Optymalizacja działania rozproszonej bazy danych na podstawie badań efektywnościowych X.500*

Maja Górecka
mgorecka@cc.uni.torun.pl

Tomasz Wolniewicz
twoln@mat.uni.torun.pl

Uniwersytet Mikołaja Kopernika w Toruniu

1 Wstęp

Celem niniejszego opracowania jest przedstawienie wniosków z badań efektywnościowych X.500 Directory, a dokładniej konkretnej implementacji standardu — oprogramowania QUIPU wersja 12.0.

Bazą dla pracy był nasz wstępny raport [4], w którym przedstawiliśmy rozważania teoretyczne nad czynnikami mającymi wpływ na sprawność działania rozproszonej bazy danych, jaką jest X.500. Zapoznanie się z tym raportem byłoby niewątpliwie pomocne przy czytaniu przedstawianego tutaj opracowania.

Obecnie chcemy podsumować wyniki testów i skonfrontować je z wcześniejszymi przewidywaniami. Od razu możemy nadmienić, że nie wszystkie testy dały oczekiwane wyniki. Głębsza analiza tych rozbieżności wykazała, że niektóre zaimplementowane w badanym oprogramowaniu metody nie działają zgodne z ich opisem, w pewnych sytuacjach natomiast wpływ niektórych czynników na działanie bazy jest tak silny, że uniemożliwia ocenę znaczenia elementu badanego.

Nasz raport opisuje, często dość szczegółowo, mechanizmy działania bazy X.500 w implementacji QUIPU. Wydaje się nam, że taka głębsza wiedza może być bardzo przydatna administratorom serwisu X.500, a nie spotkaliśmy dotychczas opracowań, które przybliżałyby rozważane przez nas tematy dostatecznie wyczerpująco.

Nie będziemy omawiali samego schematu bazy X.500. Powiemy tylko, że analizujemy rozproszoną bazę danych opartą na serwerach (DSA), do której dostęp zapewnia oprogramowanie klienta (DUA). Dokładny opis funkcjonowania bazy i używanej terminologii można znaleźć między innymi w naszych raportach [4], [5], [6], a także w [2] i [1], natomiast opis użytkowy oprogramowania QUIPU zawarty jest w [3].

W raporcie zamieszczamy też wyniki pomiarów działania polskiej bazy X.500, które zebraliśmy za pomocą specjalnie do tego celu napisanego oprogramowania. Niektóre pomiary dokonywane były również na testowo uruchamianych instalacjach, robiliśmy to w przypadkach, gdy konieczne było dokładne wydzielenie wpływu jednego badanego czynnika na zachowanie serwera bazy.

*Praca wykonana w ramach statutowej działalności NASK

2 Obsługa operacji rozproszonych

System rozproszony (jakim jest X.500) przechowuje gromadzoną informację na wielu serwerach (DSA). Dostęp klienta (DUA) do bazy rozpoczyna się od dowiązania do wybranego DSA i zadania pierwszego zapytania. W zależności od tego, czy realizacja odpowiedzi może być dokonana lokalnie, czy też niezbędne jest odwołanie do innych serwerów będziemy mieli do czynienia z występowaniem operacji rozproszonego dostępu lub nie. Do wymiany informacji pomiędzy DUA a DSA służy protokół DAP (*Directory Access Protocol*), serwery DSA komunikują się w ramach protokołu DSP (*Directory System Protocol*).

W części tej opisujemy jak oprogramowanie QUIPU podejmuje decyzje o nawiązaniu kolejnych połączeń i jaki ma to wpływ na efektywność działania całej bazy.

2.1 Wybór serwera

DSA obsługujący zlecenie nie dające się zrealizować lokalnie może albo przekazać je kolejnemu serwerowi poprzez tworzenie łańcucha (*chaining*), albo zwrócić DUA odesłanie do innego serwera DSA (*referral*). Identyczny protokół komunikacyjny obowiązuje również w trakcie wymiany informacji pomiędzy serwerami.

Wybór właściwego DSA jest bardzo istotny z punktu widzenia efektywności pracy X.500. Na ogół istnieje kilka serwerów, które prawdopodobnie są w stanie obsłużyć zlecenie. QUIPU DSA w tej sytuacji dokonuje uporządkowania listy serwerów według jakości operacyjnej i następnie przekazuje żądanie do DSA uznanego jako najlepsze, w przypadku niepowodzenia, wybiera kolejny serwer z listy itd.

Sposób postępowania przy ocenie serwerów jest określony w implementacji QUIPU w ramach algorytmu, który bazuje na informacjach o parametrach własnych DSA oraz zapamiętanych danych, dotyczących aktywności serwerów. Dodatkowo umożliwia się administracyjny wpływ na wybór serwera docelowego poprzez podanie w zbiorze konfiguracyjnym QUIPU wartości tzw. `preferDSA`.

