분산 OCSP에서의 효율적인 로드 밸런싱 기법에 관한 모델링 및 시뮬레이션*

최지혜**, 조대호**

Modeling and Simulation of Efficient Load Balancing Algorithm on Distributed OCSP

Ji-Hye Choi, Tae-Ho Cho

Abstract

The distributed OCSP (Online Certificate Status Protocol), composed of multiple responders, is a model that enhances the utilization of each responder and reduces the response time. In a multi-user distributed environment, load balancing mechanism must be developed for the improvement of the performance of the whole system. Conservative load balancing algorithms often ignore the communication cost of gathering the information of responders. As the number of request is increased, however, fail to consider the communication cost may cause serious problems since the communication time is too large to disregard. We propose an adaptive load balancing algorithm and evaluate the effectiveness by performing modeling and simulation. The principal advantage of new algorithm is in their simplicity: there is no need to maintain and process system state information. We evaluated the quality of load balancing achieved by the new algorithm by comparing the queue size of responders and analyzing the utilization of these responders. The simulation results show how efficiently load balancing is done with the new algorithm.

Key Words: Distributed OCSP, Load Balancing, DEVS Formalism, Simulation

* 본 연구는 정보통신부 및 정보통신연구진흥원의 대학 IT연구센터 육성·지원사업의 연구결과로 수행되었음

** 성균관대학교 컴퓨터공학과

1. 서론

분산 OCSP (Distributed Online Certificate Status Protocol)는 기존 환경에서 인증서 상태 응답 서버가 하나인 경우 집중되는 부하를, 동일한 기능을 하는 여러 서버가 분담하여 효율적인 상태 검증을 가능하게 한다. 따라서 다중 사용자를 가진 분산 환경에서는 작업을 처리하는 시스템의 성능과 작업을 분배할 때의 통신비용을 고려하여 부하를 적절히 할당하는 로드 밸런싱에 관한 연구가 선행되어야 한다[1]. 로드 밸런싱 알고리즘은 시스템의 처리 능력에 관한 정확한 정보를 알아내기 힘들기 때문에 경험적인 방법에 기초를 둔 연구가 많이 진행되어 왔으며, 정적 기법(static load balancing)과 적응형 기법(adaptive load balancing)으로 나눌 수 있다. 정적 로드 밸런싱 기법은 주로 시스템 구성의 초기 단계에서 사용되는 방법으로 시스템의 일반적인 특성에 대한 정보만을 이용하는 반면, 적응형 로드 밸런싱 기법은 최근의 시스템의 상태에 관한 정보를 계속 유지하는 방법으로 정적인 정책보다는 복잡하지만 상당한 성능 향상을 제공한다[1-2].

현재의 분산 OCSP 솔루션에서는 구체적인 로드 밸런싱에 관한 연구가 이루어지지 않은 상태로서 정적 로드 밸런싱 기법을 적용하고 있다. 그러나 신뢰 당사자의 영역을 정적으로 설정하는 것은 사용자들의 움직임 특성상 시간이 지남에 따라 서버에 가해지는 작업 부하를 불균형하게 만든다. 즉, 다수의 사용자들에게 서비스 요청을 받은 서버는 많은 작업을 처리해야 하는 반면, 상대적으로 사용자의 상태 확인 요청이 적은 서버는 적은 작업을 처리한다. 이는 전체 시스템의 효율성을 떨어뜨려 분산 서버의 효과를 감소시키는 결과를 가져올 수 있으므로, 분산 OCSP에서의 로드 밸런싱은 시스템의 처리 능력에 따라 동적으로 부하를 재분배하는 기법이 적용되어야 할 것이다[3].

분산 OCSP 환경과 같은 대규모 네트워크 환경에서의 로드 밸런싱은 정확한 부하의 분배 뿐 아니라 통신비용도 함께 고려하여 이루어져야 한다. 그 이유는, 대규모 거래가 이루어지는 B2B (Business to Business) 사이트나 B2B 인터넷 뱅킹의 경우 완벽한 인증서 상태 검증이 요구되므로, 부하 분배에 소요되는 통신비용이 크면 인증서 상태 검증을 실시간으로 제공할 수 없기 때문이다[4-5]. 이와 같이 네트워크 환경에서의 통신비용의 중요성에도 불구하고, 대부분의 로드 밸런싱 기법은 네트워크 트래픽에 큰 영향을 줄 수 있는 통신 지연 문제를 고려하고 있지 않다[4, 6].

따라서 본 논문에서는 부가적인 통신비용 없이 효율적인 로드 밸런싱을 수행하는 방법을 제안하며, 이를 분산 OCSP 환경에 적용한다. 제안하는 기법은 경험적인 방법에 기반을 둔 간단한 알고리즘이므로 작업 분배에 필요한 처리 시간이 짧아 인증서의 상태 검증에 큰 영향을 미치지 않으므로 분산 OCSP 환경에 적합하며, 시뮬레이션을 통한 실험으로 그 효율성을 검증한다.

