GROUP ADDRESS REGISTRATION PROTOCOL

Plan prezentacji

- Trochę historii
- Moduły definiowane przez protokół
- Opis możliwych akcji w ramach protokołu
- Model
- Weryfikacja
- Konkluzja

1. Trochę historii.

Protokół GARP został zaprojektowany na potrzeby rozsyłania komunikatów do wielu odbiorców znajdujących się w jednej sieci LAN, składającej się z segmentów połączonych mostami. Pierwsze prace na jego temat pojawiły się w 1993(!) roku. Jego weryfikację przeprowadzono w 1997 roku i mniej więcej wtedy włączono do standardów IEEE. Opis tego protokołu znalazł się w dokumencie IEEE P802.1p „Standard for Local and Metropolita Area Networks – Suplement to MAC Bridże: Traffic Expediting and Dynamic Multicast Filtering" Dość szybko powstała odmiana tego protokołu dla sieci VLAN – GVRP, zmieniono też wówczas nazwę protokołu na Group Attribute Registration Protocol. Jednym z typowych atrybutów, do rejestracji których używa się tego protokołu jest numer

wirtualnej sieci lokalnej (VLAN) do której należy dana stacja robocza.

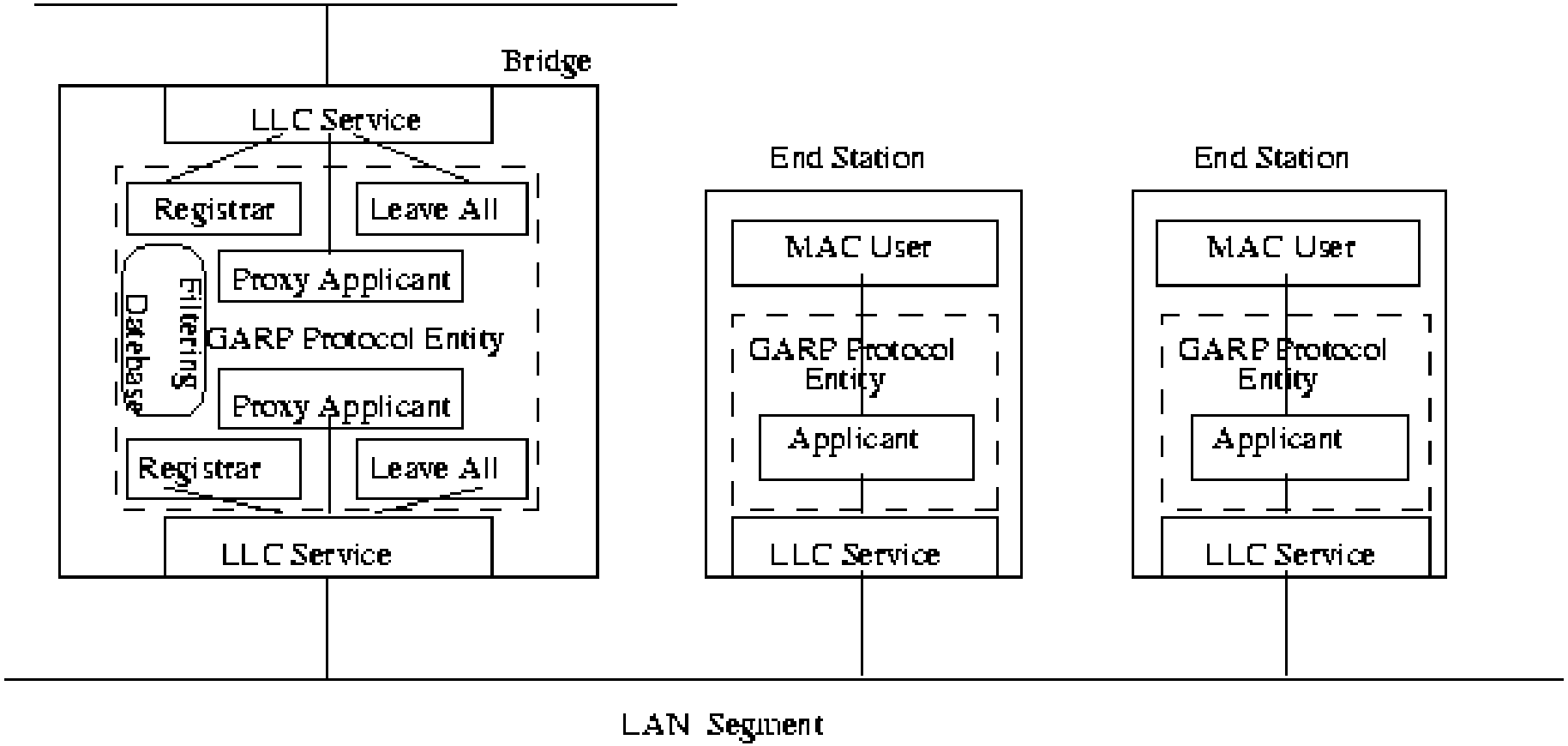
GARP umożliwia dynamiczne tworzenie grup odbierających ten sam ‘broadcast’owany’ sygnał w obrębie sieci LAN z mostami, bez jego niepotrzebnej multiplikacji. Do tego celu budowane jest drzewo rozpinające graf połączeń pomiędzy segmentami sieci.

