ISTITUTO NAZIONALE DI FISICA NUCLEARE

Sezione di Lecce

INFN/code-xx/xxx
dd/mm/yy

CCR-31/2009/P

INFN-AAI TDR: CORE SERVICES

AAI-WG

Abstract

Il Disegno Concettuale del progetto INFN-AAI (CDR) è stato approvato dalla Commissione Calcolo e Reti, nella seduta del giugno del 2008.

Contrariamente a quanto previsto dal CDR di INFN-AAI, e principalmente a causa della mancanza di man-power, si è deciso di dividere il Documento di Disegno Tecnico (TDR) in vari moduli. In questo modulo del TDR di INFN-AAI saranno analizzati e descritti tutti i principali aspetti tecnici relativi alla implementazione ed alla gestione del nucleo di INFN-AAI. In altri moduli saranno affrontati gli aspetti tecnici relativi ai dettagli delle integrazioni dei sistemi di Autenticazione ed Autorizzazione delle varie sedi dentro la INFN-AAI.

Verrà infine analizzato un possibile piano di implementazione.

Pubblicato da **SIS–Pubblicazioni**
Laboratori Nazionali di Frascati

AAI-WG

Silvia Arezzini[1], Claudio Bisegni[2], Luca Carbone[3], Enrico M.V. Fasanelli[4], Roberto Lulli[5], Dael Maselli[2], Fulvio Ricciardi[4], Stefano Stalio[6], Francesco M. Taurino[7], Emanuele Turella[2]

[1)]INFN-Sezione di Pisa, Largo B. Pontecorvo, 3, I-561270 Pisa, Italy
[2)]INFN-Laboratori Nazionali di Frascati Via E. Fermi 40, I-00044 Frascati, Italy
[3)]INFN-Sezione di Milano Bicocca, Piazza della Scienza, 3, I-20126 Milano, Italy
[4)]INFN-Sezione di Lecce, Via prov.le per Arnesano, I-73100 Lecce, Italy
[5)]INFN-Sezione di Roma2, Via Della Ricerca Scientifica, 1, I-00133 Roma, Italy
[6)]INFN-LNGS, S.S. 17 BIS km. 18.910, I-67010 Assergi, Italy
[7)]CNR/INFM ed INFN-Sezione di Napoli, Via Cintia, I-80126 Napoli, Italy

INDICE

1 INTRODUZIONE

Il comitato di revisione del CDR di INFN-AAI ha prodotto un documento nel quale ha indicato un certo numero di richieste e suggerimenti sia per il progetto INFN-AAI che per la Commissione Calcolo e Reti. Alcune richieste sono state rapidamente raccolte ed il risultato è stato integrato nella versione definitiva del CDR. Altre, che richiedevano un maggior lavoro di analisi e studio, sono state valutate in seguito ed hanno prodotto alcune variazioni rispetto a quanto definito nel CDR.

In questo capitolo introduttivo descriveremo brevemente tali variazioni, mentre nei successivi capitoli saranno esposti i dettagli tecnici relativi alla implementazione dei servizi che costituiranno il nucleo di INFN-AAI: i *"Core Services"*.

Infatti, a differenza di quanto precedentemente affermato sia nel CDR che nelle varie sedi istituzionali (comitato di revisione e CCR) il TDR di INFN-AAI non potrà essere un unico documento. La mancanza di personale da dedicare alle attività legate alla definizione di tutte le specifiche tecniche per l'implementazione di INFN-AAI nelle varie sedi, ci ha di fatto obbligati a dividere il TDR in moduli che fossero il più indipendenti possibile tra essi.

1.1 Architettura di INFN-AAI

L'architettura della INFN-AAI, come definita nel CDR, è funzionale alla necessità di unificare due categorie di esigenze ben precise: quelle relative agli accessi ai servizi centralizzati (forniti ad esempio da DataWeb, ma non solo) e quelle relative agli accessi ai sistemi informatici locali delle sedi INFN.

I rilievi del comitato di revisione del CDR hanno imposto una valutazione approfondita di due aspetti fondamentali dell'architettura della INFN-AAI: il sistema di gestione delle Identità e dei relativi diritti di accesso alle risorse (Identity and Access Management: IAM) ed il disegno dell'albero LDAP (Directory Information Tree: DIT).

Questa valutazione ha prodotto sia una variazione dell'architettura globale di INFN-AAI, che adesso comprende anche il sistema di IAM oltre all'Autenticazione ed Autorizzazione, sia ad una nuova struttura del DIT.

1.1.1 Identity and Access Management: IAM

Uno dei rilievi mossi al CDR da parte del comitato di revisori nominato allo scopo è stato quello di aver avuto una visione troppo ristretta del bacino di utenza interessato dall'operazione. Sostanzialmente è stato fatto notare che gli utenti della INFN-AAI sarebbero stati non tutti i fruitori di servizi informatici dell'ente, ma solo un sottoinsieme: quello degli utenti più facilmente raggiungibili. E sarebbe comunque rimasto un doppio e forse triplo sistema di archiviazione dei dati utente (Autenticazione) ed almeno altrettanti sistemi di Autorizzazione in base ai servizi da offrire.

Per poter unificare tutti i sistemi di Autenticazione ed Autorizzazione è cioè necessario avere anche un unico Sistema di Gestione delle Identità o Identity and Access Management (IAM).

Per la realizzazione di un IAM occorrono tre elementi fondamentali che sono intimamente correlati tra loro:

- definire il DB di riferimento (o i DB di riferimento perché possono essere più di uno, ma in questo caso occorre stabilire i criteri di interrelazione tra i più);
- definire un sistema preciso per l'inserimento e la gestione dei dati nei DB di riferimento e fornire uno strumento per tale scopo (che sia facile da usare);
- definire i processi (o workflow) di gestione delle identità e delle assegnazioni di diritti di accesso alle varie risorse (informatiche e non).

È evidente che mentre i primi due aspetti sono principalmente tecnici, l'ultimo ha un carattere più legale o amministrativo. Per questo la Commissione Calcolo e Reti ha promosso la costituzione di un apposito gruppo di lavoro, che definirà un procedimento unico di gestione delle Identità, compatibile con tutte le procedure di identificazione degli utenti che sono in essere nelle varie sedi INFN, e rispondente ai requisiti legali ed amministrativi previsti per tali procedure. Tale procedimento o workflow, dovrà quindi essere "inserito" nella implementazione del sistema di IAM.

1.1.2 Directory Information Tree: DIT

Il disegno del DIT proposto nel CDR prevedeva una struttura di tipo *"alto e magro"*, che fornisce informazioni relative alle entry in modo implicito, attraverso il DN. Una struttura di questo tipo ricalca la struttura dell'INFN ed era stata pensata anche per permette una più facile adozione da parte delle sedi.

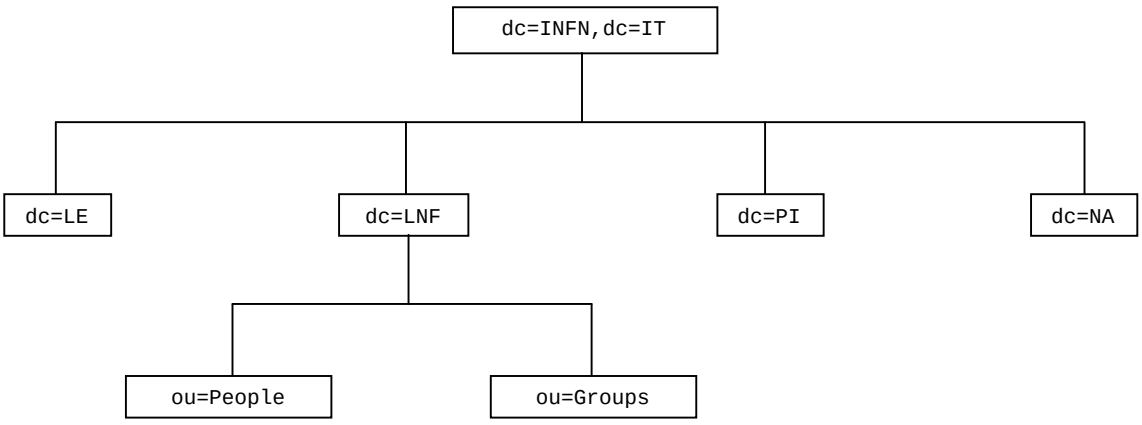


Figura 1. Il disegno originale del DIT come definito nel CDR

Una volta che si è deciso di includere il sistema di IAM all'interno del progetto INFN-

AAI, è venuto naturale accogliere il suggerimento del comitato di revisione di considerare il modello *"basso e grosso"*. In tale modello le caratteristiche delle entry sono determinate da opportuni valori di opportuni attributi, invece che dalla posizione della entry nell'albero, cosa che rende immediate alcune operazioni come ad esempio il cambio di sede di un utente.

Siccome però l'INFN continua ad avere la sua struttura divisa in varie sedi, si è deciso di adottare un modello misto in cui la parte del DIT di tipo *"basso e grosso"* contiene le informazioni legate alle persone (e quindi è la parte corrispondente al sistema di IAM) mentre la parte di tipo *"alto e magro"* contiene le informazioni relative agli account e gruppi nelle sedi.

