논문 01-26-2A-1 한국통신학회논문지 '01-2 Vol.26 No.2A

리눅스 서버 시스템에서 커널 무변경 역할기반접근제어 보안 시스템 설계

정회원 오 석 균*, 김 성 열**

Design of Role-Based Access Control Secure System on LINUX Server Systems without Kernel Changes

Sugkyun, Oh*, Seong Ryeol, Kim** *Regular Members*

요 약

본 논문은 리눅스 서버 환경에서 다양한 업무를 운영하려고 할 때에 발생되는 보안상의 문제를 해결하기 위해서 역할기반접근제어(Role-Based Access Control : RBAC) 기법을 이용하여, 리눅스 환경에서 운영 가능한 RBAC_Linux 보안 시스템을 설계하였다. RBAC_Linux는 RBAC 기법을 리눅스 환경에 적용하여 설계되었으며, 적용한 모형은 Sandhu 등이 제안한 RBAC96 모형을 이용한다. 따라서 리눅스 서버 시스템에 설계 제안된 RBAC_Linux 보안 시스템을 이용하면 서버의 소스코드를 수정하지 않고 구현 가능하고, 이식성이 높으며, 보안 관리가 단순하고 용이하다는 장점을 갖는다.

ABSTRACT

This paper applies Role-Based Access Control(RBAC) policy for solving on the security problems when it will be operated several business on the Linux server environments and designed the RBAC_Linux security systems that it is possible to manage security systems on the Linux environments. The RBAC_Linux is security system which is designed for applicable on the Linux environment. The applying RBAC model is based on RBAC96 model due to Sandhu et al. Therefor, the using designed RBAC_Linux security system on the Linux server system have the advantage of the following: it can be implemented server system without modifying its source code, high migration, easy and simple of secure managing.

Ⅰ. 서 론

최근의 정보 시스템들은 분산 시스템으로 개발되어 인터넷을 기본적으로 이용하는 업무들이 날로 증가되고 있다. 그러나 이러한 시스템들은 업무들의 운영과 수행, 인터넷상에서 사용자들의 업무 접근과 통제에 대한 보안상의 문제가 커다란 문제점으로 대두되고 있다. 따라서, 본 논문에서는 리눅스 서버 시스템의 보안을 위해 RBAC(Role-Based Access Control) 기술을 이용하였다.

RBAC의 기본 개념은 허가가 역할과 관련되고, 사용자는 적절한 역할에 할당된다는 것이다. 이 개념은 허가 관리를 매우 단순화 시켰으며, 역할이 조직 내에서 다양한 작업의 기능에 따라 생성되고 사용자의 책임과 자격을 근거로 사용자에 할당된다[1]. 또한 역할은 특정 업무 수행 능력을 나타낼 수도 있고, 권한과 책임을 구체적으로 명기할 수 있어, 여러 사용자에게 특정 임무를 순환 시켜가며 할당할 수 있다는 특징을 갖고 있다.

일반적으로 RBAC는 특별히 상업적 응용에 적용할 수 있다는 기대[2]로 임의 접근 제어(discretionary access control : DAC)와 위임 접근 제어

* 충청대학 컴퓨터학부, osk58@chch.ac.kr ** 청주대학교 컴퓨터정보공학과srkim@comnet.chongju.ac.kr
논문번호 : 00300-0729, 접수일자 : 2000년 7월 29일

(mandatory access control : MAC) 정책을 대신할 수 있다. RBAC의 기본적인 배경[3]은 기업이나 기관의 조직 구조에 자연스럽게 접근하여 하부조직의 제어와 전형적인 접근제어 리스트 등과 같은 보안 정책을 강조하여 명세화를 요구할 수 있다는 데 있다.

본 논문에서는 리눅스 서버 시스템 환경에서 인터넷으로 업무를 운영할 때에 발생되는 보안상의 문제를 해결하기 위해서 RBAC 기술을 이용한 RBAC_Linux 보안 시스템을 설계제안하고, 분산 정보 시스템에 적용할 수 있도록 RBAC_Linux의 특징과 구현 환경에 대하여 기술한다. 설계시 적용한 RBAC 모형으로는 Sandhu 등이 제안한 RBAC96 모형[4]을 기본 모형으로 이용하였다.

