OpenSER Admin Course

Voice System SRL

http://www.voice-system.ro

http://www.openser.org

Access Control Lists

Access Control List (ACLs)s

- help implementing authorization mechanisms
- it is very important to be fast and reliable, being the way to allow the access to resources in the system
- have in mind the provisioning system, ACL update should apply in real-time
- having a well-designed ACL system can be extended to be used as a user capability list

- openser capabilities for ACLs
  - group membership
  - binary acl
  - string acl
  - custom acl

ACL – Group Membership

- unix user authorization mechanism
- group module
  - SQL database backend
  - RADIUS backend
  - same model can be implemented in the script with LDAP backend (in openser 1.3.x)
- for each user is kept list of groups it belongs to
- easy to provision
- drawback: backend (e.g., database) access every time
  - for SQL backend optimization can be achieved by loading all the group names in AVPs and check membership in script
  - usage of avp_db_query() and script comparison

ACL – Group Membership

- example of usage: group module with SQL backend

```
loadmodule “group.so”
modparam("group",
    "db_url","mysql://openser:openserrw@localhost/openser")
....
if (method=="INVITE") {
      if (uri=~"sip:00[1-9][0-9]+@.*") {
            if (!is_user_in("From", "international")) {
                  sl_send_reply("403", "No permission for internaitonal calls");
                  exit;
            }
      }
}
```

ACL – Group Membership

- prototype
  - is_user_in(user_id, group_name)

- parameters
  - user_id – part of SIP message or variable to be used to get username and domain (optionally) of the user to be checked:
    - Request-URI – Request-URI.
    - To - Use To URI
    - From - Use From URI
    - Credentials - Use digest credentials username.
    - $avp(name) - Use the URI from the AVP
  - group_name – name of the group to check.

ACL – Binary ACL

- usage of integer value to store it
- each bit set has a meaning for ACL
- example:
  - fist bit set: user account disabled
  - second bit set: user allowed do VoIP calls
  - third bit set: user allowed to do PSTN national calls
  - fourth bit set: user allowed to do PSTN international calls
  - ...
- very fast to check via script operations
- compact in storage, if 32 values are not enough, an extra integer value can be used
- mechanisms to load the value from backend:
  - avpops, avp_radius, ldap

- example of usage: avpops module with SQL backend

```
avp_aliases=”acl_binary=i:11”
....
loadmodule “avpops.so”
modparam("avpops",
    "db_url","mysql://openser:openserrw@localhost/openser")
modparam("avpops", "avp_table","usr_preferences")
....
avp_db_query(“select acl_binary from subscriber where username='$au'”,
      “$avp(acl_binary)”);
```

```
if ( [ $avp(acl_binary) & 1 ] ) {
        sl_send_reply("403", "No permission – account suspended");
        exit;
}
....
if (method=="INVITE") {
        if (uri=~"sip:00[1-9][0-9]+@.*") {
                if ( ! [ $avp(acl_binary) & 8 ] ) {
                        sl_send_reply("403", "No permission for international calls");
                        exit;
                }
        }
}
```

ACL – String ACL

- usage of string value to store it
- each character or group of characters has a meaning for ACL
- example – the position does not matter - the character is unique:
  - 'd' exists in the string: user account disabled
  - 'v' exists in the string: user allowed do VoIP calls
  - 'n' exists in the string: user allowed to do PSTN national calls
  - 'i' exists in the string: user allowed to do PSTN international calls
  - ...
- pretty fast to check via script operations
- still compact in storage
- by using position in string, incremental values can be used
- mechanisms to load the value from backend:
  - avpops, avp_radius, ldap

- example of usage: avpops module with SQL backend

```
avp_aliases=”acl_string=i:22”
....
loadmodule “avpops.so”
modparam("avpops",
    "db_url","mysql://openser:openserrw@localhost/openser")
modparam("avpops", "avp_table","usr_preferences")
....
avp_db_query(“select acl_string from subscriber where username='$au'”,
      “$avp(acl_string)”);
```

```
if ( $avp(acl_string) =~ “d” ) {
        sl_send_reply("403", "No permission – account suspended");
        exit;
}
....
if (method=="INVITE") {
        if (uri=~"sip:00[1-9][0-9]+@.*") {
                if ( $avp(acl_string) !~ “i” ) {
                        sl_send_reply("403", "No permission for international calls");
                        exit;
                }
        }
}
```

