

EAM 인증을 활용한 통합관제시스템 구축

팀 명 : 8층에서 살아남기
지도 교수 : 양 환 석 교수님
팀 장 : 김성윤
팀 원 : 박영진
박광우
김인권
임종철
윤새민

2018. 11.
중부대학교 정보보호학과

목 차

1. 서론

- 1.1 프로젝트 제목
- 1.2 프로젝트 선정 이유

2. 관련 연구

- 2.1 IDS
- 2.2 iptables
- 2.3 EAM
 - 2.3.1 EAM이란?
 - 2.3.2 EAM 도입 배경
- 2.4 vnstat
- 2.5 shellinabox

3. 본론

- 3.1 시스템 구성
- 3.2 프로그램 구성
 - 3.2.1 사용자 페이지
 - 3.2.2 웹페이지 기능
 - 3.2.3 관리자 페이지

4. 결론

5. 첨부

- 5.1 프로그램 소스코드 일부
 - 5.1.1 웹 소스
 - 5.1.2 인증 소스

1. 서 론

1.1 프로젝트 제목

EAM인증방식을 활용한 통합인증시스템

1.2 프로젝트 선정 사유

최근 많은 금융권과 대기업을 중심으로 한 번 로그인으로 여러 서버에 접근할 수 있는 SSO(Single Sign On) 기능과 권한에 따라 차등적으로 자원에 접근하게 해주는 보안성을 제공하는 솔루션을 이용하고 있다. 이 기술은 SSO와 권한관리, 자원관리 및 보안정책 수립을 합한 것으로 한 번의 로그인으로 다양한 시스템을 사용 할 수 있다. 또한 사용자의 정보에 대한 접근을 중앙에서 통제할 수 있어 비인가 클라이언트의 내부망 접근을 제어 할 수 있다. EAM 기술 뿐 만 아니라 IDS 시스템을 구축하여 실제로 네트워크와 서버 상에서 작동하게 하고 다양한 패턴을 가진 공격기법들을 분석하여 탐지를 하고, 방화벽을 이용하여 차단 을 한다. 다양한 공격을 탐지, 차단하여 쌓인 유해로그와 트래픽 관제를 통해 추후 공격예방 과 대응을 할 수 있다.

2. 관련연구

2.1 IDS

IDS란 Intrusion Detection System의 약자로서

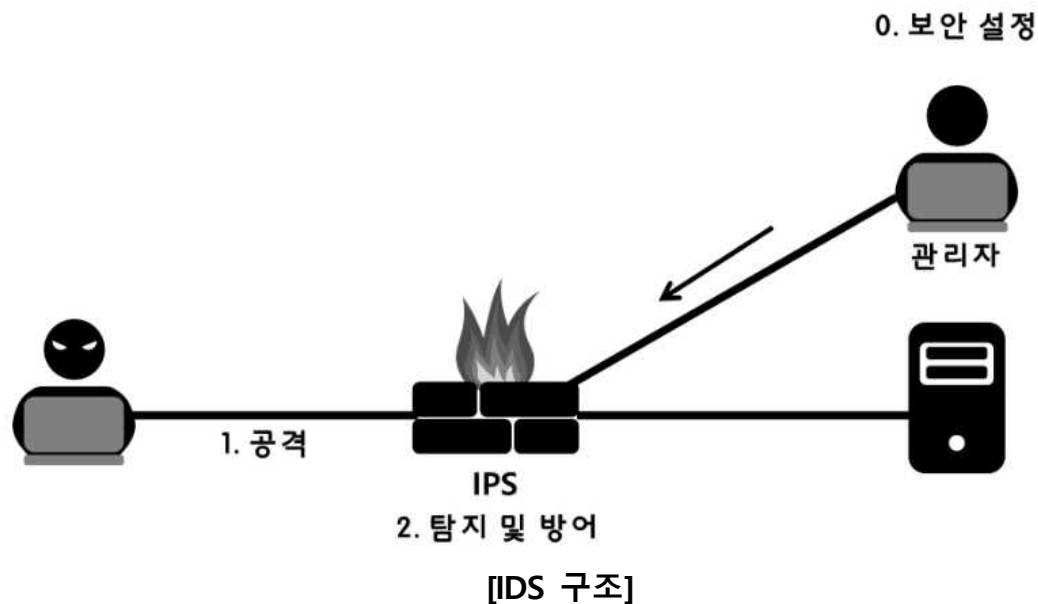
Intrusion : 침입, 침범

Detection : 발견, 간파, 탐지

System : 제도, 체제

이며 합치면 침입 탐지 시스템이라고 한다.

해커들과 같은 비인가된 사용자가 시스템을 조작하는 것을 탐지하며, 기본적인 방화벽 같은 탐지 시스템이 탐지할 수 없는 여러 종류의 악의적 네트워크 트래픽을 탐지하여 로그를 남기게 되고 관리자는 그 로그를 바탕으로 보안대책을 다시 정한다.



2.2 iptables

리눅스상에서 방화벽을 설정하는 도구로서 커널 2.4 이전 버전에서 사용되던 ipchains를 대신하는 방화벽 도구이다.

iptables는 커널상에서의 netfilter *패킷필터링 기능을 사용자 공간에서 제어하는 수준으로 사용 할 수 있다.

* 패킷필터링

패킷 필터링이란 지나가는 패킷의 Header를 살펴보고 그 전체 패킷의 운명을 결정하는 소프트웨어의 일부이다. 이것은 패킷을 'DROP(즉, 마치 전혀 전달되지도 않았던 것처럼 패킷

을 거부)하던가, 'ACCEPT(즉, 패킷이 지나가도록 내버려둠)'를 하던가 또는 다른 더욱 복잡한 무엇을 할 것인가를 결정하는 것이다. 주요 사용 목적은 패킷의 제어와 보안이다. 패킷을 제어함으로써 자신이 원하는 것만을 외부 네트워크에 공개할 수 있고 또 자신이 허용하는 것만을 내부 네트워크 사용자에게 전해줄 수 있다. 이렇게 함으로써 허가되지 않은 정보가 외부로 누출되는 것을 막을 수 있으며, 외부에서 내부로 크래킹을 시도하는 것을 막을 수 있게 된다.