3 maja 2005 roku ukazał się dokument określający protokół Multiple Registration Protocol (MRP), razem z procedurami i definiowanymi modułami. Protokół MRP został zaprojektowany, by zastąpić GARP, którego wadą, jak okazało się w trakcie użytkowania, był długi czas rejestrowania zmian w sieci i relatywnie duży narzut komunikacyjny, zmniejszający przepustowość sieci. Szczególnie było to widoczne w dużych

sieciach, w sytuacji gdy zmianie ulegała topologia niewielkiej części sieci.

2. Moduły definiowane przez protokół

- **Applicant:** Moduł zawarty w każdej stacji końcowej sieci. Pozwala użytkownikowi stacji końcowej na dynamiczne rejestrowanie i wyrejestrowanie się z grupy.
- **Registrar:** Moduł obsługujący jeden port mostu sieci LAN. Odpowiada na żądania modułów **Applicant** w stacjach roboczych segmentu sieci podłączonego do tego portu.
- **Proxy Applicant:** Również obsługuje pojedynczy port mostu sieci LAN. Przekazuje do segmentu sieci LAN podłączonego do tego portu żądania **Applicant**'ów z innych segmentów, o ile dana krawędź istnieje w drzewie rozpinającym graf sieci.
- **Leave All:** Skojarzony z jednym portem mostu sieci LAN, odpowiada za okresowe odświeżanie przynależności do grup, czyli *garbate collection*.



Schemat połączeń i rozmieszczenia modułów GARP w jednym segmencie sieci LAN łączonej mostami.

3. Opis możliwych akcji w ramach protokołu

1. Użytkownik zgłasza chęć dołączenia do grupy.

   Moduł **Applicant** wysyła komunikat JOIN i od tej chwili uważa się już za członka grupy. Po wysłaniu komunikatu **Applicant** czeka przez *JoinTime (zalecana wartość dla sieci 100Base-T, to w zależności od konfiguracji sieci – od 10 do 75 ms)* na ewentualne zgłoszenia dołączenia do grupy przez innych użytkowników w danym segmencie. Jeśli takie zgłoszenie się pojawi, **Applicant** nie podejmuje żadnej akcji. W przeciwnym razie ponownie wysyła komunikat JOIN.

2. **Registrar** odpowiada na żądanie dołączenia do grupy

   Każdy **Registrar,** który odbierze komunikat JOIN aktualizuje zawartość bazy danych przechowującej adresy członków poszczególnych grup. Jeśli komunikat JOIN jest żądaniem przyłączenia do nieistniejącej grupy, grupa taka jest tworzona.

3. Wyrejestrowanie rozpoczęte przez **Applicant**'a

   Ten przypadek ma miejsce, gdy użytkownik stacji końcowej zgłosi żądanie opuszczenia grupy. **Applicant** wysyła wówczas komunikat LEAVE i uznaje, że opuścił grupę. Każdy inny **Applicant** w tym segmencie sieci, jeśli chce pozostać w grupie, musi wysłać komunikat

JOIN, a następnie zachowywać się tak jak przy zwykłej rejestracji.