본 논문의 구성은 다음과 같다. 2장에서는 본 연구의 배경이 되는 이론인 분산 OCSP와 로드 밸런싱 기법에 대해서 기술하고, 3장에서는 제안하고자 하는 로드 밸런싱 기법의 개념과 DEVS 방법론을 기반으로 설계한 시뮬레이션 모델을 설명한다. 시뮬레이션을 통해 얻은 실험 결과를 4장에서 분석하며, 5장에서는 결론을 맺는다.

2. 관련 연구

2.1 분산 OCSP

공개키 암호 시스템(Public Key Cryptography)은 전자 상거래 등의 정보 보호를 위한 분야에서 대표적으로 사용되는 보안 메커니즘이며, 이를 효율적으로 적용 및 활용하기 위해 등장한 개념이 공개키 기반 구조(PKI;

Public Key Infrastructure)이다[7]. PKI는 기업 보안 구조에 있어서 중요한 부분으로서 보안 관리 측면에 있어서 핵심적인 기능을 제공하며, 안전한 전자 상거래 환경을 위해 인증서를 발급하여 이를 통해 사용자와 메시지의 인증을 수행한다[8-10].

PKI에서의 신뢰 당사자는 인증서를 받은 후, 이 인증서가 유효한지를 검사하는 인증서의 상태 검증 과정을 수행해야 하며[11]. OCSP 서버를 사용하는 실시간 인증서 상태 검증 방법이 대표적이다. OCSP는 온라인상에서 OCSP 서버와 인증서 상태 검증을 요구하는 신뢰 당사자인 OCSP 클라이언트 간에 수행되는 프로토콜로서, 신뢰 당사자의 해당 인증서의 상태를 묻는 질의에 대해 인증서 폐지 여부를 실시간으로 확인시켜준다[5, 12].

이와 같이 실시간의 인증서 상태 확인을 위해 등장한 OCSP는 대부분의 PKI 기반 환경에서 사용되는 방법이나, 인증서의 상태 검증 요청이 많아지면 응답 서버에 가해지는 부하가 커지며 이로 인해 신뢰 당사자의 응답 시간이 지연된다는 단점이 있으며[5], 또한 응답 서버가 집중화되어 있음으로 인해 서비스 거부(DoS; Denial-of-Service) 공격에 쉽게 노출될 수 있다. 이와 같은 문제점을 해결하기 위해 동일한 응답 서버를 여러 개 두어 부하를 분담하는 분산 OCSP 개념이 등장하였다[13-16].

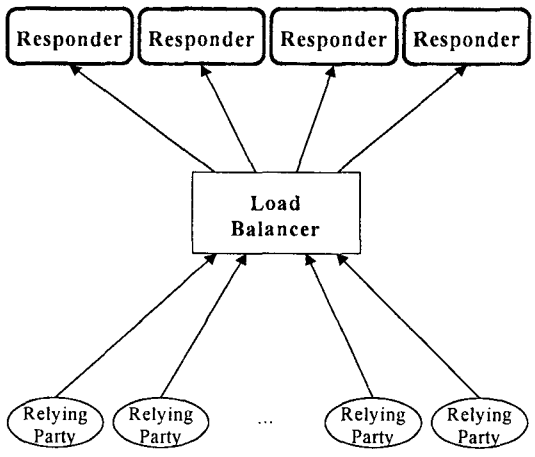
2.2 로드 밸런싱

2.2.1 로드 밸런싱 기법

로드 밸런싱 정책은 정적 기법과 적응형 기법으로 나눌 수 있으며, 적응형 기법은 다시 집중형 정책(centralized policy)과 분산형 정책(distributed policy)으로 분류할 수 있다. 집중형 정책은 로드 밸런싱을 위한 정보를 하나의 서버가 모아서 부하의 분배를 결정하는 방식이고, 분산형 정책은 모든 서버가 서로의 부하에 관한 정보를 감시하여, 로드 밸런싱 알고리즘에 따라 높은 부하를 가진 서버의 작업을 낮은 부하의 서버에게 전송시켜 부하를 분배하는 방식이다. 이 방식은 부하를 나누기 위한 두 서버 사이의 시스템 상태에 관한 많은 정보가 요구된다[17].

로드 밸런싱 기법은 몇 가지 정책에 의해 구성되는데, 전송 정책(transfer policy)은 작업의 전송에 적당한 상태인 서버를 결정하는 정책으로 임계치 정책을 주로 사용하며, 선정 정책(selection policy)은 전송할 작업을 선정하는 정책으로 전송 오버헤드보다 얻어지는 성능 향상이 클 때 작업을 선정한다. 위치 정책(location policy)은 폴링을 이용하여 부하 균형을 위한 서버를 선정하는 정책이고, 정보 정책(information policy)은 시스템의 상태 정보를 언제, 어디에서 모을 것인지 결정하는 정책이다[2, 18].

