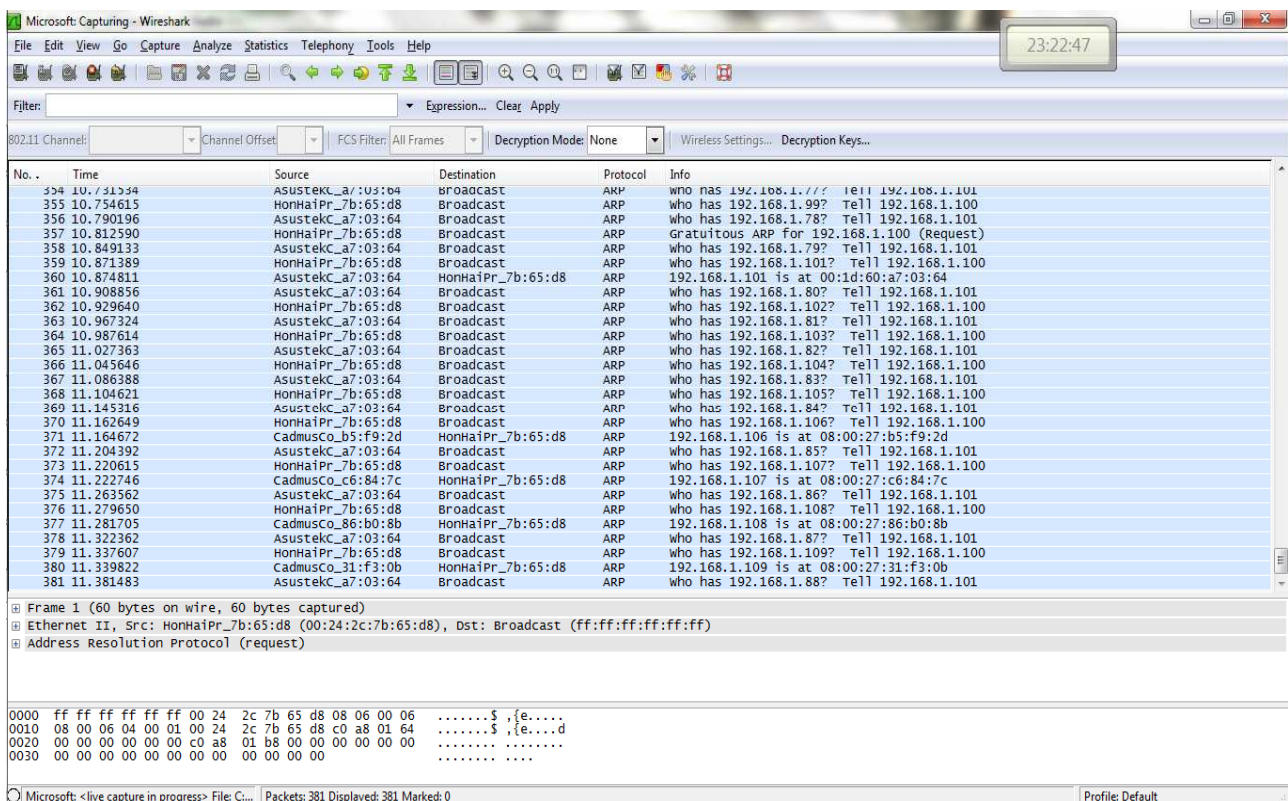

```
pcap_compile(fp_, &fcode, packet_filter, 1, netmask);

pcap_setfilter(fp_, &fcode);
```

Służy do skompilowania i ustawienia filtra przechwytywania. Dzięki temu analizujemy tylko pakiety ARP;

```
pcap_next_ex( fp_, &header, &pkt_data));
```

Przy użyciu tej funkcji odrębny wątek przechwytuje pakiety ARP, z których następnie selekcjonowane są odpowiedzi i pozyskiwane informacje o adresach sieciowym i sprzętowym stacji.



Pakiety ARP (żądania i odpowiedzi) w sniffer'ze Wireshark

POŁĄCZENIA TCP

ROLA	Komunikacja pomiędzy instancjami aplikacji w różnych sieciach odbywa się poprzez połączenia TCP.
ZASTOSOWANIE	W celu realizacji możliwości monitorowania wielu podsieci z poziomu zarządcy zdecydowaną się na realizację prostej architektury p2p. Każda instancja programu oczekuje na porcie 20011 na połączenia przychodzące od innych instancji aplikacji. Po ustanowieniu połączenia aplikacja która nasłuchiwała wysyła informacje o wszystkich aktywnych hostach poczym kończy połączenie (przesyłając odpowiedni ciąg