Algorytm ustala następujące uporządkowanie elementów decydujących o wyborze DSA:

1. Istnienie nawiązanego połączenia z odpowiednim serwerem.

2. Docelowe DSA jest QUIPU DSA, czyli w `objectClass` znaleziono wartość `quipuDSA`. Dodatkowo sprawdzany jest atrybut specyfikujący akceptowane typy protokołów, zwane kontekstem aplikacji (`SupportedApplicationContext`), tzn. te wszystkie elementy warstwy aplikacji (ASE — Application Service Element), które serwer implementuje. Pierwszeństwo otrzymuje DSA z kontekstem QUIPU, następnie DSA według protokołu Internet, wszystkie pozostałe traktowane są jednakowo, z wyjątkiem serwera DSA, który podtrzymuje wyłącznie protokół DAP.

3. Docelowe DSA zostało uznane jako bardziej niezawodne. Ocena ta bazuje na śledzeniu informacji o czasie ostatniej próby dostępu do DSA oraz jej efekcie, tzn. czy połączenie zakończyło się sukcesem oraz czy wystąpiły błędy. W ten sposób preferowany jest np. ten serwer, który nie był ostatnio aktywny w sytuacji, gdy konkurencyjny miał nawiązane połączenie, ale w trakcie dostępu zawiódł. Z kolei w przypadku, gdy dwa serwery były aktywne, zwycięża ten, który nie zawiódł, a jeżeli oba miały błędy — serwer, dla którego nie minął czas między kolejnymi próbami nawiązania połączenia (*retrytime*). Wiele przypadków traktowanych jest jako nierozstrzygalnych, m.in. gdy oba serwery nie były aktywne, gdy oba były, jeden zawiódł, ale dla obu upłynął czas *retrytime*, gdy jeden był aktywny, ale zawiódł i upłynął czas ponowienia próby połączenia.

4. DSA dostępne jest we właściwej kategorii protokołu sieciowego (*community*), a w przypadku, gdy dwa DSA znajdują się w tej samej kategorii, wybierany jest serwer uznany za lokalny na bazie analizy wyróżnionej nazwy DSA (*Distinguished Name*).

5. DSA jest uprzywilejowany na podstawie listy `preferDSA`.

Jak widać algorytm wyboru DSA nie jest rozbudowany. W minimalnym stopniu wykorzystuje „historię” połaczeń, ogranicza się wyłącznie do sprawdzania, czy DSA był aktywny i czy wystąpiły błędy w trakcie pracy. Nie jest zapamiętywany czas niezbędny do nawiązania łączności, czy średni czas połączenia. Oczywiście może to prowadzić do częstego korzystania z serwerów, które spełniają warunki kwalifikacyjne, natomiast kontakt z nimi jest czasochłonny.

Wątpliwości budzi określona kolejność czynników decydujących o doborze DSA. Małą rolę odgrywa administracyjne ustalenie listy uprzywilejowanych serwerów; jedynie w przypadku niemożliwości rozstrzygnięcia na podstawie niezawodności i lokalizacji DSA wybierany jest serwer wskazany przez administratora. W przypadku zapytań pochodzących np. z polskich serwerów regionalnych zapewnienie wysokiego priorytetu wyborowi DSA według listy wskazanej przez `preferDSA`, na której znajdowałby się serwer krajowy, miałoby zdecydowanie korzystny wpływ na obsługę zapytań dotyczących danych replikowanych przez DSA Polski. Typowym rozwiązaniem mającym na celu usprawnienie działania bazy jest replikowanie fragmentów drzewa geograficznie odległych. Na przykład lista instytucji polskich jest replikowana przez krajowy serwer australijski. Fakt ten oznacza jednak, że serwer australijski może być wybierany jako źródło informacji o Polsce przez inne serwery europejskie (a nawet polskie (!)).

QUIPU daje najwyższy priorytet DSA, do którego jest już aktualnie nawiązane połączenie. Intencja autorów oprogramowania jest łatwo zrozumiała, gdy uświadomimy sobie, że nawiązanie nowego kontaktu jest zawsze kosztowne czasowo (potwierdzają to wyraźnie wyniki wykonanych przez nas testów). Rozumiejąc to zauważamy jednocześnie negatywny wpływ takiego rozwiązania. Może się zdażyć, że nawiązane połaczenie z odległym serwerem w celu uzyskania od niego informacji dla niego lokalnej przesądzi o pobieraniu z tego miejsca danych, które, gdyby nie otwarte połączenie, mogłyby zostać udostępnione z bliższego, a nawet lokalnego serwera.