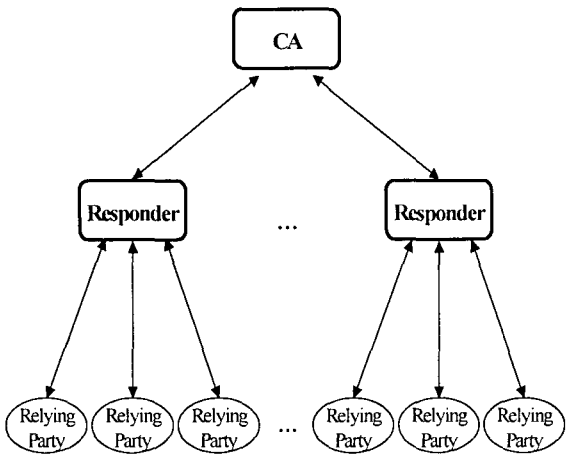
2.2.2 분산 OCSP에서의 로드 밸런싱



<그림 1> 집중형 로드 밸런싱 정책을 사용한 분산 OCSP

분산 OCSP에서의 로드 밸런싱을 위해 처음에 시도된 접근 방법은 그림 1과 같은 집중형 정책이었다. 하나의 서버가 전체 신뢰 당사자의 요청을 모아 적절한 응답 서버에게로 전송시키는 방법으로 전체 응답 능력을 향상시킨

다. 그러나 모든 메시지가 하나의 서버에게 몰리기 때문에 네트워크에 병목현상(bottleneck)이 발생하여 전체 시스템의 효율을 저하시키고, 부하의 분배를 결정하는 서버에 결함이 생길 경우 시스템 전체의 동작이 마비된다[14].



<그림 2> 정적 로드 밸런싱 정책을 사용한 분산 OCSP

<그림 2>와 같이 신뢰 당사자의 물리적인 위치에 따라 응답 서버를 할당하는 연구도 진행되었다. 이 방법은 빠르고 신뢰할만한 응답을 제공하였으나 각 응답 서버에게 가해지는 부하가 불균형적이라는 문제점이 있다[14].

2.3 DEVS 형식론

DEVS (Discrete EVent system Specification) 형식론은 집합 이론에 근거하여, 연속적인 시간상에서 발생하는 이산 사건 시스템을 시뮬레이션하기 위해 정립된 모델링 방법론이다. 이는 모델의 구조와 행동을 시뮬레이션 수행으로부터 추상화시키기 위해 모델을 집합 이론적 방법으로 이용한 것으로, 시스템을 계층적(hierarchical)이고 모듈화(modular)된 형식으로 기술한다[19].

DEVS에서는 기본 모델(basic model)과 결합 모델(coupled model)을 정의한다. 기본 모델은 시간 명세 상태 천이 레벨(timed state transition level)에서 시스템의 동작을 표현하는 모델로서 다음의 항으로 명세할 수 있다.

$M = < X, S, Y, \delta_{int}, \delta_{ext}, \lambda, ta >$
- X : 입력 사건 집합
- S : 상태들의 집합
- Y : 출력 사건 집합
- δ_{int} : $S \rightarrow S$ 내부 상태 전이 함수
- δ_{ext} : $Q \times X \rightarrow S$ 외부 상태 전이 함수
- λ : $S \rightarrow Y$ 출력함수
- ta : $S \rightarrow \mathbb{R}^+$ 시간 진행 함수

결합 모델은 시스템 구성 요소간의 상호작용을 표현하기 위한 모델로서 기본 모델이나 결합 모델로 기술된 구성 요소 모델들을 연결하여 만든 모델이다. 이 모델은 다음과 같은 항으로 명세가 가능하다.

$N = < X, Y, D, \{M_d \mid d \in D\}, EIC, EOC, IC, Select >$
- X : 입력 사건 집합
- Y : 출력 사건 집합
- D : 구성 요소 이름의 집합
- M_d : DEVS 모델의 구성 요소
- EIC : 외부 입력 관계
- EOC : 외부 출력 관계
- IC : 내부 입출력 관계
- $Select$: $2^D - \{\} \rightarrow D$

3. 제안하는 로드 밸런싱 기법

3.1 제안 기법의 개념

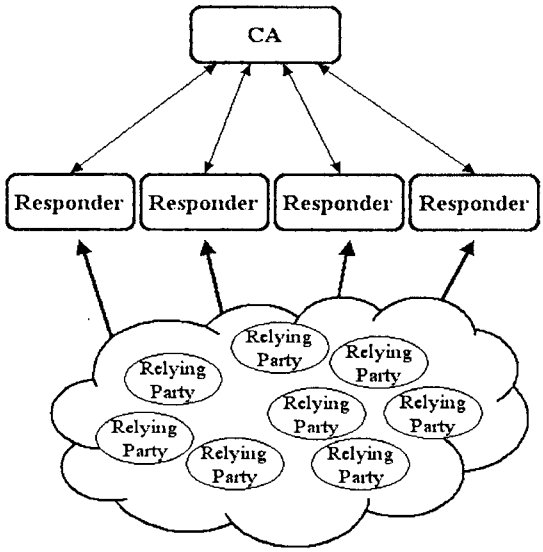
본 논문에서는 별도의 통신비용 없이 분산된 서버 간의 부하를 적절히 분배시키기 위한 알고리즘을 제안한다. 이 알고리즘은 우리가 상품을 주문하여 배달받을 때의 경험에 기초한 것이다.