2.3 EAM

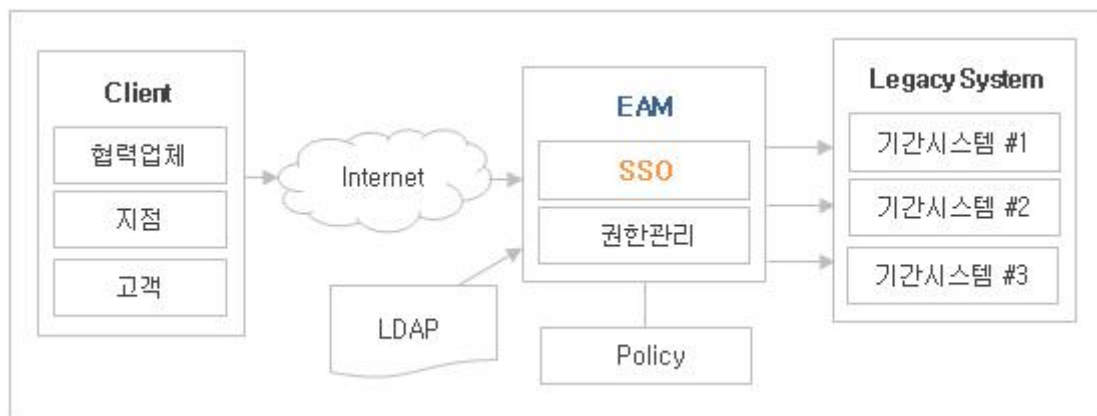
2.3.1 EAM이란?

인트라넷, 엑스트라넷 및 일반 클라이언트/서버 환경에서 자원의 접근 인증과 이를 기반으로 자원에 대한 접근 권한을 부여, 관리하는 통합 인증 관리 솔루션. 한 번 로그인으로 여러 서버에 접근할 수 있는 SSO(Single Sign-On) 기능과 각 ID에 따라 사용 권한을 차등 부여하는 통합 인증과 권한 관리 시스템이다.

일반 기업과 금융권, 포털 등 기업 내 사용자와 일반 사용자에게 적용 가능하며 인터넷 뱅킹, 쇼핑 등 서비스 편의성과 운영비 절감 및 기업 내 보안 효과가 있다.

2.3.2 EAM 도입 배경

사내 어플리케이션이 다양해지면서 기업의 Portal구축 증가 → 시스템별 권한 제어 필요
기업의 "정보공유"마인드 확산에 따라 공개된 정보의 접근통제 요구



[EAM 구조]

2.4 vnstat

vnStat 는 Linux 운영 체제 용 네트워크 유틸리티이다 . 명령 줄 인터페이스를 사용한다. vnStat 명령은 콘솔 기반 네트워크 트래픽 모니터이다. 패킷 스니핑 도구와 같이 직접적으로 트래픽을 모니터 하지 않고 커널에서 제공해주는 네트워크 인터페이스 정보를 기반으로 정보를 보여준다.

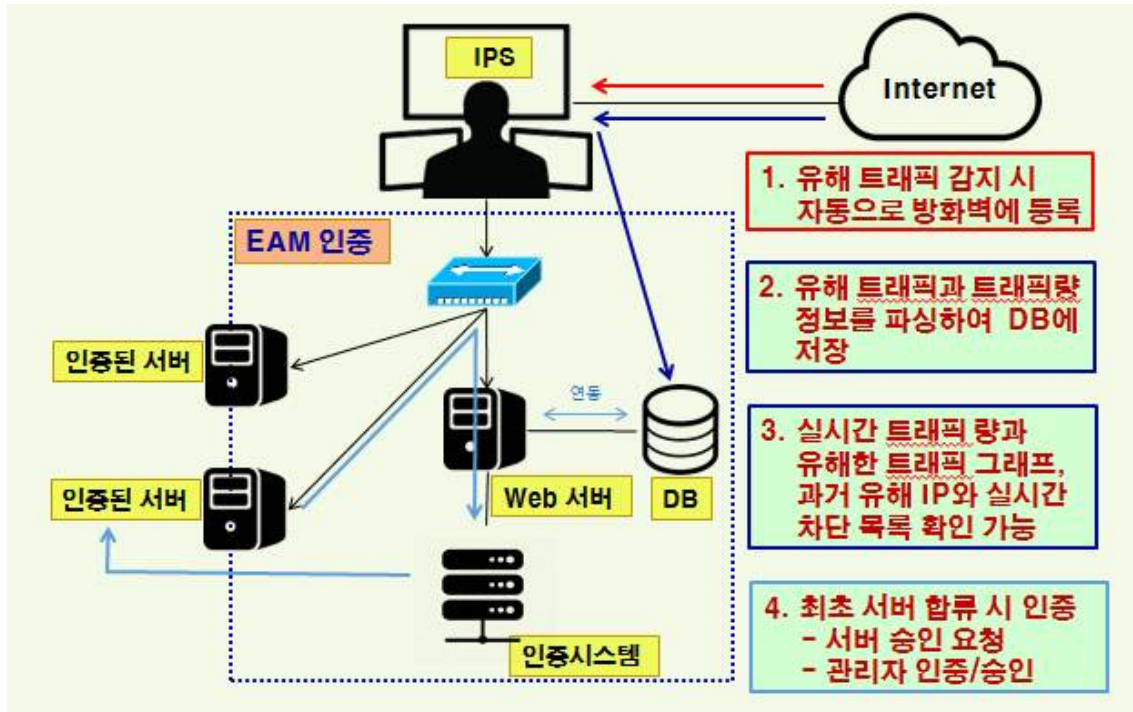
선택한 인터페이스에 대한 시간별, 일별 및 월별 네트워크 트래픽 로그를 유지 하지만 패킷 스니퍼는 아니다. 교통 정보는 proc 파일 시스템 에서 분석된다. 그런 식으로 vnStat는 루트 권한 없이도 사용할 수 있다.

2.5 shellinabox

Shell In A Box (cellinabox로 발음됨)는 Markus Gutschke가 만든 웹 기반 터미널 에뮬레이터이다. 지정된 포트에서 웹 기반 SSH 클라이언트로 실행되고 AJAX/JavaScript 및 CSS 브라우저가 활성화되지 않은 상태에서 리눅스 서버 SSH 셸을 원격으로 액세스하고 제어하도록 웹 터미널 에뮬레이터를 요청하는 웹 서버가 내장되어 있다.

3. 본론

3.1 시스템 구성



3.2 프로그램 구성

3.2.1 사용자 페이지

회원이입을 위해 Register를 클릭 한 후 ID와 Password를 입력한다.

Auth System

[Login](#)[Register](#)

[회원가입 정보 입력]

회원가입이 완료 되면 아래와 같은 팝업창이 뜬다.



