

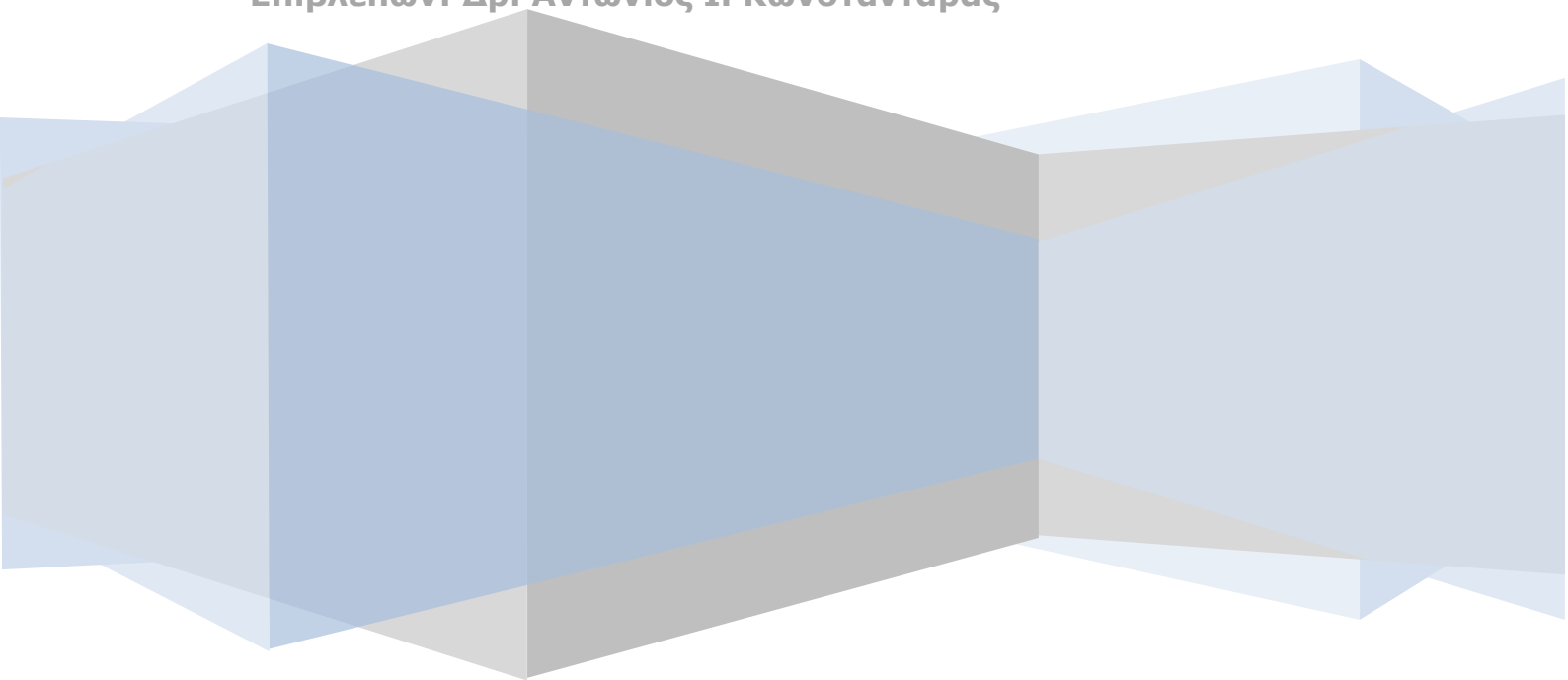
**ΤΕΙ ΚΡΗΤΗΣ – Παράρτημα Χανίων**  
**Τμήμα Ηλεκτρονικής**

**ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ**

**Συστήματα και Λογισμικό Κατανεμημένης και  
Παράλληλης Επεξεργασίας**

**Μανώλης Κατσιφαράκης**

**Επιβλέπων: Δρ. Αντώνιος Ι. Κωνσταντάρας**





## ΠΕΡΙΛΗΨΗ

Η διπλωματική αυτή εργασία έχει ως ευρύτερο αντικείμενο τα συστήματα παράλληλης και κατανεμημένης επεξεργασίας ενώ επικεντρώνεται στα υπολογιστικά συστήματα Πλέγματος (Grid) και συγκεκριμένα στην Ελληνική Υποδομή Πλέγματος, HellasGrid.

### Λέξεις Κλειδιά

Πλέγματα υπολογιστών, HellasGrid, EGEE, διεπαφή χρηστών, υποβολή εργασιών, διαχείριση δεδομένων

## **ABSTRACT**

The subject of this diploma thesis is parallel and distributed computing systems with a focus on grid computing environments through the Hellas Grid, Hellenic Grid Infrastructure.

### Keywords

Grid computing, HellasGrid, EGEE, User Interface, job submission, data management

## **ΕΥΧΑΡΙΣΤΙΕΣ**

Θα ήθελα να εκφράσω τις ευχαριστίες μου στον επιβλέποντα καθηγητή μου Δρ. Αντώνιο Ι. Κωνσταντάρια για το αμείωτο ενδιαφέρον του και τη καθοδήγησή του καθ' όλη τη διάρκεια διεκπεραίωσης της παρούσας πτυχιακής εργασίας, καθώς και για την ευκαιρία που μου έδωσε να ασχοληθώ ενεργά με έναν από τους πιο σημαντικούς τομείς της πληροφορικής.

Επίσης θα ήθελα να ευχαριστήσω τον κύριο Χρήστο Παπαχρήστο για τη βοήθεια του και τη ξενάγηση του κατά την επίσκεψή μας στο Ίδρυμα Τεχνολογίας και Έρευνας στο Ηράκλειο.

Τέλος θα ήθελα να ευχαριστήσω την οικογένειά μου για την υποστήριξή και την υπομονή τους .

## Περιεχόμενα

1. ΠΑΡΑΛΛΗΛΗ ΚΑΙ ΚΑΤΑΝΕΜΗΜΕΝΗ ΕΠΕΞΕΡΓΑΣΙΑ.....	6
1.1. ΕΙΣΑΓΩΓΗ .....	6
1.2. Τι είναι η παράλληλη και η κατανεμημένη επεξεργασία .....	6
1.3. Ιστορία .....	7
2. ΤΕΧΝΟΛΟΓΙΕΣ ΠΛΕΓΜΑΤΟΣ (GRID).....	9
2.1. Τι είναι οι τεχνολογίες πλέγματος.....	9
2.2. Ομοιότητες και διαφορές με άλλους τύπους παράλληλων υπολογιστικών συστημάτων .....	9
2.3. Ενδιάμεσο λογισμικό (middleware) .....	10
2.4. Δομή των grids .....	10
2.5. Ιστορία .....	11
3. ΓΕΝΙΚΑ ΣΤΟΙΧΕΙΑ ΓΙΑ ΤΟ EGEE .....	11
3.1. Εισαγωγή στο EGEE .....	11
3.2. Ιστορία .....	12
3.3. Στόχοι.....	12
3.4. Οργάνωση.....	13
3.5. Το HellasGrid .....	14
4. ΘΕΜΕΛΙΩΔΗ ΣΤΟΙΧΕΙΑ ΑΡΧΙΤΕΚΤΟΝΙΚΗΣ ΤΟΥ EGEE .....	15
4.1. Εργασίες (Jobs).....	15
4.2. Ασφάλεια – ψηφιακά πιστοποιητικά.....	15
4.3. Ιδεατοί Οργανισμοί (VOs) .....	16
4.4. Υπηρεσία Συμμετοχής σε Εικονικούς Οργανισμούς (VOMS). 17	
4.5. Η Διεπαφή Χρηστών (UI).....	17
4.6. Το Υπολογιστικό Στοιχείο (CE).....	18
4.7. Το Αποθηκευτικό Στοιχείο (SE).....	19
4.8. Η Υπηρεσία Πληροφοριών (IS) .....	19

4.9. Το Σύστημα Διαχείρισης Εργασιών (WMS).....	19
4.10. Διαχείριση Δεδομένων.....	20
5. ΠΡΟΣΒΑΣΗ ΣΤΟ HELLASGRID.....	20
5.1. Διαδικασία απόκτησης πρόσβασης σε βήματα.....	20
5.1.1. Απόκτηση ψηφιακού πιστοποιητικού χρήστη.....	21
5.1.2. Απόκτηση πρόσβασης σε ένα User Interface του Hellas Grid.....	23
5.1.3. Εγγραφή σε Ιδεατό Οργανισμό (Virtual Organization – VO).....	23
5.2. Πρόσβαση στο User Interface.....	24
Εγκατάσταση και ρύθμιση PuTTY SSH Client για Windows.....	24
5.2.1. Μεταφορές αρχείων από / προς το UI.....	25
5.3. <i>Εγκατάσταση ψηφιακού πιστοποιητικού χρήστη στο UI.....</i>	27
5.3.1. Εξαγωγή ψηφιακού πιστοποιητικού χρήστη από τον Web Browser.....	27
5.3.2. Εξαγωγή ψηφιακού πιστοποιητικού χρήστη από τον Mozilla Firefox.....	27
5.3.3. Εξαγωγή ψηφιακού πιστοποιητικού χρήστη από τον Google Chrome.....	30
5.3.4. Μεταφορά αρχείου ψηφιακού πιστοποιητικού χρήστη στο UI.....	41
5.3.5. Μετατροπή ψηφιακού πιστοποιητικού σε μορφή .pem...	41
5.3.6. Ολοκλήρωση εγκατάστασης πιστοποιητικού χρήστη στο UI.....	43
5.3.7. Πληροφορίες εγκατεστημένου πιστοποιητικού χρήστη...	43
6. ΧΡΗΣΗ ΥΠΗΡΕΣΙΩΝ ΠΙΣΤΟΠΟΙΗΣΗΣ ΤΑΥΤΟΤΗΤΑΣ ΚΑΙ ΕΞΟΥΣΙΟΔΟΤΗΣΗΣ ΠΡΟΣΒΑΣΗΣ ΣΤΟ HELLASGRID.....	44
6.1. Πληρεξούσια πιστοποιητικά Grid (Grid Proxy Certificates) ..	44
6.1.1. Δημιουργία Proxy Certificate.....	44
6.1.2. Πληροφορίες Proxy Certificate.....	45
6.1.3. Ακύρωση Proxy Certificate.....	46
6.2. Πληρεξούσια πιστοποιητικά VOMS (VOMS Proxy Certificates)	46

6.2.1. Δημιουργία VOMS Proxy Certificate .....	46
6.2.2. Πληροφορίες VOMS Proxy Certificate .....	47
6.2.3. Ακύρωση VOMS Proxy Certificate.....	47
6.3. Υπηρεσία MyProxy .....	48
6.3.1. Χρήση υπηρεσίας MyProxy .....	48
6.3.2. Πληροφορίες χρήσης υπηρεσίας MyProxy .....	48
6.3.3. Διακοπή υπηρεσίας MyProxy.....	48
7. ΥΠΟΒΟΛΗ ΕΡΓΑΣΙΩΝ ΣΤΟ HELLASGRID .....	49
7.1. Η υποβολή εργασιών σε βήματα .....	49
7.2. Αρχεία Περιγραφής Εργασιών JDL.....	49
7.2.1. Executable.....	50
7.2.2. JobType.....	51
7.2.3. Arguments.....	51
7.2.4. StdInput.....	51
7.2.5. StdOutput .....	52
7.2.6. StdError .....	52
7.2.7. InputSandbox .....	52
7.2.8. OutputSandbox .....	53
7.2.9. CPUNumber .....	54
7.2.10. Requirements.....	55
7.3. ΔΙΑΧΕΙΡΙΣΗ ΕΡΓΑΣΙΩΝ ΣΤΟ GRID.....	55
7.3.1. Έλεγχος .jdl αρχείου και εμφάνιση διαθέσιμων CE.....	55
7.3.2. Υποβολή job στο WMS.....	56
7.3.3. Έλεγχος κατάστασης jobs .....	58
7.3.4. Λήψη αποτελεσμάτων από ολοκληρωμένα jobs .....	60
7.3.5. Ακύρωση job.....	61
7.3.6. Απλουστευμένο παράδειγμα υποβολής και διαχείρισης ενός job.....	61
7.4. MPI Jobs .....	65
7.4.1. Ο μηχανισμός MPI-START .....	65
8. ΔΙΑΧΕΙΡΙΣΗ ΔΕΔΟΜΕΝΩΝ ΣΤΟ GRID .....	69