4. **Registrar** odpowiada na komunikat wyrejestrowania

   Registrar po otrzymaniu komunikatu LEAVE dotyczącego grupy zarejestrowanej na porcie tego **Registrar'**a czeka przez czas *LeaveTime (zalecana wartość waha się od 400 ms do 1 s, w zależności od rodzaju sieci i jej obciążenia)* na możliwe komunikaty JOIN. Jeśli nadejdzie chociaż jeden taki komunikat, **Registrar** nic więcej nie robi. Jeśli w tym czasie przyjdzie kolejny komunikat LEAVE, **Registrar** znowu czeka przez *LeaveTime.* Jeśli przez ten czas nie

przyjdzie żaden komunikat nie przyjdzie, **Registrar** ponownie czeka przez *LeaveTime* i jeśli nie dostanie żadnych komunikatów to wyrejestrowuje segment sieci LAN podłączony do jego portu z grupy.

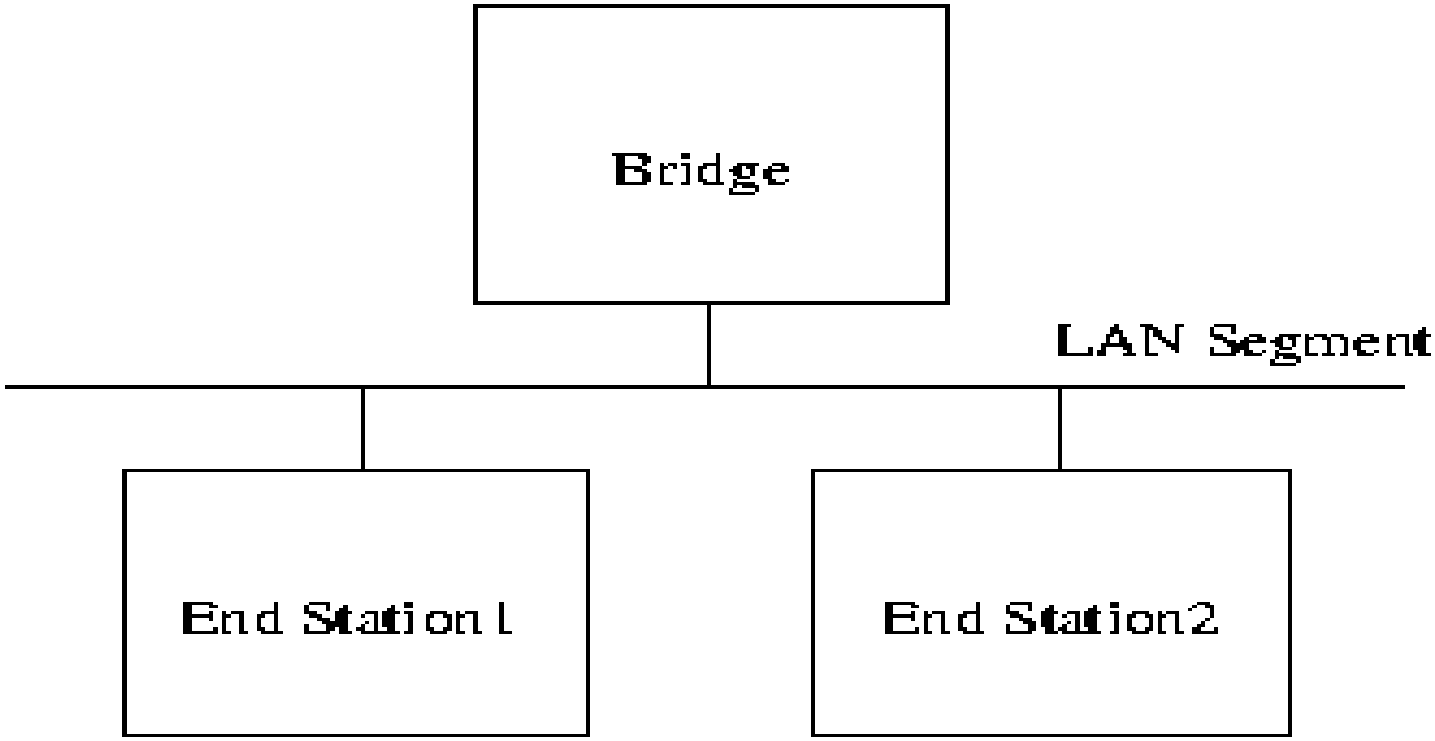
5. Most inicjuje usuwanie grupy

   Moduł **LeaveAll** co czas *LeaveAllTime* wymusza ponowną rejestrację wszystkich MAC-adresów podłączonych do tego portu mostu do którego jest podłączony. Odbywa się to poprzez wysłanie komunikatu LEAVE do tej podsieci. Wszystkie moduły **Applicant** które chcą pozostać w grupie muszą odpowiedzieć tak jak w przypadku wysłania komunikatu LEAVE przez moduł **Applicant**. Zachowanie modułu

Registrar jest również takie samo jak w przypadku wysłania komunikatu LEAVE przez moduł **Applicant.** Ta procedura ma na celu zmniejszenie obciążenia sieci, przez eliminację z grupy stacji końcowych, które nie są zainteresowane członkostwem w grupie, ale nie wyrejestrowują się poprzez swoje moduły **Applicant**.