Ⅱ. RBAC_Linux 설계

Sandhu에 의해 정의된 RBAC 모형들을 RBAC96이라 부르며, 이 모형들 중에서 가장 일반적인 것은 그림 1과 같다.

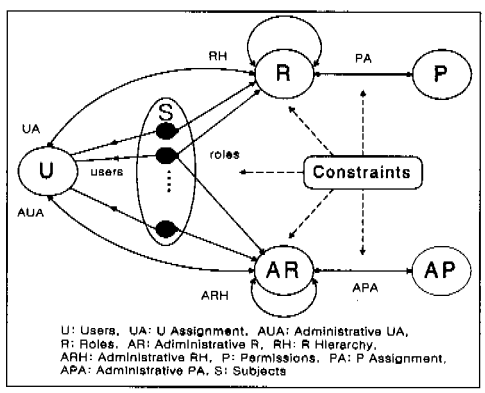

그림 1. RBAC96 모형

그림에서 상위 부분은 데이터와 자원의 액세스를 통제하는 정규 역할과 정규 허가를 담당하고, 상위 부분을 관리하는 하위 부분은 관리 역할과 관리 허가를 담당한다. 역할은 순서쌍으로 표현하며, x>y이면 역할 x는 역할 y의 허가를 상속하고 x의 멤버는 y의 멤버를 내포한다. 이 때 x는 y의 상위(senior)이라고 하고, y는 x의 하위(junior)라고 말한다.

RBAC96 모형을 이용하여 리눅스 운영체제 시스템에 적합하도록 RBAC 기술을 확장 추가하여 RBAC_Linux 보안 시스템을 설계 제안한다.

1. RBAC_Linux 모형 정의

RBAC_Linux 보안 시스템 모형 표현은 RBAC96 모형 표현을 이용하여 정의하며, 사용자 세션에 있는 활동역할집합(ARS:active- roles sets) 정의, 사용자할당 역할집합, 역할상속관계, 정적직무분리(ssd), 상호배타적직무분리(msd), 개방적직무분리(lsd), 역할 이용권한이 부여된 사용자 최대수(cardinality)를 추가적으로 설계 정의한다.

<정의 1> RBAC_Linux 모형 표현

- ○ U ; 사용자 집합
  R ; 교차하지 않는 정규역할집합
  AR ; 교차하지 않는 관리역할집합
  P ; 교차하지 않는 정규허가집합
  AP ; 교차하지 않는 관리허가집합
  S ; 세션집합
- ○ $UA \subseteq U \times R$; 역할-사용자 할당관계
  $AUA \subseteq U \times AR$; 관리역할-사용자 다대다 할당관계
- ○ $PA \subseteq P \times R$; 역할-허가 할당관계
  $APA \subseteq AP \times AR$; 관리역할-허가 다대다 할당관계
- ○ $RH \subseteq R \times R$; 순서쌍 역할 계층
  $ARH \subseteq AR \times AR$; 순서쌍 관리역할 계층
- ○ user : $S \rightarrow U$; 단일 사용자 user(s_i)에 각 세션 s_i를 매핑
- ○ roles : $S \rightarrow 2^{R \cup AR}$;
  roles(s_i)와 admin_roles(s_i)$\subseteq \{r \mid (\exists r' \geq r)[(user(s_i), r') \in UA \cup AUA]\}$에 각 세션 s_i를 매핑, 세션 s_i는 $\cup_{r \in roles(si)}\{p \mid (\exists r'' \geq r)[(p,r'') \in PA \cup APA]\}$인 허가를 가짐
- ○ $assigned-roles$: $2^R \rightarrow U$;
  $assigned-roles$(u)는 사용자 u에 할당된 역할집합
- ○ $active-roles$: $2^R \rightarrow U$;
  $active-roles$(u)는 사용자 u의 세션에 있는 활동역할집합
- ○ $inherits \subseteq R \times R$; $\rightarrow^a$ 는 비대칭 상속, $\rightarrow^t$ 는 추이적 상속, $\rightarrow^r$ 은 반사적 상속.
- ○ $SSD \subseteq \frac{R \times (R-1)}{2}$; 역할간의 ssd 관계
- ○ $MSD \subseteq \frac{R \times (R-1)}{2}$; 역할간의 msd 관계
- ○ $LSD \subseteq \frac{R \times (R-1)}{2}$; 역할간의 lsd 관계
- ○ $cardinality$: $R \rightarrow N \cup \{\infty\}$:
  $cardinality$(r)은 역할 r의 이용 권한이 부여된 사용자의 최대 수
- ○ 제약조건 : 여러 구성요소 값을 받아들일지 여

부를 결정하며, 각 요소에 제약을 가 하고, 받아들일 수 있는 값만 허용.