La coerenza delle informazioni presenti nei due segmenti di albero, e le relazioni tra di esse è garantita dal sistema di gestione di IAM.

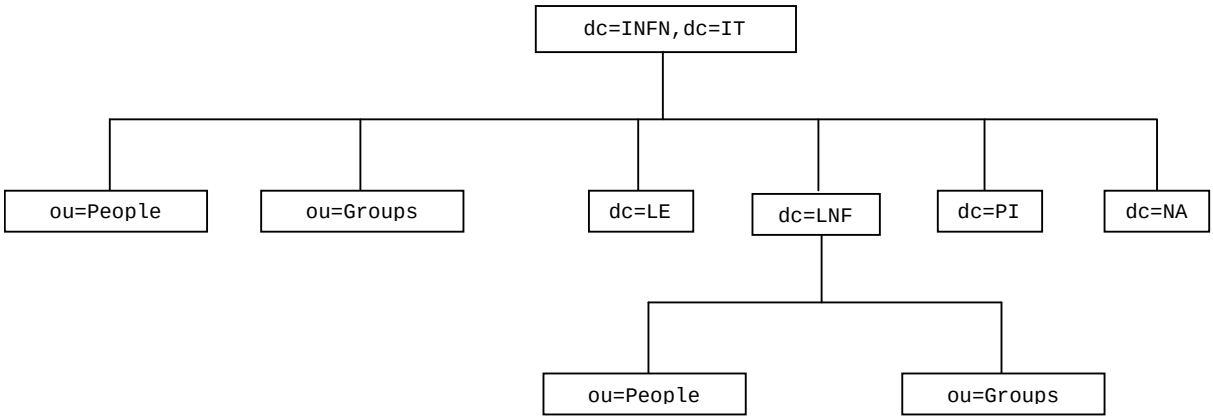


Figura 3. La struttura definitiva del DIT LDAP di INFN-AAI

I due rami **di ente** identificati da

- `ou=People, dc=INFN, dc=IT`
- `ou=Groups, dc=INFN, dc=IT`

saranno popolati e gestiti soltanto attraverso il sistema di gestione delle identità, mentre per i rami relativi alle varie sedi sarà permessa, per un numero limitato di attributi, anche la gestione diretta via query LDAP.

1.1.3 Applicazioni centralizzate

Tutti i dati utilizzabili dalle applicazioni centralizzate andranno memorizzati all'interno degli attibuti delle *entry* presenti in uno dei due rami **di ente** definiti sopra. In particolare all'interno di `ou=People` andranno inseriti i dati relativi alle persone (IAM) ma anche i dati relativi all'account che le persone useranno per l'accesso alle applicazioni centralizzate. FIXME.

2 INFN-IAM: GOVA

Il Sistema di Identity and Access Management è basato su GOVA (Gestione Ospiti Visitatori ed Associati) che è l'evoluzione del sistema GO sviluppato da Claudio Bisegni ed Antonino Passarelli dei LNF, in risposta alle richieste del decreto Pisanu.

GOVA è un sistema software a tre livelli (DB, Application Server e Client Java) pensato oltre che per la gestione delle identità di Ospiti, Visitatori ed Associati (le identità dei Dipendenti sono gestite dal sistema HR dell'Amministrazione Centrale, e saranno importate nel DB di riferimento di GOVA) anche per:

- la gestione dell'arricchimento delle identità di tutte le persone presenti nel DB, compresi i Dipendenti;
- la popolazione dei 4 server LDAP che costituiscono il nucleo di INFN-AAI ed in particolare la popolazione e gestione dei gruppi.

2.1 GOVA: Architettura software

Come descritto nel capitolo precedente, GOVA sarà un sistema a tre livelli composto da un client desktop, un Application Server che regolamenterà la lettura e scrittura dei dati fornendo servizi (WebServices) accessibili dall'esterno, e un database per la persistenza dei dati.

Il *framework* su cui sarà sviluppato è diviso principalmente in due parti **Client** e **Server**.

La parte server si basa sul concetto di **processi** che sono costituiti da una o più **azioni** su un determinato tipo di dato comune. Per esempio il processo **gestioneAnagrafica** avrà le azioni **inserisciAnagrafica**, **aggiornaAnagrafica** etc. Ogni azione, per manipolare i dati, usa una o più classi DAO (Data Access Object), che usano la connessione al back-end di persistenza assegnata dal *framework* al momento della creazione di ogni processo.

Per fare in modo che GOVA possa trattare contemporaneamente dati memorizzati su DB Oracle e dati memorizzati sul sistema LDAP di INFN-AAI, saranno sviluppate apposite classi di oggetti DAO. I processi in questo modo regolamenteranno le scritture e letture dei dati usando DB, LDAP o entrambi contemporaneamente, sfruttando una chiave di relazione comune tra i due back-end di persistenza. Questa chiave comune deve essere un identificativo univoco legato alla persona, ed a questo punto del disegno del sistema si pensa ad un identificativo del tipo UUID (Universally Unique IDentifier).

Questa suddivisione (Processo, Azione, DAO) permette di avere una più semplificata gestione delle autorizzazioni che potranno essere capillari sino al dettaglio dell'azione (vedi paragrafo successivo).

Il Client infine, per ogni processo lato server, usa una vista per permettere all'utente di usare tutte le azioni che esso mette a disposizione. Prendendo come esempio il processo **gestioneAnagrafica** sopra descritto, il Client avrà una vista che permetterà di inserire o modificare i dati anagrafici di una persona.

2.2 Gestione delle Autorizzazioni

Essendo GOVA un sistema nazionale centralizzato, la gestione delle autorizzazioni deve essere capillare e multi-dominio.

Esempio: un ufficio di segreteria di una sede X che svolge anche funzioni di segreteria per, ad esempio, una Commissione Nazionale, dovrà avere accesso completo a tutti i dati relativi alle persone della sede X, ma solo alcuni dati (in questo caso solo ai dati relativi alla Commissione Nazionale in considerazione) di alcune persone (delle sole persone che afferiscono a tale Commissione Nazionale) appartenenti a tutte le altre sedi.

L'architettura software descritta nel precedente paragrafo permette di semplificare una gestione così complessa di privilegi, usando i processi come un insieme di funzioni su una tipologia di dato e le azioni come operazioni atomiche su di esso. Per GOVA si sta pensando ad un sistema di autorizzazioni "astratto" in modo che possa essere esteso anche ad altre applicazioni, riuscendo così ad usare il sistema stesso come gestore di autorizzazioni centrale per tutte le applicazioni che si conformeranno a questa logica.

Una autorizzazione può essere espressa come l'autorità che un soggetto possiede ad eseguire una determinata azione su un dato, a volte ristretta ad alcuni domini. Il soggetto è colui che effettua l'autenticazione su un applicazione. INFN-AAI mette a disposizione tre tipi di autenticazione:

- **Username e Password (temporaneamente)**
- **Credenziali Kerberos5**
- **Certificati X.509**

Qualsiasi sia il modo di autenticazione GOVA avrà accesso al DN dell'utente autenticato, che sarà usato dall'applicazione per la ricerca delle autorizzazioni. Trovato il soggetto dobbiamo ora definire le azioni che esso può compiere.

Nel contesto dell'architettura software precedentemente descritta bisognerà descrivere l'autorizzazione di un soggetto ad eseguire un processo e una o più azioni dello stesso. Nel contesto di un azione si potrà definire anche il dettaglio (esempio: l'autorizzazione a modificare solo alcuni dati di un'anagrafica).

Uno dei metodi usati per implementare una tale granularità rispetto alle autorizzazioni è quello di definire e memorizzare nel modo opportuno gli *"Entitlements"* ossia i diritti che l'utente ha nei riguardi dell'azione che si vuole compiere.

Il supporto migliore per memorizzare informazioni di questo tipo è certamente un server LDAP in quanto tali informazioni non cambiano frequentemente, ma devono essere lette un numero elevato di volte. Ogni volta che l'appicazione dovrà effettuare una operazione per conto dell'utente.

Per questo si è definito un *"naming"* o *"namespace"* che rimanga univoco e che possa essere utilizzato come valore di attributi opportuni.

L'unicità del nome può essere garantita dall'anteporre il nome del dominio ed il nome dell'applicazione alla descrizione delle azioni e dei diritti su tali azioni. Ad esempio, il diritto ad aggiornare i numeri di telefono delle persone presenti nella anagrafica può essere definito dalla stringa:

infn.it:IAM:gestioneAnagrafica:modifica:numeroTelefono

che può anche essere formata, per restringere la validità del diritto di cui sopra, da un dominio di applicabilità codificato ad esempio come

infn.it:IAM:gestioneAnagrafica:
modifica:numeroTelefono+gid=CCR,ou=Groups,dc=INFN,dc=IT

Una tale struttura garantisce l'univocità dei nomi e la possibilità di definire il dettaglio dell'azione fino all'azione più atomica possibile.

L'implementazione di una tale struttura è quasi sempre stata effettuata attraverso la definizione di attributi opportuni e relative classi di oggetti secondo le specifiche della definizione degli OID di IANA, ossia attraverso la definizione di uno *"schema"* ad hoc per ogni applicazione.