[회원가입 완료]

로그인이 완료 되면 아래와 같은 메인페이지로 접속이 가능하다.



[웹 페이지 메인]

로그인 후 서비스이용을 위해 메뉴를 선택하면
'서버 등록이 필요합니다.'라는 팝업창이 뜬다.



[서버 등록 전]

3.2.2 웹 페이지 기능

로그인 후 나의 서버 정보에서 서버 등록을 위한 키 값 확인 후 관리자에게 요청해야한다.
서버 미등록 시 서버 등록이 필요합니다. 라는 안내창이 발생한다.



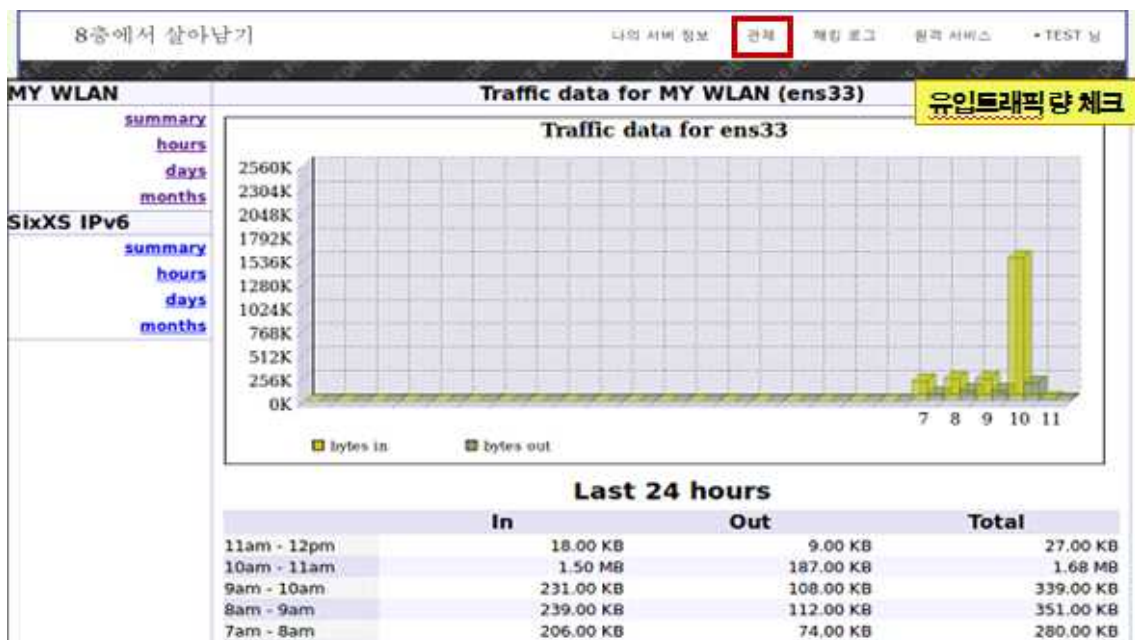
[나의 서버 정보]

ID마다 부여된 키값을 보내 관리자에게 인증을 요청한다.

```
sy@ubuntu:~/Desktop/agent$ nodejs cli.js test10 test ROv5hwsnFv
```

[서버등록 요청]

관제 메뉴를 클릭하게 되면 유입되는 트래픽량을 체크 할 수 있다.



[관제]

해킹 로그 메뉴를 클릭하면 서버로 오는 공격의 대해 로그 정보를 확인 할 수 있다.

8층에서 살아남기

나의 서버 정보

관제

해킹 로그

원격 서비스

→ TEST10 님

test10님 서버의 해킹 로그 정보

ID	당지시간	프로토콜	출발지	목적지	당지유형
2	2	3	4	192.168.253.129	6

[해킹 로그]

원격 서비스 메뉴를 클릭하면 SSH원격 서비스를 이용할 수 있다.

8층에서 살아남기

나의 서버 정보

관제

해킹 로그

원격 서비스

→ TEST10 님

SSH원격서비스

```
ubuntu login: kim
Password:
Last login: Thu Sep 13 22:26:03 PDT 2018 from 10.100.114.209 on pts/18
Welcome to Ubuntu 16.04.4 LTS (GNU/Linux 4.15.0-34-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

145 packages can be updated.
0 updates are security updates.

kim@ubuntu:~$
```

[SSH 원격 서비스]

3.2.3 관리자 페이지

Auth System

LoginRegister

LOGIN

[관리자 로그인]

관리자로 로그인하면 현재 승인 요청중인 서버의 목록을 확인 할 수 있다.

IDS Admin Queue List Monitor Log SSHadmin ▾									
승인 요청 서버									
ID	소유자	아이피	맥 주소	호스트이름	운영체제	인증시간	권한레벨	승인	거부
9	b	192.168.253.129	00:0c:29:74:b5:cf	ubuntu	linux	2018-10-14 12:04:20	4	승인 확인	거부
10	test1	10.100.114.125	00:0c:29:e9:71:33	ubuntu	linux	2018-10-14 12:17:20	4	승인 확인	거부
11	j	192.168.253.129	00:0c:29:74:b5:cf	ubuntu	linux	2018-10-14 12:48:58	4	승인 확인	거부
12	ds	192.168.253.129	00:0c:29:74:b5:cf	ubuntu	linux	2018-10-14 12:50:18	4	승인 확인	거부
13	cs	10.100.114.233	00:0c:29:44:fb:e2	ubuntu	linux	2018-10-14 12:51:59	4	승인 확인	거부
14	er	10.100.114.233	00:0c:29:44:fb:e2	ubuntu	linux	2018-10-14 12:54:06	4	승인 확인	거부

[승인 요청 서버]

승인 확인을 클릭하면 아래와 같은 팝업창이 뜬다.

10.100.114.149 내용:
정상적으로 승인되었습니다.

확인

[승인 요청 승인]

관리자는 승인 완료된 서버들의 목록을 확인 할 수 있다.

IDS Admin Queue List Monitor Log SSH admin							
승인된 서버							
ID	소유자	아이피	맥 주소	호스트이름	운영체제	인증시간	권한레벨
1	qw	192.168.253.129	00:0c:29:74:b5:cf	ubuntu	linux	2018-10-14 06:55:49	4
2	qwe	192.168.253.129	00:0c:29:74:b5:cf	ubuntu	linux	2018-10-14 07:35:58	4
3	test10	192.168.253.129	00:0c:29:74:b5:cf	ubuntu	linux	2018-10-14 07:58:07	4
4	fq	192.168.253.129	00:0c:29:74:b5:cf	ubuntu	linux	2018-10-14 09:18:29	4

[승인된 서버 확인]

현재 등록된 서버들의 SSH List를 확인 할 수 있다.