주문해서 상품을 받기 위해 우리는 선택과

행동의 과정을 거친다 — 선택은 행동을 위한 여러 대안들이 있을 경우 요구되며, 대안에 관한 정보나 선호에 따른 예측을 통해 이루어진다[20]. 만약, 같은 서비스를 제공하는 상점이 여러 개라면 우리는 그 중의 한 상점을 선택하여 주문 요청을 해야 한다. 이 선택은 선험적 결과에 따른 것이며, 만약 상점에 대한 정보가 없는 상태라면 임의로 하나를 선택하게 된다. 상점이 결정되면, 상품을 주문하고 배달받을 때까지 기다리는 과정을 거친다. 추후에 다시 상품을 주문할 경우, 우리는 이전의 경험을 토대로 앞으로 받을 서비스에 대해 예측하여, 다음에 이용할 상점을 선택한다. 이 때의 선택 과정에서 우리는 상점에 대한 별도의 정보 없이, 이전의 서비스 응답에 대한 만족도만을 이용하여 결정한다. 같은 서비스를 제공하는 상점이라고 가정하였으므로, 이때의 만족도를 결정짓는 요소는 응답 시간이 대표적일 것이다. 이 기법은 소비자에게 각 상점의 처리 능력이나 작업량에 대한 부가적인 정보 없이 서비스를 제공받을 수 있게 한다. 알고리즘의 반복은 각 상점의 부하를 적절히 나누고 서비스 처리율을 비슷한 상태로 유지시켜, 서비스를 요청하는 모든 소비자들이 비슷한 응답 시간을 갖도록 할 것이다.

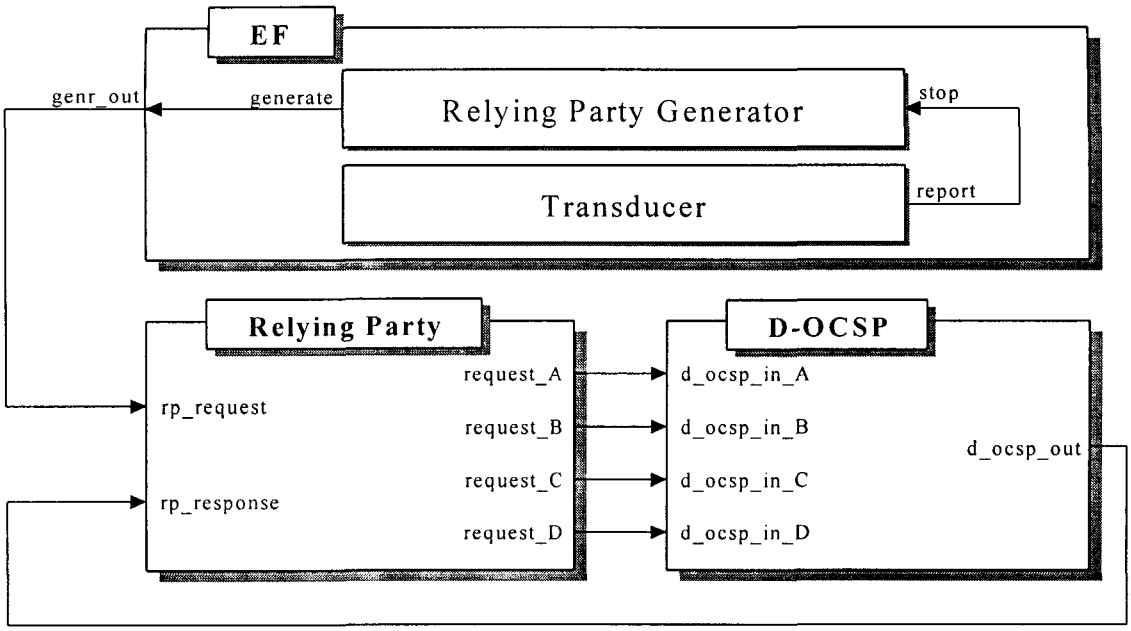
이 알고리즘은 간단한 개념으로서 그 처리 시간이 짧아 실시간의 상태 확인이 요구되는 분산 OCSP 환경에 적합한 기법이며, 부하의 재분배를 위한 통신이 필요하지 않으므로 네트워크 트래픽에 영향을 끼치지 않는다.

<그림 3>은 제안한 기법을 분산 OCSP 환경의 인증서 상태 검증 과정에 적용한 것이다. 인증서의 폐지 유무를 알려주는 응답 서버는 인증기관(CA; Certificate Authority)이 발급한 인증서 취소 목록(CRL; Certificate Revocation List)을 가지고 있으며, 분산 OCSP에서는 중복된 응답 서버가 시스템에 분산되어 있다. 신뢰 당사자는 검증하고자 하는 인증서의 상태 확인을 위해 여러 서버 중 하나를 임의로 택하여 상태 확인 요청 메시지를 전송하며 서버로부터 응답 메시지를 받으면 응답 시간을 기록하고 인증서의 상태에 따른 처리를 한다. 이후에 인증서의 상태 확인이 다시 필요한 경우에는 이전의 서비스 요청에 대한 응답 시간을 고려하여 기존의 응답 서버에게 상태 확인 질의를 할 것인지 다른 서버로 바꿀 것인지를 결정한다.