Questo approccio però va in contrasto con la tendenza, sempre più in uso anche per le federazioni di AAI, di utilizzare gli *"Entitlements"* in modo che identifichino il diritto che un utente ha di compiere una determinata azione, indipendentemente dalla specifica applicazione attraverso la quale si tenta di compiere l'azione.

L'idea è quindi quella di usare l'attributo **eduPersonEntitlement** della classe di oggetti **eduPerson.** Il valore di tale attributo è un **URN** con formato definito, che nel momento in cui si MACE delegerà lo spazio di nomi **urn:mace:infn.it** all'INFN, avrà il formato:

urn:mace:infn.it:<iNSS>

dove **<iNSS>** è una *"Namespace Specific String"* come definita nell'RFC2141, ma non *"case sensitive"*. Nel caso di diritto di modificare il numero telefonico, tale URN diventerebbe:

urn:mace:infn.it:IAM:gestioneAnagrafica:modifica:numeroTelefono

2.3 GOVA: Architettura hardware

Come accennato, il sistema GOVA è l'evoluzione del sistema GO che si basa su un semplice DB Oracle.

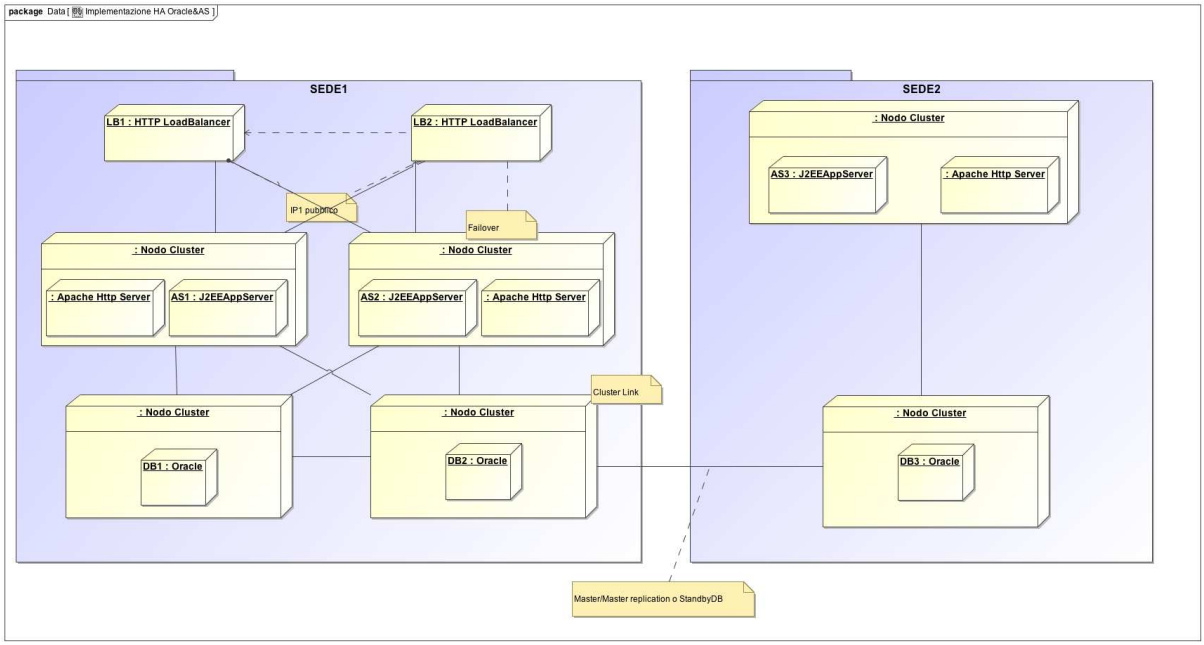
Per la configurazione finale di GOVA, data la criticità che esso avrà nei confronti delle operazioni di gestione delle Identità e degli accessi, si userà una configurazione in Real Application Cluster (Oracle RAC) tra due database server installati presso i LNF, oltre ad un server esterno di failover installato presso il CNAF.

L'accesso ai DB Oracle in configurazione RAC avverrà solo attraverso gli Application

Server che, utilizzando le funzionalità dei driver J2EE, potranno accedere in modo trasparente al DB Oracle attivo, anche in caso di indisponibilità di uno di essi.

Anche per l'accesso ai due Application Server è prevista una configurazione di *failover*. Attraverso una coppia di nodi che permetteranno il *Load Balance*.

Figura 3. Schema hardware del nucleo del sistema GOVA



Il disegno dell'architettura hardware di Figura 3 mostra anche che l'intero sistema di GOVA è configurato in modo tale che in caso di irraggiungibilità della SEDE1 (nel nostro caso i LNF) il sistema continuerà a funzionare utilizzando i server di *failover* installati nella SEDE2 (nel nostro caso il CNAF).

Le specifiche del sistema di Alta Disponibilità su Area Geografica (HA-WAN) sono in fase di definizione, e saranno verificate al più presto insieme al GARR. L'idea è quella di definire una VLAN che si estenda oltre i confini geografici delle singole sedi, e sulla quale sia quindi possibile utilizzare strumenti standard quali l'Heartbeat per far migrare i servizi informatici dall'host primario a quello di *failover*. Questa estensione di una VLAN su area geografica è un servizio VPLS (Virtual Private Lan Service) disponibile su reti IP/MPLS.

3 AUTENTICAZIONE

Per rendere il più versatile possibile la INFN-AAI i sistemi di autenticazione supportati saranno Kerberos5 (sia in modalità nativa che con coppia username/password) e certificati X.509. Inoltre per il periodo transitorio, fino a quando le sedi non avranno completato la migrazione dei loro sistemi di autenticazione a Kerberos5, sarà supportata anche l'autenticazione con coppia username/password Unix (ma basata su server LDAP).

Tutto questo è reso possibile dal fatto che il Directory Server LDAP scelto, il "389 Directory Server", supporta i meccanismi di autenticazione SASL oltre al nativo *bind* LDAP, ed è inoltre configurabile attraverso *plug-in* (che non sono altro che apposite librerie che il server carica a *run-time*).

Prima di procedere alla descrizione delle varie implementazioni del meccanismo di *Client Authentication*, è necessario svolgere preliminarmente alcune brevi considerazioni sul significato dell'operazione di *binding* ad un Directory Server (per brevità indicato come DS nel seguito).

In generale *binding* è sinonimo di Autenticazione: l'operazione di *binding* consiste infatti nel fornire al DS delle credenziali in base alle quali il DS stesso decide se il client è abilitato ad accedere al DS e con quali privilegi.

LDAPv3 supporta almeno due tipi di operazioni di *bind* distinte: il *simple bind*, che prevede l'utilizzo di credenziali nella forma *(DN, password)*; ed il *SASL bind*, che prevede l'utilizzo delle credenziali relative al particolare meccanismo di autenticazione SASL supportato (*PLAIN*, *GSSAPI*, *EXTERNAL*).

Tralasciando per il momento il fatto che entrambi i succitati meccanismi prevedono *bind* anonimi, nei quali il cliente **non fornisce credenziali di sorta e di fatto non viene quindi autenticato**, sia per il *simple bind* sia per il SASL *bind* è necessario che il cliente si identifichi, mediante un DN nel primo caso, o mediante credenziali che debbono essere collegate al relativo DN nel secondo (attraverso un *"mapping"*).

3.1 Kerberos5

Il supporto per Autenticazione Kerberos5 è reso obbligatorio dal fatto che tale protocollo è adottato in un numero sempre maggiore di sedi INFN, visto anche le caratteristiche di sicurezza e di supporto per Single Sign-On.

Infatti, nei casi in cui sia i servizi a cui l'utente voglia accedere, sia le applicazioni client di tali servizi siano compilate con il supporto per SASL e l'utente dispone nella sua cache delle credenziali di un *Ticket Granting Ticket* valido per chiedere un ticket di servizio, verrà negoziata un'autenticazione GSSAPI. In tal caso quindi, l'utente non dovrà digitare alcuna password poiché sfrutterà il *Ticket* già acquisito.

Per permettere che la stessa operazione di *bind* al directory server possa avvenire mediante GSSAPI, è necessario configurare il Directory Server affinché possa decriptare il

Ticket di servizio e, una volta avvenuta l'autenticazione, effettui il *mapping* tra il principal contenuto nel ticket e *l'entry* LDAP con cui è richiesta l'operazione di *bind*.

Siccome il meccanismo di Autenticazione GSSAPI è proprio di SASL (e non di LDAP) è necessario che la configurazione del *mapping* sia fatta su SASL. Tale configurazione per il "389 Directory Server" si effettua facilmente attraverso la console di management, ed inoltre la possibilità di usare *regular-expressions* per definire il criterio di ricerca, rende questo strumento altamente flessibile.

Per alcune applicazioni e su alcune piattaforme non è disponibile un client Kerberizzato (cioè con supporto per SASL-GSSAPI). Per questo motivo è stato scritto un plug-in che converte una richiesta di autenticazione LDAP con username e password in una richiesta di *Ticket* Kerberos5. Tale funzionalità facilita l'uso della INFN-AAI allargando il bacino d'utenza a tutti i client che non supportano autenticazione kerberos5, ma siccome il suo utilizzo espone la coppia di username/password al Directory Server ed alla rete, dovrà essere garantita la messa in sicurezza di tutti i server LDAP che offrono questo servizio, e dovrà essere garantito l'uso di un canale crittografato SSL.

