

# JCU NetAccess

**In late 2008, James Cook University decided to replace their out-of-date Internet Access system.**

**After product investigation, discussion and site referee's reports, JCU decided to commission Obsidian Consulting Group to install their Je tproduct to provide a managed Internet service.**

**The new service would be known to users as the NetAccess system.**

## Aims

- **Manage the University's Internet bill**
- **Continue to use differential charging to allow unrestricted traffic for high volume research**
- **Restrict activities of no more than 5-10% of students**
- **Authentication**
  - **No client installation required on user's workstation**
  - **Pre-authenticate users where possible**
  - **Transparent redirection to authenticate the balance**

# Jet Traffic Management & Billing

The Jet Billing system is an Internet and Telephony management system.

It supports a number of edge devices, such as the Procera PacketLogic, Cisco Service Control Engine and the Bluecoat devices.

It can source it's account management from a number of systems, such as LDAP and Novell.

## Implementation

The standard Jet install uses a number of methods to allocate traffic from IP addresses to users or groups:

- IP Manager
- Popup
- Novell (not used at JCU)

The installation at JCU took the base Jet functionality and extended the functionality to support a number of pre-authentication methods:

- Microsoft Active Directory
- 802.1x / Radius / eduroam

# IP Manager

Managed internally within Jet.

Allows servers or non-compliant devices to gain access or be accounted for:

user_id	ipaddr	macaddr	provider	start_time	last_seen
ccjsc	137.219.19.24		IPManager		

# Popup

Unauthenticated users attempting to access standard web resources off-campus are intercepted by the PacketLogic and re-directed to the Jet server on port 80.

This page then redirects again to an SSL protected login page.

All other traffic, such as non-web (port 80) and traffic to known problem external hosts are not redirected, and are instead dropped in the bit bucket.

# Problem traffic

The volume of problem traffic was so high that we found it overwhelmed Jet's standard configuration.

While other Jet sites had already experienced this problem for various reasons they did not wish to share their fixes. We had to write our own.

Reconfiguring the initial page from an Apache redirect to JavaScript, reduced the system load dramatically.

# Findings

What application classes caused the problems:

1. Operating System updates
2. Applications
3. Application updates
4. Malware
5. Antivirus

## Popup 2

Once users correctly provide a username and password, Jet moves the User's IP into an access category based on their status (staff, student, no quota).

A popup window is opened to both inform the user and maintain their session via a periodic refresh.

## Popup 3

After a small amount of time the page originally requested is returned.

The Popup expires after a preset time since the most recent last\_seen time.

At JCU, our popup reaper process is set to run at 10 minute intervals.

This handles logout, closure of the browser and machine failures.

# Brainstorming

To determine a way to tie a user to an IP for a period of time.

What processes already have this information:

- Users on our central Active Directory
- Users on our 802.1x wireless (JCU & Eduroam)

# Microsoft Active Directory

Aim: no client installed on user's machine

Workstations login to an AD domain.

Kerberos ticket is created or updated on the Domain Controller.

These are the useful events for this process.

# MS-AD 2 - Events

Event ID	Account Logon Event Type	Event Information
672	Success audit Windows 2000 Windows Server 2003 Failure audit Windows Server 2003	Authentication Ticket Request: User Name Supplied Realm Name Service Name Client Address
673	Success audit Windows 2000 Windows Server 2003 Failure audit Windows Server 2003	Service Ticket Request: User Name User Domain Service Name Client Address

<http://technet.microsoft.com/en-us/library/cc738673.aspx>

# MS-AD 3 - Event Translator

This new feature allows an administrator to specify SNMP events to be translated as SNMP traps. The frequency of event translation can also be specified, along with log file options.

A command line tool, Evntcmd.exe, or a user interface, Evntwin.exe, can be used for configuration. Both files, along with the event translator Evntagnt.dll, are created in the % SystemRoot %\system32 directory when the SNMP service is installed, and can be launched through a Windows 2000 command window.

The event translator uses the SNMP service to generate the trap. By default, no events are translated. For information about how to use and configure this utility, see the SNMP online documentation.