<그림 3> 제안한 로드 밸런싱 기법을 적용한 분산 OCSP

본 논문에서 제안하는 로드 밸런싱 기법은 신뢰 당사자가 기존 서비스의 만족도에 따라 서버를 선택하는 방법이므로 서버의 부하나 작업 처리 능력에 관한 정보를 요구하지 않는다. 응답 서버는 모두 동일한 인증서 취소 목록을 가지고 상태를 확인하기 때문에 신뢰 당사자가 얻게 되는 정보에는 차이가 없다. 따라서 새로운 요청 메시지를 보낼 서버를 선택하는데 고려되어야 할 요소는 응답시간만으로 충분하여, 작업의 균등한 분배를 위해 필요한 시간이 기존의 기법에 비해 짧다. 이는 신뢰 당사자의 인증서 상태 검증 과정이 실시간으로 수행되는데 큰 장점으로 작용한다.

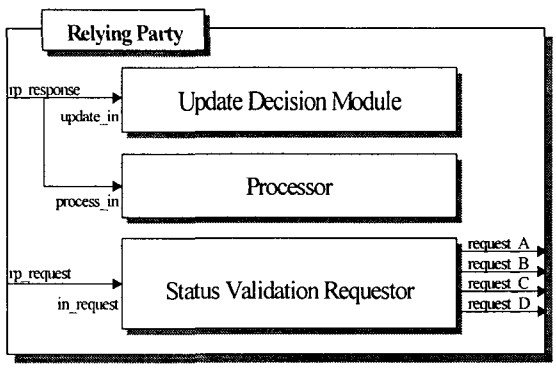


<그림 4> 제안 기법을 적용한 분산 OCSP의 DEVS 시뮬레이션 모델 구조

3.2 시스템의 모델 구성

제안 시스템을 구성하는 모델은 DEVS 형식론에 근거하여 모델링 하였으며 그림 4와 같다. EF 모델은 시뮬레이션 수행의 시작과 종료를 담당하며 신뢰 당사자의 서비스 요청이 일어나는 간격을 지수 분포에 따라 발생시킨다. Relying Party 모델과 D-OCSP 모델은 결합 모델로 구성되며, Relying Party 모델은 인증서 상태에 대한 요청을 D-OCSP 모델에게 전송하여, 응답 메시지를 받은 후 해당 인증서에 대한 처리를 한다. D-OCSP 모델은 Relying Party 모델의 요청에 따라 인증서의 폐지 유무를 확인하여 응답 메시지를 전송하는 역할을 담당한다.

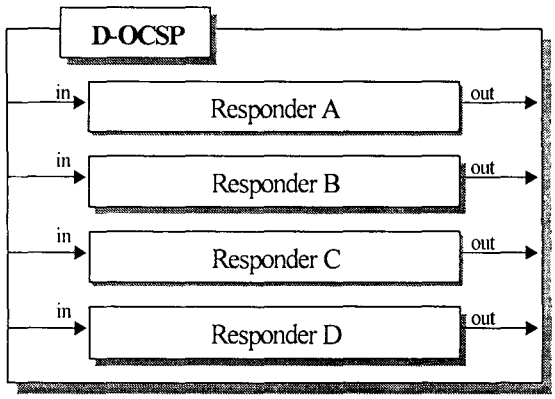
Relying Party 모델은 <그림 5>와 같으며, EF 모델의 서비스 요청 시간을 입력받아 인증서의 상태 확인 요청을 하도록 구성하였다. 이는 실험하고자 하는 신뢰 당사자의 수만큼 모델을 생성하지 않고도 시뮬레이션을 수행할 수 있도록 한다. 응답 서버로부터 응답 메시지를 받으면 인증서 상태의 유효성에 따라 적절한 처리를 하며, 동시에 이전의 응답 시간과 비교하여 다음에 사용할 응답 서버를 결정하는 역할을 한다.



<그림 5> Relying Party 모델의 구성

<그림 6>은 D-OCSP 모델의 세부적인 구성도이며, 본 논문에서는 분산되어 있는 응답 서버를 네 개로 가정하였다. 각각의 응답 서버

는 신뢰 당사자로부터 받은 인증서에 대한 정보를 이용하여 인증서의 폐지 유무를 응답 메시지로 전송한다.



<그림 6> D-OCSP 모델의 구성

3.3 시스템의 동작 과정

3.3.1 초기 동작 과정

- 사용자와 신뢰 당사자는 공인 인증기관(CA)에 등록하여 인증서를 발급 받는다.
- 모든 응답 서버는 동일한 인증서 취소 목록을 가지고 있으며, 새로운 취소 목록이 발급될 때마다 갱신 받는다.
- 신뢰 당사자는 최초로 질의할 응답 서버를 임의로 선택하여 해당 정보를 저장한다.