8.1. Εισαγωγή .....	69
8.2. Βασικές έννοιες.....	70
8.2.1. Αναγνωριστικά αρχείων GUID.....	70
8.2.2. Αναγνωριστικά αρχείων PFN.....	71
8.2.3. Αναγνωριστικά αρχείων LFN .....	71
8.2.4. Ο Λογικός Κατάλογος Αρχείων (Logical File Catalog – LFC) .....	72
8.3. Διαχείριση αρχείων .....	72
8.3.1. Προαπαιτούμενα .....	72
8.3.2. Απαραίτητες μεταβλητές περιβάλλοντος στο UI .....	73
8.3.3. Ρύθμιση μεταβλητών περιβάλλοντος στο UI.....	73
8.3.4. Τα σετ εντολών διαχείρισης αρχείων.....	74
8.3.5. Λίστες αρχείων και υποκαταλόγων .....	75
8.3.6. Δημιουργία νέου υποκαταλόγου.....	75
8.3.7. Αποθήκευση αρχείου σε ένα SE .....	76
8.3.8. Δημιουργία αντιγράφων (replication) αρχείων .....	76
8.3.9. Επιστροφή αρχείου από το Grid στο UI .....	77
8.3.10. Διαγραφή αρχείων και υποκαταλόγων.....	77
9. ΥΠΗΡΕΣΙΑ ΠΛΗΡΟΦΟΡΙΩΝ .....	78
9.1. Εισαγωγή .....	78
9.2. Αναζήτηση γενικών πληροφοριών πόρων.....	79
9.2.1. Αναζήτηση πληροφοριών για υπολογιστικούς πόρους....	79
9.2.2. Αναζήτηση πληροφοριών για αποθηκευτικούς πόρους...	79
9.2.3. Αναζήτηση του κοντινότερου SE για κάθε CE.....	80
9.2.4. Αναζήτηση των τοπικών εξυπηρετητών LFC.....	81
9.3. Αναζήτηση πληροφοριών πόρων βάση κριτηρίων .....	81
9.3.1. Χαρακτηριστικά (attributes) .....	81
9.3.2. Αναζήτηση CE βάση κριτηρίων.....	82
9.3.3. Εμφάνιση διαθέσιμων πακέτων λογισμικού .....	83
ΕΠΕΞΗΓΗΣΗ ΣΥΝΤΟΜΟΓΡΑΦΙΩΝ.....	85
10. ΕΥΡΕΤΗΡΙΟ ΕΝΤΟΛΩΝ .....	87

11. ΕΥΡΕΤΗΡΙΟ ΕΙΚΟΝΩΝ .....	88
12. ΒΙΒΛΙΟΓΡΑΦΙΑ .....	89

# 1. ΠΑΡΑΛΛΗΛΗ ΚΑΙ ΚΑΤΑΝΕΜΗΜΕΝΗ ΕΠΕΞΕΡΓΑΣΙΑ

## 1.1. ΕΙΣΑΓΩΓΗ

---

Παρόλο που οι επιδόσεις των υπολογιστικών συστημάτων αυξάνονται με αλματώδεις ρυθμούς τα τελευταία 50 χρόνια, υπάρχουν ερευνητικοί και άλλοι τομείς των οποίων οι ανάγκες πάντα αδυνατούσαν να καλυφθούν από μεμονωμένα παραδοσιακά υπολογιστικά συστήματα. Η παράλληλη επεξεργασία, πρωτοεμφανιζόμενη σε ακριβά υπολογιστικά συστήματα υψηλών επιδόσεων (HPC – High Performance Computing) και εφαρμοζόμενη πια ακόμα και σε καθημερινούς οικιακούς υπολογιστές, αποτελεί έναν από τους συνηθέστερους τρόπους επίτευξης μεγαλύτερης υπολογιστικής και αποθηκευτικής ισχύος.

Τα τελευταία χρόνια, η ραγδαία ανάπτυξη του Διαδικτύου δημιούργησε τεράστιες απαιτήσεις σε αποθήκευση και διανομή δεδομένων και έδωσε με τη σειρά της σημαντική ώθηση στα συστήματα παράλληλης επεξεργασίας, παρουσιάζοντας μια νέα σειρά υπολογιστικών προβλημάτων που αναζητούσαν καινοτόμες λύσεις. Μια σειρά από νέες ανερχόμενες τεχνολογίες φαίνεται να αλλάζουν τον τρόπο που δουλεύουμε σήμερα με τους υπολογιστές παρέχοντας στο ευρύ κοινό υπηρεσίες που μέχρι πρότινος ήταν διαθέσιμες μόνο σε πολύ μικρές ομάδες ατόμων. Χαρακτηριστικό παράδειγμα αποτελεί το Cloud Computing που υπόσχεται πρόσβαση σε υπολογιστικούς πόρους τεράστιας ισχύος και αποθηκευτικού χώρου από οποιοδήποτε τερματικό περιορισμένων δυνατοτήτων, οπουδήποτε στον κόσμο.

## 1.2. Τι είναι η παράλληλη και η κατανεμημένη επεξεργασία

---

Η Παράλληλη και η Κατανεμημένη επεξεργασία στην επιστήμη υπολογιστών, είναι προσεγγίσεις για την επίλυση προβλημάτων βασισμένες στην αρχή ότι μεγάλα προβλήματα μπορούν συχνά να αναλυθούν σε μικρότερα, τα οποία στη συνέχεια επιλύονται ταυτόχρονα (παράλληλα). Οι διαφορές μεταξύ των δύο όρων είναι λεπτές, με έμφαση να δίνεται άλλοτε στον σχεδιασμό και την

ανάλυση αλγορίθμων, άλλοτε στην κατασκευή υποστηρικτικού λογισμικού και άλλοτε στη σχεδίαση των υποδομών υλικού που απαιτούνται για την επίτευξη του παραλληλισμού.

Η κατανεμημένη επεξεργασία αποτελεί ένα υποσύνολο της παράλληλης επεξεργασίας, στο οποίο όλοι οι επεξεργαστές έχουν ιδιωτικές, τοπικές μνήμες με ξεχωριστούς χώρους διευθύνσεων (είναι δηλαδή ανεξάρτητοι, δικτυωμένοι υπολογιστές) και παρέχουν στον χρήστη την ψευδαίσθηση του ενιαίου, μοναδικού συστήματος. Ο όρος «κατανεμημένο σύστημα» αναφέρεται τόσο στο υλικό (επεξεργαστές, μνήμες, δίκτυο) όσο και στο λογισμικό (λειτουργικό σύστημα, εφαρμογές) που είναι απαραίτητα για να υλοποιηθούν η κατανομή και αυτή η ψευδαίσθηση, με έμφαση όμως στο λογισμικό. Στα κατανεμημένα συστήματα, σε αντίθεση με τα παράλληλα, η εικόνα συνεκτικότητας των γεωγραφικά διεσπαρμένων πόρων είναι συνήθως σπουδαιότερος στόχος από την αύξηση των υπολογιστικών επιδόσεων που επιτυγχάνεται με τον παραλληλισμό. Δεν είναι σπάνιο μάλιστα να μη συμμετέχουν καν οι κόμβοι σε κάποιον από κοινού υπολογισμό αλλά να εκτελούν διαφορετικές επιμέρους εργασίες, παρουσιαζόμενοι όμως στο εξωτερικό περιβάλλον (π.χ. σε πελάτες οι οποίοι ζητούν υπηρεσίες) ως ενιαίο σύστημα. Αυτά ονομάζονται μη συνεκτικά κατανεμημένα συστήματα (π.χ. ο Παγκόσμιος Ιστός ή οι ομότιμες εφαρμογές ανταλλαγής αρχείων μέσω Διαδικτύου), σε αντίθεση με τα συνεκτικά κατανεμημένα συστήματα, τα οποία αξιοποιούνται περισσότερο ως συνήθη παράλληλα συστήματα για τη μεγιστοποίηση των υπολογιστικών επιδόσεων.

### 1.3. Ιστορία

---

Το ενδιαφέρον ανάπτυξης παράλληλων συστημάτων χρονολογείται από τα τέλη της δεκαετίας του 1950, με εξελίξεις που είχαν τη μορφή υπερυπολογιστών καθ' όλη τη δεκαετία του '60 και του '70. Οι υπερυπολογιστές αυτοί ήταν πολυεπεξεργαστές διαμοιρασμένης μνήμης, με πολλούς επεξεργαστές που εργάζονταν ταυτόχρονα σε κοινόχρηστα δεδομένα. Στα μέσα της δεκαετίας του 1980, ξεκίνησε ένα νέο είδος παράλληλης επεξεργασίας όταν το Caltech Concurrent Computation project κατασκεύασε έναν υπερυπολογιστή για επιστημονικές εφαρμογές αποτελούμενο από 64 επεξεργαστές Intel 8086/8087. Το σύστημα αυτό έδειξε ότι ακόμα και μικροεπεξεργαστές ευρέως διαθέσιμοι στο εμπόριο ήταν σε θέση να επιτύχουν σημαντική απόδοση. Αυτοί οι Μαζικά Παράλληλοι

Επεξεργαστές (MPPs – Massively Parallel Processors) κυριάρχησαν στην ανώτατη βαθμίδα της πληροφορικής, με τον υπερυπολογιστή ASCI Red να σπάει το φράγμα του ενός τρισεκατομμυρίου πράξεων κινητής υποδιαστολής (FLOPs – Floating Point Operations) ανά δευτερόλεπτο το 1997. Από τότε, τα Μαζικά Παράλληλα Επεξεργαστικά συστήματα συνέχισαν να αυξάνονται σε μέγεθος και ισχύ.

Ξεκινώντας στα τέλη της δεκαετίας του '80, οι συστοιχίες (Clusters) υπολογιστών ανταγωνίστηκαν και τελικά να εκτόπισαν τα Μαζικά Παράλληλα Επεξεργαστικά συστήματα σε πολλές εφαρμογές. Η συστοιχία είναι ένα είδος παράλληλων υπολογιστικών συστημάτων που αποτελούνται από μεγάλους αριθμούς υπολογιστών διαθέσιμους στο εμπόριο που συνδέονται μέσω τύπων δικτύου επίσης διαθέσιμων στο εμπόριο. Τα τελευταία χρόνια οι συστοιχίες αποτελούν τη κινητήριου δύναμη της πληροφορικής της επιστημονικής έρευνας όντας η κυρίαρχη αρχιτεκτονική στα κέντρα δεδομένων που δίνουν ώθηση στην σύγχρονη εποχή της πληροφορίας.

Επιπλέον, η παράλληλη επεξεργασία γνωρίζει σήμερα μεγάλη επιτυχία χάρη στους ευρέως διαδεδομένους πολυπύρηνους επεξεργαστές. Τα περισσότερα συστήματα desktop και laptop υπολογιστών διαθέτουν διπύρηνους ή τετραπύρηνους επεξεργαστές και σύντομα επεξεργαστές με ακόμα περισσότερους πυρήνες θα είναι διαθέσιμοι με χαμηλό κόστος. Οι κατασκευαστές τσιπ ρίχνουν σημαντικό βάρος στην αύξηση της συνολικής απόδοσης επεξεργασίας με την προσθήκη επιπλέον πυρήνων CPU. Ένα από τα βασικά πλεονεκτήματα αυτής της στρατηγικής είναι ότι η αύξηση των επιδόσεων μέσω της παράλληλης επεξεργασίας μπορεί να επιφέρει σημαντικά μεγαλύτερη ενεργειακή απόδοση σε σχέση με παλαιότερες μεθόδους όπως η αύξηση της συχνότητας ρολογιού του μικροεπεξεργαστή. Η συνεχιζόμενη κλιμάκωση του αριθμού τρανζίστορ ανά επεξεργαστή που προβλέπει ο Νόμος του Moore εξασφαλίζει τη κλιμάκωση της μετάβασης από λίγους πυρήνες σε περισσότερους για αρκετό καιρό ακόμα.

Ακόμα μια καινοτομία που προωθεί την παράλληλη επεξεργασία σήμερα είναι η ραγδαία ανάπτυξη των δικτύων και η σχετικά φθηνή δημιουργία υπερ-ταχέων τυποποιημένων δικτύων τοπικής εμβέλειας ή ακόμη και ευρείας περιοχής. Τέτοια παραδείγματα είναι το GigaBit Ethernet, το ATM, και το WDM. Η διάδοση τους έδωσε τη δυνατότητα ανάπτυξης ενός νέου υπολογιστικού μοντέλου όπου

πολλοί, απλοί υπολογιστές γραφείου συνδεδεμένοι μέσα από το ταχύ δίκτυο μπορούν να λειτουργούν ως μια μεγάλη, εικονική παράλληλη μηχανή. Αυτή η τεχνολογία είναι γνωστή ως υπολογιστικά συστήματα Πλέγματος (Grid Computing) και αποτελεί μια γέφυρα μεταξύ παράλληλης και κατακευματισμένης τεχνολογίας.