3.2 Simple LDAP bind (coppia Username/Password)

La semplice autenticazione LDAP, o *simple bind* con coppia (DN, password) è l'unica che garantisce un passaggio praticamente indolore, da una autenticazione di tipo Unix (distribuita sulla LAN via YellowPages o NIS o in altri modi proprietari) ad autenticazione LDAP. Essa infatti avviene attraverso la verifica che la password immessa, sia la stessa memorizzata nell'attributo userPassword della entry che corrisponde al DN.

I valori di DN e della password sono normalmente inviati in chiaro al Directory Server. È quindi necessario garantire anche in questo caso l'uso di un canale di comunicazione crittografato.

3.3 krb5 plug-in

Per facilitare il passaggio ad autenticazione Kerberos5 è stato disegnato e scritto un plug-in per il "389 Directory Server" che ha due funzioni:

1) simula una Autenticazione LDAP *simple bind*
2) automatizza il popolamento di un KDC.

Il plug-in Krb5 per 389 Directory Server ribalta le richieste di autenticazione LDAP con username e password (simple authentication) verso il KDC Kerberos5 autoritativo nel riconoscimento dell'utente. E' ovvio, che benché si utilizzi Kerberos5 come backend di convalida delle credenziali, non si tratta comunque di autenticazione Kerberizzata poiché tra client e directory server non avviene alcuno scambio di *Ticket*. Per le applicazioni l'attività del *plug-in* e il conseguente ricorso a Kerberos è trasparante nel senso che queste scambiano con il Directory Server solo messaggi LDAP di *Simple Authentication*, mentre è il server, che

invece di convalidare la password, come consuetudine, tramite l'hash memorizzato nell'attributo userPassword, interroga il KDC e risponde di conseguenza al client.

L'associazione tra una entry LDAP che può eseguire un'operazione di bind e il principal Kerberos5 avviene tramite l'attributo *infnKerberosPrincipal* in cui è contenuto il principal completo di REALM (es. pippo@LE.INFN.IT) con cui il plug-in tenterà la convalida Kerberos5. Benché fosse possibile, per buona parte delle entry che rappresentano gli account utente, stabilire automaticamente il principal di autenticazione, basandosi sulla sede di appartenenza, si è preferito esplicitarlo inequivocabilmente all'interno della entry. Questo permette di definire account che appartengono a REALM esterni alla sede, cosa che avviene adesso nelle sedi che gestiscono account appartenenti alla cella nazionale AFS infn.it.

Oltre al compito di autenticazione, il plug-in svolge anche la funzione di migrazione automatica degli utenti delle Sedi che non usano Kerberos 5 come sistema di autenticazione. Per tali realtà, poiché non è possibile importare direttamente nei KDC gli hash delle password presenti nel NIS perché incompatibili con le chiavi di cifratura usate da Kerberos, si procede come segue:

- Si importa l'hash di ogni utente nell'attributo userPassword della entry che costituisce il suo account;
- La prima volta che l'utente cerca di autenticarsi e il plug-in si accorge che il principal Kerberos contenuto nell'attributo *infnKerberosPrincipal* non esiste nel database del KDC di competenza, cerca di autenticarlo per confronto con l'hash presente in userPassword;
- Se l'autenticazione ha esito positivo, l'utente ottiene l'accesso e come ulteriore effetto, il plug-in crea il principal Kerberos sfruttando la password in chiaro digitata dall'utente nel tentativo di autenticarsi. Si noti, che in una simple authentication, la password arriva in chiaro al directory server, ma comunque viaggia in un tunnel TLS se richiesto;
- Creato il principal, il plug-in cancella l'attributo userPassword in quanto da quel momento l'utente è kerberizzato.

3.4 Certificati X.509

La *Client Authentication* mediante certificati X.509 differisce sostanzialmente dai casi sopra citati per il fatto che l'autenticazione, parzialmente demandata a SSL per quanto riguarda la verifica del certificato presentato dal client, pur richiedendo comunque la necessità di definire una corrispondenza tra uno o più DN contenuti nel certificato ed i corrispondenti DN presenti sul DS, non implica automaticamente il .

Di più: a meno di non richiedere esplicitamente una bind operation basata su credenziali non nulle, anche se il cliente viene correttamente mappato in un record contenuto nel DS, il bind di default che segue l'avvenuta autenticazione è di tipo *anonymous*.

Una volta verificata quindi l'identità del client, che normalmente consiste in

1) verificare se il certificato sia stato emesso da una delle CA riconosciute dal DS e
2) verificare se esiste una corrispondenza (*mapping)* tra il Subject DN presente nel certificato ed un DN presente nel DS,

per effettuare un bind vero e proprio (cioe' non anonimo) al DS è necessario richiedere un SASL bind con meccanismo EXTERNAL. Tramite quest'ultima operazione e' infatti possibile autenticare un cliente sulla base di credenziali non direttamente riconducibili a SASL (cioè - in altre parole - esterne ad esso: è una palese tautologia, ma è la prescrizione quasi testuale desunta dalla RFC 2222).

Vale la pena di notare che, allo stato, sono pochissimi i clienti in grado di implementare completamente *out of the box* questo sistema di autenticazione; è però opportuno ricordare che il *binding* non anonimo è in generale necessario solo per operare direttamente sui dati contenuti nel DS (cambio password, modifica/cancellazione di attributi personali e via dicendo).

Nella stragrande maggioranza dei casi ciò che realmente interessa è la semplice autenticazione di un cliente, e si è quindi ritenuto opportuno includere anche questo tra i possibili meccanismi di autenticazione supportati ufficialmente da AAI per aumentarne la flessibilità e le possibilità operative (si pensi per esempio alle smart card).

3.5 Architettura hardware

In questa fase di implementazione di INFN-AAI, il supporto per certificati X.509 è semplicemente dato dalla possibilità di effettuare autenticazione usando tali tipi di certificati, rilasciati dalla INFN-CA. Quindi non ci sarà una parte di architettura hardware dedicata a questo.

Mentre per quanto riguarda l’autenticazione Kerberos5, per garantire che il processo di autenticazione possa essere sempre fattibile (deve essere contattabile almeno un server KDC affinché ciò sia garantito) è necessario installare un KDC server slave per ogni REALM Kerberos5, nelle vicinanze dei sistemi che potrebbero richiedere tale processo (e quindi nei due centri che ospitano i servizi centralizzati: LNF e CNAF).

Sono previsti almeno due server fisici che conterranno le 26 macchine virtuali ognuna delle quali svolgerà il ruolo di KDC slave per uno dei REALM Kerberos5 dell’INFN. I due server fisici saranno configurati in modalità cluster, in modo che le macchine virtuali potranno essere spostate da un server fisico all’altro sia per esigenze di manutenzione, che per sopperire ad eventuali malfunzionamenti dell’hardware.

4 AUTORIZZAZIONE

In generale, una applicazione concede l'autorizzazione affinché un utente autenticato possa eseguire operazioni, passando attraverso due livelli:

- il primo corrisponde alla verifica dell'appartenenza a gruppi, che normalmente rappresentano il “ruolo” o la funzione esercitata dalla persona
- il secondo livello (normalmente definito attraverso ACL) indica il “diritto” (*entitlement*) che il gruppo a cui la persona appartiene, ha sulla risorsa in questione.

La maggior parte delle applicazioni sono in grado di delegare la gestione del solo primo livello ad un sistema esterno (tipicamente una Directory) e gestiscono internamente il secondo livello (e questo avviene essenzialmente perché sono applicazioni che si adattano ad un servizio esterno e non sono disegnate tenendo in considerazione le potenzialità di una Direcory).

Applicazioni che sono invece pensate e disegnate per funzionare in ambienti di federazioni di AAI, possono delegare alla AAI anche la funzione di verifica del diritto dell'utente ad utilizzare la risorsa richiesta.

4.1 I Gruppi

Le classi di oggetti usate per la gestione dei gruppi e gli attributi utilizzati per la gestione dei membri di un gruppo, dipendono in generale dal tipo di server LDAP.

Ad esempio, Windows Active Directory usa **group** come classe di oggetti per i gruppi e **member** come attributo per identificare i membri del gruppo, mentre “389 Directory Server” usa rispettivamente **groupOfUniqueNames** e **uniquemember**.

Normalmente, si utilizza lo *schema* tipico del Directory Server in uso, e si adattano ad esso le *query* LDAP fatte dai client. Nel nostro caso, siccome le informazioni nella Directory saranno inserite solo attraverso il sistema GOVA, è possibile utilizzare contemporaneamente tutti i modelli più usati. I vari metodi di definizione dei gruppi saranno comunque tenuti sincronizzati attraverso il sistema GOVA. Questo renderà meno complicata la modifica delle configurazioni dei client nelle sedi che usano già LDAP (al limite sarà necessario modificare la *base* di ricerca nell'albero LDAP e non il metodo di ricerca).