2.2 Obsługa zlecania — tworzenie łańcucha lub przekazywanie odesłania do serwera

QUIPU DSA realizując zlecenie musi podjąć decyzję, w jaki sposób ma ono zostać obsłużone. Jeżeli polecenie otrzymane od DUA lub innego DSA nie może być rozstrzygnięte lokalnie jest ono albo przekazywane innemu serwerowi poprzez formowanie łańcucha zleceń (*chaining*), albo DSA zwraca DUA odesłanie do innego serwera (*referral*) i DUA samodzielnie kontynuuje zapytania.

Wybór metody może mieć istotny wpływ na efektywność pracy X.500. Decyzja *chaining* lub *referral* jest podejmowana po ustaleniu listy możliwych serwerów oraz uporządkowaniu jej zgodnie z algorytmem opisanym w rozdziale 2.1 i odbywa się w następujących krokach:

- W przypadku, gdy zgłaszająca się do DSA aplikacja legitymuje się jako proces używający protokołu DAP — służący do wymiany informacji pomiędzy DUA i DSA) zakłada się, że stosowana będzie obsługa poprzez tworzenie łańcucha zleceń, druga forma — zwrot odesłania pozostaje również w mocy. Uporządkowanie listy serwerów nie ulega zmianie. Na bazie tej listy tworzone są odpowiednie parametry dla operacji, tzw. *continuation references*, czyli wskazania dokąd kierować zlecenie po wystąpieniu błedów. Następnie sprawdzane są argumenty związane z obsługą operacji (*service control*). Jeżeli proces zgłosił się z zaleceniem, by nie przekazywać zadania do wykonania innym serwerom (*chainingprohibited*), lub by ograniczyć przeszukiwania do bazy lokalnej (*localscope*), zostaje wybrana metoda zwrotu odesłania do DUA, w przeciwnym razie stosowana jest metoda tworzenia łańcuchów zleceń.

- Jeżeli do DSA zwraca się proces o protokole DSP, wówczas następuje wybór serwera DSA priorytetowego przy wskazywaniu odesłania. Odbywa się to przez określenie czy proces wysyłający zlecenie i docelowy serwer należą do tej samej kategorii sieciowej — *community*. Oczywiście jest to możliwe jedynie w sytuacji, gdy lokalnie można zdobyć informację, jaki adres prezentacyjny posiada zlecający. Pierwszy serwer z listy dopuszczalnych DSA o *community* zgodnej z aplikacją zlecającą jest wybierany jako najlepszy. Jeżeli żaden serwer nie znajduje się w tej samej sieci, blokowana jest możliwość zwracania odesłania. W przypadku, gdy DSA nie jest w stanie ustalić adresu zgłaszającego, pierwszy serwer z listy delegowany jest jako adres ewentualnego odesłania.

  Po ustaleniu listy DSA sprawdzana jest możliwość wykonywania operacji *chaining* w ramach obsługi zlecenia według protokołu DSP. Jeżeli wystąpił jeden z następujących przypadków:

  - *chaining* został zakazany administracyjnie przez ustawienie parametru konfiguracyjnego `dspchaining` jako `off` lub `WhenNeeded`,
  - proces zgłaszający wymusza obsługę bez łańcuchowania (*chainingprohibited* lub *localscope*)

  DSA będzie obsługiwało zlecenia przez zwrot odesłania, chyba że wcześniej zablokowano stosowanie tej metody — w tym przypadku operacja zakończy się niepomyślnie z powodu zakazu tworzenia łańcucha do innych DSA i niemożliwości wysłania wskazania innego serwera.

  W pozostałych sytuacjach stosowane jest przekazywanie niezrealizowanych żądań do kolejnych DSA.

Jak wynika z opisu, decyzja o metodzie obsługi zlecenia zależy od takich czynników jak:

- środowisko pracy serwera i możliwość współpracy z innymi DSA,
- parametry konfiguracyjne serwera,
- otrzymane razem ze zleceniem argumenty operacji.

Dla zapytań od DUA typowo stosowana jest metoda budowania łańcucha, jednak interfejs użytkowy może wymusić obsługę poprzez wskazania za pomocą odpowiednich argumentów.

Jeżeli DSA pośredniczy w realizacji zlecenia, *chaining* dokonywany jest do kolejnych DSA z listy, stąd większe prawdopodobieństwo sukcesu. Dodatkowo DSA jest w posiadaniu informacji statystycznej, na podstawie której będzie w stanie podejmować kolejne akcje w imieniu DUA, samo DUA skazane jest na ścisłe podążanie za podawanymi mu odesłaniami.