IDS Admin Queue List Monitor Log SSH admin							
SSH List							
List							
qw SSH							
qwe SSH							
test10 SSH							
fq SSH							

[SSH List]

현재 등록된 서버들의 Traffic량을 체크 할 수 있다.

IDS Admin	Queue	List	Monitor	Log	SSH	admin ▼
Monitor List						
List						
qw Traffic						
qwe Traffic						
test10 Traffic						
fq Traffic						

[Monitor List]

현재 등록된 서버들의 Total IDS Log를 확인 할 수 있다.

IDS Admin Queue List Monitor Log SSH admin ▾

Total IDS Log

ID	탐지시간	프로토콜	출발지	목적지	탐지유형
1	2	3	4	5	6
2	2	3	4	192.168.253.129	6

[Total IDS Log]

4. 결론

EAM인증 방식을 통한 통합관제 시스템을 개발 하면서 SSO보다 보안성이 뛰어난 EAM에 대해 알게 되었고, 웹 페이지에서 실시간으로 로그를 불러와 관제하거나 서버에 접근하여 관리를 하는 방법을 공부할 수 있었다.

SSO는 권한을 모두에게 동일하게 주기 때문에 보안성에 문제가 있어 보여, EAM을 통해 각자에게 다른 권한을 부여해서 보안성을 강화하였다.

또한 회원가입 후에도 로그인마다 하나의 키값을 부여하여 관리자에게 다시 한 번 인증을 받게 하여 비인가 접근자에 대하여 접근 방지를 하였다.

웹 페이지에서 SSH 원격 서비스를 지원을 하여 Putty와 같은 애플리케이션을 따로 설치하지 않아도 편리하게 서버로의 접근과 관리가 편리하게 구현하였다.

그리고 인가된 각 서버마다 유입 트래픽을 분석하고 탐지된 유해 IP를 방화벽에 등록하여 안전한 네트워크 환경을 제공하고, 서비스 중인 서버의 Dos공격을 빠르게 탐지하여 예방 할 수 있게 하였다.

5. 첨부

5.1 프로그램 소스코드 일부

5.1.1 웹 소스

(1) 에이전트 제공 및 인증 관리 php

```
<?php
    include 'db.php';
    $sql = "select * from servers where username = '". $_SESSION['username']. "'";
    $query = $db->query($sql) or die($sql . " ". $db->error);
    $selectUserSql = "select * from users where username = '".
$_SESSION['username']. "'";
    $selectUserQuery = $db->query($selectUserSql) or die($db->error);
    $selectUserResult = mysqli_fetch_assoc($selectUserQuery);
?>

<?php if($query->num_rows == 1) { ?>
<div class="container">
    <div class="scrollable">
        <table>
            <thead>
                <tr>
                    <th>아이피</th>
                    <th>맥 주소</th>
```

```

        <th>호스트 이름</th>
        <th>운영체제</th>
        <th>인증시간</th>
        <th>관리자 승인 여부</th>
    </tr>
</thead>
<tbody>
    <?php while($result = mysqli_fetch_array($query)) { ?>
    <tr>
        <td><?php echo $result['ip']; ?></td>
        <td><?php echo $result['mac']; ?></td>
        <td><?php echo $result['hostname']; ?></td>
        <td><?php echo $result['platform']; ?></td>
        <td><?php echo $result['auth_date']; ?></td>
        <td><?php echo ($result['permit'] == 0 ? "미승인": "승인") ; ?></td>
    </tr>
    <?php } ?>

</tbody>
</table>
</div>
</div>
<?php } ?>

<?php if($query->num_rows == 0) { ?>
<div class="container">
<div id="comments">
    <h2>서버 등록 및 인증 방법</h2>
    <ul>
        <li>
            <article>
                <header>
                    <center><h1>nodejs      cli.js      Username      Password      Key
Level</h1></center>
                </header>
                <div class="comcont">
                    <code style="color: red;">

                    <center><h2>
                        username : "<?php echo $_SESSION['username']; ?>",

```

```

        password : "your password",
        key : "<?php echo $selectUserResult['authkey']; ?>",
        level : "(1~4)"</h2></center>

<?php
if ($handle = opendir('./uploadfile/')) {
    while (false !== ($entry = readdir($handle))) {
        if ($entry != '.' && $entry != '..'){
            if(strchr($entry, ".tar") ==
true){

                                                    echo "<a
href='./uploadfile/$entry' download>";

                                                    e c h o

"<center><img src='./uploadfile/agent.jpg' /></center>";

                                                    e c h o

"</a>";

                echo "<br /><center><div>File name : $entry (Click the
image.)</div></center><br />";

                                                    }

                                                    }

                }
            closedir($handle);
        }
    }
?>

</code>
</div>
</article>
</li>
</ul>
</div>
<?php } ?>

```

(2) 트래픽 관제 php

```

<?php
include './config/db.php';
include './util/util.php';

```

```

    $adminsqli = "select is_admin from users where username = '".
$_SESSION['username']. "'";
    $adminquery = $db->query($adminsqli) or die($db->error);
    $adminresult = mysqli_fetch_row($adminquery);
    $sql = "select * from servers where username = '". $_SESSION['username']. "'";
    $query = $db->query($sql) or die($db->error);
    $result = mysqli_fetch_assoc($query);

if($_SESSION['username'] != 'admin'){
    if ($query->num_rows == 0) {
        go_error("서버 등록이 필요합니다.", "./index.php?menu=server");
    } else {
        if ($result['permit'] == 0) {
            go_error("최고 관리자의 승인이 필요합니다.", "./index.php?menu=server");
        }
    }
}

if ($adminresult[0] == 1){
    $sql = "select * from servers where permit=1";
    $query = $db->query($sql) or die($db->error);
?>
<div class="container">
<br>
<h1><?php echo "Monitor List" ?></h1>
<br>
<div style="overflow-x: auto;">
<table class="table">
    <thead class="thead-dark">
        <tr>
            <th scope="col">List</th>

        </tr>

    </thead>

    <tbody>

        <?php