2. 직무 분리 설계

시스템 관리자가 RBAC 메커니즘을 수행하기 위해서는 직무가 분리되어야 한다. 여러 작업들이 가지고 있는 관련된 능력들이 서로 협동하여 작업 처리를 하다보면 예기치 못했던 사건이 발생할 수 있다. 이러한 사건을 막기 위해서는 직무를 분리해야만 한다[5]. 그 이유는 트랜잭션의 부분 집합에서 한 사람이 그 집합 내의 모든 트랜잭션을 수행하는 것을 허용하지 않아야 한다는 것이다. 시스템 관리자는 기업의 사업 관리 방법을 자연스럽게 추상적인 개념 레벨에서 접근을 제어할 수 있다. 이는 역할, 역할계층, 관계, 제약조건을 정의하고 수립함으로서 사용자 활동을 통제할 수 있다. 따라서 RBAC96 모형의 직무분리 표현에 각종 직무분리 추가 및 삭제 기능을 추가하여 직무분리의 유연성을 확보할 수 있도록 RBAC_Linux 직무분리를 다음과 같이 정적, 배타적, 개방적 직무분리에 대한 각각의 추가 및 삭제에 대하여 설계한다.

<정의 2> 정적 직무분리의 추가 및 삭제

○ add_SSD

$\forall u \in U, \forall r, r_i, r_j \in R, r_i \neq r_j, (r_i, r_j) \notin SSD$ and MSD, $\{r_i, r_j\} \not\subseteq$ assigned_roles(u), $r \rightarrow_{\pi} r_i \Rightarrow (r, r_j) \in SSD$ or $r \rightarrow_{\pi} r_j \Rightarrow (r, r_i) \in SSD$

==> SSD ← SSD ∪ {$(r_i, r_j), (r_j, r_i)$}

○ del_SSD

$\forall r, r_i, r_j \in R, (r_i, r_j) \in SSD, r_i \rightarrow_{\pi} \Rightarrow (r, r_j) \notin SSD$ or $r_j \rightarrow_{\pi} \Rightarrow (r, r_i) \notin SSD$

==> SSD ← SSD - {$(r_i, r_j), (r_j, r_i)$}

<정의 3> 배타적 직무분리 추가 및 삭제

○ add_MSD

$\forall u \in U, \forall r, r_i, r_j \in R, r_i \neq r_j, (r_i, r_j) \notin$ SSD and MSD, $\{r_i, r_j\} \not\subseteq$ active_roles(u), $r \rightarrow_{\pi} r_i \Rightarrow (r, r_i) \in MSD$ or $r \rightarrow_{\pi} r_j \Rightarrow (r, r_i) \in MSD$

==> MSD ← MSD ∪ {$(r_i, r_j), (r_j, r_i)$}

○ del_MSD

$\forall r, r_i, r_j \in R, (r_i, r_j) \in MSD, r_i \rightarrow_{\pi} \Rightarrow (r, r_j) \notin MSD$ or $r_j \rightarrow_{\pi} \Rightarrow (r, r_i) \notin MSD$

==> MSD ← MSD - {$(r_i, r_j), (r_j, r_i)$}

<정의 4> 개방적 직무분리 추가 및 삭제

○ add_LSD

$\forall u \in U, \forall r, r_i, r_j \in R, r_i \neq r_j, (r_i, r_j) \notin SSD$, MSD and LSD, $\{r_i, r_j\} \not\subseteq$ active_roles(u), $r \rightarrow_{\pi_i} \Rightarrow (r, r_j) \in LSD$ or $r \rightarrow_{\pi_j} \Rightarrow (r, r_i) \in LSD$

==> LSD ← LSD ∪ {$(r_i, r_j), (r_j, r_i)$}

○ del_LSD

$\forall r, r_i, r_j \in R, (r_i, r_j) \in LSD, r_i \rightarrow_{\pi} \Rightarrow (r, r_j) \notin LSD$ or $r_j \rightarrow_{\pi} \Rightarrow (r, r_i) \notin LSD$

==> LSD ← LSD - {$(r_i, r_j), (r_j, r_i)$}

Ⅲ. RBAC 데이터베이스 일관성 특성

RBAC에서 사용자와 역할에 관한 관계정보를 가지고 있는 RBAC 데이터베이스가 한 상태에서 일관성을 유지하기 위해서는 <정의 5>에서와 같이 정의한 특성을 만족하여야 한다.