Widać z tego, że tworzenia łańcucha ma szereg zalet. Niewątpliwie jest to najwłaściwszą metodą dla obsługi połączenia realizowanego w protokole DAP (inicjowanego przez DUA), a operacje według protokołu DSP (pomiędzy DSA) powinny mieć zablokowanie łańcuchowanie zgłoszeń. Aby to uzasadnić przyjmijmy najbardziej typową sytuację, w której DUA łączy się z bliskim sobie (w sensie geograficznym) DSA. DSA to posiada lokalną wiedzę o możliwości nawiązywania sprawnych połączeń z innymi DSA, które poprzednio wykorzystywało, może też preferować pewne DSA zgodnie z ustawieniami administracyjnymi. Pytania zadawane przez DUA obsługiwane są zatem zawsze w optymalny dla tego DUA sposób. Przekazanie DUA odesłania spowoduje, że nastąpi dowiązanie do obcego (być może odległego) DSA, który będzie stosował swoje mechanizmy doboru najwłaściwszych DSA, sam będzie podejmował decyzje o tym, czy tworzyć łańcuchy czy nie i w efekcie może doprowadzić do tego, ze DUA będzie korzystało z niego do końca nawiązanej sesji. Utworzenie łańcucha zawierającego więcej niż jedno DSA powoduje, że kolejne DSA nie mają właściwej informacji o tym skąd pochodzi zapytanie i mogą przekazywać nieoptymalne adresy kolejnych DSA. Towarzyszy temu oczywiście niepotrzebny transfer danych przez kilka serwerów.

Oczywiście tworzenie łańcuchów protokołu DSP daje większe prawdopodobieństwo powodzenia operacji, ale równocześnie obniża sprawność całej bazy i dlatego, naszym zdaniem, nie powinno mieć miejsca. Z tego powodu wydaje się korzystne wymuszanie administracynego zakazu tworzenia łańcuchów dla protokołu DSP. W sytuacji podłaczenia DSA tylko do jednego *community*, całkowicie zbędna jest funkcja łańcuchowania protokołu DSP, funkcja ta ma sens, gdy serwer jest podłączony do kilku *communities* i może działać jako most (*relay DSA*). W takiej sytuacji włączenie lub wyłączenie opcji zezwalającej na łańcuchówanie może być podyktowane np. argumentami finansowymi (routowanie obcego ruchu).

Wydaje się nam, że najwłaściwsze w sytuacji polskiego X.500 byłoby zmodyfikowanie algorytmu generowania odesłań, tak aby polskie DSA zwracały się do polskiego serwera krajowego, zawsze wtedy gdy serwer ten posiada poszukiwaną przez nie informację, z kolei serwer krajowy powinien blokować łańcuchowanie protokołu DSP, tak aby promować obsługiwanie lokalnych DUA, przez ich lokalne DSA.

3 Wpływ wielkości bazy na pracę serwera

Serwer DSA w implementacji QUIPU w swojej podstawowej konfiguracji ładuje do pamięci operacyjnej całą lokalną bazę danych, łącznie z wszystkimi replikacjami. Przy dużej ilości haseł prowadzi to do tworzenia bardzo pamięciochłonnego procesu.

Rozmiar bazy danych w pamięci zależy od ilości i wielkości haseł. Średnio określa się, że jedno hasło potrzebuje około 2KB pamięci.

Przeprowadzone testy wykazały m.in., że:

- Baza o ilości haseł 3500, większość haseł rozbudowana — wymaga ok. 6.5MB pamięci operacyjnej (na dysku zajmuje ok. 14MB), czyli średnio na jedno hasło przypada 1.8MB.
- Baza o ilości haseł ok. 30000, hasła o małej ilości atrybutów — wymaga ok. 33MB pamięci operacyjnej (na dysku — 27MB), czyli średnio jedno hasło zajmuje 1.1MB.
- Baza o 20000 małych haseł zajmuje 24MB pamięci operacyjnej (ok. 1.2MB na hasło), po rozbudowanu haseł — 34MB (ok. 1.7MB na hasło).

Pamięć zajmowana przez serwer, to suma wielkości samego procesu DSA i rozmiaru bazy danych powiększana przez bieżącą alokację dla potrzeb buforowania, czy wykonywania operacji replikacji danych. Zarządzanie pamięcią nie jest mocną stroną oprogramowania QUIPU. Wszystkie wersje, na bazie których pracujemy już od 3 lat m.in. nie wykonują prawidłowo zwalniania pamięci, co powoduje, że rozmiar długo pracującego serwera znacznie wzrasta.