4.2 Diritti (Entitlements)

Un entitlement puo' essere rappresentato da una semplice URI che indica l'oggetto di un'azione predefinita, ad esempio una URL per indicare l'accesso in lettura ad una pagina web. Tuttavia per definire in modo più granulare i permessi si ricorre all'utilizzo delle URN (Uniform Resource Name) che rappresentano il privilegio su una risorsa all'interno di un namespace. Un esempio di entitlement URN per INFN-AAI potrebbe essere

urn:mace:infn.it:entitlement:gova:visitatori:admin

Per inserire gli *entitlement* nella directory LDAP si usa un attributo della classe di oggetti eduPerson, ossia eduPersonEntitlement. Le applicazioni potranno quindi, tramite una query LDAP sulla entry dell'utente che tenta l'accesso ad una risorsa, controllare il livello di accesso da consentire.

La gestione degli *entitlement* potrà essere effettuata in due modi:

1)tramite ereditarietà dai gruppi: Tramite l'applicazione GOVA sarà possibile associare degli *entitlement* predefiniti ai membri di determinati gruppi, in modo da legare i diritti ai ruoli.

2)associando direttamente un *entitlement* ad una persona.

Normalmente si gode di un diritto in quanto aventi un ruolo all'interno dell'ente, la modalità 1) gestisce questo meccanismo automaticamente.

Se invece si dovessero gestire tutti i permessi associandoli direttamente alla persona e non ad un ruolo (modo 2) e ad esempio si rimuovesse un ruolo da una persona, ci si troverebbe nella difficoltà di individuare manualmente tutti gli *entitlement* non più necessari. Il rischio, in quest'ultimo caso, sarebbe di eliminare i permessi in comune con altri ruoli che appartengono ancora all'individuo. La modalità 2 quindi, seppur più semplice, è da sconsigliare se non in casi eccezionali.

L'applicazione GOVA permetterà la gestione degli *entitlement*, ma li utilizzerà anche per discriminare i permessi al proprio interno. Esisteranno quindi degli *entitlement* che permetteranno ad alcuni individui la gestione di altri *entitlement*.

4.3Architettura hardware

Il supporto all’Autorizzazione sarà fornito attraverso un pool di server LDAP. Sono previsti 4 server sui quali il “389 Directory Server” sarà configurato in modalità Multi-Master. Per garantire la raggiungibilità di almeno uno di questi server, ne saranno installati 2 sulla LAN dei LNF e 2 presso il CNAF.

5 BACKUP E DISASTER RECOVERY

Data la presenza presso i LNF di un sistema di backup Tivoli TSM, i servizi presenti sui server dei LNF godranno di tale servizio. Quindi sarà definita una politica di backup sia per i DB server Oracle, che per gli Application Server, per i server di Load Balancer e per i Directory Server. Gli unici sistemi che non godranno del servizio di backup dei LNF saranno i server KDC, in quanto server slave di strutture che per poter funzionare devono aver previsto almeno un paio di server di quel tipo sulla propria LAN.

5.1“Time-Machine” plug-in

Per quanto frequenti possano essere i backup, è possibile che vengano perse informazioni relative allo stato della Directory (delle *entry* della Directory) se avvengono più modifiche tra un backup ed il successivo.

Per poter recuperare lo stato di una qualunque *entry* in un qualunque istante, è stato progettato (ed è in fase di codifica) un plug-in per il “389 Directory Server”.

Tale plug-in registra sul DB Oracle (un apposito DB gestito dagli stessi server che ospitano il DB di GOVA) ogni nuovo stato di una *entry*, nel momento in cui viene effettuata la modifica

L'intero processo di memorizzazione dei dati per la sotricizzazione delle operazioni effettuate su l'LDAP server verrà eseguito da due software:

- Plug-In per il server LDAP
- Software per la scrittura sul database Oracle

5.1.1 Plug-In

Il Plug-In verrà scritto in modo da leggere la sua configurazione direttamente sull'albero LDAP. Si potranno così configurare dinamicamente due tipologie di informazioni:

- Rami da escludere nel backup
- DN che effettuano un'operazione da escludere nel backup

Questo ci permetterà innanzitutto di fare in modo che il solo nodo (dei 4 master) su cui viene fatta l'operazione scriva il log, usando il blocco del DN associato all'utente di replica. La configurazione sui rami permetterà di bloccare il backup durante un ripristino(time-machine) da un backup (per ovviare al problema di effettuare backup di dati già memorizzati). Tutte le operazioni effettuate sul server LDAP saranno scritte su una coda del kernel, questo renderà il plug-in estremamente veloce, e nel caso ci siano problemi a contattare il server oracle, il processo su cui esso viene eseguito non verrà bloccato.

5.1.2 Software per la scrittura su Oracle

Il software al punto due, sarà costituito da due *thread* differenti:

- Consumatore della coda
- Scrittura sul database Oracle

Il *thread* consumatore toglierà dalla coda del *kernel* i messaggi di log scritti dal plug-in, ognuno dei quali verrà archiviato momentaneamente in un database locale (SQlite http://www.sqlite.org/) usato come cache su disco temporanea. Il *thread* di scrittura preleverà il record dal database di cache più vecchio e tenterà di scriverlo sul database Oracle di backup remoto. Il record verrà tolto dal database di cache solo quando la transazione sul database di backup sarà stata effettuata con successo. Questo schema permette in caso di problemi (crash server LDAP, server Oracle non accessibile, etc.) di non perdere nessun log da archiviare.

6 LO SCHEMA DI INFN-AAI

Con “schema” si intende l'insieme delle Classi di Oggetti (ObjectClass) utilizzate nel Directory Server.

Una delle direttive di buon disegno di un Directory Server è quella che consiglia di non definire Classi di Oggetti nuove, se è possibile utilizzare Classi di Oggetti già definite. Per questo motivo, nonostante nelle prime fasi pre-prototipali di INFN-AAI siano state definite delle Classi di Oggetti ad-hoc per attributi specifici per l'INFN, uno studio attento delle Classi di Oggetti definite dalle collaborazioni MACE di Internet2 (eduPerson) e TF-EMC2 di TERENA (SCHAC) ha evidenziato la possibilità di utilizzare tali Classi di Oggetti.

6.1objectClasses

Oltre quelle standard incluse in 389-DS é supportata l'objectclass eduPerson, quelle SCHAC, ossia schacPersonalCharacteristics, schacContactLocation, schacEmployeeInfo, schacLinkageIdentifiers, schacEntryMetadata, schacEntryConfidentiality, schacUserEntitlements, schacGroupMembership e infnPerson.

Gli attributi obbligatori per ogni entry:

attributo	objectclass	descrizione
infnUUID	infnPerson	Identificativo univoco della entry (**utilizzato come RDN**)
uid	inetOrgPerson	login name
mail	inetOrgPerson	indirizzo e-mail principale
schacPersonalUniqueID	schacLinkageIdentifiers	identificativo legale, Codice Fiscale in formato URN
schacPersonalUniqueCode	schacLinkageIdentifiers	codice identificativo univoco della persona assegnato dalla AAI o dall'ente. INVARIABILE nel tempo
infnKerberosPrincipal	infnPerson	Principal Kerberos5 per l’autenticazione
schacPersonalPosition	schacEmployeeInfo	Posizione lavorativa nei confronti dell’ente

Gli **attributi consigliati** in caso di necessità:

attributo	Objectclass	descrizione
mailAlternateAddress	mailRecipient	alias di posta
schacExpiryDate	schacEntryMetadata	data e ora di scadenza dell'account
eduPersonEntitlement	eduPerson	diritti dell’utente
infnCertSubjectDN	infnPerson	Subject del certificato x509