## **2.ΤΕΧΝΟΛΟΓΙΕΣ ΠΛΕΓΜΑΤΟΣ (GRID)**

### **2.1. Τι είναι οι τεχνολογίες πλέγματος**

---

Τα υπολογιστικά συστήματα πλέγματος (grid) αποτελούν ένα είδος κατακευματισμένων υπολογιστικών συστημάτων. Ένα grid συνδυάζει τους πόρους των υπολογιστών που το αποτελούν για να εκτελέσει πολύ μεγάλες υπολογιστικές εργασίες που θα έπαιρναν πολύ χρόνο ή δεν θα μπορούσαν καν να ολοκληρωθούν από ένα μόνο υπολογιστή. Μια από τις κύριες στρατηγικές των grids είναι η διαίρεση μιας υπολογιστικής εφαρμογής σε κομμάτια και ο διαμοιρασμός αυτών των επιμέρους κομματιών σε πολλούς (ακόμα και χιλιάδες) υπολογιστές.

Το μέγεθος των grids, δηλαδή ο αριθμός των υπολογιστικών συστημάτων που τα απαρτίζουν, ποικίλλει. Υπάρχουν μικρά σε μέγεθος που απαρτίζονται για παράδειγμα από τους σταθμούς εργασίας (workstations) του δικτύου μιας επιχείρησης, αλλά και μεγάλα που αποτελούν συνεργασίες μεταξύ πολλών οργανισμών και εκτίθενται σε ένα μεγάλο αριθμό συνδεδεμένων μεταξύ τους (συνήθως μέσω του Internet) δικτύων.

### **2.2. Ομοιότητες και διαφορές με άλλους τύπους παράλληλων υπολογιστικών συστημάτων**

---

Παρόλο που έμμεσα το grid μοιάζει με ένα εικονικό υπέρ-υπολογιστή (supercomputer), αφού αποτελείται από πολλούς επεξεργαστές (CPUs) που δουλεύουν παράλληλα, έχει και κάποιες σημαντικές διαφορές. Το υλικό που αποτελεί το grid, είναι ολοκληρωμένοι υπολογιστές του εμπορίου που διαθέτουν πέρα από έναν ή περισσότερους κεντρικούς επεξεργαστές, τροφοδοτικό ρεύματος, αποθηκευτικά μέσα και τέλος διεπαφή δικτύου (network interface)

με την οποία και επικοινωνούν μεταξύ τους. Αντιθέτως, ένα supercomputer αποτελείται από πολλούς κεντρικούς επεξεργαστές που επικοινωνούν μεταξύ τους με ένα υψηλής ταχύτητας δίκτυο υπολογιστών.

Αρκετές ομοιότητες υπάρχουν επίσης μεταξύ των grid συστημάτων και τις συστοιχίες υπολογιστικών συστημάτων (clusters). Οι σημαντικότερες διαφορές τους είναι ότι τα grids αποτελούνται από περισσότερο ετερογενή συστήματα, με χαλαρή μεταξύ τους συνδεσιμότητα (loosely-coupled), που είναι σε πολλές περιπτώσεις γεωγραφικά διασκορπισμένα.

### **2.3. Ενδιάμεσο λογισμικό (middleware)**

---

Το στρώμα του λογισμικού που μετατρέπει ένα δίκτυο ετερογενών υπολογιστικών συστημάτων σε ένα grid ονομάζεται ενδιάμεσο λογισμικό (middleware). Το middleware λειτουργεί παρέχοντας ομοιογενή πρόσβαση σε ανομοιογενείς υπολογιστικούς και αποθηκευτικούς πόρους και μπορεί να θεωρηθεί ως ένα επίπεδο μεταξύ του υλικού και του λογισμικού εφαρμογών (application software).

Τα πακέτα middleware που υπάρχουν σήμερα αναλαμβάνουν να καλύψουν αρκετές ανάγκες ενός grid σε διαφορετικό βαθμό το καθένα, χρησιμοποιώντας μια πληθώρα από νέες και ήδη υπάρχουσες τεχνολογίες. Σε κάποιες υλοποιήσεις το middleware είναι τόσο ολοκληρωμένο ώστε καλύπτει όλες τις ανάγκες ενός grid, ενώ σε άλλες οι διαχειριστές είναι ελεύθεροι να επιλέξουν ανεξάρτητο λογισμικό για ορισμένους τομείς.

### **2.4. Δομή των grids**

---

Ένα grid απαιτεί ένα ελάχιστο αριθμό από βασικές υπηρεσίες για να λειτουργήσει κατάλληλα και να διακρίνεται από άλλους τύπους υπολογιστικών συστημάτων. Παρόλο που οι βασικές ανάγκες της κοινότητας που θα χρησιμοποιήσει το grid μπορεί να υπαγορεύουν επιπρόσθετη ή πιο λεπτομερή λειτουργικότητα, οι παρακάτω υπηρεσίες grid αποτελούν μια συνήθη βάση:

- Διεπαφή χρηστών (User Interface – UI)

- Διαχείριση πρόσβασης (πιστοποίηση και εξουσιοδότηση)
- Εντοπισμός και διαχείριση πόρων
- Διαχείριση δεδομένων
- Διαχείριση και χρονοδιαγράμματα εργασιών
- Διαχείριση grid
- Επίβλεψη (monitoring)

## 2.5. Ιστορία

---

Αν και η ιστορία των καταμετρημένων υπολογιστικών συστημάτων ξεκινά από τη δεκαετία του 1960, η ιδέα των υπολογιστικών συστημάτων πλέγματος καθώς και μια σειρά από απαραίτητες τεχνολογίες αναπτύχθηκαν αρχικά από τους Ian Foster, Carl Kesselman, και Steve Tuecke με τη δημιουργία του πακέτου λογισμικού ανοιχτού κώδικα Globus Toolkit το 1997. Το project αυτό συνεχίζεται μέχρι σήμερα από το Globus Alliance (μια διεθνή συνεργασία με στόχο την έρευνα και ανάπτυξη θεμελιωδών τεχνολογιών grid) και αποτελεί τη βάση πολλών σημαντικών grid.

## 3. ΓΕΝΙΚΑ ΣΤΟΙΧΕΙΑ ΓΙΑ ΤΟ EGEE

### 3.1. Εισαγωγή στο EGEE

---

Το Enabling Grids for E-sciencE (EGEE) είναι το κυριότερο ευρωπαϊκό project πλέγματος υπολογιστικών συστημάτων. Η υπολογιστική υποδομή που παρέχει χρησιμοποιείται από περισσότερους από 13.000 ερευνητές ανά τον κόσμο, σε πλήθος επιστημονικών πεδίων όπως φυσική υψηλών ενεργειών, βιολογία και σεισμολογία.

Αν και χρηματοδοτείται κυρίως από την Ευρωπαϊκή Ένωση, η αποστολή του είναι παγκόσμια και η συνεισφορά που δέχεται από τις Η.Π.Α., την Αυστραλία, την Ταϊβάν και άλλα μη Ευρωπαϊκά κράτη είναι σημαντική. Το EGEE consortium αποτελείται από 42 δικαιούχους από τον ακαδημαϊκό και τον ιδιωτικό τομέα οι οποίοι



προκύπτουν από ομαδοποίηση ανά κράτος περισσότερων από 120 επιμέρους συνεργατών. Αυτού του είδους η ιεραρχική ομαδοποίηση αποτελεί ορόσημο για την οργάνωση και τη διατήρηση ενός μοντέλου υποδομής πλέγματος.

## 3.2. Ιστορία

---

Το EGEE αποτελεί επέκταση του δικτύου έρευνας GEANT2 της Ε.Ε. και επωφελείται από την εξειδίκευση που προήλθε από προγενέστερα projects της Ε.Ε. όπως το European DataGrid καθώς και πρωτοβουλίες επιμέρους κρατών όπως το αγγλικό e-Science, το INFN Grid, το Nordugrid και το αμερικανικό US Trillium. Η υποδομή παρέχει διαλειτουργικότητα με άλλα grid ανά τον κόσμο συνεισφέροντας έτσι στην προσπάθεια δημιουργίας μιας παγκόσμιας υποδομής.

Η πρώτη φάση του EGEE ξεκίνησε το Μάρτιο του 2004 με αρχική ονομασία "Enabling Grids for E-science in Europe" η οποία τελικά τροποποιήθηκε όπως είναι σήμερα, όταν άρχισαν να συμμετέχουν μη ευρωπαϊκά κράτη στο project. Η δεύτερη φάση (EGEE-II) ξεκίνησε τον Απρίλιο του 2006 αμέσως μετά το τέλος της πρώτης και τελείωσε στις 30 Απριλίου του 2008. Σήμερα διανύουμε την τρίτη φάση του project (EGEE-III) η οποία ξεκίνησε στις 1 Μαΐου του 2008 και αναμένεται να ολοκληρωθεί στις 30 Απριλίου του 2010.

## 3.3. Στόχοι

---

Ο κυριότερος στόχος του EGEE project από τη δημιουργία του είναι να παρέχει σε ερευνητές, τόσο του ακαδημαϊκού όσο και του βιομηχανικού τομέα, δυνατότητα πρόσβασης σε υπολογιστικούς και αποθηκευτικούς πόρους μεγάλης κλίμακας, ανεξαρτήτου γεωγραφικής τοποθεσίας. Επίσης στόχο αποτελεί η προσέλκυση ενός ευρέως φάσματος νέων χρηστών στο grid.

Επιμέρους στόχοι του project στην τρίτη φάση που διανύει σήμερα είναι:

- Η επέκταση και βελτίωση της υποδομής του EGEE grid μέσω της διαρκής δράσης, συμπεριλαμβάνοντας περισσότερους

πόρους και περισσότερους χρήστες από την επιστημονική κοινότητα.

- Η προετοιμασία για μετανάστευση από ένα project-based μοντέλο σε μια διατηρητέα ομοσπονδιακή υποδομή βασισμένη στις επιμέρους εθνικές πρωτοβουλίες grid.

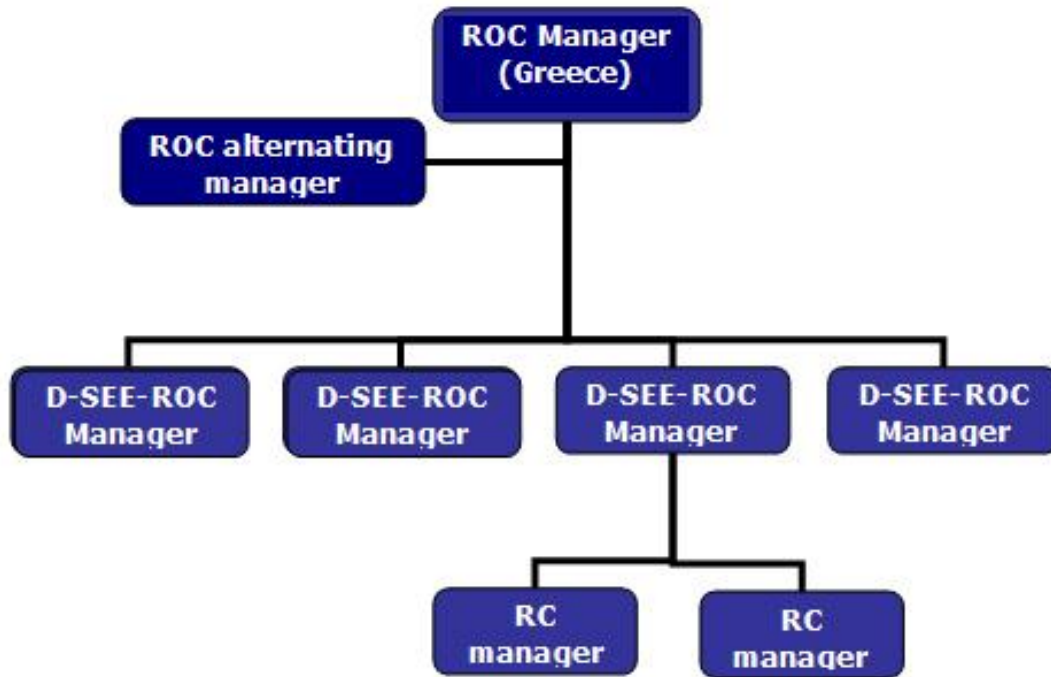
### 3.4. Οργάνωση

---

Το EGEE project εκμεταλλεύεται ένα grid καταμεμημένο σε περισσότερα από 200 επιμέρους δίκτυα (sites) ανά τον κόσμο, με περισσότερες από 40000 CPUs και 5 PB αποθηκευτικού χώρου. Η ποσότητα των πόρων που διαθέτει το κάθε ένα από αυτά τα sites στο grid ποικίλλει και το δίκτυο που χρησιμοποιείται για τη μεταξύ τους διασύνδεση είναι το Internet. Για αποτελεσματικότερη οργάνωση του EGEE, τα sites τελικά ομαδοποιούνται ανά γεωγραφικές περιοχές που συντονίζονται από επιμέρους Τοπικά Κέντρα Επιχειρήσεων (Regional Operations Centers – ROCs).