Jeżeli proces serwera nie mieści się w całości w wolnej pamięci komputera stosowana jest metoda tworzenia pamięci wirtualnej poprzez przenoszenie części procesu na dysk (stronicowanie pamięci). W tej sytuacji istotnie obniża się efektywność pracy serwera, gdyż większość operacji wiąże się z koniecznością kontaktu z dyskiem w celu pobrania właściwego fragmentu procesu.

Z powyższych ustaleń wynika, że serwery QUIPU obsługujące duże bazy lokalne, dla zapewnienia sprawności działania wymagają odpowiedniej wielkości pamięci operacyjnej, tak by cały proces mógł rezydować w pamięci. W przeciwnym razie doprowadza się do bardzo dużych opóźnień czasowych w realizacji zapytań. Testy wykazały, że operacja przeszukiwania lokalnej bazy zawierającej 20000 haseł, obsługiwana przez serwer o pamięci operacyjnej 64MB jest wykonywana w przeciągu 1-2 sekundy, podczas gdy w przypadku uruchomienia tego serwera na tym samym komputerze, ale przy rozmiarze pamięci 32MB oczekiwanie na rezultat trwa ok. 1.5 minuty.

W pakiecie oprogramowania Isode Consortium począwszy od wersji IC R2.0 istnieje możliwość przechowywania części bazy X.500 na dysku w trakcie pracy serwera. Wydzielony fragment drzewa informacji DIT jest zarządzany przez specjalny proces zwany *delegate DSA*. Baza dyskowa posiada szereg dodatkowych zbiorów indeksowych, pomocnych przy operacjach przeszukiwania i listowania.

Stosowanie bazy dyskowej jest zalecane przy przechowywaniu dużych poddrzew, autorzy oprogramowania zapewniają sprawne działanie przy ilości haseł do jednego miliona. Korzystanie z dysku zacznie skraca czas ładowania procesu i istotnie ogranicza wymagania odnośnie pamięci operacyjnej. Usprawnienie to odbywa się jednak kosztem zwiększenia zapotrzebowania na pamięć dyskową. Bardzo czasochłonny jest też proces budowania baz indeksowych.

Testowanie wersji serwera z obsługą dyskową dla bazy gromadzącej w „delegowanym” poddrzewie ok. 3500 haseł wykazało blisko dwukrotne powiększenie zużycia obszaru dyskowego, zapotrzebowanie na pamięć spadło o ok. 3MB, przy czym w trakcie pracy rozmiar serwera w pamięci rósł stosunkowo szybko (buforowanie wyników). Sprawność reakcji przy realizacji zleceń dotyczących lokalnych zasobów w sposób zauważalny maleje, w porównaniu z sytuacją, gdy cała baza rezyduje w pamięci (opóźnienie rzędu 3-5 sekund).

W przypadku, gdy serwer ma obsługiwać rozbudowane poddrzewo danych i wiadomo, że proces serwera nie będzie dysponował wystarczającą pamięcią operacyjną, podejście dyskowe powinno znacznie podnieść efektywność.

4 Indeksowanie a wielkość bazy i sprawność jej działania

Oprogramowanie QUIPU dopuszcza indeksowanie bazy X.500 według zadanych atrybutów `optimized_attr` w ramach wskazanych obszarów drzewa danych (poddrzewo — *subtree* lub jeden poziom — *siblings*). Tworzenie indeksu odbywa się w pamięci operacyjnej, podczas ładowania bazy danych serwera. Indeks zawiera dla każdego optymalizowanego atrybutu wskazania miejsc rezydowania w pamięci haseł mających przypisane tym atrybutom wartości.

W celu przyspieszenia realizacji zapytań QUIPU używa do zapamiętywania haseł w pamięci struktur wyważonego drzewa binarnego, zwany AVL[1]. Obsługa takiej struktury jest optymalna — przy N hasłach średnio potrzeba nie więcej niż $logN$ operacji, by odnaleźć hasło szukane. Odpowiednie algorytmy dokonują wyrównania drzewa po dodaniu lub usunięciu węzłów. Również indeksy budowane są na bazie struktur AVL.

Wprowadzenie indeksowania w bazie prowadzi do dodatkowego zapotrzebowania na pamięć operacyjną. Nie jest to wprawdzie drastyczny wzrost (np. dla indeksacji według dwóch atrybutów, względem jednego poddrzewa przy ilości haseł 3500 zaobserwowano przyrost ok. 500KB, przy zwiększeniu ilości atrybutów do 6 — ok. 1.8MB), jednak w niektórych sytuacjach można indeksując przekroczyć barierę dostępnej pamięci i w efekcie spowodować znaczne pogorszenie efektywności.