7 FEDERAZIONI

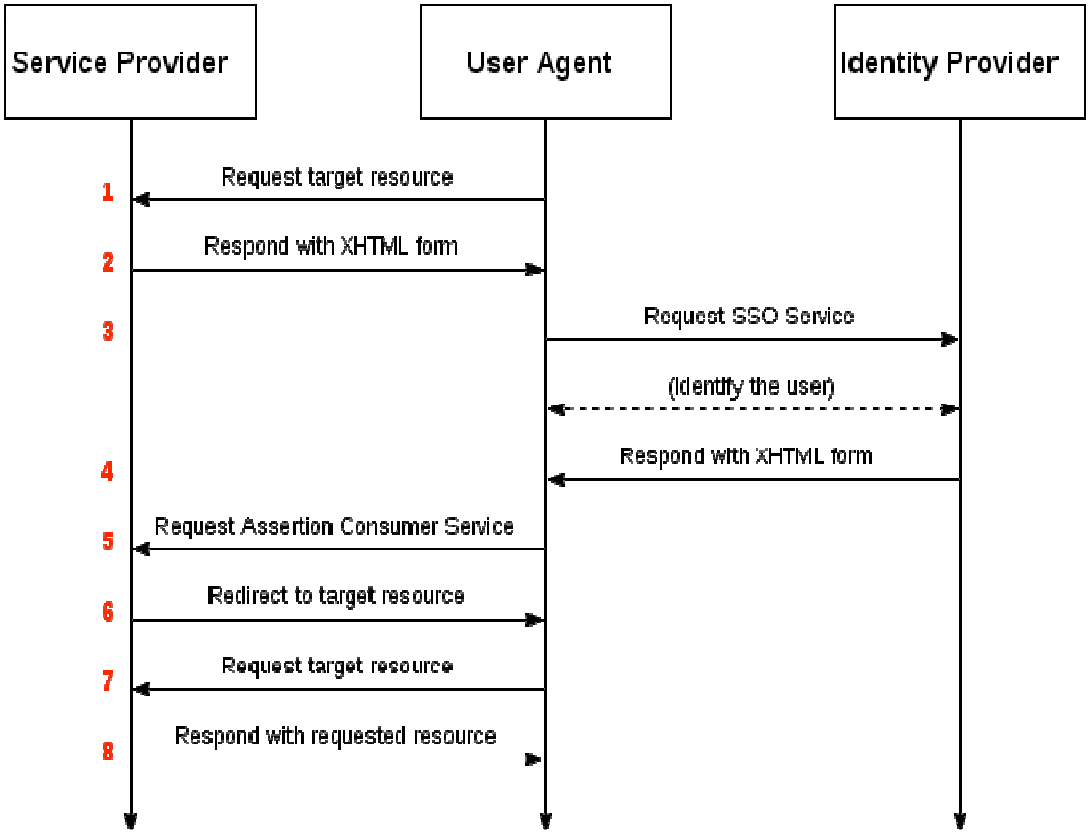
Grazie all'implementazione della INFN-AAI si potrà partecipare alle federazioni internazionali di AAI. Questo permetterà di accedere ad applicazioni federate utilizzando i sistemi di autenticazione e autorizzazione INFN senza bisogno di ulteriori registrazioni da parte degli utenti.

7.1 SAML

Il sistema utilizzato dalle attuali federazioni è basato sul protocollo SAML 2.0, evoluzione e standardizzazione del software Shibboleth 1.3, che infatti nella versione 2.0 aderisce agli standard SAML 2.0.

Nell'architettura SAML ogni istituto ha un suo Identity Provider, che è autoritativo per il proprio dominio e può essere utilizzato dai Service Provider, ossia l'applicazione a cui si vuole accedere, per controllare l'identità e le autorizzazioni degli utenti.

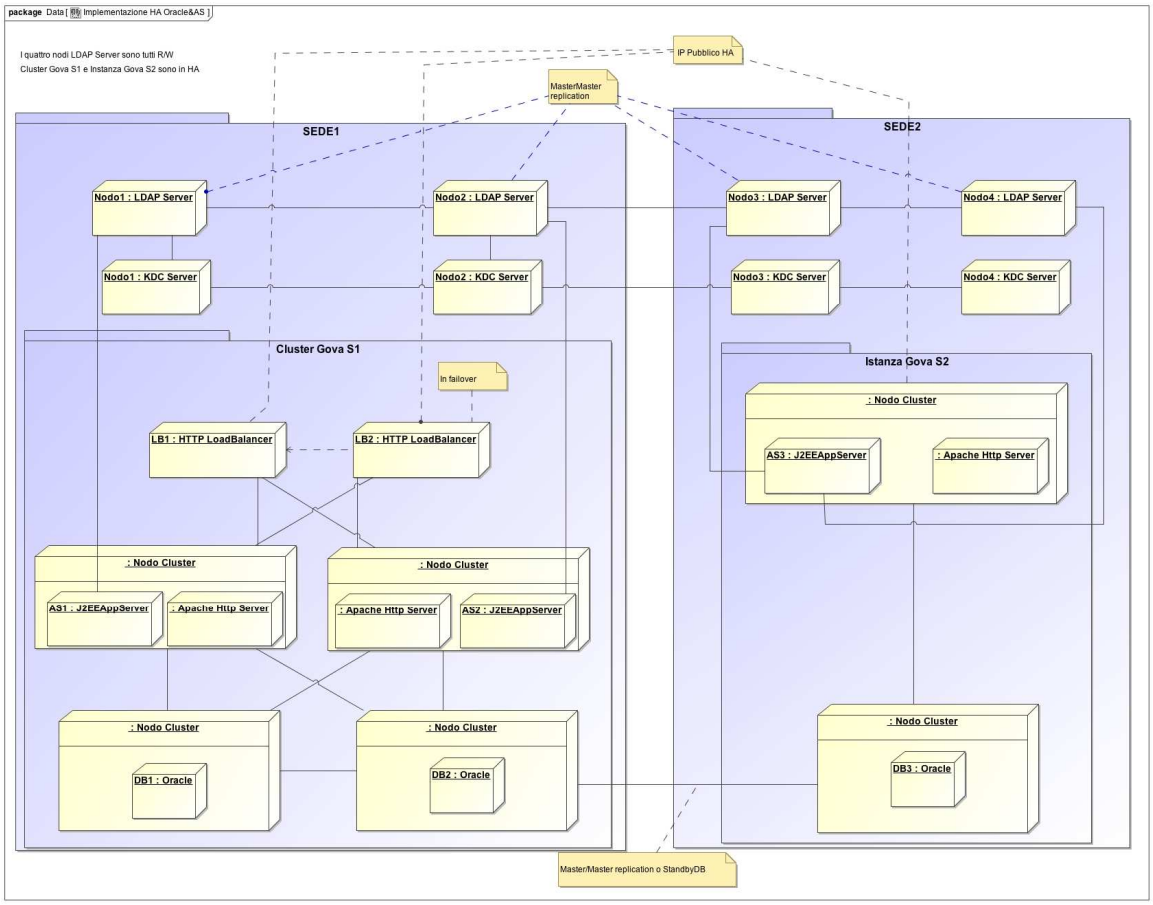
Uno dei vantaggi dell'utilizzo di SAML è che il client non presenta mai le proprie credenziali di accesso al SP, quest'ultimo infatti quanto un utente tenta il login ridirige il browser del client verso l'IdP, che a sua volta dopo un'autenticazione corretta effettua un altro redirect del browser verso il SP insieme ad una assertion, ovvero un pacchetto di dati XML firmato tramite RSA che contiene le informazioni sull'utente autenticato.



Un’altra impartante caratteristiche che si ottiene con l’utilizzo di SAML è il Single Sign On, l’IdP infatti ricorderà, attraverso un cookie di sessione, l’autenticazione dell’utente quando questo tenterà l’accesso ad una secondo SP.

8 SPECIFICHE HARDWARE

Il disegno globale dell'architettura in figura, mostra che per l'implementazione completa sono necessari 14 server fisici (3 per il DB Oracle, 3 per Application Server, 4 per Directory Server e 4 per KDC server) oltre ad un paio almeno di server su macchine virtuali.



Tutti i server fisici dovranno avere un set di caratteristiche comuni, che sono illustrate in tabella.

Componente	caratteristiche
CPU	1 processore 4-core
RAM	8GB
Alimentatore	ridondato
Rete	2x1GB Ethernet
Montaggio	Rack standard 19"

8.1 DB Server Oracle

Sono previsti 3 DB Server Oracle. 2 in configurazine RAC da installare presso i LNF ed 1 di *failover* da installare presso il CNAF.

I server da installare presso i LNF dovranno essere dotati di HBA Fiber Channel per poter essere collegati alla SAN (la configurazione RAC prevede un filesystem di tipo cluster, condiviso tra gli host, e si prevede di utilizzare OCFS2). Il server da installare presso il CNAF invece dovrà avere un proprio sistema di disco.

utilizzo	collocazione	caratteristiche specifiche
Oracle RAC #1 & #2	LNF	HBA Fiber Channel
Oracle failover	CNAF	-Almeno 3 dischi interni da 15Krpm in configurazione RAID 1 con hot spare -Scheda RAID con almeno 256MB di cache e batteria tampone

8.2 Application Server

Come evidenziato dal disegno della configurazione hardware di GOVA, sono previsti 3 Application Server: due da installare presso i LNF, ed uno presso il CNAF. Per questo tipo di server non sono richieste configurazioni differenti per le due sedi. Quindi le caratteristiche specifiche da aggiungere a quelle comuni sono:

utilizzo	collocazione	caratteristiche specifiche
Application Server 1,2,3	2@LNF + 1@CNAF	- Almeno 3 dischi interni da 15Krpm in configurazione RAID 1 con hot spare - Scheda RAID con almeno 256MB di cache e batteria tampone

8.3 Load Balancer

I server di Load Balancer previsti per la corretta gestione delle connessioni in caso di fail di uno dei due Application Server presenti ai LNF, non richiedendo elevate prestazioni, possono essere comodamente alloggiati in un paio di macchine virtuali. Siccome presso i LNF esiste una infrastruttura di servizi informatici su macchine virtuali, sarà possibile trovare risorse all'interno di tale infrastruttura senza dover acquistare ulteriore hardware.