4. Model

Jako wystarczająco kompletny model dla weryfikacji protokołu przyjęto system składający się z jednego segmentu sieci LAN z jednym mostem i dwiema stacjami końcowymi.



Modelowanie objęło następujące funkcje:

1. Stacja końcowa:
   a. **Applicant**
   b. Usługa LLC
   c. Użytkownik adresu MAC
2. Most:
   a. **Proxy Applicant**
   b. **Registrar**
   c. **Leave All**
   d. Usługa LLC

Usługa LLC (Logical Link Control) reprezentuje podwarstwę warstwy drugiej protokołów sieciowych wg OSI i odpowiada m.in. za sterowanie przepustowością sieci, za to jakie pakiety są nadawane i do którego protokołu warstwy trzeciej trafiają odebrane dane.

Szczegóły modelu:

1. Kolejki komunikatów na wyjściu i wejściu usługi LLC zostały wymodelowane jako synchroniczne. Kolejki wyjściowe usługi LLC mają rozmiar 1, by modelować bufory wyjściowe kart sieciowych, a kolejki wejściowe mają rozmiar 0.

2. Modelowana sieć gubi pakiety, tzn. proces usługi LLC przekazuje komunikaty od nadawcy do 0-2 odbiorców niedeterministycznie.
3. Timeouty. Moduły **Applicant, Registrar, Proxy Applicant** i **Leave All** mają zegary. Ponieważ jednak PROMELA nie ma reprezentacji czasu, wprowadzono flagi dla każdego timera. Wystąpienie timeoutu jest rozpoznawane gdy odpowiednia kolejka komunikatów jest pusta.
4. Aby poprawnie rozpoznawać sytuację zakleszczenia, w modułach **Applicant, Registrar, Proxy Applicant** i Usługa LLC do stanów oczekiwania na komunikat dołączono etykiety końcowe (*end labels*)
5. Aby rozpoznawać odebranie niepoprawnego komunikatu, w miejscach, komunikat może się pojawić i