```

```

while($result = mysqli_fetch_assoc($query))
{
    $url = "http://". $result['ip'].
"/vnstat_php_frontend-1.5.1/index.php?if=ens33&graph=large&style=light&page=d";
    ?>

    <tr>
        <td><a href="<?php echo $url ?>"><?php echo $result['username'] ?>
Traffic</a></td>
    </tr>
<?php
}
?>
</tbody>
</table>
</div>
</div>
<?php
}
else {
    $url = "http://". $result['ip'].
"/vnstat_php_frontend-1.5.1/index.php?if=ens33&graph=large&style=light&page=d";
    ?>
<div class="container">
<br>
<center><h1><?php echo $_SESSION['username'] . "님 서버의 트래픽 정보"
?></h1></center>
<br>
<iframe src="<?php echo $url; ?>" height="130%" width="100%">
</iframe>
</div>
<?php
}?>

```

(3) 해킹 로그 PHP

```

<html>
<head>
<title>Attack_log </title>
</head>
<body>

```

```

<?php
$objConnect = mysqli_connect("localhost","id","password", "dbname") or die("Error
Connect to Database");
$strSQL1 = "SELECT * FROM servers where username='". $_SESSION['username']. "'";
$objQuery1 = $objConnect->query($strSQL1) or die ("Error Query [".$strSQL1."]);
$objResult1 = mysqli_fetch_assoc($objQuery1);
$ip = $objResult1['ip'];
$strSQL = 'SELECT * FROM attacklog where dst=' . "'$ip'";
$objQuery = $objConnect->query($strSQL) or die ("Error Query [".$strSQL."]);

$sql = "select * from servers where username = '". $_SESSION['username']. "'";
$query = $objConnect->query($sql) or die("error");
$result = mysqli_fetch_assoc($query);
if ($query->num_rows == 0) {
    ?>
    <script>
        alert("서버 등록이 필요합니다.");
        location.replace('/index.php?menu=server');
    </script>
<?php
}
else {
    if ($result['permit'] == 0) {
        ?>
        <script>
            alert("최고 관리자의 승인이 필요합니다.");
            location.replace('/index.php?menu=server');
        </script>
<?php
    }
?>
<div class="container">
<br>
<center> <h1><?php echo $_SESSION['username'] . "님 서버의 해킹 로그 정보"
?> </h1> </center>
<br>
<div style="overflow-x: auto;">
<table class="table">
    <thead class="thead-dark">
        <tr>

```

```

<th scope="col"> <div align="center">ID </div> </th>
<th scope="col"> <div align="center">탐지시간 </div> </th>
<th scope="col"> <div align="center">프로토콜 </div> </th>
<th scope="col"> <div align="center">출발지 </div> </th>
<th scope="col"> <div align="center">목적지 </div> </th>
<th scope="col"> <div align="center">탐지유형 </div> </th>
</tr>
</thead>
<tbody>
<?php
while($objResult = mysqli_fetch_array($objQuery))
{
?>
<tr>
<td><div align="center"><?php echo $objResult["id"];?> </div> </td>
<td><div align="center"><?php echo $objResult["time"];?> </div> </td>
<td><div align="center"><?php echo $objResult["Protocol"];?> </div> </td>
<td><div align="center"><?php echo $objResult["src"];?> </div> </td>
<td><div align="center"><?php echo $objResult["dst"];?> </div> </td>
<td><div align="center"><?php echo $objResult["rule"];?> </div> </td>
</tr>
</tbody>
<?php
}
?>
</table>
</div>
</div>
<?php
mysqli_close($objConnect);
?>
</body>
</html>
<?php
}
?>

```

(4) SSH 원격 서비스 php

```

<?php
include './config/db.php';

```

```

include './util/util.php';

$adminsqli = "select is_admin from users where username = '".
$_SESSION['username']. "'";
$adminquery = $db->query($adminsqli) or die($db->error);
$adminresult = mysqli_fetch_row($adminquery);
$sql = "select * from servers where username = '". $_SESSION['username']. "'";
$query = $db->query($sql) or die($db->error);
$result = mysqli_fetch_assoc($query);

if($_SESSION['username'] != 'admin'){
    if ($query->num_rows == 0) {
        go_error("서버 등록이 필요합니다.", "./index.php?menu=server");
    } else {
        if ($result['permit'] == 0) {
            go_error("최고 관리자의 승인이 필요합니다.", "./index.php?menu=server");
        }
    }
}

if ($adminresult[0] == 1){
    $sql = "select * from servers where permit=1";
    $query = $db->query($sql) or die($db->error);
?>
<div class="container">
<br>
<h1><?php echo "SSH List" ?> </h1>
<br>
<div style="overflow-x: auto;">
<table class="table">
    <thead class="thead-dark">
        <tr>
            <th scope="col">List</th>
        </tr>
    </thead>
    <tbody>
        <?php
        while($result = mysqli_fetch_assoc($query))
        {
            $url = "https://". $result['ip']. ":4200";

```



```

    host: config.db.host,
    user: config.db.user,
    password: config.db.password,
    database: config.db.database
  });
  var array = {
    platform: os.platform(),
    hostname: os.hostname(),
    uptime: os.uptime(),
    cpus: os.cpus(),
    network: os.networkInterfaces(),
    totalmem: os.totalmem(),
    loadavg: os.loadavg()
  };
  function exit(message) {
    console.log(message);
  }
  connection.connect(function (err) {
    if (err) throw err;
    exit('connected!');
  });
  io.sockets.on('connection', function (socket) {
    socket.emit('example message 1', { hello: 'world 1' });

    socket.on('example message 2', function (data) {
      console.log(data);
    });
    socket.on('aa',function (data){
      console.log(data);
    });

    var info = [];
    if (data.platform === 'linux') {
      data.network.ens33.forEach(function (item) {
        if (item.family === 'IPv4') {
          info.push({platform: data.platform, hostname: data.hostname, ip:
item.address, mac: item.mac, username: data.username, password: data.password, key:
data.key,lev: data.lev})
        }
      })
    }
  });
};

```

```

console.log(info[0]);

connection.connect(function (err) {

    connection.query("select * from users where username = ?",
[info[0].username], function (err, result) {
        if (err) throw err;
        if (md5(info[0].password) === result[0].password) {
            if (info[0].key === result[0].authkey) { // auth compare
                console.log("auth success");
                connection.query("select * from servers where username = ?",
[info[0].username], function (err, result) {
                    if (err) throw err;
                    if (result.length === 0) {
                        connection.query("insert into servers (username, ip, mac,
hostname, platform, permit) values (?, ?, ?, ?, ?, ?)",
[info[0].username, info[0].ip, info[0].mac, info[0].hostname,
info[0].platform, false], function (err) {
                            if (err) throw err;
const text = info[0].ip+" "+info[0].lev;
fs.writeFileSync("target.txt",      'Wufoff'      +      text,      {encoding:      'utf8'});

                exit('insert db success');
            });
        } else {
            exit('insert db failed');
        }
    });
} else {
    exit('auth failed');
}
} else {
    exit('password incorrect');
}
});
});
});
});