Κάθε κρατική πρωτοβουλία grid είναι με τη σειρά της υπεύθυνη για τα Κέντρα Πόρων (Resource Centers – RCs) στην χώρα που αντιπροσωπεύει. Τέλος κάθε RC έχει δικούς του διαχειριστές (Administrators) και τεχνικούς υπεύθυνους για την σωστή λειτουργία του και τη συμμόρφωση με τους κανόνες του grid.

Για παράδειγμα όλες οι πρωτοβουλίες grid των κρατών της Νοτιοανατολικής Ευρώπης όπως η Ελλάδα (HellasGrid), η Βουλγαρία, η Κροατία (CroGrid), η Τουρκία (Tr-Grid) και άλλες συγκεντρώνονται υπό την ομπρέλα του SEE (South Eastern Europe) ROC. Ο ελληνικός φορέας ΕΔΕΤ (GRNET - υπεύθυνος και για το HellasGrid) ο διαχειριστής του συγκεκριμένου ROC (ROC Manager), δηλαδή αναλαμβάνει να συντονίσει τους τομείς επιχειρήσεων, εκπαίδευσης και την προσπάθεια διασποράς στην περιοχή της Νοτιοανατολικής Ευρώπης. Ταυτόχρονα είναι και ο Εναλλακτικός Διαχειριστής του ROC (ROC Alternating Manager). Όλες οι χώρες που συμμετέχουν διαχειρίζονται μαζί τη λειτουργία του grid δημιουργώντας συνολικά ένα Καταμεμημένο SEE ROC (Distributed SEE ROC – D-SEE-ROC). Κάθε χώρα που συμμετέχει έχει και ένα επιμέρους Διαχειριστή D-SEE-ROC (τον τοπικό εκπρόσωπο του ROC) και κάθε RC έχει και το δικό του RC διαχειριστή (RC Manager).



Εικόνα 3-1 - Οργανωτική δομή SEE ROC

### 3.5. Το HellasGrid

Το HellasGrid είναι η ελληνική υποδομή grid που εντάσσεται στο EGEE. Είναι η μεγαλύτερη υποδομή πλέγματος στην νοτιοανατολική Ευρώπη, και μια από τις σταθερότερες υποδομές σε Ευρωπαϊκό επίπεδο. Πόροι της υποδομής χρησιμοποιούνται από Έλληνες ερευνητές και από ευρωπαϊκά προγράμματα. Τα τελευταία χρόνια σημαντικός και αυξανόμενος αριθμός χρηστών από διάφορα επιστημονικά πεδία (φυσική, βιοτεχνολογία, υπολογιστική χημεία, πληροφορική, μετεωρολογία, κ.α.) χρησιμοποιεί την υποδομή HellasGrid για τις υπολογιστικές τους ανάγκες. Η πρόσβαση είναι ελεύθερη στην ελληνική ακαδημαϊκή και ερευνητική κοινότητα με μία απλή διαδικασία εγγραφής.

Στόχοι του HellasGrid είναι:

- Η παροχή υπολογιστικών υπηρεσιών υψηλής απόδοσης (High Performance Computing, High Throughput Computing) στην ελληνική ακαδημαϊκή και ερευνητική κοινότητα.
- Η εγκατάσταση, λειτουργία και υποστήριξη 6 υπολογιστικών και αποθηκευτικών κόμβων ανά την Ελλάδα – HellasGrid

(Αθήνα (3), Θεσσαλονίκη, Πάτρα, Ηράκλειο). Λειτουργία Αρχής Πιστοποίησης (Certification Authority) HellasGrid για έκδοση πιστοποιητικών χρηστών.

- Η υποστήριξη χρηστών στη μεταφορά των εφαρμογών τους στην υποδομή.
- Η υποστήριξη εφαρμογών και η παροχή βιβλιοθηκών και λογισμικού υποστήριξης στην κοινότητα της.

## **4.ΘΕΜΕΛΙΩΔΗ ΣΤΟΙΧΕΙΑ ΑΡΧΙΤΕΚΤΟΝΙΚΗΣ ΤΟΥ EGEE**

### **4.1. Εργασίες (Jobs)**

---

Οι χρήστες του EGEE χρησιμοποιούν το Grid για την εκτέλεση των εφαρμογών τους με τη μορφή εργασιών (Jobs) . Κάθε job που υποβάλλεται στο Grid, περιγράφει κάποια απαραίτητα χαρακτηριστικά του και προσδιορίζει τις απαιτήσεις που έχει σε πόρους της υποδομής. Βάση αυτών των στοιχείων, το middleware προγραμματίζει ένα χρονοδιάγραμμα εκτέλεσης των jobs ώστε να πετυχαίνει τη βέλτιστη αξιοποίηση των διαθέσιμων πόρων παρέχοντας παράλληλα τους μικρότερους δυνατούς χρόνους ολοκλήρωσης των jobs.

### **4.2. Ασφάλεια – ψηφιακά πιστοποιητικά**

---

Η Υποδομή Ασφάλειας του Grid (Grid Security Infrastructure – GSI) αναλαμβάνει τα ζητήματα πιστοποίησης (authentication) και επικοινωνιών μέσω ενός ανοιχτού δικτύου (στην προκειμένη περίπτωση του Internet). Βασίζεται κυρίως σε ευρέως διαδεδομένες τεχνολογίες όπως η κρυπτογράφηση δημοσίων κλειδιών (public key encryption), τα ψηφιακά πιστοποιητικά X.509 και το πρωτόκολλο επικοινωνίας SSL (Secure Sockets Layer).

Για να μπορέσει ένας χρήστης να πιστοποιήσει την ταυτότητά του (authentication) στους πόρους του grid και άρα να έχει πρόσβαση σε αυτούς, είναι απαραίτητο να έχει ένα ψηφιακό πιστοποιητικό X.509 από μια Αρχή Πιστοποίησης (Certification Authority – CA) που να

αναγνωρίζει και να εμπιστευτεί το EGEE. Με τον ίδιο τρόπο οι επιμέρους πόροι του grid διαθέτουν και αυτοί με τη σειρά τους ψηφιακά πιστοποιητικά X.509 για να πιστοποιούν την ταυτότητά τους στους χρήστες καθώς και σε άλλες υπηρεσίες.

### **4.3. Ιδεατοί Οργανισμοί (VOs)**

---

Οι Ιδεατοί Οργανισμοί (Virtual Organizations – VOs) αποτελούν βασική έννοια της οργάνωσης των grids. Όπως υπονοεί η ονομασία τους είναι εικονικές ομάδες χρηστών που έχουν μεταξύ τους ορισμένα κοινά σημεία όπως στόχους, ενδιαφέροντα, ανάγκες κ.α.

Στην περίπτωση του EGEE υπάρχουν εξειδικευμένα VOs για ένα πλήθος πεδίων επιστημονικής έρευνας καθώς και κάποια βοηθητικά, γενικότερης φύσεως στα οποία εντάσσονται χρήστες ανάλογα με τη γεωγραφική περιοχή από την οποία προέρχονται (για παράδειγμα οι χρήστες του HellasGrid αποκτούν γίνονται κατά την εγγραφή τους αυτόματα μέλη στο VO Νοτιοανατολικής Ευρώπης – South Eastern Europe SEE VO).

Κάθε VO έχει πρόσβαση σε συγκεκριμένους πόρους και υπηρεσίες που έχουν καθοριστεί κατά τη δημιουργία του ανάλογα με τις ανάγκες των χρηστών του. Για να μπορέσει ένας χρήστης να έχει πρόσβαση στους πόρους του grid (π.χ. να υποβάλλει ένα job ή να αποθηκεύσει δεδομένα σε κάποιο storage element) είναι απαραίτητο να είναι μέλος κάποιου VO. Καθώς το κάθε VO έχει δικαίωμα να επιλέγει τα μέλη του, η διαδικασία εγγραφής δεν η ίδια για όλα τα VOs και συνήθως περιγράφεται στις επιμέρους ιστοσελίδες τους. Στη πύλη επιχειρήσεων του EGEE στη διεύθυνση <http://cic.gridops.org/index.php?section=home&page=volist#12> διατηρείται μια λίστα με όλα τα υπάρχοντα VOs και ορισμένες γενικές πληροφορίες γύρω από το καθένα.

Σε περίπτωση που μια ομάδα χρηστών θεωρεί ότι το αντικείμενο έρευνάς της δεν καλύπτεται από κανένα ήδη υπάρχον VO είναι δυνατή η δημιουργία ενός νέου. Το πρώτο βήμα για τη δημιουργία ενός νέου VO είναι η συμπλήρωση της φόρμας αίτησης καταχώρησης στην παρακάτω ηλεκτρονική διεύθυνση:

<https://cic.gridops.org/index.php?section=vo&page=newvoregistration>

Να σημειωθεί ότι κάθε νέο VO που δημιουργείται αναμένεται να παρέχει στην υποδομή του EGEE υπολογιστικούς πόρους ισοδύναμους με αυτούς που καταναλώνει ανά μέσο όρο (αν και ο κανόνας αυτός είναι λιγότερο αυστηρός σε λίγες εξαιρετικές περιπτώσεις).

#### **4.4. Υπηρεσία Συμμετοχής σε Εικονικούς Οργανισμούς (VOMS)**

---

Η Υπηρεσία Συμμετοχής σε Εικονικούς Οργανισμούς (Virtual Organization Membership Service - VOMS) είναι ένα σύστημα διαχείρισης δεδομένων εξουσιοδότησης σε συνεργασίες πολλών επιμέρους οργανισμών. Το VOMS παρέχει μια βάση δεδομένων με «ρόλους» και «δυνατότητες» χρηστών και ένα σετ εργαλείων πρόσβασης και διαχείρισης της βάσης δεδομένων αυτής. Επιπλέον παρέχει εργαλεία που βασίζονται τα στοιχεία της βάσης δεδομένων για να παρέχουν διαπιστευτήρια για τους χρήστες του Grid ώστε να μπορούν να έχουν πρόσβαση στους πόρους του.

Οι διαχειριστές των VOs φροντίζουν να παρέχουν «ρόλους» στους χρήστες των VOs και «δυνατότητες» στους πόρους που είναι διαθέσιμοι στα VOs και να καθορίζουν τις «δυνατότητες» που έχει ο κάθε «ρόλος». Για να μπορέσουν οι χρήστες να έχουν πρόσβαση στις υπηρεσίες του Grid, θα πρέπει πρώτα να λάβουν τα απαραίτητα διαπιστευτήρια από το VOMS (τα οποία δίνονται με τη μορφή προσωρινών ψηφιακών πιστοποιητικών x509) βάση των οποίων αναγνωρίζονται οι «ρόλοι» των χρηστών και οι «δυνατότητες» που έχουν μέσα στο σύστημα.

#### **4.5. Η Διεπαφή Χρηστών (UI)**

---

Το σημείο πρόσβασης των χρηστών στο grid είναι η Διεπαφή Χρηστών (User Interface – UI). Ένα UI μπορεί να είναι ένας οποιοσδήποτε υπολογιστής με πρόσβαση στο Internet, στον οποίο ο χρήστης έχει λογαριασμό και έχει εγκατεστημένο το ψηφιακό του πιστοποιητικό. Μέσω του UI ο χρήστης μπορεί να πιστοποιήσει την ταυτότητά του στο grid και να λάβει εξουσιοδότηση να εκμεταλλευτεί τους πόρους και τις υπηρεσίες του EGEE.