	<p>znaków). Najważniejsze metody klasy CConnectionMgr realizujące łączność TCP to:</p> <ul style="list-style-type: none"> • listen (port) – oczekiwanie na połączenia, domyślnie port 20011, metoda uruchamiana w wątku przez metodę startListening • connect (ip, port) – próba połączenia z hostem • sendInfo (tcpsocket) – wysłanie informacji o aktywnych hostach, uruchamiane, w metodzie listen, po ustanowieniu połączenia TCP. Działa w oddzielnym wątku. • receiveInfo (tcpsocket) – odbieranie informacji o hostach w innej podsieci, uruchamiane gdy connect zakończy się ustanowieniem połączenia. Działa w oddzielnym wątku. • connections (port) – metoda wywołująca sekwencyjnie connect dla wszystkich obserwowanych podsieci. Działa w oddzielnym wątku. <p>Komunikacja sieciowa realizowana jest przy użyciu biblioteki SDL_net umożliwiającej programowanie na poziomie gniazdek sieciowych.</p>
--	--

POŁĄCZENIE Z BAZĄ SQLITE

ROLA	CDataBaseWrapper zapewnienia dostęp do bazy SQLite i realizację zapytań SQL, a także pilnuje, aby hosty odłączone usuwać z kontenera hostów aktywnych.
ZASTOSOWANIE	<pre>sqlite3_open("../sqlite/ARPreCORD.sqlite", &database);</pre> <p>Funkcja ta otwiera połączenie z bazą danych.</p> <pre>sqlite3_prepare_v2(database, query.str().c_str(), -1, &statement, NULL);</pre> <p>i</p> <pre>sqlite3_step(statement);</pre> <p>Używane są przy obsługiwaniu zapytań sql, a zatem przy zapisie host'ów i pobieraniu danych z bazy.</p> <p>Dodatkowo metody enqueueReceived i handleReceived dbaja o obsługę kontenerów aktywnych hostów i hostów z innych podsieci.</p>

PODSUMOWANIE

Dostarczona aplikacja stanowi propozycję konstrukcji monitora aktywności w oparciu o protokół ARP i analizę jego ramek. Jego konstrukcja zapewnia sprawną wizualizację stanu

sieci do 200 stacji i monitorowanie sieci znacznie większych. Dzięki budowie aplikacji bez wyraźnego podziału na role klient/zarządca, można sprawnie modyfikować konfigurację hierarchii bez potrzeby istotnych zmian (fizyczne przenoszenie klientów i zarządcy).

Sprawną konsolą zapewnia szybki dostęp do wszystkich funkcjonalności, a proste odpytania bazy danych pozwalają na ograniczoną analizę aktywności poszczególnych użytkowników i podsieci.

Implementacja ta miała jednak głównie cel dydaktyczny i cel ten spełniła.

Obserwując efekt naszej pracy po jej zakończeniu, dostrzegamy słabości niektórych rozwiązań, możliwości optymalizacji działania i zwiększenia łatwości obsługi.

Do najistotniejszych zalicza się eliminacja zjawiska zalewania sieci pakietami ARP poprzez poprawę algorytmu wysyłania pakietów ARP.

Obecny algorytm wysyłania pakietów polega na ciągłym zalewaniu sieci pakietami ARP i analizowaniu odpowiedzi. Jest to algorytm nieoptymalny.

Propozycją jego zmiany jest wysyłanie pakietów na wszystkie adresy tylko na początku działania programu lub w rzadkich odstępach czasu. Przez pozostały czas należałoby analizować pakiety ARP wysyłane przez stacje robocze na adres broadcast'owy w czasie 'normalnej' aktywności sieciowej.

Inne udoskonalenie może polegać na wysyłaniu pakietów tylko do aktywnych hostów.

Nowe hosty zaraz po podłączeniu do sieci lokalnej wysyłają kilka pakietów ARP w celu rozpoznania adresu fizycznego bramy domyślnej oraz w celu otrzymania adresu IP (lub sprawdzenia, czy przydzielony adres statyczny nie występuje w sieci).

Tak samo ma się sprawa transmisji sieciowej, która zamiast po jednym porcie, mogłaby być oparta o dwa. Na jednym odbywałby się nasłuch, na drugim przesyłanie danych.

W przypadku transmisji, niezbędnym, a obecnie nieobecnym, jest również zabezpieczenie całego procesu.

Dodatkowo, gdyby zdecydować się na inne biblioteki graficzne/opisu GUI, można by było zapewnić znacznie łatwiejszy, interakcyjny interfejs graficzny, dający łatwiejszy dostęp do historii konkretnych hostów, a także zapewniający lepszą wizualizację stanu hostów w poszczególnych podsieciach.

Koniec