```

(2) 인증(Client) nodejs

```
var io = require('socket.io-client');
const os = require('os');
var commanda = process.argv[2];
var commandb = process.argv[3];
var commandc = process.argv[4];
var commandd = process.argv[5];

var array = {
  platform: os.platform(),
  hostname: os.hostname(),
  uptime: os.uptime(),
  cpus: os.cpus(),
  network: os.networkInterfaces(),
  totalmem: os.totalmem(),
  loadavg: os.loadavg(),
  username: commanda,
  password: commandb,
  key: commandc,
  lev: commandd
};
var socket = io.connect('http://127.0.0.1:8000');
socket.emit('aa', array);
```



EAM 인증을 활용한 통합관제시스템 구축

2018. 11. 7

담당 교수 : 양환석 교수님

8층에서 살아남기

(김성윤, 박영진, 박광우, 김인권, 임종철, 윤세민)

1



목 차

- 조원 편성
- 주제 선정
- 구 상 도
- 추진 경과
- 개발 환경 및 시스템 개발
- 개발 결과 및 시스템 운영
- 결론 및 기대효과

2



조원 편성

성명	담당
김성윤	팀장 / EAM 인증체제 구현
박영진	웹 페이지 구축 / EAM 인증체제 구현
박광우	웹 페이지 구축 / DB 설계
김인권	웹 프로그래밍 / IDS Rules 결정
임중철	방화벽 / DB 설계
윤새민	웹 및 UI 프로그래밍

3



주제 선정

- 현재 널리 사용되는 ID/PW 인증방식은 각 서버마다 별도의 ID/PW 사용으로 기억에 의존하거나 메모 방식으로 관리
보안관리에 한계가 있고 사용 불편
- 또한 클라이언트별 권한관리가 어려워 대형 서버 등에
적용이 제한

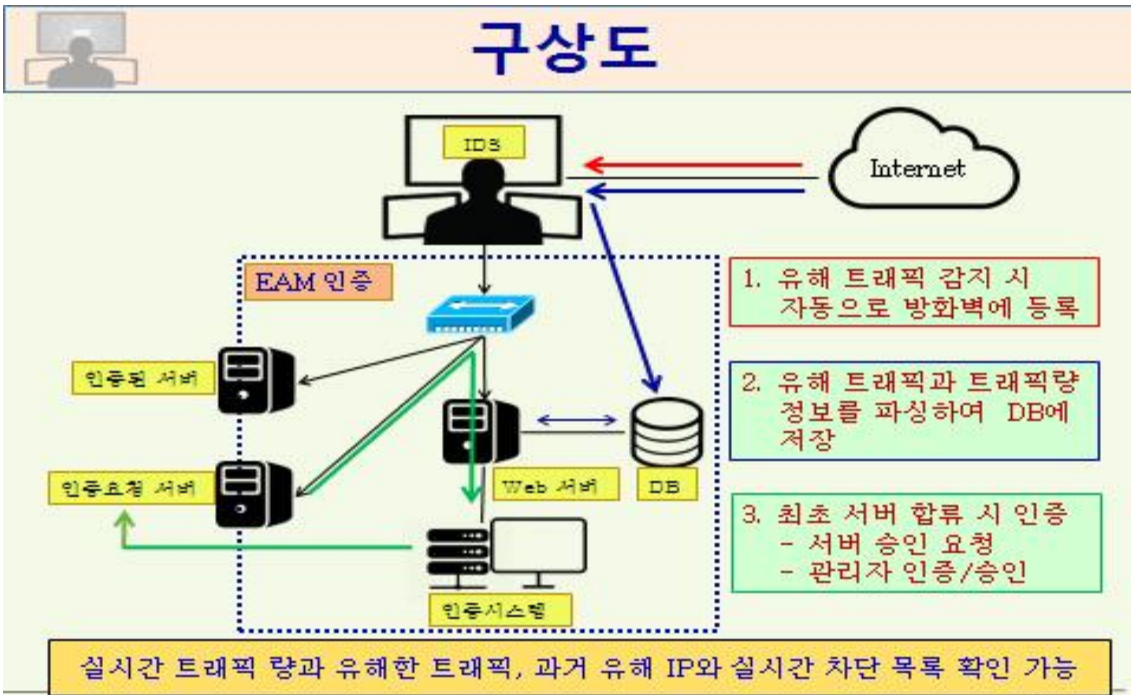


EAM 인증/권한관리 방식을 적용, SSO 인증, 자동적인 권한관리 및
유해 트래픽을 탐지/차단하는 방안을 연구

EAM(Extranet Access Management) 단일 ID와 PW로 인증과 접근
권한을 동시에 관리하는 기술(SSO + 권한관리)

※ 단일 인증으로 여러 서비스를 이용할 수 있고, 권한관리를 통해 각 계정별
자동적 권한 적용이 가능

4



5

추진 경과

(단위 : 월)

구분 \ 기간 (2018년)	3	4	5	6	7	8	9	10	11
환경구축									
IS & DB									
웹 프로그래밍									
EAM 인증									
성능 점검/보완									

6



개발 환경 및 시스템 개발(1/8)

개발 환경



운영체제

Ubuntu 16.04 LTS



WEB

Html5, PHP7, CSS, J query



침입탐지(IDS)

Suricata



공격 차단

IPtables, Shell script



EAM 인증

Node.js, Mysql_DB

7



개발 환경 및 시스템 개발(2/8)

IDS 구축(Suricata)

○ IDS 탐지를 위한 Rule 작성

```

kin@ubuntu:/etc/suricata/rules$ ls
app-layer-events.rules      emerging-scada.rules
botcc.portgrouped.rules    emerging-scan.rules
- decoder-events.rules      # available in suricata sources under rule
- stream-events.rules       # available in suricata sources under rule
- http-events.rules         # available in suricata sources under rules dir
- smtp-events.rules         # available in suricata sources under rules dir
- dns-events.rules          # available in suricata sources under rules dir
- tls-events.rules          # available in suricata sources under rules dir
- modbus-events.rules       # available in suricata sources under rules dir
- app-layer-events.rules    # available in suricata sources under rules dir
- dnp3-events.rules         # available in suricata sources under rules dir
- ntp-events.rules          # available in suricata sources under rules dir

classification-file: /etc/suricata/classification.config
reference-config-file: /etc/suricata/reference.config
# threshold-file: /etc/suricata/threshold.config
        
```