<정의 5> RBAC 데이터베이스 일관성 특성

○ 특성-1

$\forall r \in R$, |authorized_users(r)| ≤ cardinality(r).

○ 특성-2

$\forall r \in R$, $\neg(r \rightarrow^{+} r)$.

○ 특성-3

$\forall r_1, r_2 \in R, \forall u \in USERS, r_1, r_2 \notin LSD$, $r_1, r_2 \in$ assigned_roles(u) $\Rightarrow \neg(r_1 \rightarrow^{s} r_2)$ and $(r_1, r_2) \notin ssd$.

○ 특성-4

$\forall r \in R \Rightarrow (r, r) \notin SSD$ and MSD.

○ 특성-5

$\forall r_1, r_2 \in R, (r_1, r_2) \in SSD \Rightarrow (r_2, r_1) \in SSD$ or $(r_1, r_2) \in MSD \Rightarrow (r_2, r_1) \in MSD$.

○ 특성-6

$\forall r_1, r_2 \in R, \forall u \in U, r_1, r_2 \in$ active_roles(u) $\Rightarrow (r_1, r_2) \notin$ msd.

○ 특성-7

$\forall r, r_1, r_2 \in R, r \rightarrow_{\pi_1} (r_1, r_2) \in SSD \Rightarrow (r, r_2) \in SSD$ or $r \rightarrow_{\pi_1} (r_1, r_2) \in MSD \Rightarrow (r, r_2) \in MSD$.

○ 특성-8

$\forall u \in U$, active_roles(u) $\subseteq$ authorized_roles(u).

○ 특성-9

$\forall r_1, r_2 \in R, (r_1, r_2) \in MSD \Rightarrow (r_1, r_2) \notin SSD$.

Ⅳ. RBAC_Linux 모형구성과 구현방법

리눅스 서버에 역할기반 접근제어를 구현하려면 설계 제안하고 있는 RBAC_Linux가 필요하다. 이는 RBAC_Linux를 수행하는 서버에서 사용하는 어떠한 브라우저도 추가요구 없이 설치할 수 있어야 본래의 RBAC 구현 목적[6]에 부합된다.

1. RBAC_Linux 모형 구성 요소

RBAC_Linux를 구현하려면 기본적인 RBAC96 모형의 구성요소를 사용할 수 있으며, API 라이브러리에 Visual BASIC, Visual C, C++ 라이브러리 등을 추가하고 CGI 뿐만 아니라 PHP, Perl, Tcl 등을 사용할 수 있도록 하였으며, RBAC_Linux 모형의 구성요소는 <정의 6>와 같이 정의하여 사용한다.

<정의 6> RBAC_Linux 구성요소

○ 데이터베이스 : 사용자와 역할, 역할계층, 제약, 활동중인 역할간의 관계와 역할과 동작간의 관계를 지정하는 파일

○ 데이터베이스 서버 : 사용자와 역할, 역할계층, 제약간의 관계를 정의하는 파일을 관리하는 호스트 이 파일들은 관리도구에 의해 생성되고 유지되며 이들 파일이 변경되면 데이터베이스 서버는 인터넷 서버에 수정된 사항을 통보한다.

○ API 라이브러리 : 인터넷 서버와 RBAC_Linux에 접근하기 위해서 사용하며, Visual Basic, Visual C, C++언어와 Perl Library를 이용하여 API Library를 만드는데 이는 RBAC_Linux_API 구현에 이용한다.

○ CGI/PHP : 인터넷 서버를 소스코드를 변경하지 않고 구현하기 위해 CGI와 PHP 기법을 제공한다.

○ 세션 관리자 : 사용자의 ARS를 생성하고 제거한다.

○ 관리 도구 : 서버 관리자에 사용자, 역할, 사용자와 역할 연관짓기, 사용자/역할 관계 제약조건 지정, RBAC 데이터베이스 유지 허용한다.