3.3.2 인증서 상태 검증 및 로드 밸런싱 동작 과정

제안한 시스템은 인증서의 상태 확인이 요구될 때마다 아래의 단계를 반복적으로 수행하여 각 서버의 부하를 균형적으로 분배한다.

[Step 1]

- 신뢰 당사자는 확인하고자 하는 사용자의 인증서에 대한 정보를 얻는다.
- 저장되어 있는 응답 서버에 관한 자신의 정보를 참조하여 해당 서버에게 상태 확인을 요청하며, 이 때의 시간 정보를 기록한다.

[Step 2]

- 신뢰 당사자로부터 요청 메시지를 받은 응답 서버는 해당 인증서의 폐지 여부를 확인한다.
- 상태 확인 결과를 응답 메시지로 작성하여, 신뢰 당사자에게 전송한다.

[Step 3]

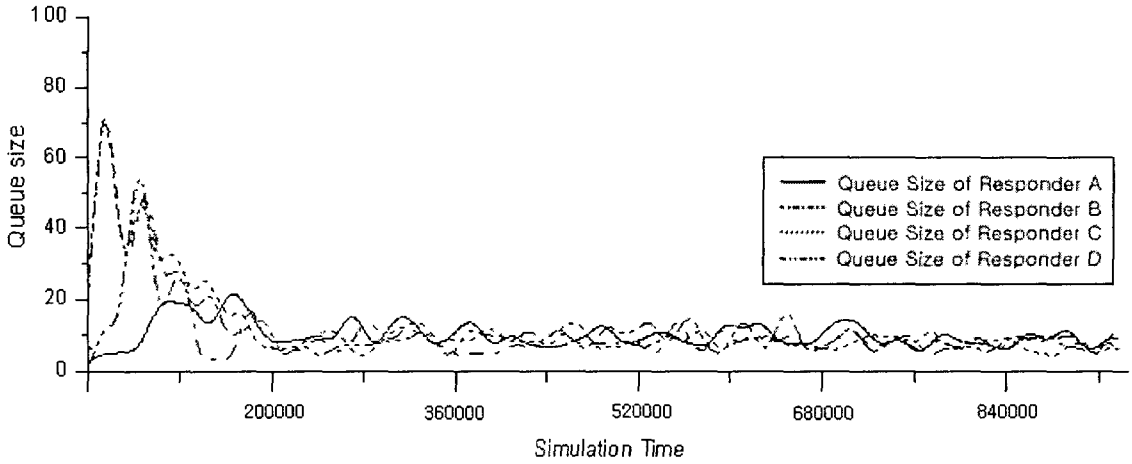
- 신뢰 당사자는 응답 메시지의 내용을 확인하여 인증서 상태에 따른 처리를 한다.
- 요청 메시지를 전송했을 때의 시간 정보를 이용하여, 응답 시간을 계산한다.
- 이전의 응답 시간과 비교하여 다음에 서비스를 제공받을 응답 서버를 결정하고 관련 정보를 저장한다.

4. 실험 및 결과 분석

본 논문에서 제안한 분산 OCSP 환경을 위한 로드 밸런싱 기법의 성능은 각 응답 서버의 사용효율(utilization)과 작업대기 큐의 크기, 신뢰 당사자의 응답 시간으로 평가한다. 실험을 위한 시뮬레이션 환경은 본 연구진이 개발한 DEVS-ObjC를 사용하였다. 시뮬레이션 시간은 1000000, 신뢰 당사자의 수는 1000으로 정하여 실험하였으며, 응답 서버에 대한 신뢰 당사자의 서비스 요청 횟수를 증가시켜 대규모 분산 네트워크 환경을 실험하였다.

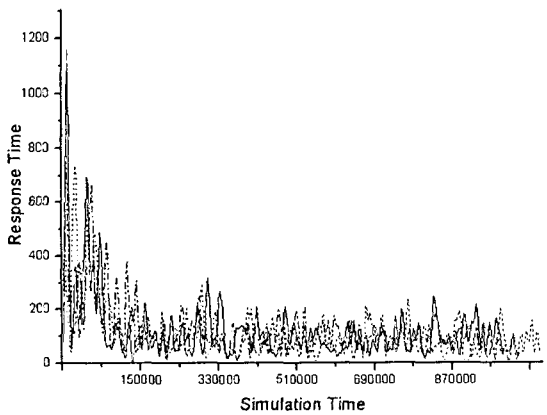
<표 1> 각 응답 서버의 사용효율

Responder	Utilization
Responder A	0.916926
Responder B	0.921042
Responder C	0.934360
Responder D	0.916326



<그림 7> 시뮬레이션 시간에 따른 각 응답 서버에서의 작업대기 큐의 크기

<그림 7>은 질의할 응답 서버를 신뢰 당사자의 임의대로 선택하여 초기화시킨 경우의 시뮬레이션 결과로서 각 서버가 선택될 확률은 모두 동일하다. 각 응답 서버의 작업대기 큐의 크기를 측정하여 그 수치를 평균화하였으며, 시간이 지남에 따라 큐의 크기가 비슷해짐을 확인할 수 있고, <표 1>에서 보여주는 응답 서버의 사용 효율에 대한 수치 또한, 값의 차이가 크지 않은 것으로 보아 각 서버에 가해진 작업의 부하가 균형적임을 알 수 있다. 전체 시스템에서 처리해야 할 작업이 응답 서버로 골고루 할당된다는 것은 분산된 서버가 효율적으로 일을 처리하는 것을 의미하며, <그림 8>의 신뢰 당사자의 응답 시간을 나타낸 그래프로도 확인할 수 있다. 그림은 시뮬레이션이 진행됨에 따라 응답 시간의 편차가 줄고 전체 신뢰 당사자의 응답 시간이 일정 수준을 유지하게 됨을 보이고 있다.