Suricata

Rule 목록

Rule 작성

Suricata 구축 및 유해 Traffic를 걸르기 위한 Rule 적용

8



개발 환경 및 시스템 개발(3/8)

IDS 구축(유해트래픽 탐지)

○ 로그파일에 유해트래픽 기록

```
root@ubuntu:/home/kin# cat fast.log
05/13/2018-22:25:21.698802  [**] [1:2240001:2] SURICATA DNS Unsolicited response [**]
[Classification: Generic Protocol Command Decode] [Priority: 3] {UDP} 192.168.233.2:53
-> 192.168.233.148:33715
05/13/2018-22:25:21.698802  [**] [1:2240001:2] SURICATA DNS Unsolicited response [**]
[Classification: Generic Protocol Command Decode] [Priority: 3] {UDP} 192.168.233.2:53
-> 192.168.233.148:33715
05/13/2018-22:39:09.564876  [**] [1:2240001:2] SURICATA DNS Unsolicited response [**]
[Classification: Generic Protocol Command Decode] [Priority: 3] {UDP} 168.126.63.1:53
-> 192.168.233.148:49169
05/13/2018-23:12:27.585699  [**] [1:2240001:2] SURICATA DNS Unsolicited response [**]
[Classification: Generic Protocol Command Decode] [Priority: 3] {UDP} 168.126.63.1:53
-> 192.168.233.148:40729
05/13/2018-23:47:39.748349  [**] [1:10001:1] kimsungyonn rule [**] [Classification: (n
ull)] [Priority: 3] {TCP} 192.168.233.148:35662 -> 112.175.50.171:80
05/13/2018-23:47:39.798710  [**] [1:10001:1] kimsungyonn rule [**] [Classification: (n
ull)] [Priority: 3] {TCP} 192.168.233.148:35662 -> 112.175.50.171:80
05/13/2018-23:47:39.824377  [**] [1:10001:1] kimsungyonn rule [**] [Classification: (n
ull)] [Priority: 3] {TCP} 192.168.233.148:35662 -> 112.175.50.171:80
```

IDS 로그파일

지정한 Rule에 의해 탐지된 유해트래픽 로그파일에 기록



개발 환경 및 시스템 개발(4/8)

IDS 구축(유해트래픽 저장)

○ 실시간 로그 변화 체크

```
root@ubuntu:/home/kin# ./newdb.sh
file check
mysql select * from attacklog;
4423
file c
4423
```

id	time	Protocol	src	dst	rule
1	05/13/2018-22:25:21.698802	UDP	192.168.233.2:53	192.168.233.148:33715	SURICATA DNS Unsolicited re
2	05/13/2018-22:39:09.564876	UDP	168.126.63.1:53	192.168.233.148:49169	SURICATA DNS Unsolicited re
3	05/13/2018-23:12:27.585699	UDP	168.126.63.1:53	192.168.233.148:40729	SURICATA DNS Unsolicited re
4	05/13/2018-23:47:39.748349	TCP	192.168.233.148:35662	112.175.50.171:80	kimsungyonn rule
5	05/13/2018-23:47:39.798710	TCP	192.168.233.148:35662	112.175.50.171:80	kimsungyonn rule

로그파일 변화 체크

변화가 생긴 로그 DB에 정보 저장

변화가 생긴 로그파일의 기록을 데이터베이스에 저장

10



개발 환경 및 시스템 개발(5/8)

IPtables 구축(유해 IP 방화벽 등록)

- 저장된 로그 정보를 이용하여 방화벽에 유해 IP를 등록

```
#!/bin/bash
echo "dynamic(yes) static(no) default(key)"
read key
Chain INPUT (policy DROP)
case "$1" in
yes) ACCEPT tcp -- 0.0.0.0/0 0.0.0.0/0 tcp dpt:22
echo "ACCEPT all -- 0.0.0.0/0 0.0.0.0/0 state RELATED,ESTABLISHED"
read dpt
echo "ACCEPT tcp -- 0.0.0.0/0 0.0.0.0/0 tcp dpt:53"
while read dpt
do
echo "ACCEPT udp -- 0.0.0.0/0 0.0.0.0/0 udp dpt:53"
iptables -A INPUT -s 0.0.0.0 -d 0.0.0.0 -p tcp --dport $dpt -j ACCEPT
done < 4
echo "ACCEPT tcp -- 0.0.0.0/0 0.0.0.0/0 tcp dpt:21"
iptables -A INPUT -s 0.0.0.0 -d 0.0.0.0 -p tcp --dport $dpt -j ACCEPT
echo "ACCEPT tcp -- 0.0.0.0/0 0.0.0.0/0 tcp dpts:1024:65535"
echo "ACCEPT tcp -- 0.0.0.0/0 0.0.0.0/0 tcp dpt:80"
echo "ACCEPT tcp -- 0.0.0.0/0 0.0.0.0/0 tcp dpt:443"
echo "ACCEPT tcp -- 192.168.1.1 0.0.0.0/0 tcp dpt:99"
read ip
echo "ACCEPT tcp -- 192.168.0.1 192.168.0.8 tcp dpt:22"
echo "DROP tcp -- 192.168.233.148 115.175.50.171 tcp dpt:80"

Chain FORWARD (policy DROP)
target prot opt source destination

Chain OUTPUT (policy ACCEPT)
target prot opt source destination
iptables -A OUTPUT -s 0.0.0.0 -d 0.0.0.0 -p tcp --sport 21 -j ACCEPT
iptables -A OUTPUT -s 0.0.0.0 -d 0.0.0.0 -p tcp --sport 1024:65535 -j ACCEPT
*)
```

Suricata Rule에 걸린 유해 IP를 방화벽(IPtables)에 등록

11



개발 환경 및 시스템 개발(6/8)

인증 소스코드(클라이언트)

```
var io = require('socket.io-client');
const os = require('os');
var commanda = process.argv[2];
var commandb = process.argv[3];
var commandc = process.argv[4];
var commandd = process.argv[5];
var array = {
platform: os.platform(),
hostname: os.hostname(),
uptime: os.uptime(),
cpus: os.cpus(),
network: os.networkInterfaces(),
totalmem: os.totalmem(),
loadavg: os.loadavg(),
username: commanda,
password: commandb,
key: commandc,
lev: commandd
};
```