Wykonane testy wykazały, że wprowadzenie indeksu nie wpływa zauważalnie na przyspieszenie operacji na bazie lokalnej. Krytycznym elementem pozostaje pamięć, jaką dysponuje serwer. Jeżeli proces serwera nie musi korzystać z technik stronicowania i *swapowania*, jego reakcja, jest zadawalająca, nawet przy przeglądaniu dużego poddrzewa i stosowaniu rozbudowanych filtrów przeszukania. Niewystarczająca pamięć wprowadza długi czas odpowiedzi, a dodanie metody indeksacji nie wnosi istotnych zmian.

Przeprowadzone próby z nowym mechanizmem bazy opartej na dysku pokazują, że stosowanie jego wydaje się bardzo ograniczać możliwości przeszukań całych gałęzi bazy. Serwer nie zezwala na dokonywanie przeszukań ze względu na atrybuty nieindeksowane. Mechamizm działania serwera z tą dodatkową opcją nie jest dostatecznie dobrze opisany w dokumentacji i prawidłowe zrozumienie algorytmów możliwe będzie dopiero po analizie kodu źródłowego.

5 Wpływ na efektywność poprzez stosowanie techniki buforowania resultatów operacji

QUIPU DSA pozwala utrzymywać w „podręcznej pamięci” typu *cache* otrzymane dane z innych serwerów DSA. Technika *„cachowania”* jest stosowana zarówno po stronie DSA jak i DUA. W trakcie pracy serwera w pamięci *cache* gromadzone są przeczytane hasła, tj. nazwa (DN — *Distinguished Name*) oraz wartości tych atrybutów, które zostały przekazane pytającemu. DSA implementuje, niezbędne dla wiarygodnej pracy serwera, algorytmy przedawniania danych w pamięci podręcznej oraz decydowania, czy informacja ta może zadowolić odbiorcę. W tym celu wprowadzane są odpowiednie interwały czasowe, w ramach których zakładana jest poprawność danych z pamięci typu *cache*. Dodatkowo zlecający razem z typem wykonywanej operacji może ustalić warunki obsługi (*service control*) i m.in. zażądać nie używania kopii przy przekazywaniu wyników.

Stosowanie techniki buforowania znacznie przyspiesza wszelkie operacje związane z ponownym odczytem informacji z bazy X.500. Z punktu widzenia serwera wprowadza jednak dodatkowe zapotrzebowanie na pamięć operacyjną i wymaga poprawnych algorytmów przedawniania informacji oraz prawidłowego administracyjnego ustalenia interwałów czasowych dotyczących czasu życia danych.

Należy podkreślić, że niezbędne jest również korzystanie z dobrze przygotowanych in-

[1] Adelson-Velskii i Landis zdefiniowali tę strukturę jako takie drzewo, w którym dla każdego węzła wysokości dwóch jego poddrzew różnią się co najwyżej o 1.

terfejsów użytkowych — DUA. DSA implementuje utrzymywanie kopii rezultatu operacji listowania tylko dla poleceń według protokołu DSP (czyli pomiędzy serwerami), poza tym w pamięci podręcznej umieszczane są odczytane hasła. Uzyskane wyniki listingu powinny więc być *cachowane* w ramach DUA. Dzięki temu można np. dodatkowo, przed zapamiętaniem w pamięci podręcznej przeprowadzić uporządkowanie. DUA działające w oparciu o prawidłowo zrealizowaną technikę korzystania z pamięci *cache* mogą znacznie podnieść efektywność pracy serwisu postrzeganą przez użytkowników. Jednocześnie, w przypadku interfejsów DUA, będących procesami krótkotrwałymi, niebezpieczeństwo przekazywania odbiorcy nieaktualnej informacji, wnoszone przez buforowanie danych jest niewielkie.

Nasze dotychczasowe doświadczenia związane z realizacją programów użytkowych opierają się na pracach nad modyfikacjami w celu dostosowania do korzystania z zapamiętanych w bazie X.500 poprawnych polskich danych. Przy obsłudze zlecenia odczytu uzyskiwana jest wyróżniona nazwa konkretnego obiektu w przeszukiwanym drzewie. Nazwa ta składa się z szeregu relatywnych nazw wyróżnionych, które pozbawione są polskich znaków narodowych. Aby uzyskać pełną, poprawną polską nazwę należy odczytać odpowiednie atrybuty każdego obiektu występującego w gałęzi prowadzącej do otrzymanego wyniku. Oczywiście, tego typu działania mogą powodować istotne opóźnienia w odbiorze rezultatu. Technika buforowania polskich nazw wyróżnionych może w opisanym przypadku istotnie poprawić efektywność.