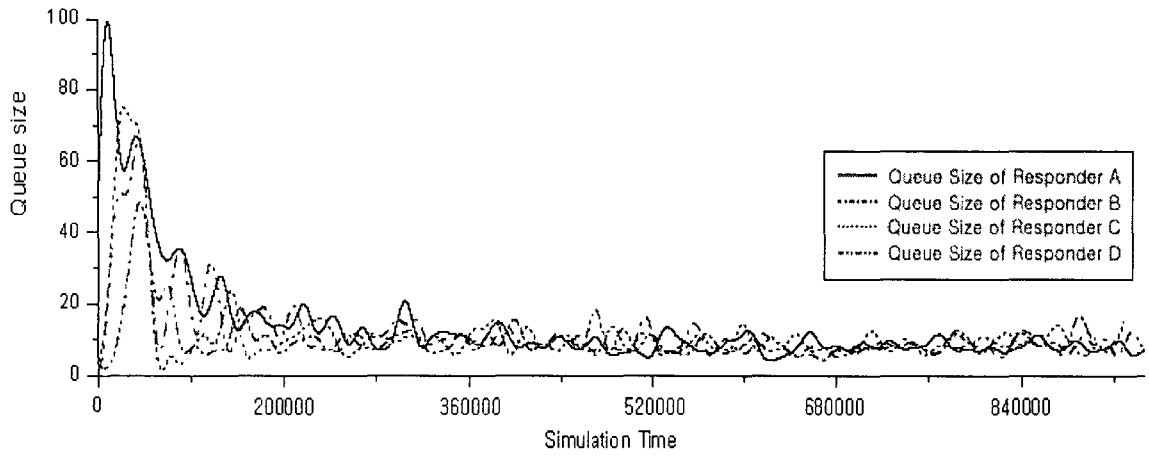


<그림 8> 신뢰 당사자의 응답 시간

<그림 9>는 초기 응답 서버의 부하가 불균형한 경우 — 각 서버에 가해지는 부하가 전체 부하의 50%, 30%, 15%, 5%가 되도록 설정 — 에 시뮬레이션을 수행한 결과이다. 각 응답 서버는 크게 다른 작업량을 처리해야 하므로, 시뮬레이션이 시작될 때에는 각 서버의 큐에 쌓인 작업 수의 차이가 크다. 그러나 로드 밸런싱이 진행됨에 따라 각 응답 서버에 가해지는 부하의 크기는 균형을 이루며, 이는 실험을 통해 얻어낸 결과로 확인할 수 있다.

5. 결론

본 논문에서는 분산 OCSP 환경에서 부가적인 통신비용 없이 효율적으로 로드 밸런싱을 수행하는 기법을 제안하였다. 분산 OCSP는 대규모 거래가 이루어지는 네트워크 환경의



<그림 9> 초기 부하가 불균형적인 경우의 시뮬레이션 결과

특성을 지니고 있으므로 기존의 로드 밸런싱 기법을 적용하기에는 적합하지 않다. 로드 밸런싱에 관한 기존 연구는 정확한 부하를 분배하기 위한 것이 대부분으로, 알고리즘이 복잡하여 처리 시간이 길고 시스템의 작업 처리 능력이나 작업에 대한 정보를 얻기 위한 통신 비용이 크게 요구되기 때문이다. 분산 OCSP 환경에서의 로드 밸런싱에 높은 통신비용이 요구된다는 것은, 인증서의 상태 확인에 대한 요청이 많아질 경우 신뢰당사자의 응답 지연율이 커지고, 결국 인증서 상태 검증을 실시간으로 제공할 수 없다는 것을 의미한다.

따라서 본 논문에서는 로드 밸런싱 과정에 별도의 통신비용이 발생하지 않도록 신뢰 당사자가 직접 부하를 분배하도록 하였다. 신뢰당사자는 서버의 상태나 처리 능력에 관한 정보를 구하지 않고, 자신의 정보를 이용하여 응답 서버를 선택하는 기법을 사용한다.

본 논문에서 제안한 기법은 시뮬레이션을 통해 실험하였으며, 실험 결과로 얻어낸 응답 서버의 사용효율과 작업대기 큐의 수는 각 응답 서버가 갖게 되는 작업 부하의 양이 균형적임을 보인다. 이는 제안한 로드 밸런싱 기법이 네트워크에 부가적인 트래픽을 발생시키지 않고도 부하의 분배를 효율적으로 수행함을 증명한다.