메이컨트

클라이언트의 정보
값 비교를 위한
입력 배열

클라이언트의 PC
정보를 수신

요청 시 PC정보를 같이 받아와서 클라이언트 PC파악에 유리함

12



개발 환경 및 시스템 개발(7/8)

인증 소스코드(인증서버)

```
socket.on('example message 2', function (data) {
  // 인증 서버
  connection.connect(function (err) { // 디비 연결
    connection.query("select * from users where username = ?", [info[0].username], function (err, result) {
      if (err) throw err;
      if (md5(info[0].password) === result[0].password) { // 비밀번호 비교
        if (info[0].key === result[0].authkey) { // auth key 비교
          console.log("auth success");
          connection.query("select * from servers where username = ?", [info[0].username], function (err, result) {
            if (err) throw err;
            if (result.length === 0) {
              connection.query("insert into servers (username, ip, mac, hostname, platform, permission) values (?, ?, ?, ?, ?, ?)", [info[0].username, info[0].ip, info[0].mac, info[0].hostname, info[0].platform, info[0].permission], function (err, result) {
                if (err) throw err;
              }
            }
          }
        }
      }
    }
  });
  const text = info[0].ip + " " + info[0].lev;
  fs.writeFileSync("target.txt", '\u000a' + text, {encoding: 'utf8'}); // 후후 방화벽 등록을 위해서 텍스트파일로 정보를 저장
  exit('insert db success');
});
```

클라이언트 정보 비교 확인 후, 승인 대기열에 추가하고 추가 정보를 파일로 기록

13



개발 환경 및 시스템 개발(8/8)

방화벽 등록(권한별 자동 등록)

```
if [ 4 = $autha ]
then
echo "4 insert" > "date +%H-%M-%S".txt
fi

if [ 2 = $autha ]
then
echo "2 insert" > "date +%H-%M-%S".txt
iptables -t nat -A POSTROUTING -s IP -o ens33 -j MASQUERADE
sshpas -p 'pw' scp -o StrictHostKeyChecking=no /home/kim/desktop/agent/target.txt mail@192.168.1.1
sshpas -p 'pw' scp -o StrictHostKeyChecking=no /home/kim/desktop/agent/target.txt dns@192.168.1.1
fi

if [ 1 = $autha ]
then
echo "1 insert" > "date +%H-%M-%S".txt
iptables -t nat -A POSTROUTING -s IP -o ens33 -j MASQUERADE
sshpas -p 'pw' scp -o StrictHostKeyChecking=no /home/kim/desktop/agent/target.txt web@192.168.1.1
sshpas -p 'pw' scp -o StrictHostKeyChecking=no /home/kim/desktop/agent/target.txt dns@192.168.1.1
sshpas -p 'pw' scp -o StrictHostKeyChecking=no /home/kim/desktop/agent/target.txt mail@192.168.1.1
fi
```

인증페이지에서 관리자가 권한 체크 후 승인을 하면, 권한 별로 자동 방화벽 등록

14



개발 결과 및 시스템운영(3/6)

메인 홈페이지

8층에서 살아남기

나의 EAM 관리 | 관제 | 해당 로그 | 원격 서비스 | * 180321 일

EAM방식을 활용한 인증/관제시스템

EAM을 활용하였다.
4000건에 달하는 로그 데이터를 효율적인
관리를 제공한다.

트래픽 증가시 웹에서 그
레프트로 보거나 쉽게 관제 가
능하다.

해킹로그를 볼 수 있다.
이전과 비교 로그를 확인 할 수 있다.

접속한 장비가 없습니다.

클라이언트의 메인 홈페이지

17



개발 결과 및 시스템운영(4/6)

메인 홈페이지(관제)



관제 메뉴에서 자신의 트래픽 유입량을 확인

18



개발 결과 및 시스템운영(5/6)

메인 홈페이지(해킹로그)

로그에서 살펴보기						유해트래픽 정보 확인
ID	발착시간	프로토콜	출발지	도착지	원격 서비스	원격 서비스
1	05/13/2018-22:25:21.698802	UDP	192.168.233.2:53	192.168.233.148:33715	SURICATA DNS Unsolicited response	
2	05/13/2018-22:39:09.564876	UDP	168.126.63.1:53	192.168.233.148:49169	SURICATA DNS Unsolicited response	
3	05/13/2018-23:12:27.585699	UDP	168.126.63.1:53	192.168.233.148:40729	SURICATA DNS Unsolicited response	
4	05/13/2018-23:47:39.748349	TCP	192.168.233.148:35662	112.175.50.171:80	kimusgyoon rule	
5	05/13/2018-23:47:39.798710	TCP	192.168.233.148:35662	112.175.50.171:80	kimusgyoon rule	

해킹 로그 메뉴에서 자신에게 유입된 유해트래픽 정보를 확인

19



개발 결과 및 시스템운영(6/6)

메인 홈페이지(SSH)

로그에서 살펴보기						SSH 원격 서비스
<pre> ubuntu login: kim Password: Last login: Thu Sep 13 22:26:03 PDT 2018 from 10.100.114.209 on pts/18 Welcome to Ubuntu 16.04.4 LTS (GNU/Linux 4.15.0-34-generic x86_64) * Documentation: https://help.ubuntu.com * Management: https://landscape.canonical.com * Support: https://ubuntu.com/advantage 145 packages can be updated. 0 updates are security updates. kim@ubuntu:~\$ </pre>						

원격 서비스 메뉴를 이용하여 쉽고 빠르게 자신의 서버에 접근가능

20



결론 및 기대효과

○ EAM 인증방식의 통합관제 시스템 개발 성과

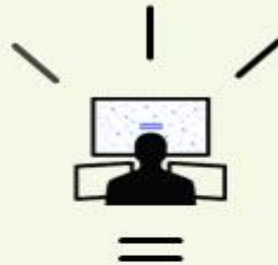
- 인가된 각 서버마다 유입 트래픽을 탐지/분석하고 탐지된 유해 IP를 방화벽에 등록/관제하여 안전한 네트워크 환경을 제공
- SSO 방식의 인증체계를 구현하여 클라이언트의 작업 편리성을 높이고 차등적 권한관리가 가능함을 확인



○ 기대 효과

- 시스템 개발과정에서 팀워크를 최대한 발휘함으로써 개인 역량보다 조직 역량을 힘을 직접 체험
- 특히 모든 조원들이 임무를 분담하여 인증체계와 웹서버 등을 구현, 실무 능력 향상시키는 계기

21



Q & A

Thank you

22