W ramach opisywanych prac dokonaliśmy modyfikacji atrakcyjnego interfejsu POD poprzez wprowadzenie funkcji buforowania wyników i poprawę algorytmu sortowania list. Zmiana sposobu sortowania przyspieszyła działanie tego interfejsu o ok. 100 razy. Zwracamy na to uwagę, aby podkreślić jak zupełnie prosty błąd w realizacji końcowego interfejsu może osłabić działanie całego systemu.

6 Wyniki pomiarów sprawności polskiej bazy X.500

W ramach badania efektywności bazy X.500 przeprowadzilismy szereg pomiarów funkcjonowania polskiej bazy. W tym celu napisane zostało oprogramowanie monitorujące bazę o następujących funkcjach:

1. sprawdzenie średnigo czasu dostępu do komputera, na którym działa DSA (za pomocą internetowego `ping`),
2. dowiązanie do DSA (bind),
3. dokonanie kilku wyszukań lokalnych dla serwera, z którym się połączyliśmy,
4. dokonanie kilku wyszukań nielokalnych,
5. wyświetlenie wartości odszukanych haseł lokalnych,
6. wyświetlenie wartości wyszukanych haseł nielokalnych.

Oprogramowanie monitorujące oparte jest na wersji DUA o nazwie `dish`, które pozwala na wydawanie dowolnych komend DUA bezpośrednio z linii komend systemu operacyjnego. Interfejs ten jest bardzo wygodny do oprogramowania, obarczony jednak pewnym opóźnieniem uruchomienia każdej komendy (oceniamy ją na ok. 0.5s). Pomiary czasów prowadzone były z dokładnością ok. 0.1 s. Trzeba podkreślić, że toruński serwer centralny ma jeszcze obecnie połaczenie satelitarne z Warszawą, co wprowadza opóźnienia sieciowe. Sytuacja ta jest w zasadzie sprzyjająca pomiarom efektywności, gdyż pozwala łatwiej mierzyć wpływ opóźnień sieciowych na zachowanie się bazy.

Pomiar był dokonywany według następującego algorytmu:

1. dowiązanie do mierzonego serwera,

2. wydanie komendy przeszukania (typowo dwa do trzech przeszukań zwracających niekiedy pojedyncze, a niekiedy wielokrotne trafienia), czasy przeszukań były dodawane, a następnie uśredniane,

3. wydanie komendy wypisania wybranych atrybutów znalezionych haseł poprzez wielokrotne wydanie rozkazu `showentry` odpowiadającemu protokołowemu `read`, czasy były dodawane, a następnie uśredniane; ponieważ wyników operacji przeszukania było z reguły więcej niż samych operacji przeszukania, dla operacji czytania mieliśmy większą próbkę statystyczną,

4. wydanie komendy przeszukania w zakresie organizacji obługiwanej przez inny serwer (dla wszystkich przypadków były to te same dwa hasła znajdujące się na serwerze warszawskim),

5. wydanie komendy wypisania atrybutów tych haseł.

Operacja ta wykonywana jest 7 razy. Z każdego ciągu wyników odrzucany jest wynik najlepszy i najgorszy, pozostałe są uśredniane. Dodatkowo odnotowywane są również wyniki skrajne, jak też czas pierwszego dowiązania do serwera.

Dowiązanie do serwera prowadzone było zarówno w trybie anonimowym jak i ze sprawdzeniem tożsmości. Ta druga sytuacja zmusza serwery do dokonania poświadczania (*authentication*), a zatem przesłania dodatkowej informacji (co oczywiście musi opóźnić operację dowiązania).

Poza możliwościa przeprowadzenia pojedynczego cyklu prób, napisane przez nas oprogramowanie (po ewentualnym rozbudowaniu) będzie służyło jako system ciągłego monitorowania polskiego projektu X.500.

Wyniki przeprowadzonych pomiarów wykazują generalnie dobre zachowanie się bazy. Spodziewaliśmy się uzyskania odpowiedzi na następujące pytania:

1. Jaki jest czas startu bazy (czy pierwsze dowiazanie jest wolniejsze od następnych)?

2. Jaka jest różnica pomiędzy wykonaniem dowiązania anonimowego i autoryzowanego?

3. Jaka jest róznica pomiędzy otrzymaniem wyniku przeszukania instytucji, której dane są dostępne lokalnie od tych, których pozyskanie wymaga nawiązania dodatkowego połączenia?