2. RBAC_Linux 구현 방법

리눅스 서버에서 RBAC_Linux을 구현하려면 위에서 정의한 구성요소들이 필요하며, 다음의 두 가지 방법으로 구현할 수 있다.

첫 번째 방법으로는 가장 일반적인 방법으로 CGI 기법이나 PHP 기법을 이용하는 것이다. 이 기법은 현재 사용하고 있는 리눅스 서버의 소스 코드를 수정하지 않고 사용할 수 있다. 이 기법을 사용하여 제작되는 RBAC_Linux 프로시저를 RBAC_Linux_CGI 또는 RBAC_Linux_PHP라고 한다. 이 프로시저를 사용하면 RBAC url이 인터넷 서버를 통해 전달되고 처리된다. 이는 RBAC_Linux 환경 파일이 사용자의 역할을 기반으로 하는 접근제어를 수행하는 것으로 url을 파일이름으로 매핑하는 것이다. RBAC_Linux_CGI나 RBAC_Linux_PHP의 설치는 인터넷 서버를 설치[7]하는 것과 유사하며, 구현시 별도의 추가 비용이 발생하지는 않는다는 장점이 있다.

두 번째 방법으로는 API 기법을 사용하는 것으로 이 방법은 RBAC_Linux를 접근하기 위해서 API를 호출해야하기 때문에 리눅스 운영체제를 탑재한 인터넷 서버의 원시코드 일부를 수정해야 한다. 이 기법에서 사용되는 url은 사용자가 입력한 url이 인터넷 환경 파일에서 파일이름으로 매핑되는 것으로 RBAC_Linux에 의해 제어되는 url이다.

3. RBAC_Linux 운영

RBAC_Linux를 설치한 인터넷 서버와 사용자간에 이루어지는 상호활동과 기존 방식에 의한 인터넷 서버와 사용자간에 이루어지는 상호활동은 표면상으로는 같아 보이지만 내부적으로 많은 부분이 다르다. 두 방식이 같게 동작하는 경우에는 RBAC_Linux를 설치한 인터넷 서버의 접근이 RBAC_Linux의 통제를 받기 전 단계까지이다. 그러나 RBAC_Linux의 통제가 이루어지기 시작하면 접근하기 전에 사용자가 RBAC 세션을 설정해야만 url이 요구한 접근을 허가한다. 사용자가 활동중인 ARS 중에서 선택함으로써 RBAC 세션이 설정되고, 그에 따라 해당 ARS가 할당된다. ARS는 사용자가 RBAC에서 통제한 url에서 수행할 수 있는 동작의 허가를 결정하고, 사용자가 새로운 ARS를 설정할 때까지 유지된다. 사용자가 msd 관계인 경우에 RBAC_Linux 운영의 동작과정 알고리즘은 다음과 같다.

<알고리즘 1> RBAC_Linux 동작과정

```
while ( count < init )
  { gets username and password;
    if match username and password
     { if set session
          search role_table(username)
          case  found :
               display active_role_sets;
```

```
                select ARS;
                assign selected_ARS;
        case not_found :
                puts "You have not the
                assigned role.";
                return;
    }
    otherwise  {
        increase  count;
        puts  "try again, mismatch
              username and password";
    }
}
return;
```

4. RBAC_Linux 인증

RBAC_Linux는 패스워드, SSL(Secure Socket Layer), Secure HTTP, Private Communication Technology Protocol들이 포함하고 있는 인터넷 인증과 기밀 서비스를 결합하여 사용할 수 있는 접근제어 메커니즘을 이용할 수 있다. 사용자 id 정보는 인터넷 서버에서 RBAC_Linux에 전달된다. 이는 사용자 id 정보를 인증하기 위해서 인터넷 서버가 갖추어야할 의무이며, 인터넷 서버 관리자에 의해 이루어지는 것처럼 신뢰할 수 있게 데이터를 전송한다.

또한 인증된 사용자만이 RBAC 세션을 설정할 수 있어서 사용자의 인증이 삭제되면 RBAC 통제를 받는 url에서 접근이 거부된다. 따라서 사용자 인증과 RBAC 세션의 설정이 완벽하게 동작이 분리되므로 RBAC_Linux는 제공되는 인증 메커니즘 내에서 작업이 이루어진다.