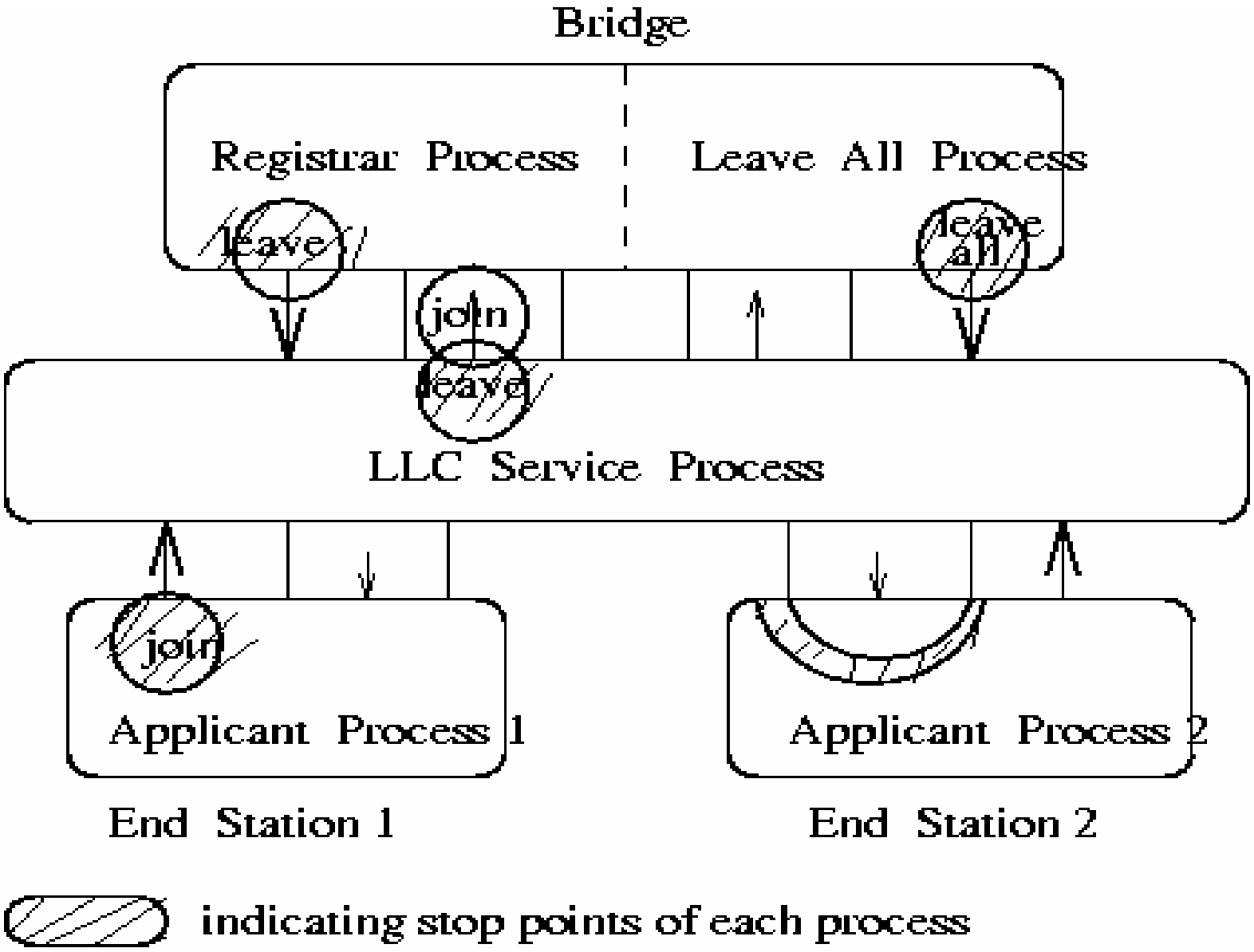
nie jest żadnym z możliwych komunikatów umieszczono ‘assertion(0)’

6. By zachować dostateczną ogólność, pozwolono by stacje końcowe nie były symetryczne.

5. Weryfikacja

Najpierw sprawdzano brak nieznanych komunikatów, nieosiągalnego kodu i blokad.

Uruchomiono SPINa w trybie Supertrace/Bitstate z blokującymi kolejkami komunikatów. SPIN szybko znalazł stan blokady i wskazał scenariusz jego powstania. Przyczyną blokady było równoczesne wysyłanie komunikatów przez proces **Registrar** i Usługa LLC gdy kolejka odbiorcza jednego z tych procesów jest pełna.


Bridge
Registrar Process
Leave All Process
leave
leave all
join
leave
LLC Service Process
join
Applicant Process 1
Applicant Process 2
End Station 1
End Station 2
indicating stop points of each process

Mając kody modułów odtworzyłem sytuację wystąpienia błędu, rzeczywiście występuje blokada, SPIN znalazł ją po około 60 krokach.

W rzeczywistym systemie, mosty i stacje końcowe mają z reguły dostatecznie pojemne bufory, by to takich sytuacji nie dochodziło. Ale, ponieważ teoretycznie może się to zdarzyć, więc zmieniono zachowanie kolejek, by nie były blokujące, tylko gubiły komunikaty przychodzące, gdy są pełne.

Dotyczy to jednak **tylko** kolejek wejściowych do procesu Usługi LLC, ponieważ pozostałe kolejki mają zerową pojemność.

Po wprowadzeniu takiej zmiany SPIN nie znalazł żadnych błędnych zachowań, żadnych blokad i żadnego nieosiągniętego kodu. Jednakże pokrycie przestrzeni stanów było niewielkie – współczynnik hash factor był równy 5.25. Maszyna na której wykonywano testy miała około 100 MB RAM i procesor Pentium-120. Na komputerze wyposażonym w procesor Athlon ~1800 MHz i 512 MB RAM współczynnik hash factor wyniósł około 97, a po pozwoleniu SPINowi na kompresję wzrósł do około 194.

Autorzy pracy [1] zdecydowali się jednak nie czekać na lepszy sprzęt. Zamiast tego podzielili model tak, by móc sprawdzać dwa mniejsze modele osobno, lecz zachować ogólność.