<http://technet.microsoft.com/en-us/library/cc959662.aspx>

# MS-AD 4 – SNMP Traps

Event log set to send SNMP trap on specific events:

- 672 Kerberos TGT creation
- 673 Ticket Service Request

SNMP traps sent to Jet *snmptrapd.py*

*snmptrapd.py* daemon then sends these updates to Jet's login process *tradius.py*

user_id	ipaddr	macaddr	provider	start_time	last_seen
ccjsc	137.219.19.240		ActiveDirectory	Jul 09 08:45:18	Jul 09 16:15:21

# MS-AD 5 - Kerberos Policy

## Maximum lifetime for service ticket

Determines the maximum amount of time (in minutes) that a ticket granted for a service (that is, a session ticket) can be used to access the service. If the setting is zero minutes, the ticket never expires. Otherwise, the setting must be greater than ten minutes and less than the setting for Maximum lifetime for user ticket. By default, the setting is 600 minutes (10 hours).

## Maximum lifetime for user ticket

Determines the maximum amount of time (in hours) that a user's TGT can be used. When a user's TGT expires, a new one must be requested or the existing one must be renewed. By default, the setting is ten hours.

## Maximum lifetime for user ticket renewal

Determines the longest period of time (in days) that a TGT can be used if it is repeatedly renewed. By default, the setting is seven days.

<http://technet.microsoft.com/en-us/library/cc738673.aspx>

# MS-AD 6 - Implementation

At JCU, our MS-AD reaper process is set to run every hour, watching for **now() - last\_seen** times greater than 10 hours.

## 'Pure' Radius

Jet supports normal radius with User-Name and Calling-Station-Id fields:

```
Wed Jun 10 10:08:06 2009
  User-Name = "jc123456"
  NAS-Port = 28847
  Service-Type = Framed-User
  Framed-Protocol = PPP
  Framed-IP-Address = 137.219.12.103
  Class = 0x00
  Calling-Station-Id = "123.45.67.89"
  Acct-Status-Type = Start
  Acct-Session-Id = "abcdef"
  Tunnel-Client-Endpoint:0 = "123.45.67.89"
  Acct-Authentic = RADIUS
  Acct-Delay-Time = 0
  NAS-IP-Address = 137.219.12.10
  NAS-Port-Type = Virtual
  Client-IP-Address = 137.219.12.10
  Acct-Unique-Session-Id = "abcdef"
  Timestamp = 1244592486
```

# Radius 2

Simply fill in normal details:

user_id	ipaddr	macaddr	provider	start_time	last seen
jc123456	137.219.12.103		radius	Jul 09 10:08:06	Jul 09 10:08:06

## 802.1x wireless & eduroam

802.1x packets deal with authenticating the user.

They can provide VLAN steering based on user attributes.

They generally do not allocate IP addresses.

Set an accounting proxy on the RADIUS server to forward accounting records to Jet.

# 802.1x packet

## Accounting Start Record

Wed Jun 10 09:20:03 2009

Called-Station-Id = "0012.da94.1f22"  
Calling-Station-Id = "0022.5f18.d7fd"  
Cisco-AVPair = "ssid=walkabout"  
Cisco-AVPair = "vlan-id=988"  
Cisco-AVPair = "nas-location=Student Mall"  
WISPr-Location-Name = "Student Mall"  
User-Name = "jc234567"  
Cisco-AVPair = "connect-progress=Call Up"  
Acct-Authentic = RADIUS  
Acct-Status-Type = Start  
NAS-Port-Type = Wireless-802.11  
NAS-Port = 2881  
Service-Type = Framed-User  
NAS-IP-Address = 172.26.2.10  
Acct-Delay-Time = 15  
Client-IP-Address = 172.26.2.10  
Freeradius-Proxied-To = 137.219.2.40  
Timestamp = 1244589638

## 802.1x 3 – Radius Details



Fill in the details:

user_id	ipaddr	macaddr	provider	start_time	last_seen
jc234567		00:22:5f:18:d7:fd	radius	Jul 09 10:20:03	Jul 09 10:20:03

Well, that didn't work so well ...