4. Jakie składowe typowego zapytania mają szczególny wpływ na czas trwania operacji?

5. Czy rozpraszanie bazy na wiele serwerów jest korzystne i w jakich warunkach?

W wyniku testów otrzymaliśmy:

ad. 1 W przypadku polskiego systemu serwerów regionalnych czas pierwszego dowiązania nie jest zauważalnie dłuższy od następnych. Z doświadczeń lokalnych wiemy, że serwer bardzo rzadko eksploatowany ma dłuższy czas rozruchu (konieczność załadowania do pamięci operacyjnej znacznej ilości danych z dysku). Serwery regionalme są jednak eksploatowane stosunkowo intensywnie.

ad. 2 Opóźnienie powodowane poświadczaniem jest sumą operacji dowiązania do DSA przechowującego dane dowiązującego się, przesłania podanego hasła i wykonanie porównania z przechowywanym w bazie, przesłania wyniku operacji. Na wszystko to nakładają się oczywiście opóźnienia sieciowe. W niekorzystnych sytuacjach może to oczywiście prowadzić do zauważalnego pogorszenia działania bazy, typowo jednak poświadczanie dokonywane jest przez DSA bliskie DUA, a następne operacje rozproszone oparte są na wzajemnym zaufaniu DSA (ta ewidentna luka w bezpieczeństwie może być usuwana przy stosowaniu wyższych poziomomów zabezpieczeń). Przy okazji warto wspomnieć, że istnieje możliwość dowiązania z podaniem nazwy, ale bez poświadczania. Takie dowiązanie ma charakter jedynie informacyjny. Serwer dokonuje jedynie bardzo pobieżnej weryfikacji, jeżeli dowiązujący się powinien występować w bazie lokalnej danego serwera, to sprawdzane jest istnienie takiego obiektu i w przypadku stwierdzenia fałszu połączenie jest blokowane, nie sprawdza sie jednak danych nielokalnych, aby nie przedłużać operacji. Ten rodzaj dowiązania nie pozwala oczywiście na dokonywanie żadnych modyfikacji bazy, ani na czytanie zastrzeżonych danych.

ad. 3 Zgodnie z przewidywaniami, dostęp do danych lokalnych jest na ogół szybszy, chociaż w przypadku serwera krakowskiego, w pewnych okresach dane nielokalne były dostępne znacznie szybciej. Musi to być związane z obciążeniem maszyny. Dostęp do danych nielokalnych był jednak w sumie bardzo szybki (średnio przeszukanie lokalne trwa 2–3 sekund, nielokane o 1 do 2 sekund dłużej).

ad. 4 Jak wynika z załączonych wyników testów, najdłuższą operacją jest dokonanie początkowego dowiązania do serwera (zwłaszcza autoryzowanego na znaczną odległość). Operacje przeszukania są stosunko sprawne. Należy przyjąć kryterium, że operacja przeszukania bazy nie powinna przekraczać 3 sekund. Dłuższy czas wskazuje na nadmierne obciążenie serwera, bądź tą samą bazą X.500, bądź innymi procesami. Najszybsza (zgodnie z przewidywaniem) jest operacja czytania. Przy konstrukcji DUA należy zwracać uwagę na to, aby prosić o przesłanie jedynie tych atrybutów, które nasze DUA jest w stanie obrabiać, w szczególności nie ma sensu robić transferu obrazu dla DUA tekstowego. Jednocześnie prawidłowo realizowane buforowanie tego typu danych może bardzo znacznie przyspieszyć funkcjonowanie systemu.

ad. 5 Stosunkowo szybki dostęp do danych nielokalnych wydaje się wskazywać na celowość rozpraszania bazy. Jednocześnie jednak rozpraszanie pojedynczych instytucji, w ramach których dokonywane jest przeszukiwanie, wyraźnie osłabia funkcjonowanie systemu i należy go unikać. Wydaje się zatem, że właściwym modelem jest rozproszone zarządzanie danymi skupionymi na jednym serwerze. Zarządzanie takie polega na powierzeniu pieczy nad wybranymi domenami określonym zarządcom. Mogą oni dokonywać modyfikacji danych po autoryzowanym dowiązaniu do bazy. Zarządzanie takie może być dokonywane jedynie w czasie pracy serwera i nie może dotyczyć modyfikacji danych bezpośrednio na dysku.

Bibliografia

[1] Data Communication Networks: Directory, Recommendations X500-X.521, CCITT, Fascile VIII.8 of Blue Book

[2] Marshall T. Rose (1992) *The Little Black Book: Mail Bounding with OSI Directory Services*

[3] C.J. Robbins, S.E. Kille (1992) *The ISO Development Environment: User's Manual, Volume 5: QUIPU*

[4] M. Górecka, T. Wolniewicz (1994) *Ocena efektywności rozproszonej bazy danych na podstawie X.500 Directory*, Raport NASK

[5] M. Górecka, T. Wolniewicz (1994) *Stosowanie znaków diakrytycznych w systemach baz danych X.500*, Raport NASK

[6] M. Górecka, T. Wolniewicz (1995) *X.500 — standard i usługi katalogowe*, Materiały konferencyjne, POLMAN '95