Τα UIs του EGEE χρησιμοποιούν λειτουργικό σύστημα Linux (συνήθως τη διανομή Scientific Linux που έχουν δημιουργήσει το CERN και το Fermilab την οποία προτείνει το gLite) το οποίο τρέχει την κατάλληλα διαμορφωμένη έκδοση του gLite middleware. Το gLite παρέχει εργαλεία μέσω της γραμμής εντολών (CLI) για την εκτέλεση βασικών λειτουργιών στο grid όπως:

- Εμφάνιση όλων των πόρων που είναι κατάλληλη για την εκτέλεση ενός job.
- Υποβολή jobs για εκτέλεση.
- Ακύρωση job.
- Ανάκτηση αποτελεσμάτων από jobs που ολοκληρώθηκαν.
- Εμφάνιση της κατάστασης jobs που έχουν υποβληθεί για εκτέλεση.
- Αντιγραφή και διαγραφή αρχείων στο grid.
- Ανάκτηση κατάστασης διαφορετικών ειδών πόρων από το Σύστημα Πληροφοριών (Information System).

Επιπλέον, μέσω του UI είναι διαθέσιμες οι Διεπαφές Προγραμματισμού Εφαρμογών (Application Programming Interfaces – APIs) του EGEE για την ανάπτυξη εφαρμογών που εκμεταλλεύονται τις υπηρεσίες του grid (grid-enabled applications).

## 4.6. Το Υπολογιστικό Στοιχείο (CE)

---

Στην ορολογία των grid ένα Υπολογιστικό Στοιχείο (Computing Element – CE) είναι ένα σύνολο υπολογιστικών πόρων (π.χ. ένα cluster, ένα δίκτυο υπολογιστών). Το CE περιλαμβάνει μια Πύλη προς το Grid (Grid Gate – GG), που δρα ως μια διεπαφή γενικής χρήσης προς το cluster, ένα Σύστημα Διαχείρισης Τοπικών Πόρων (Local Resource Management System – LRMS ή αλλιώς Batch System), και το ίδιο το cluster που είναι μια συλλογή από Κόμβους-Εργάτες (Worker Nodes – WNs), οι υπολογιστές στους οποίους τελικά εκτελούνται τα διάφορα jobs.

Το Grid Gate είναι υπεύθυνο για την αποδοχή των jobs και τη διανομή τους, μέσω του Local Resource Management System, στα Worker Nodes προς εκτέλεση. Τα Worker Nodes συνήθως έχουν

εγκατεστημένες τις ίδιες εντολές και βιβλιοθήκες που υπάρχουν και στο User Interface, εκτός από αυτές που απευθύνονται προς τους χρήστες και έχουν να κάνουν με τη διαχείριση εργασιών. Σε ορισμένα sites υπάρχει εγκατεστημένο λογισμικό που απευθύνεται ειδικά για κάποια VOs. Συνήθως αυτό βρίσκεται σε ένα Διαμοιρασμένο Σύστημα Αρχαιοθήκης (Shared File System) που είναι προσβάσιμο από όλα τα Worker Nodes.

#### **4.7. Το Αποθηκευτικό Στοιχείο (SE)**

---

Ένα Αποθηκευτικό Στοιχείο (Storage Element - SE) παρέχει ομοιόμορφη πρόσβαση σε ανομοιογενείς αποθηκευτικούς πόρους. Το Storage Element μπορεί να ελέγχει απλούς εξυπηρετητές δίσκων (disk servers), μεγάλες συστοιχίες δίσκων (large disk arrays) ή ακόμα και Συστήματα Μαζικής Αποθήκευσης (Mass Storage Systems - MMS) βασισμένα σε μαγνητικές ταινίες (tape-based). Τα περισσότερα sites του EGEE παρέχουν τουλάχιστον ένα Storage Element.

#### **4.8. Η Υπηρεσία Πληροφοριών (IS)**

---

Η Υπηρεσία Πληροφοριών (Information Service - IS) παρέχει πληροφορίες σχετικά με τους πόρους του grid και την κατάστασή τους. Αυτές οι πληροφορίες είναι απαραίτητες για τη λειτουργία ολόκληρου του grid, αφού μέσω του Information Service γίνεται η αναζήτηση πόρων. Οι πληροφορίες αυτές χρησιμοποιούνται επίσης για επίβλεψη και συλλογή στατιστικών στοιχείων. Περισσότερα για την Υπηρεσία Πληροφοριών στο κεφάλαιο 9.

#### **4.9. Το Σύστημα Διαχείρισης Εργασιών (WMS)**

---

Το Σύστημα Διαχείρισης Εργασιών (Workload Management System - WMS) είναι ένα σετ εργαλείων του middleware, υπεύθυνα για τη διαχείριση των εργασιών του Grid και την αποτελεσματική κατανομή τους στους διαθέσιμους πόρους της υποδομής.

Το βασικό συστατικό του WMS είναι ο Διαχειριστής Εργασιών (Workload Manager - WM). Σκοπός του είναι να δέχεται και να ικανοποιεί αιτήσεις Διαχείρισης Jobs από τους πελάτες του. Για ένα



Job υπολογιστικής φύσεως υπάρχουν δύο είδη αιτήσεων: υποβολή και ακύρωση.

Το νόημα της υποβολής ενός Job είναι η μεταβίβαση της ευθύνης του Job αυτού στον WM. Ο WM με τη σειρά του θα μεταβιβάσει το Job στα κατάλληλα CEs για εκτέλεση και θα επιστρατεύσει τα κατάλληλα SEs για αποθήκευση δεδομένων, λαμβάνοντας υπόψη τις απαιτήσεις και τις προτιμήσεις που εκφράζονται στην περιγραφή του Job. Η τελική επιλογή πόρων που θα χρησιμοποιηθούν είναι το αποτέλεσμα μιας διαδικασίας αντιστοίχισης μεταξύ αιτήσεων υποβολής Jobs και διαθέσιμων πόρων.

#### **4.10. Διαχείριση Δεδομένων**

---

Η πρωταρχική μονάδα διαχείρισης δεδομένων στο grid είναι, όπως και στους παραδοσιακούς υπολογιστές, το αρχείο. Σε ένα περιβάλλον grid το κάθε αρχείο μπορεί να έχει πανομοιότυπα αντίγραφα του (replicas) σε πολλά διαφορετικά sites. Καθώς όλα αυτά τα αντίγραφα πρέπει να είναι ενημερωμένα και σύμφωνα μεταξύ τους, μετά τη δημιουργία ενός αρχείου στο grid αυτό μπορεί μόνο να αναγνωσθεί ή να σβηστεί αλλά όχι να τροποποιηθεί. Μια εκτενέστερη αναφορά στη διαχείριση δεδομένων του Grid πραγματοποιείται στο κεφάλαιο 8.

## **5. ΠΡΟΣΒΑΣΗ ΣΤΟ HELLASGRID**

### **5.1. Διαδικασία απόκτησης πρόσβασης σε βήματα**

---

Για την απόκτηση πρόσβασης στο HellasGrid είναι απαραίτητες οι παρακάτω διαδικασίες:

- Απόκτηση ψηφιακού πιστοποιητικού χρήστη.
- Απόκτηση πρόσβασης σε ένα από τα User Interfaces του Hellas Grid.
- Εγγραφή σε ένα Ιδεατό Οργανισμό (Virtual Organization - VO).

### 5.1.1. Απόκτηση ψηφιακού πιστοποιητικού χρήστη

Όπως αναφέρθηκε παραπάνω, για να έχει πρόσβαση ένας χρήστης στις υπηρεσίες του HellasGrid και του EGEE γενικότερα, είναι απαραίτητο να διαθέτει ένα προσωπικό ψηφιακό πιστοποιητικό τόσο για πιστοποίηση της ταυτότητάς του, όσο και για ασφαλή ανταλλαγή δεδομένων με την υποδομή. Τα ψηφιακά πιστοποιητικά των χρηστών έχουν διάρκεια ισχύος ενός έτους. Λίγο πριν τη λήξη της διάρκειας ισχύος του πιστοποιητικού του, ο χρήστης ειδοποιείται ώστε να ανανεώσει το πιστοποιητικό του μέσω της ιστοσελίδας της εθνικής Αρχής Πιστοποίησης (Certification Authority – CA). Μέσω της ίδιας ιστοσελίδας μπορεί να γίνει και η απόκτηση του ψηφιακού πιστοποιητικού, πατώντας στην πρώτη επιλογή που εμφανίζεται στη διεύθυνση <https://access.hellasgrid.gr/>.

Αφού ο ενδιαφερόμενος συμπληρώσει και αποστείλει μια ηλεκτρονική φόρμα με τα προσωπικά του στοιχεία, θα λάβει ένα e-mail που θα επιβεβαιώνει την καταχώρηση των στοιχείων του στη βάση δεδομένων του HellasGrid. Ο ενδιαφερόμενος οφείλει να επιβεβαιώσει ότι έλαβε το e-mail μέσω σχετικού link που συμπεριλαμβάνεται στο μήνυμα εντός επτά ημερών. Σε περίπτωση που δεν γίνει επιβεβαίωση στο συγκεκριμένο χρονικό διάστημα, η διαδικασία ακυρώνεται.

Μετά την επιβεβαίωση λήψης του e-mail, η διαδικασία απόκτησης του πιστοποιητικού συνεχίζεται και ζητείται από τον ενδιαφερόμενο να εγκαταστήσει το ψηφιακό πιστοποιητικό της αρχής πιστοποίησης HellasGrid CA στον περιηγητή ιστού (web browser) που θα χρησιμοποιεί για πρόσβαση στις web υπηρεσίες του HellasGrid. Η ελληνική αρχή πιστοποίησης λειτουργεί στο Τμήμα Φυσικής του Αριστοτελείου Πανεπιστημίου Θεσσαλονίκης.



### 5.1.2. Απόκτηση πρόσβασης σε ένα User Interface του Hellas Grid.

Οι χρήστες του HellasGrid μπορούν να δημιουργήσουν το δικό τους User Interface (UI) εγκαθιστώντας τα απαραίτητα πακέτα λογισμικού σε έναν υπολογιστή με λειτουργικό σύστημα Linux. Εάν επιθυμούν μπορούν όμως να χρησιμοποιήσουν και ένα από τα ήδη υπάρχοντα UIs του HellasGrid.

Αυτή τη στιγμή το Hellas Grid διαθέτει έξι UIs για εξυπηρέτηση χρηστών:

- Τρία στην Αθήνα  
(ui01.isabella.grnet.gr, ui02.isabella.grnet.gr,  
ui01.marie.hellasgrid.gr)
- Ένα στη Θεσσαλονίκη  
(ui01.afroditi.hellasgrid.gr)
- Ένα στο Ηράκλειο  
(ui01.ariagni.hellasgrid.gr)
- Ένα στην Πάτρα  
(ui01.kallisto.hellasgrid.gr)

Οι χρήστες μπορούν να ζητήσουν πρόσβαση στο κοντινότερό τους UI από την ιστοσελίδα <https://access.hellasgrid.gr/> πατώντας τη δεύτερη επιλογή (με την προϋπόθεση ότι διαθέτουν ήδη ψηφιακό πιστοποιητικό χρήστη).

### 5.1.3. Εγγραφή σε Ιδεατό Οργανισμό (Virtual Organization – VO)

Για να μπορέσει ένας χρήστης να εκτελέσει ουσιαστικές εργασίες στο Grid πρέπει να είναι μέλος ενός VO ανάλογα με το είδος της έρευνας με την οποία ασχολείται. Οι νέοι χρήστες του HellasGrid μπορούν να εγγραφούν στο VO της Νοτιοδυτικής Ευρώπης (South Eastern Europe VO – SEE VO) που δεν έχει συγκεκριμένο αντικείμενο έρευνας και δημιουργήθηκε ως σημείο εκκίνησης των νέων χρηστών από τη συγκεκριμένη γεωγραφική περιοχή. Η εγγραφή στο SEE VO μπορεί να πραγματοποιηθεί από την ιστοσελίδα <https://access.hellasgrid.gr/> πατώντας στην τρίτη επιλογή (με την προϋπόθεση ότι υπάρχει διαθέσιμο εγκατεστημένο ψηφιακό πιστοποιητικό χρήστη).

## 5.2. Πρόσβαση στο User Interface

---

Ο προτεινόμενος τρόπος πρόσβασης των χρηστών στα διάφορα UIs του HellasGrid είναι μέσω της λειτουργίας ασφαλούς κελύφους (Secure Shell – SSH). Ο συγκεκριμένος τρόπος πρόσβασης παρέχει στον χρήστη ένα τερματικό γραμμής εντολών (Command-Line Terminal) στο λειτουργικό σύστημα Linux του υπολογιστή στον οποίο είναι εγκατεστημένο το UI. Η τεχνολογία SSH προσπαθεί να διασφαλίσει ότι η σύνδεση μεταξύ δύο σημείων είναι ασφαλής (δηλαδή είναι δύσκολη η υποκλοπή των δεδομένων που μεταφέρονται) μέσω κρυπτογραφικών τεχνικών.