# 802.1x 4 – Finding an IP

Once a machine is authenticated, the AP or controller steers it into a VLAN.

Then the machine will negotiate an IP address via DHCP on that VLAN.

We run standard Internet Systems Consortium DHCP Servers.

Normally, hosts on campus are allocated a normal dynamic hostname simply constructed from the IP address.

The trick here is to update a DNS zone which does the mapping to join the MAC address to the IP address of the client.

## dhcpcd.conf

```
ddns-updates off;
on commit {
    set new-ddns-fwd-name = concat(
        binary-to-ascii(16, 8, ":", hardware), ".",
        pick(config-option server.ddns-domainname, config-option domain-name));
    switch(ns-update(delete(IN, A, new-ddns-fwd-name, null),
        add(IN, A, new-ddns-fwd-name, leased-address, lease-time / 2))) {
    case NOERROR:
        unset ddns-fwd-name;
        on expiry or release;
    }
    unset new-ddns-fwd-name;
}
```

# named.conf

We runs the BIND DNS server, so it has the ability to send a DNS\_NOTIFY to another server which is listed in it's configuration.

```
zone "translation.table.mac" IN {  
    type master;  
    file "master/translation.table.mac";  
    allow-update { key DHCP_UPDATER; key HOSTS; };  
    notify explicit;  
    also-notify { jet.jcu.edu.au port 1053; };  
};
```

## 802.1x 7 - DNS\_NOTIFY

The Jet server runs a *dnssd.py* listener process on port 1053, which only acts on messages of type DNS\_NOTIFY.

It then runs down the list of users with a MAC address but no IP address and attempts to resolve the hostname constructed from the MAC address.

# 802.1x 8 - Looking up a host

user_id	ipaddr	macaddr	provider	start_time	last_seen
jc234567		00:22:5f:18:d7:fd	eduroam	Jul 09 10:20:03	Jul 09 10:20:03

```
$ host 1:0:22:5f:18:d7:fd.translation.table.mac
```

```
1:0:22:5f:18:d7:fd.translation.table.mac has address  
137.219.189.158
```

user_id	ipaddr	macaddr	provider	start_time	last_seen
jc234567	137.219.189.158	00:22:5f:18:d7:fd	eduroam	Jul 09 10:20:03	Jul 09 10:20:03

So now we have a user `jc234567` and an IP address `137.219.189.158`, so the `dnsd.py` then runs further processes to log the user in.

# 802.1x 9 - Implementation

Radius STOP records from the 802.1x server presents the MAC address again, so `tradius.py` simply matches this from the internal table and logs the user out.

At JCU, we expire MAC address only records that are older than 5 minutes.

# Results

Authentication load is spread using different methods

Usage within bounds

Costing of traffic on the PacketLogic

## Typical usage:

Provider	Number of Users
ActiveDirectory	1048
eduRoam	703
IPManager	10
popup	1085

# Quota

Off-Net usage is charged at \$5 per gig (inc GST)

Staff and Postgrads: \$24 per year

allocated yearly

reset yearly

Students: \$24 per year,

allocated monthly

reset monthly

# Usage patterns

Student Usage %	February	March	April	May	June
no usage	91	57	58	57	68
0	7	25	19	17	17
10	1	4	5	5	3
20	0	3	3	3	2
30	0	2	2	2	1
40	0	1	1	2	1
50 – 90	0	5	5	5	5
100	0	5	7	9	5

# Costing traffic on the PL

Use outbound route-maps on border router to add an extra (private) AS-path to the PacketLogic peer, based on AARNet charging community, eg: 7575:1000 => 65000

Send this to the PL, eg:

137.219.0.0/16 65000 7575

Make a charging or shaping rule based on the presence or otherwise of the private AS

65000 => On-Net => Free

! 65000 => Off-Net => Charged

# Thanks

## **Microsoft Active Directory**

Adrian Tarca

## **Eduroam**

Jeffrey Bird

<http://www.cesnet.cz/doc/techzpravy/2007/eduroam-accounting/>

## **Obsidian**

Simon Hookway

Mark Dawes

# JCU NetAccess

**Questions?**