# Summary

## Meeting time and location

Madrid conference room @ Advance

## Participants

* Advance
  + Roelof
  + Loutjie
* Dark nITes
  + Ruslynn
  + Jeandre
  + Muhammed
  + Sisa
  + Christo

## Decisions/Amendments made/adopted

* Logging system will form part of the system. A log entry will include *time*, *username*, and the *action taken*
* User accounts/Log ins will be included as a security feature
* IPv6 support not to be added now; peg as a possible future feature
* Implementation of “unique algorithm per pool” requirement to be added later

# Overview

The purview of this meeting was to touch base and reconcile our progress with the client’s expectations.

## User interface

* Interface may not need to be exposed outside the working service
* User interface should allow the manipulation of specifics in a pool
  + Process should include the listing of all pools -> select pool -> manipulate pool instances

## Security

* Log-in system will be added to add a layer of security
* Implement two user roles; Admin and User
  + Admin has read and write privileges
  + User has read privileges
* [Logging system](#_Decisions/Amendments_made/adopted) will also be included
* A notifications system alerting of attacks must also be included

## Logging system

* Store date/time, username and description of action taken
* No specific database has been specified; can use whichever we want

## Unit testing

* As root, we can make raw packets therefore specify the source IP. This can be used a strategy to test if blacklisting/packet-dropping works
* Packet generators (NPing for example) can also be used to test packet-dropping/load-balancing functionality
* Unix **nc** (netcat) command can also be used to read/write from/to network connections. This is another tool we can adapt for testing. Note: the IP addresses to be used must be resident on the machine

## Integration testing

* Integration points will determine the type of testing suites to use.

## IPv6

* IPv6 support need not be added at current, however may be looked at in future to add

## Styling

* Final application styling can follow styling of Advance webpage, or styling of tender document

## Algorithm switching requirement

* At current the system does not meet the “unique algorithm per pool” requirement; this will be implemented at a later stage

## Dynamic blacklisting of IPs strategy

* The maps used to store blacklisted IPs may be purged on a timely basis, i.e. we do not need to keep an IP if it has not committed offences greater than the stipulated restrictions.

## Standard service registration (registration of back-end instances)

The following software packages are distributed configuration services. Essentially they will help use keep track of which services we are protecting

* Apache Zookeeper
  + Synchronises services across an Apache Hadoop cluster (clusters can be seen as the pools in which the instances reside)
* Eureka
  + Netflix’s service registry, flagged to have a good UI
* ETCD
  + Written in Go, addresses the shortfalls of Zookeeper

# To-do

* Implement user interface
* Add *sudo* commands to run scripts to automatically run elevated
* Investigate the testing interface included in the **xdp-tutorial** repo