Pierwszy model składał się z:

1. Jednego mostu z jednym modułem **Registrar**
2. Dwóch stacji końcowych, każda z jednym modułem **Applicant** i jednym procesem użytkownika.
3. Jednego segmentu sieci LAN (usługi LLC)

Drugi model zawierał dodatkowo moduł **Leave All** w moście, natomiast zawierał tylko jedną stację końcową.

Dzięki takiemu podziałowi, SPIN mógł stwierdzić, że nie są osiągalne fragmenty kodu z ‘assertion(0)’ (czyli nie ma komunikatów innych niż wyspecyfikowane), nie ma blokad i wszystkie dobre stany są osiągalne. Współczynnik hash factor wynosił 14.4 i 213, co uznano za wynik akceptowalny.

Temporal Claim

Najistotniejsze wymaganie stawiane protokołowi GARP to „zgodność członkostwa”. Oznacza to, że jeśli w segmencie LAN jest chociaż jedna stacja końcowa należąca do grupy, most musi ją zobaczyć tak szybko jak to możliwe (przynajmniej w końcu).

Używając logiki temporalnej wyrażono tę własność w następujący sposób:

$$\mathbf{A(p \longrightarrow F(Aq))}$$

Gdzie:
p= (Proces P1 użytkownika MAC kończy się)
q= (Stan procesu **Registrar** jest różny od OUT)
i proces P1 został zmodyfikowany tak, by wysyłać komunikat ReqJoin tylko raz.

Formuła została przekształcona na ‘Never Claim’ w standardowy sposób, za pomocą XSPINa. Zanim ją jednak zweryfikowano, usunięto proces reprezentujący moduł **LeaveAll**, ponieważ rozumiejąc własność „zgodności członkostwa” jako zachodzącą „tak szybko jak to możliwe”, nie powinniśmy polegać na module pełniącym *de facto* rolę

„odśmiecacza”. Również moduł Usługa LLC został zmodyfikowany, by reprezentować sieć **nie** gubiącą pakietów, ponieważ w przeciwnym razie powyższa formuła w oczywisty sposób nie byłaby spełniona. Kolejki komunikatów jednak wciąż gubią komunikaty przy przepełnieniu.

SPIN w takich warunkach znalazł scenariusz w którym dochodzi do złamania tej formuły, jednak jest on wysoce nieprawdopodobny.
Scenariusz falsyfikacji formuły:
Applicant1 otrzymuje komunikaty z kolejki z takim opóźnieniem, że te komunikaty są gubione.

Spróbowano więc zwiększyć rozmiar kolejek procesu UsługaLLC z 1 do 3.

Okazało się to wystarczające – SPIN wykazał, ze współczynnikiem hash coverage równym 64.2, że formuła **nie jest fałszywa w żadnym stanie.**

6. Konkluzja

SPIN pomógł wykazać, że protokół GARP jest wolny od takich błędów jak blokady, nieosiągalne stany czy otrzymywane nieznane komunikaty. Ponadto, z pomocą SPINa wykazano, że protokół zapewnia prawdziwość formuły **temporal claim** „zgodności członkostwa”.

Autorzy [1] wyrażali podziw dla potencjału SPINa, szczególnie w kontekście możliwości modelowania w PROMELi zegarów i multicastingu, nie wspieranych bezpośrednio przez ten język.

Pomimo tego, że praca [1] kończy się stwierdzeniem, że protokół GARP się zmienia i pojawia się potrzeba zweryfikowania poprawności tych zmian, to późniejsze prace dotyczące GARP/GVRP/GMRP dotyczą implementacji, a nie weryfikacji (przynajmniej nie weryfikacji z użyciem SPINa).

Bibliografia:

1. Tadashi Natatani, Verification of Group Address Registrarion Protocol using PROMELA and SPIN, Febuary 1997
2. Fouad A. TobagiPablo Molinero-FernándezMansour J. Karam, Study of IEEE 802.1p GARP/GMRP TimerValues
3. IEEE P802.1p Supplement to ISO/IEC 15802-3 (802.1D): Information Technology - Telecommunications and information exchange between systems - Local & Metropolitan Area Networks - Common specifications - Part 3: Media Access Control (MAC) Bridges - GARP Proprietary Attribute Registration Protocol (GPRP), 1998

4. IEEE 802.1ak Standard for Local and Metropolitan Area Networks Virtual Bridged Local Area Networks - Amendment 07: Multiple Registration Protocol, maj 2005