### Εγκατάσταση και ρύθμιση PuTTY SSH Client για Windows

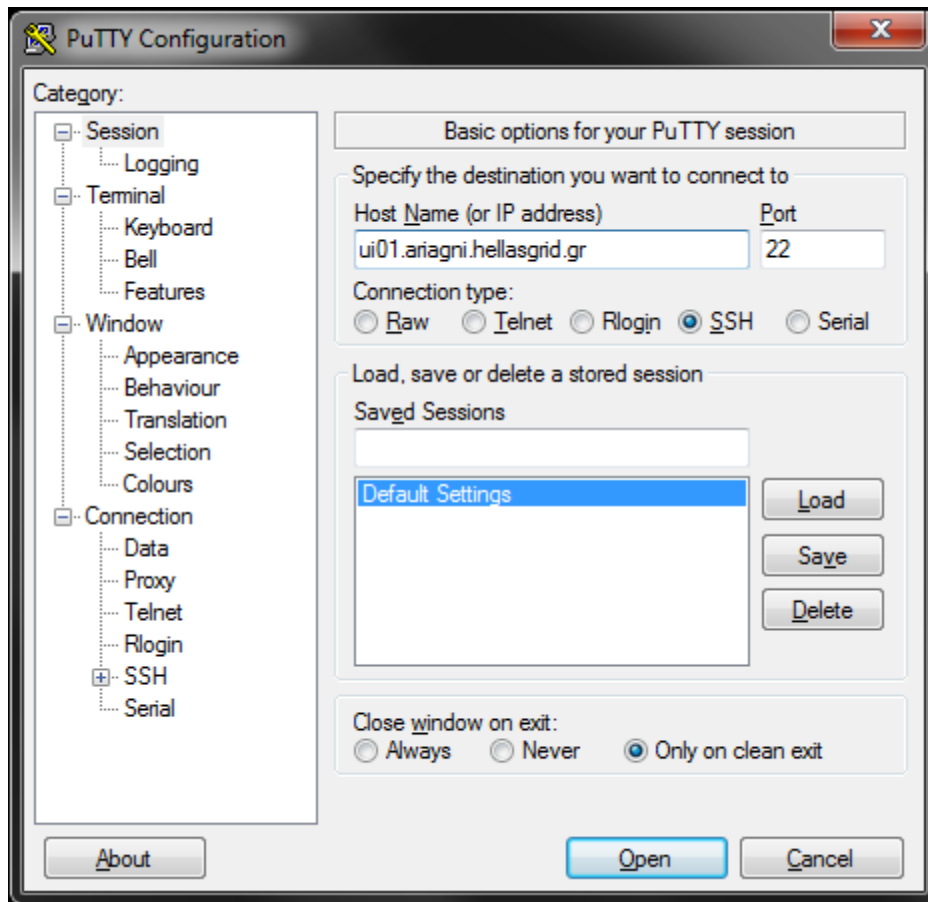
Για να συνδεθεί ένας χρήστης με το UI είναι απαραίτητη η χρήση μιας εφαρμογής πελάτη SSH (SSH Client Application). Αν και οι περισσότερες διανομές του λειτουργικού συστήματος Linux έχουν προεγκατεστημένους SSH Clients, σε περιβάλλον Windows είναι απαραίτητη η εγκατάσταση τέτοιου λογισμικού.

Προτείνεται η εγκατάσταση του client PuTTY καθώς παρέχει υποστήριξη όλων των λειτουργιών που θα χρειαστούν, είναι αρκετά εύχρηστος και παρέχεται δωρεάν για download στη παρακάτω ιστοσελίδα:

<http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>

Οι νέοι χρήστες του HellasGrid προτείνεται να κατεβάσουν το αρχείο **putty.zip** που περιέχει όλες τις απαραίτητες εφαρμογές και να το αποσυμπιέσουν σε έναν υποκατάλογο της επιλογής τους.

Για σύνδεση με ένα UI ο χρήστης πρέπει να εκτελέσει το αρχείο putty.exe και να επιλέξει από το μενού στα αριστερά την επιλογή Session. Στη συνέχεια πρέπει να εισάγει τη διεύθυνση του UI που θέλει να συνδεθεί στο πεδίο Host Name στα δεξιά χωρίς να μεταβάλλει τις υπόλοιπες επιλογές, όπως φαίνεται στο παράδειγμα της παρακάτω εικόνας.



**Εικόνα 5-2 - Ρυθμίσεις για σύνδεση στο UI ui01.ariagni.hellasgrid.gr μέσω PuTTY SSH**

Αφού γίνουν οι παραπάνω ρυθμίσεις και ο χρήστης πατήσει το κουμπί **Open** πραγματοποιείται σύνδεση στο επιλεγμένο UI όπου ο χρήστης πρέπει να εισάγει το όνομα χρήστη που του έχει δοθεί, καθώς και τον κωδικό πρόσβασής του για να βρεθεί στη γραμμή εντολών Linux του UI.

### **5.2.1. Μεταφορές αρχείων από / προς το UI**

Πέρα από τη σύνδεση στη γραμμή εντολών του UI με το PuTTY, είναι απαραίτητο για τους χρήστες να μπορούν να μεταφέρουν αρχεία από και προς το UI. Ο πιο άμεσος τρόπος είναι μέσω Ασφαλούς Αντιγραφής (Secure Copy – SCP). Το SCP όπως και το SSH χρησιμοποιεί κρυπτογραφικές τεχνικές για να επιτύχει ασφαλείς συνδέσεις. Η προτεινόμενη εφαρμογή πελάτη SCP (SCP Client) ονομάζεται PSCP και συμπεριλαμβάνεται μέσα στο πακέτο εφαρμογών PuTTY που περιγράφεται παραπάνω.

Το PSCP δεν παρέχεται με γραφικό περιβάλλον και χρησιμοποιείται από γραμμή εντολών ακόμα και σε περιβάλλον Windows. Η σύνταξη της εντολής PSCP είναι ως εξής:

### **Μεταφορά από το UI προς τον τοπικό υπολογιστή:**

```
pscp [επιλογές] [χρήστης@]διεύθυνση:προέλευση προορισμός
```

### **Μεταφορά από τον τοπικό υπολογιστή προς το UI:**

```
pscp [επιλογές] προέλευση [χρήστης@]διεύθυνση: προορισμός
```

Εάν για παράδειγμα θέλουμε να αντιγράψουμε στον τοπικό υπολογιστή στο δίσκο C:\ το αρχείο test.txt από τον φάκελο του χρήστη **mkats** που βρίσκεται στο UI **ui01.ariagni.hellasgrid.gr** θα χρησιμοποιήσουμε την παρακάτω σύνταξη:

```
pscp mkats@ui01.ariagni.hellasgrid.gr:/home/mkats/test.txt c:\
```

#### **ΣΗΜΕΙΩΣΗ:**

Συνήθως ο φάκελος χρήστη στο Linux (άρα και στα διάφορα UIs) βρίσκεται στον υποκατάλογο /home/[όνομα χρήστη] όπως στο παράδειγμα όπου ο φάκελος του χρήστη mkats βρίσκεται στο /home/mkats.

Στο παραπάνω παράδειγμα, όπως και στις περισσότερες περιπτώσεις, το PSCP θα επιχειρήσει να συνδεθεί με το UI **ui01.ariagni.hellasgrid.gr** και καθώς ο λογαριασμός του χρήστη είναι προστατευμένος με κωδικό δεν θα τα καταφέρει. Όταν συμβαίνει αυτό το PSCP θα ζητήσει από το χρήστη να παρέχει τον κωδικό του πριν προχωρήσει αυτόματα στην αντιγραφή του αρχείου.

Σε ένα δεύτερο παράδειγμα όπου θα θέλαμε να μεταφέρουμε το αρχείο c:\example.txt από τον τοπικό υπολογιστή στο φάκελο του χρήστη **mkats** στο UI **ui01.ariagni.hellasgrid.gr** , η σύνταξη της εντολής θα ήταν η εξής:

```
pscp c:\example.txt mkats@ui01.ariagni.hellasgrid.gr/home/mkats/
```

Και στο δεύτερο παράδειγμα το pscp θα ζητούσε τον κωδικό χρήστη πριν προχωρήσει σε αντιγραφή του αρχείου καθώς η πρόσβαση σε ένα από τα δύο σημεία (συγκεκριμένα στον απομακρυσμένο server - το UI) περιορίζεται με ονόματα και κωδικούς χρηστών.

### 5.3. Εγκατάσταση ψηφιακού πιστοποιητικού χρήστη στο UI

---

Αφού ο χρήστης αποκτήσει το προσωπικό ψηφιακό πιστοποιητικό του, εγγραφεί σε ένα VO και αποκτήσει πρόσβαση σε ένα UI, είναι απαραίτητο να προβεί σε ορισμένες ρυθμίσεις του UI του ώστε να είναι σε θέση να χρησιμοποιήσει την υποδομή του HellasGrid απροβλημάτιστα. Το πρώτο βήμα σε αυτή τη διαδικασία είναι η εγκατάσταση του προσωπικού ψηφιακού πιστοποιητικού του χρήστη στο UI.

Όπως αναφέρεται και παραπάνω το ψηφιακό πιστοποιητικό χρήστη είναι απαραίτητο για ταυτοποίηση του χρήστη και ασφαλή επικοινωνία με σχεδόν όλες τις υπηρεσίες του Grid. Συνεπώς ο χρήστης είναι αναγκασμένος να παρέχει το πιστοποιητικό του πριν αρχίσει να επικοινωνεί με τις κυριότερες υπηρεσίες της υποδομής. Για το λόγο αυτό, το πιστοποιητικό του χρήστη (σε μορφή .pem) πρέπει να εγκατασταθεί στον υποκατάλογο **.globus/** που βρίσκεται μέσα στον αρχικό φάκελο του χρήστη.

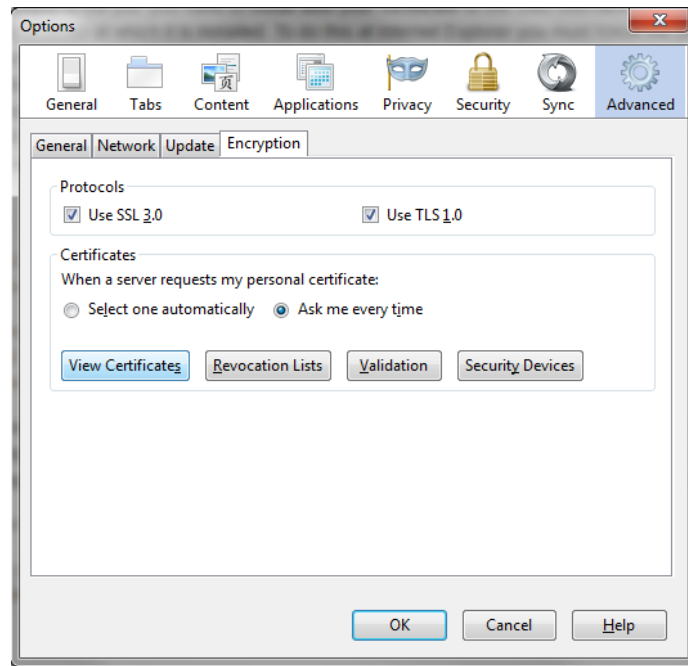
#### 5.3.1. Εξαγωγή ψηφιακού πιστοποιητικού χρήστη από τον Web Browser

Οι χρήστες του HellasGrid λαμβάνουν το ψηφιακό πιστοποιητικό τους αυτόματα μέσω εγκατάστασης του πιστοποιητικού στον Web Browser τους αμέσως μετά την απόκτησή του ώστε να έχουν πρόσβαση στις Web υπηρεσίες του HellasGrid. Για να μπορέσουν να το εισάγουν στο UI τους, θα πρέπει πρώτα να το εξαγάγουν από τον Web Browser που το έχουν εγκαταστήσει. Η διαδικασία εξαγωγής του πιστοποιητικού περιγράφεται παρακάτω για τους τρεις επικρατέστερους Web Browsers.