- another example: using fixed positions in the string for certain capabilities.
- same value can be at different positions with different meaning now

- example:
  - first character in the string is a digit and represents the access level
    - '0': user account disabled
    - '1': user allowed do VoIP calls
    - '2': user allowed to do PSTN national calls
    - '3': user allowed to do PSTN international calls
    - ...
  - convert to int do number comparison

```
$avp(s:acl_access_level) = $(avp(acl_string){s.substr,0,1}{s.int})
if ( $avp(s:acl_access_level) == 0) {
      sl_send_reply("403", "No permission – account suspended");
      exit;
}
....
if (method=="INVITE") {
      if (uri=~"sip:00[1-9][0-9]+@.*") {
            if ( $avp(s:acl_access_level) <  3 ) {
                  sl_send_reply("403", "No permission for international calls");
                  exit;
            }
      }
}
```

- use power of script language
  - combine above group membership, binary and string ACLs
  - custom mechanisms

- examples
  - use the source IP address to grant some privileges
  - allow routing to specific resources to everybody during certain intervals of time
  - user ID can be allocated to suggest some privileges

```
# deny calls from a certain IP (flooding us)
if ( src_ip == 10.10.10.10) {
        sl_send_reply("403", "No permission – stop flooding");
        exit;
}
....
# allow myself only for international
if (method=="INVITE") {
        if (uri=~"sip:00[1-9][0-9]+@.*") {
                if ( $from.user != “daniel” ) {
                        sl_send_reply("403", "No permission for international calls");
                        exit;
                }
        }
}
```

```
# everybody can call international between 18:00 and 22:00
# $Tf – current time: Wed Sep 26 10:46:09 EEST 2007
$var(hour) = $(Tf{s.substr,11,2});
....

if (method=="INVITE") {
      if (uri=~"sip:00[1-9][0-9]+@.*") {
            if ( $var(hour) < “18” || $var(hour) >=  “22” ) {
                  sl_send_reply("403", "No permission for international calls");
                  exit;
            }
      }
}
```

BREAK

- remember – openser is SIP-only application
- cannot be a gateway to PSTN
- additional components in the platform
  - owned SIP-PSTN gateway
  - using wholesale SIP-PSTN termination providers
- basically routing to PSTN is a matter of defined dialing plan
- it is recommended to have a clear way to sort local and PSTN destinations, avoiding as much as possible the DB hit
- you have to 'trust' calls from PSTN when targeting local users
- openser offers support for
  - list cost routig: lcr module
  - load balancing: dispatcher module
  - gateway failover