참고문헌

[1] J. N. C. Arabe, A. Beguelin, B. Lowekamp, E. Seligman, M. Starkey, and P. Stephan, "Dome: Parallel programming in a distributed computing environment," In *Proceedings of the 10th International Parallel Processing Symposium (IPPS 96)*, IEEE Computer Soc. Press, Silver Spring, MD, April 1996.

[2] D. L. Eager, E. D. Lazowska, and J. Zahorjan, "A Comparison of Receiver-Initiated and Sender-Initiated Adaptive Load Sharingt," *Performance Evaluation*, vol. 6, no. 1, pp. 53-68, March 1986.

[3] C. Reynolds, "Flocks, Herds, and Schools: A Distributed Behavioral Model," *Computer Graphics*, vol. 21, no. 4, July 1987.

[4] M. Schaar, K. Efe, L. Delcambre, and L. N. Bhuyan, "Load balancing with network

cooperation," In *Proceedings 11th International Conference Distributed Computer System*, pp. 328-325, May 1991.

[5] M. Myers, R. Ankney, and C. Adams, *Online Certificate Status Protocol, version 2*, IETF Draft, 2001.

[6] D. L. Eager, E. D. Lazowska, and J. Zahorjan, "Adaptive load sharing in homogeneous distributed systems," *IEEE Transactions on Software Engineering*, vol. 12, no. 5, pp. 662-675, May 1986.

[7] R. Mahadevan, "Analytical Modeling of Electrostatic Structures," In *Proceedings of IEEE Workshop on Micro Electro Mechanical Systems,* pp. 120-127, February 1990.

[8] R. Housley and T. Polk, *Planning for PKI*, Wiley, 2001.

[9] B. D. Reimers, "PKI's Are Still Tough To Deploy," *Internet Week*, April 2001.

[10] GAO, Information Security: "Advances and Remaining Challenges to Adoption of Public Key Infrastructure Techno- logy," *Item no. 0546-D; SuDocs no. GA 1.13: GAO-01-277*, February 2001.

[11] R. Housley, W. Ford, W. Polk, and D. Solo, *Internet X.509 Public Key Infrastructure: Certificate and CRL Profile*, IETF RFC3280, 2002.

[12] M. Naor and K. Nissim, "Certificate Revocation and Certificate Update," *IEEE Journal on Selected Areas in Communications,* vol. 18, issue 4, pp. 561-570, April 2000.

[13] T. Moses, *OCSP vs CRLs over HTTP,* Available at http://www.imc.org/ietf-pkix/old-archive-97/msg01377.html, December 1997.

[14] *Whitepaper: Distributed OCSP: Security, Scalability, and Availability for Certificate Validation*, Available at www. corestreet.com/whitepapers/w02_07v2_distributed_ocsp.pdf

[15] S. Koga and K. Sakurai, "A Distributed Online Certificate Status Protocol with a Single Public Key," *International Workshop on Practice and Theory in Public Key Crypto- graphy (PKC 2004)*, LNCS 2947, pp. 389-401, March 2004.

[16] S. Micali, "NOVOMODO: Scalable Certificate Validation and Simplified PKI Management," *1st Annual PKI Research Workshop*, pp. 15-25, 2002.

[17] K. Antonis, J. Garofalakis, I. Mourtos, and P. Spirakis, "A hierarchical adaptive distributed algorithm for load balancing," *Journal of Parallel and Distributed Computing*, vol. 64, no. 1, pp. 151-162, January 2004.

[18] N. Shivaratri, P. Krueger, and M. Singhal, "Load distributing for locally distributed systems," *IEEE Transaction Computer*, vol. 25, pp. 33-44, December 1992.

[19] B. P. Zeigler, H. Praehofer, and T. G. Kim, *Theory of Modeling and Simulation*, 2nd Ed., Academic Press, 2000.

[20] J. D. Schall, "Neural Basis of Deciding, Choosing and Acting," *Nature Reviews Neuroscience 2*, pp. 33-42, 2001.

주 작 성 자 : 최 지 혜
논문투고일 : 2004. 10. 11
논문심사일 : 2004. 10. 19(1차), 2004. 10. 19(2차), 2004. 10. 20(3차)
심사판정일 : 2004. 10. 20

● 저자소개 ●

최지혜

2003 성균관대학교 정보통신공학부 학사

2003~현재 성균관대학교 컴퓨터공학과 석사과정

관심분야 : 공개키 기반 구조, 분산 시스템, 네트워크 보안 시뮬레이션

조대호

1983 성균관대학교 전자공학과학사

1987 알라바마대 전자공학과 석사

1983 아리조나대 전자 및 컴퓨터공학과 박사

1993~1995 경남대학교 전자계산학과 전임강사

1995~1999 성균관대학교 전기전자 및 컴퓨터공학부 조교수

1999~2004 성균관대학교 전기전자 및 컴퓨터공학부 부교수

2004~현재 성균관대학교 정보통신공학부 교수

관심분야 : 모델링 및 시뮬레이션, 네트워크 보안, 지능제어, ERP