#### 5.3.2. Εξαγωγή ψηφιακού πιστοποιητικού χρήστη από τον Mozilla Firefox

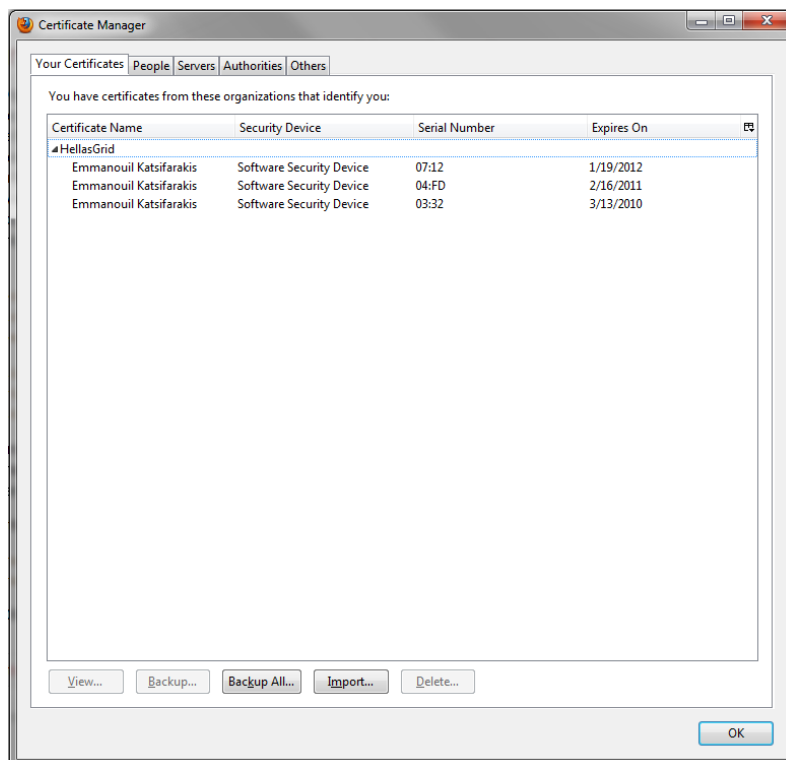
Ανοίγουμε το παράθυρο ρυθμίσεων από την επιλογή **Options** στο menu **Tools** του Mozilla Firefox. Πατάμε το κουμπί **Advanced** στην πάνω δεξιά γωνία και επιλέγουμε το tab encryption από κάτω (Εικόνα 5-3).





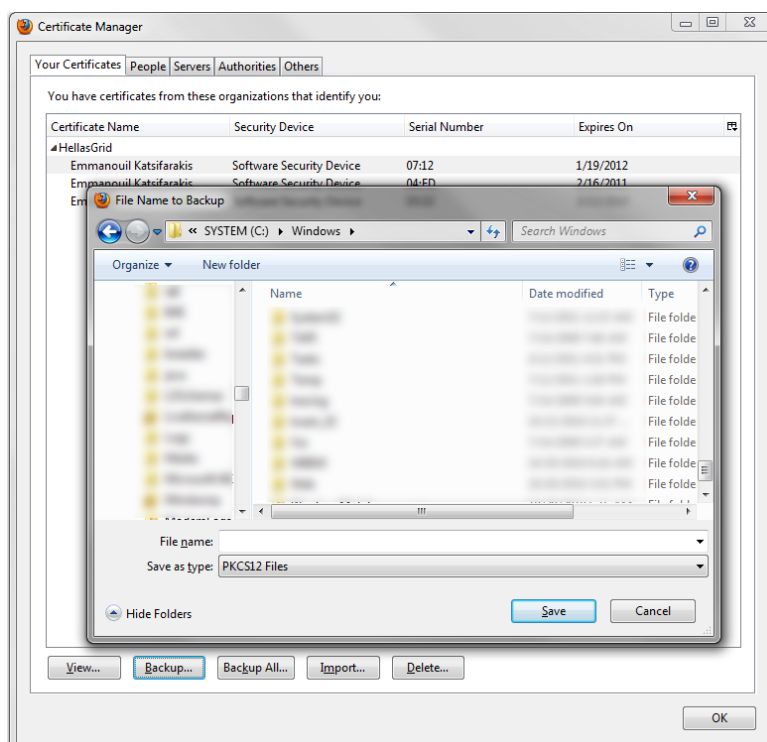
Εικόνα 5-3 - Mozilla Firefox - Μενού ψηφιακών πιστοποιητικών

Εδώ πατάμε το αριστερό κουμπί **View Certificates** για να δούμε τη λίστα εγκατεστημένων πιστοποιητικών στον browser και φροντίζουμε να είναι επιλεγμένο το tab **Your Certificates** (**Error! Reference source not found.**).



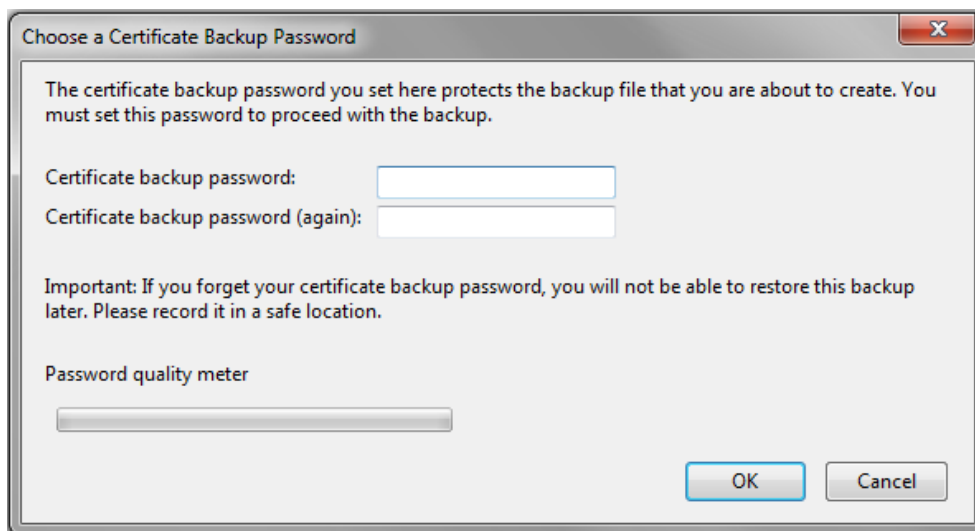
Εικόνα 5-4 - Mozilla Firefox - Λίστα ψηφιακών πιστοποιητικών

Από τη λίστα πιστοποιητικών επιλέγουμε το προσωπικό μας πιστοποιητικό για το HellasGrid. Σε περίπτωση που υπάρχουν πιστοποιητικά προηγούμενων ετών από το HellasGrid επιλέγουμε αυτό που λήγει αργότερα (στήλη *Expires On*). Και πατάμε το κουμπί **Backup**. Τέλος επιλέγουμε το σημείο στο δίσκο που θέλουμε να αποθηκευτεί το αρχείο του ψηφιακού πιστοποιητικού σε μορφή PKCS12.



**Εικόνα 5-5 - Mozilla Firefox - Εξαγωγή ψηφιακού πιστοποιητικού**

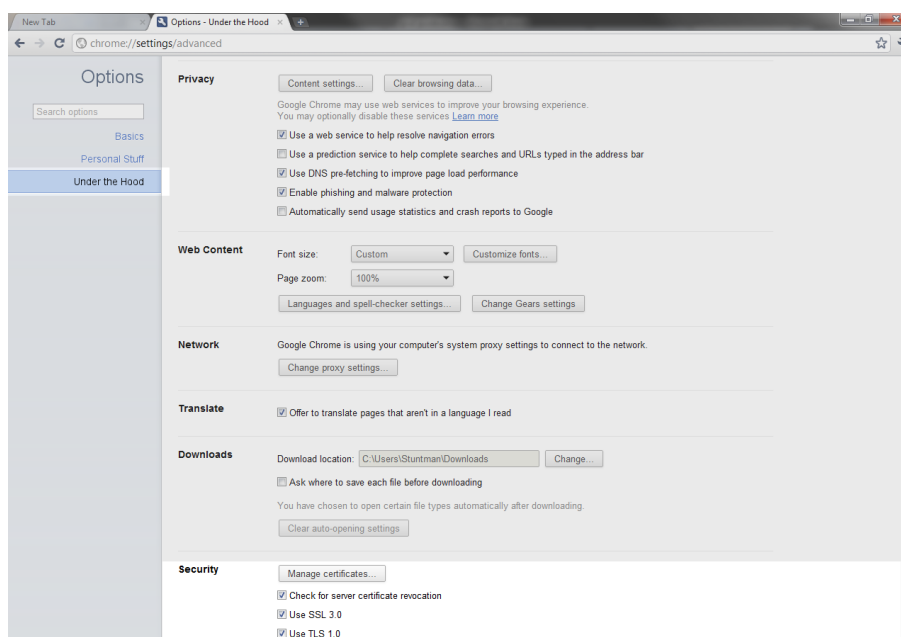
Πριν την αποθήκευση του πιστοποιητικού στο αρχείο που επιλέξαμε, θα ζητηθεί από την εφαρμογή ένας κωδικός ο οποίος θα χρησιμοποιηθεί για να προστατευθεί το ιδιωτικό κλειδί του πιστοποιητικού. Η εισαγωγή του κωδικού πραγματοποιείται δύο φορές για επιβεβαίωση.



Εικόνα 5-6 - Mozilla Firefox - Προστασία ιδιωτικού κλειδιού με κωδικό

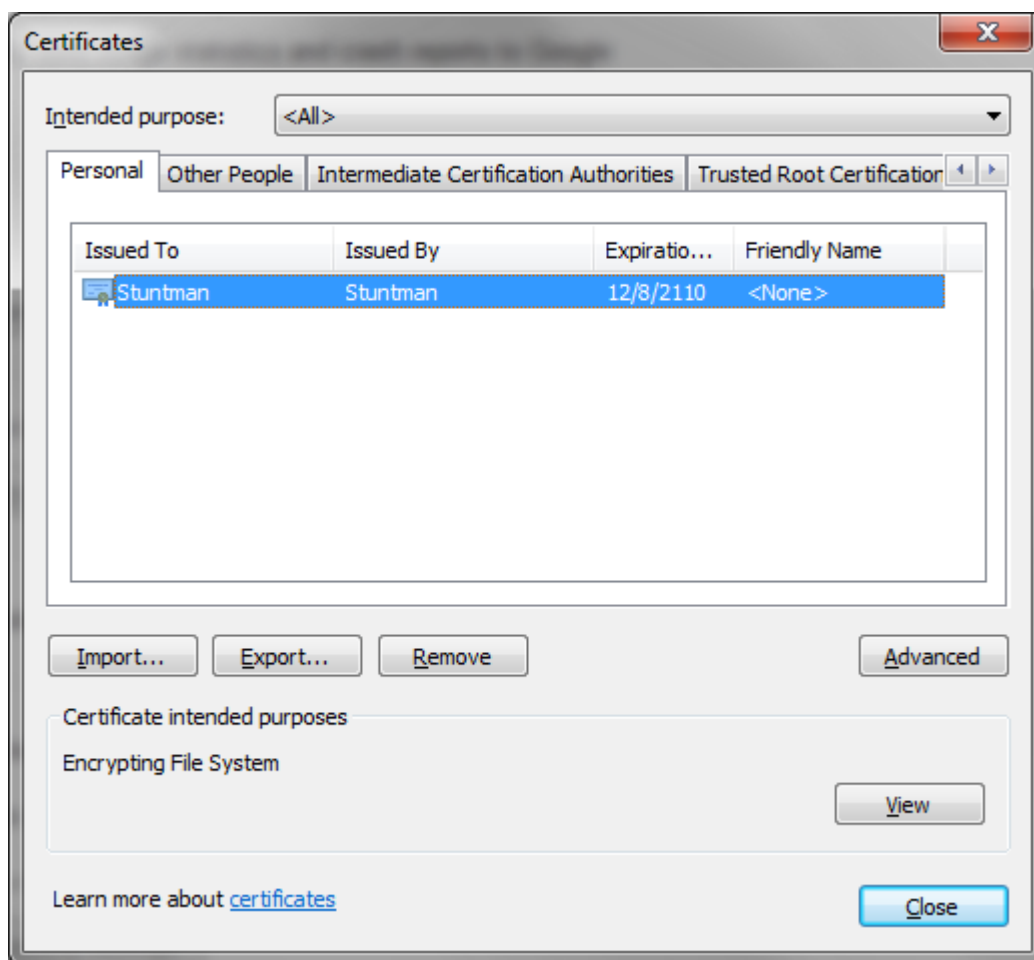
### 5.3.3. Εξαγωγή ψηφιακού πιστοποιητικού χρήστη από τον Google Chrome

Ανοίγουμε τις ρυθμίσεις του browser από το μενού **Options** και επιλέγουμε από το μενού στα αριστερά την επιλογή **Under the Hood**. Δεξιά εμφανίζονται κατηγορίες ρυθμίσεων όπως φαίνεται στην Εικόνα 5-7. Εδώ προχωράμε προς τα κάτω μέχρι να συναντήσουμε την κατηγορία **Security**. Εδώ πατάμε το κουμπί **Manage Certificates**.