8.4 Server LDAP

Come descritto precedentemente, sono previsti 4 server LDAP in configurazione Multi-

Master. Due saranno installati presso i LNF e due presso il CNAF. Anche per questi server

utilizzo	collocazione	caratteristiche specifiche
389 Directory Server 1,2,3,4	2@LNF + 2@CNAF	- Almeno 3 dischi interni da 15Krpm in configurazione RAID 1 con hot spare - Scheda RAID con almeno 256MB di cache e batteria tampone

non sono previste configurazioni differenti tra le due sedi e quindi:

8.5 Server per KDC farm

A regime i quattro server conterranno ognuno 13 macchine virtuali KDC slave. Siccome un KDC slave gira bene anche con 512MB di RAM, basterebbero in linea di principio server con 8GB di RAM. In realtà se si fornirà anche il servizio di cluster su tali server, in modo da poter spostare i KDC slave da un host fisica all'altro per effettuare le operazioni di manutenzione programmate o meno, sarà necessario raddoppiare la quantità di RAM.

Essendoci ad oggi solo una decina di sedi che usano Kerberos5, la configurazione standard è sufficiente anche a gestire gli spostamenti dei KDC slave da un host all'altro.

Quindi anche in questo caso avremo:

utilizzo	collocazione	caratteristiche specifiche
KDC server farm LNF-1,2 +CNAF-1,2	2@LNF + 2@CNAF	- Almeno 3 dischi interni da 15Krpm in configurazione RAID 1 con hot spare - Scheda RAID con almeno 256MB di cache e batteria tampone

9. APPENDICI

9.1. Installazione e configurazione del plug-in krb5

Il plug-in, scritto in C richiede per la compilazione gli header per l'accesso alle API di Kerberos (krb5.h) e di quelli per l'interfacciamento al directory server (slapi-plugin.h). La compilazione e il caricamento nel directory server avviene tramite le seguenti operazioni:

Effettuare il download del codice sorgente disponibile all'URL http://wiki.infn.it/_media/cn/ccr/aai/howto/krb5-plugin.tgz

Estrarre il contenuto del file krb5-plugin.tgz. Il risultato è contenuto nella directory krb5-plugin

Entrare nella directory krb5-plugin e digitare il comando make. Il risultato è la shared object libkrb5-plugin.so

Copiare il file libkrb5-plugin.so nella directory contenente i plug-in (su di una macchian a 64bit la directory è /usr/lib64/dirsrv/plugins/)

Dopo essersi assicurati che i processi del Directory Server non sono attivi (/etc/init.d/dirsrv stop) editare il file /etc/dirsrv/slapd-DS-UID/dse.ldif aggiungendo le seguenti linee:

```
dn: cn=KRB5 Bind,cn=plugins,cn=config
objectClass: top
objectClass: nsSlapdPlugin
objectClass: extensibleObject
cn: KRB5 Bind
nsslapd-pluginPath:/usr/lib64/dirsrv/plugins/libkrb5-plugin.so
nsslapd-pluginInitfunc: krb5bind_init
nsslapd-pluginType: preoperation
nsslapd-pluginEnabled: on
nsslapd-plugin-depends-on-type: database
nsslapd-pluginId: krb5-bind
nsslapd-pluginVersion: 1.0.4
nsslapd-pluginVendor: Fedora Project
nsslapd-pluginDescription: Kerberos 5 bind pre-operation plugin (INFN)
```

Il plug-in per l'autenticazione Kerberos 5 va installato su tutti i Directory Server, inclusi quelli RO. Il file /etc/krb5.conf, contenente la configurazione delle librerie Kerberos 5, deve avere una entry per ognuno dei REALM. Sui server RW presenti ai LNF e al CNAF sono necessarie inoltre delle chiavi di autenticazione associate al servizio autoadd/plugin@REALM mediante cui il plug-in si autentica nei confronti del KDC Master per poter aggiungere gli utenti. Tali chiavi risiedono nei file /etc/krb5-plugin/keys/REALM.keytab.

9.2. Implementazione di Client Authenticazion via certificati X.509

Dando per scontato che il server sia gia' stato opportunamente configurato per supportare SSL e riconoscere come validi i certificati emessi dalla CA INFN (in altre parole: la CA INFN deve comparire nella lista delle CA trusted - si veda a questo proposito Managing Servers with RedHat Console, pagg. 175- e 184-), vediamo come configurare il server in modo che quest'ultimo sia in grado di interpretare correttamente i DN presenti nei certificati. A questo scopo il server utilizza il file di configurazione certmap.conf, il cui scopo e' quello di definire le strategie di mapping tra i DN contenuti nei certificati e le informazioni immagazzinate nel DS stesso. Il formato di tale file e' il seguente:

```
certmap name 'issuer CA DN'
name:property [value]
name:property [value]
....

certmap name 'issuer CA DN'
....
```

È possibile definire sia un *mapping* di default, sia *mapping* multipli, ognuno dei quali relativi ad una data trusted CA. In ogni singola mappa è possibile definire, per ogni *trusted* CA, i seguenti parametri:

- il punto dell'albero da cui iniziare la ricerca;
- quali DN contenuti nel certificato utilizzare come chiavi di ricerca;
- quali attributi della directory utilizzare per il matching;
- quali verifiche ulteriori effettuare per stabilire l'identita' del cliente.

Senza entrare troppo nel dettaglio, le regole della mappa possono essere definite utilizzando una o piu' delle seguenti clausole principali: DNComps, FilterComps, VerifyCert, CmapLdapAttr, il cui significato conviene chiarire - molto brevemente - con due esempi:

```
# certmap.conf 1
certmap INFN            CN=INFN CA,O=INFN,C=IT
INFN:DNComps
INFN:FilterComps        mail,cn
INFN:verifyCert         off

# certmap.conf 2
certmap INFN            CN=INFN CA,O=INFN,C=IT
INFN:DNComps
INFN:CmapLdapAttr       CertSubjectDN
INFN:verifyCert         on
```

Tenendo presente che i certificati emessi dalla CA INFN sono, di norma, della forma

che segue:

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 10575 (0x294f)
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: C=IT, O=INFN, CN=INFN CA
        Validity
            Not Before: Jun 19 12:56:36 2008 GMT
            Not After : Jun 19 12:56:36 2009 GMT
        Subject: C=IT, O=INFN, OU=Personal Certificate, L=Milano
Bicocca, CN=Jack Tamigi
....
            X509v3 Subject Alternative Name:
                email:jack.tamigi@mib.infn.it
....
```

vediamo nel dettaglio il comportamento delle due mappe di cui sopra. Entrambe le mappe sono relative a certificati emessi dalla CA INFN (`certmap INFN CN=INFN CA,O=INFN,C=IT`) e si applicano in altre parole solo a certificati emessi dalla CA il cui DN corrisponde esattamente a quello indicato; poiché nel DIT implementato da AAI i sotto-alberi relativi alle singole sezioni sono individuati da un DN della forma dc=<sezione>,dc=infn,dc=it, non essendo presenti nel naming scheme implementato dalla CA INFN attributi dc, in entrambi i casi il DNComps è vuoto, la qual cosa istruisce il DS ad effettuare la ricerca di un matching DN su tutto l'albero.

Se da un lato questa potrebbe comportare una certa quale inefficienza, dall'altro permette - in dipendenza di quali parti dell'albero saranno accessibili alle singole sezioni - di non limitare l'autenticazione tramite certificato al personale locale. Nel caso della prima mappa gli attributi contenuti nel certificato che costituiranno la chiave di ricerca nell'albero del DS sono mail, cn: in altre parole, il DS estrarrà dal certificato i valori di cn (Jack Tamigi nel caso in esame) e mail (jack.tamigi@mib.infn.it) e li userà per cercare nell'albero del DS un record i cui attributi cn e mail coincidano con quelli indicati. Se esiste un record rispondente a queste caratteristiche:

```
dn:uid=tamigi,ou=People, dc=mib, dc=infn, dc=it

                givenName: Jack
                       sn: Tamigi
...
                     mail: jack.tamigi@mib.infn.it
              objectClass: top
                           person
...
                      uid: tamigi
                       cn: Jack Tamigi
...
```

allora, poiché non è prevista l'ulteriore verifica di uguaglianza tra il certificato presentato e quello eventualmente presente nel DS (`INFN:verifyCert off`) l'autenticazione potrà dirsi conclusa con successo con il *mapping*:

Il secondo esempio illustra invece una strategia di mapping basata su un attributo non

standard (`INFN:CmapLdapAttr CertSubjectDN`) che deve essere quindi esplicitamente incluso come estensione dello schema di default ed altrettanto correttamente inizializzato:

```
dn:uid=tamigi,ou=People, dc=mib, dc=infn, dc=it
...
          CertSubjectDN: CN=Jack Tamigi,L=Milano Bicocca,OU=Personal
Certificate,O=INFN,C=IT
           userCertificate: XXXX...
...
```

e prevede inoltre un ulteriore stadio di verifica, consistente nella richiesta che il certificato presentato dal client corrisponda a quello memorizzato nel DS (`INFN:verifyCert on`). Se, e solo se, entrambe le condizioni sono verificate (in particolare il certificato deve essere presente nel record utente), allora l'autenticazione può dirsi conclusa con successo con il medesimo *mapping* di cui sopra.

Rapporto referee

Questa relazione è il frutto del lavoro di referaggio del progetto AAI, descritto nei documenti Conceptual Design Report (CDR) e Technical Design Report (TDR), che è stato effettuato a giugno 2009 dal collegio di referee costituito da Roberto Cecchini, Riccardo Fantechi, Luciano Gaido, Fulvio Galeazzi e Massimo Pistoni.

Descrizione

Il progetto AAI si propone la realizzazione di un'infrastruttura globale di Autenticazione e Autorizzazione (AAI) per tutto l'INFN. In questa relazione il progetto non verrà descritto in dettaglio ma verranno evidenziate soltanto le caratterisitiche principali, gli elementi positivi ed i suggerimenti del collegio dei referee. Informazioni più approfondite si trovano nei due documenti menzionati e nella relazione del primo collegio di referee.

In seguito alle raccomandazioni effettuate dal primo collegio di referee nel 2008, che ha analizzato il CDR, il progetto è stato modificato ed i cambiamenti sono stati recepiti nel TDR.

Le carattersitiche principali sono le seguenti:

- Il TDR è modulare per consentire la realizzazione delle diverse parti del progetto in modo graduale, compatibilmente con il man power e le risorse disponibili;
- E' ora previsto un sistema per la gestione di identità e accessi (IAM);
- Per la gestione delle identità si utilizzerà un tool (GOVA: Gestione Ospiti Visitatori ed Associati) che costituisce l'evoluzione di un tool esistente (GO: Gestione Ospiti); le informazioni relative ai dipendenti saranno importate dai database in cui attualmente risiedono e che sono gestiti da altri servizi dell'INFN;
- E' stata definita la Struttura Directory Information Tree (DIT): ora è un mix tra "alto e grosso" e "basso e magro";
- Sono previsti servizi di Backup (in particolare la tracciabilità delle modifiche) e di Disaster Recovery;
- E' prevista l'implementazione di un Identity Provider SAML;
- E' prevista una elevata granularità di configurazione per poter rispondere alle specificità delle Sezioni e Laboratori INFN;
- E' prevista l'integrazione della struttura AAI dell'INFN con quella di altre federazioni AAI.

Attualmente la parte relativa alla gestione delle identità e degli accessi, che ha implicazioni in ambito amministrativo e legale, è in fase di definizione, mentre il tool GOVA è in fase di sviluppo. A quest'ultima attività partecipano attivamente alcune persone dei Laboratori Nazionali di Frascati ed il borsista della sezione di Lecce il cui contributo è ritenuto molto importante.

L'architettura è basata sui componenti seguenti:

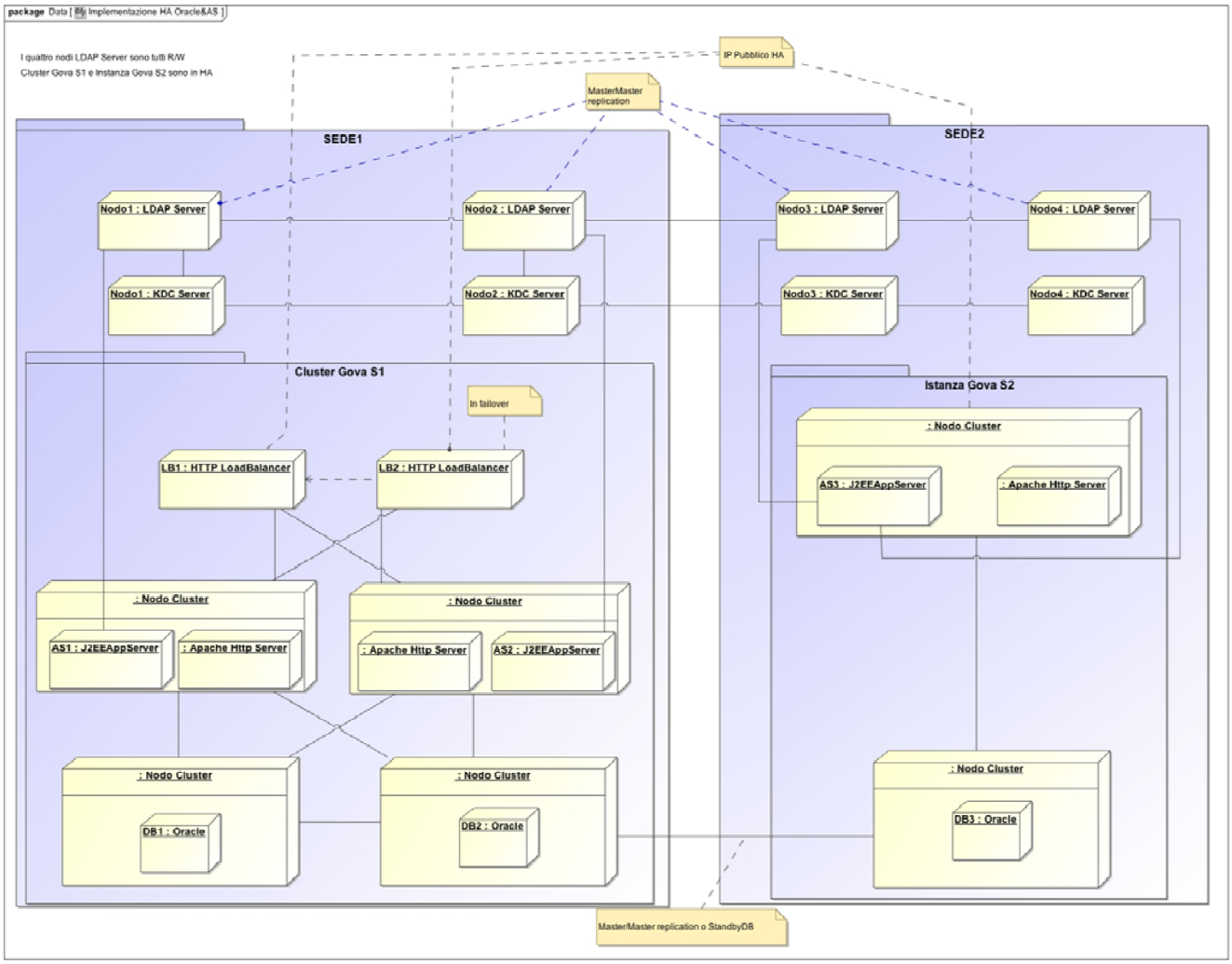
- LDAP/Kerberos per l'autenticazione (IAM e GOVA);
- LDAP (una copia di GOVA) con la definizione di gruppi e l'uso degli "entitlements" per l'autorizzazione;
- Database ORACLE e Application Server (con load balancer) per IAM e GOVA.

Per l'implementazione completa del progetto vengono richieste le seguenti risorse hardware (tutti i calcolatori sono monoprocessori quad core con 3 dischi veloci in RAID1) che saranno distribuite su due sedi (LNF, sede principale e CNAF, sede di backup):

- Per il databse Oracle: 3 server (2 a LNF ed 1 al CNAF) in configurazione RAC;
- Per l'Application Server necessario per l'accesso a Oracle (usato da GOVA): 3 server (2 a LNF ed 1 al CNAF);
- Per l'autenticazione Kerberos (per le copie slave dei principal delle sezioni): 4 server (2 a LNF e 2 al CNAF);
- Per l'autenticazione LDAP: 4 server (2 a LNF e 2 al CNAF) in configurazione Multimaster.

Sono previsti anche 2 Load Balancer che però verranno forniti dai Laboratori di Frascati.

La distribuzione dei calcolatori e dei servizi nelle due sedi è riportata nella figura seguente:



Considerazioni e raccomandazioni dei referee

Il collegio dei referee ritiene che le raccomandazioni formulate durante la precedente fase di valutazione del progetto siano state recepite in maniera soddisfacente e che il documento TDR sia completo ed esaustivo.

Tuttavia ci sono alcuni aspetti che meritano di essere chiariti in modo più approfondito. In particolare non è chiaro il ruolo dei database Oracle ed LDAP e la loro interazione. E' necessario specificare più in dettaglio quali informazioni sono contenute nei differenti database, il tipo di dato (informazione autoritativa o replica) e quali sono le interazioni previste tra i differenti database. Per una migliore comprensione si richiede di preparare un workflow che descriva il flusso di informazioni.

Per quanto riguarda il database Oracle, si ritiene opportuna e giustificata l'adozione di una configurazione RAC. Tuttavia si ritiene debba essere valutata attentamente la reale necessità della ridondanza dei servizi e dei database, soprattutto a livello geografico.

Una configurazione di backup a livello geografico richiede l'utilizzo di tecnologie e strumenti complessi che meritano una approfondita fase di test, che al momento non risulta ancora essere stata effettuata.

Si invita quindi, almeno nella fase iniziale, a semplificare la struttura prevista per quanto possibile:

- accorpando i database
- riducendo i servizi di backup
- concentrando i servizi previsti su un numero inferiore di calcolatori più potenti (biprocessori).

La valutazione del collegio dei referee è complessivamente positiva. Si invita quindi a procedere alla realizzazione delle componenti "core" della infrastruttura nazionale AAI che sono necessarie per lo startup del progetto, in modo tale da consentire l'inizio della migrazione delle Sezioni.

A questo proposito si invita ad individuare un numero limitato di sezioni pilota che possano sperimentare la migrazione dei servizi di autenticazione locali alla nuova infrastruttura con l'obiettivo di avere feedback sulle procedure previste, individuare eventuali problemi ed ottimizzare il dimensionamento delle strutture di backup.