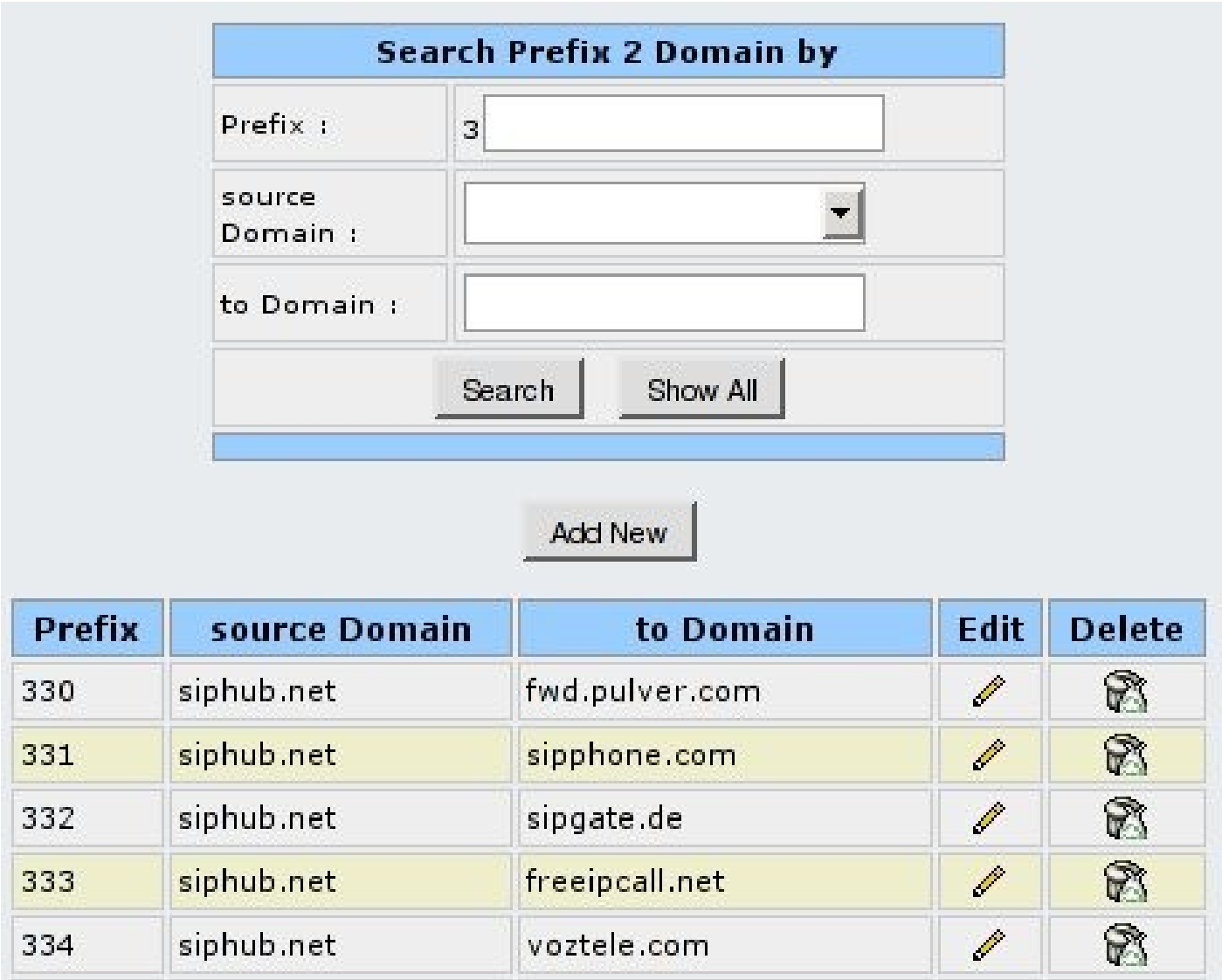
- example of routing logic to PSTN
  - very important are the authorization mechanisms
  - it is German based numbering plan
  - our users are in the range of 493012340000 to 4930123459999
  - service is located in Berlin
- destination of calls are accepted in national format or international format
- access levels are
  - voip – user can call other local users
  - national – user can call within Germany
  - international – user can call anywhere else
- we have two gateways
  - one for national calls
  - one of the other calls

walk through config file

- routing via DNS
  - no add on
  - before forwarding, DNS query will resolve to right IP
- routing via prefixes
  - script embedded

```
if($ruri.user=~"^12") {
        strip(2);
        rewritehostport("foreign.com");
        t_relay();
        exit;
}
```

  - pdt module – prefix to domain
    - use database to load prefixes and destinations
    - cache mode
    - ability to add new records at run-time