V. 결 론

분산된 정보시스템에서 인터넷을 통하여 다양한 정보를 공유함으로써 발생되는 문제들 중에서 보안상의 문제가 가장 크게 대두되고 있다. 따라서 정보 접근을 사용자로부터 적절하게 규제할 수 있는 보안체계가 필요하게 되었다. 이를 용이하게 해결할 수 있는 접근제어 메커니즘으로 RBAC 기술이 이용되고 있다. 이 RBAC 기법을 리눅스 환경의 인터넷상에서 이용할 수 있도록 RBAC_Linux 보안 시스템을 설계 제안하였다.

설계 제안된 RBAC_Linux는 보안관리 능력을 가지고 있어 데이터 보안관리와 사용자 접근에 관한 보안관리를 폭넓게 수행할 수 있다. RBAC_Linux에서 사용자는 역할로 멤버십이 허가되고, 쉽게 취소될 수 있으며, 새로운 멤버십을 설정할 수 있다. RBAC_Linux에서는 사용자에게 역할과 관련된 동작을 수행하기 위한 허가를 허락하지 않는다. 역할은 조직의 기능이 변경되어도 이에 따라 신축적으로 대응할 수 있다. 즉, 새로운 작업과 관련된 역할을 설정하고 마무리된 작업과 관련된 역할은 삭제할 수 있어 권한관리를 단순화시킬 수 있는 특징을 갖는다. 또한 RBAC_Linux는 리눅스 서버를 사용하는 인터넷 환경에서 응용할 수 있으며, 서버의 소스코드를 수정하지 않고 구현 할 수 있으며, 이식성이 높고, 단순화된 보안관리 기능을 갖고 있다는 특징을 갖고 있다.

참 고 문 헌

[1] R. S. Sandhu, V. Bhamidipati, "The URA97 Model for Role-Based User-Role Assignment, Proc. of IFIP WG 11.3 Workshop on Database Security, 1997.

[2] D. Ferrailo, R. Kuhn, "Role-Based Access Control", Proceedings of 15th National Computer Security Conference, pp.554-563, 1992.

[3] D. Ferraiolo, J. Cugini, D. R. Kuhn "Role Based Access Control : Features and Motivations", In Annual Computer Security Applications Conference, 1995.

[4] R. S. Sandhu, E. J. Coyne, H. Feinstein, C.E. Youman "Role-Based Access Control Models", IEEE Computer, Vol. 29, No. 2, pp.38-47, 1996,

[5] J. D. Moffett, "Control Principles and Role Hierarchies", Third ACM Workshop on Role-based access control, pp.63-69 Oct. 1998.

[6] J. F. Barkley, A. V. Cincotta, D. F. Ferraiolo, S. Gavrilla, D. R. Kuhn "Role Based Access Control for the World Wide Web", 20th national computer security conf., April. 1997.

오 석 균(Sug Kyun, Oh) 정회원

1981년 2월 : 숭실대학교 전자계산학과(공학사)
1983년 2월 : 숭실대학교 대학원 전자계산학과(공학석사)
2000년 2월 : 청주대학교 대학원 전산정보공학과 박사수료

1986년 3월~1991년 2월 : 기전여자대학 전자계산과 조교수
1991년 3월~현재 : 충청대학 컴퓨터학부 부교수
1994년 3월~1995년 3월 : 충청대학 전자계산소 소장
1999년 2월~현재 : 충청대학 컴퓨터학부 학부장
<주관심 분야> 컴퓨터 네트워크, 컴퓨터 보안, 운영체제, 멀티미디어

김 성 렬(Seong Ryeol, Kim) 정회원

1982년 2월 : 숭실대학교 전자계산학과(공학사)
1987년 2월 : 숭실대학교 대학원 전자계산학과(공학석사)
1992년 2월 : 숭실대학교 대학원 전자계산학과(공학박사)

1982년 1월~1984년 2월 : 한국전력공사 전자계산소 근무
1984년 3월~1990년 8월 : 오산대학 전자계산과 조교수
1990년 9월~현재 : 청주대학교 컴퓨터정보공학과 부교수
1997년 1월~1998년 1월 : 호주 QUT ISRC 객원교수
<주관심 분야> 컴퓨터 정보통신, 컴퓨터 보안, 분산 객체시스템