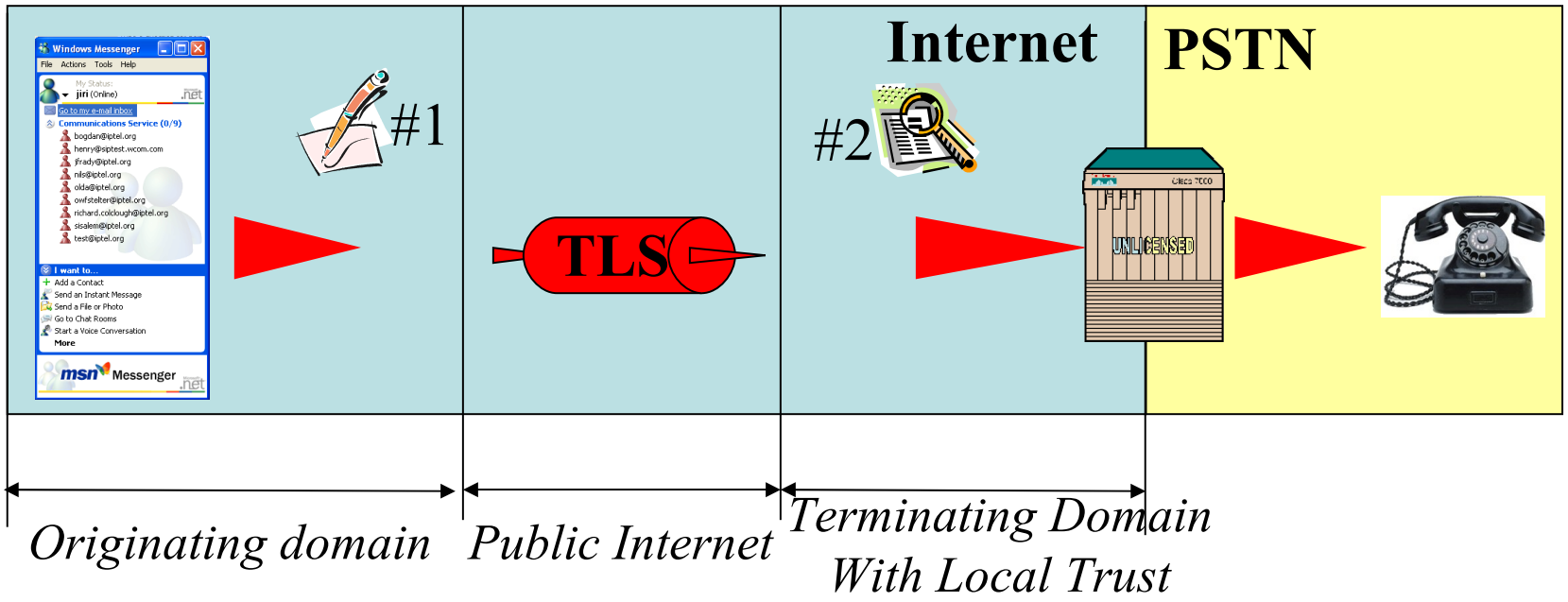
Prefix	source Domain	to Domain	Edit	Delete
330	siphub.net	fwd.pulver.com		
331	siphub.net	sipphone.com		
332	siphub.net	sipgate.de		
333	siphub.net	freeipcall.net		
334	siphub.net	voztele.com		

- trusted relationships
  - IP based
    - set of IP addresses
    - netmask
    - SIP authentication
  - TLS
    - certificates
    - server-to-server authentication
  - IPSEC
    - encrypted link

Trust: Interdomain versus Intradomain

- Within single administrative domain, trust can be implemented using physical security and knowledge of identity of local users – proxy servers verify identity of local users using digest and gateways trust local proxies.
- Interdomain scenario example: siphub.net users terminate calls to US PSTN with National Gateways Inc. How do you export the trust then?
  - The terminating provider can't verify identity of remote users and can't trust information passed over the public Internet. RPID alone can't be trusted as it can be changed anywhere on the transit. Stronger security protocols come in for interdomain operation: TLS.

TLS Use for Interdomain Security



- Assumption: target domain trusts source domain to display proper CallerID and settle incurred costs.
- Step 1: originating domain verifies identity of local user (digest). If OK, it appends RPID and uses TLS for secure inter-domain communication.
- Step 2: terminating proxy verifies incoming TLS connection against list of trustworthy domains. If ok, SIP request is forwarded to PSTN gateway.

- IP authentication for known hosts
- check From user and authentication user to prevent spam
- don't allow not-authenticated anonymous calls, only authenticated requests from local users allowed
- don't allow direct access to critical resources (gateways, media servers)
- anti-flood detection via pike
- calls from foreign domain should be always considered untrusted unless via secure channel (TLS)

break