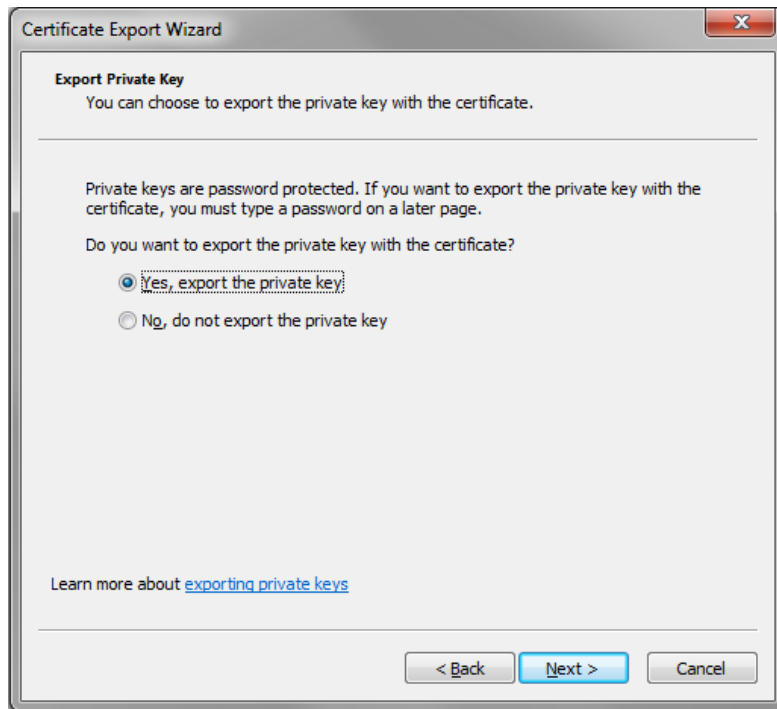
Εικόνα 5-7 - Google Chrome - Εξαγωγή ψηφιακού πιστοποιητικού

Στη λίστα πιστοποιητικών που εμφανίζεται, φροντίζουμε να είναι επιλεγμένο το tab **Personal** και επιλέγουμε το προσωπικό μας πιστοποιητικό για το HellasGrid (Εικόνα 5-8). Σε περίπτωση που υπάρχουν πιστοποιητικά προηγούμενων ετών από το HellasGrid επιλέγουμε αυτό που λήγει αργότερα (στήλη *Expiration Date*). Πατάμε το κουμπί **Export**.



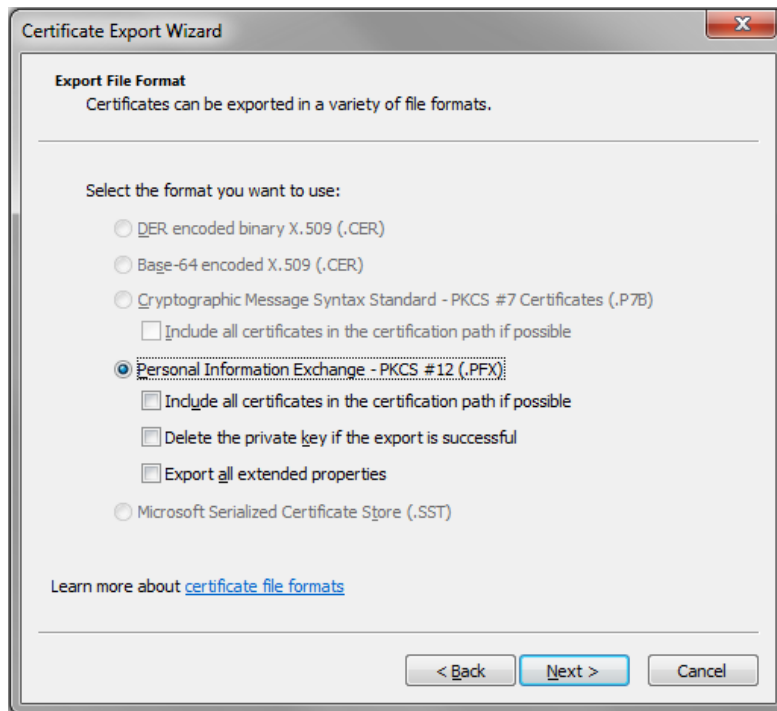
Εικόνα 5-8 - Google Chrome - Λίστα ψηφιακών πιστοποιητικών

Στη συνέχεια πατάμε **Next** μέχρι να φτάσουμε στην Εικόνα 5-9 όπου επιλέγουμε **Yes, export the private key** και πατάμε **Next**.



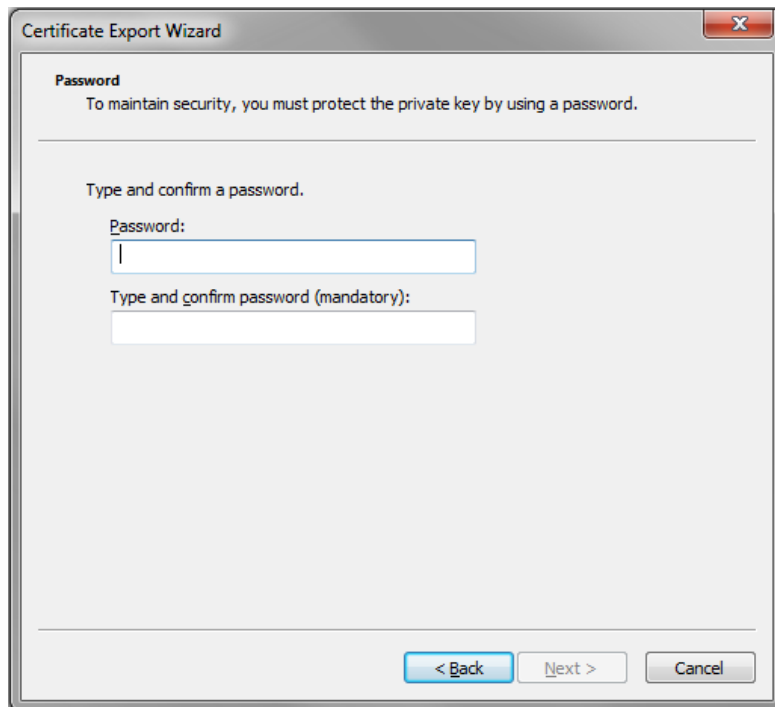
**Εικόνα 5-9 - Google Chrome - Εξαγωγή ψηφιακού πιστοποιητικού**

Στο επόμενο βήμα επιλέγουμε το είδος αρχείου στο οποίο θα γίνει η εξαγωγή του πιστοποιητικού. Εδώ επιλέγουμε Personal Information Exchange – PKCS #12 (.PFX) όπως φαίνεται στην Εικόνα 5-10 και πατάμε **Next**.



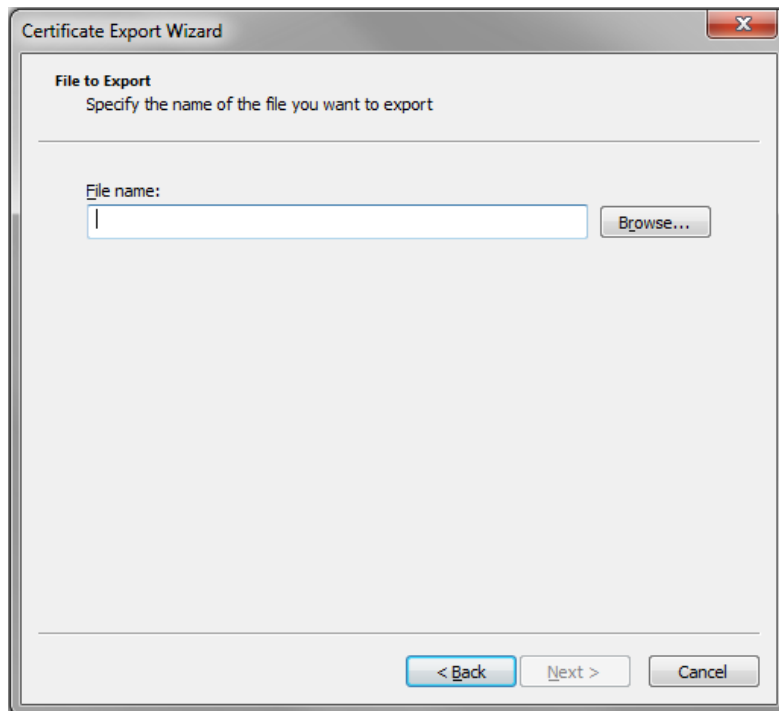
**Εικόνα 5-10 – Google Chrome – Εξαγωγή ψηφιακού πιστοποιητικού, επιλογή τύπου αρχείου**

Το επόμενο βήμα παρουσιάζεται στην Εικόνα 5-11 όπου το ιδιωτικό κλειδί του προσωπικού πιστοποιητικού πρέπει να προστατευθεί με ένα κωδικό. Ο κωδικός πρέπει να εισαχθεί δύο φορές για επιβεβαίωση και προτείνεται να περιέχει αρκετούς χαρακτήρες τόσο αλφαριθμητικούς όσο και συμβόλων για αυξημένη ασφάλεια. Αφού εισάγουμε το κωδικό, πατάμε ξανά **Next**.



**Εικόνα 5-11 - Google Chrome - Εξαγωγή ψηφιακού πιστοποιητικού, επιλογή κωδικού προστασίας ιδιωτικού κλειδιού**

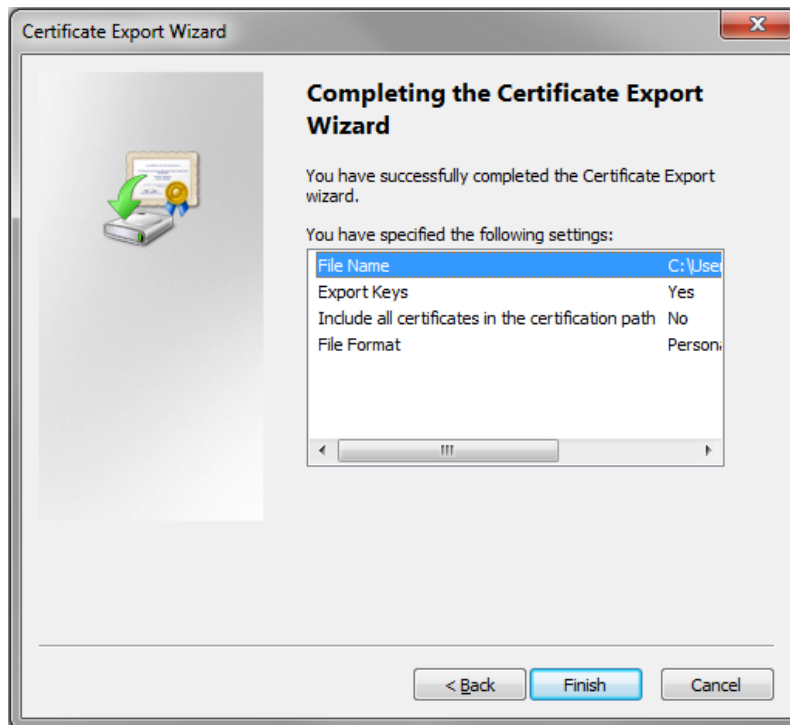
Στη συνέχεια επιλέγουμε το όνομα του αρχείου στο οποίο θα αποθηκευτεί το ψηφιακό πιστοποιητικό και πατάμε **Next** (Εικόνα 5-12).



**Εικόνα 5-12 - Google Chrome - Εξαγωγή ψηφιακού πιστοποιητικού, επιλογή ονόματος αρχείου**

Το τελευταίο βήμα της διαδικασίας παρουσιάζεται στην Εικόνα 5-13 όπου εμφανίζεται μια σύνοψη των επιλογών μας. Εάν οι επιλογές είναι έγκυρες πατάμε ***Finish*** για να ολοκληρωθεί η εξαγωγή του πιστοποιητικού.

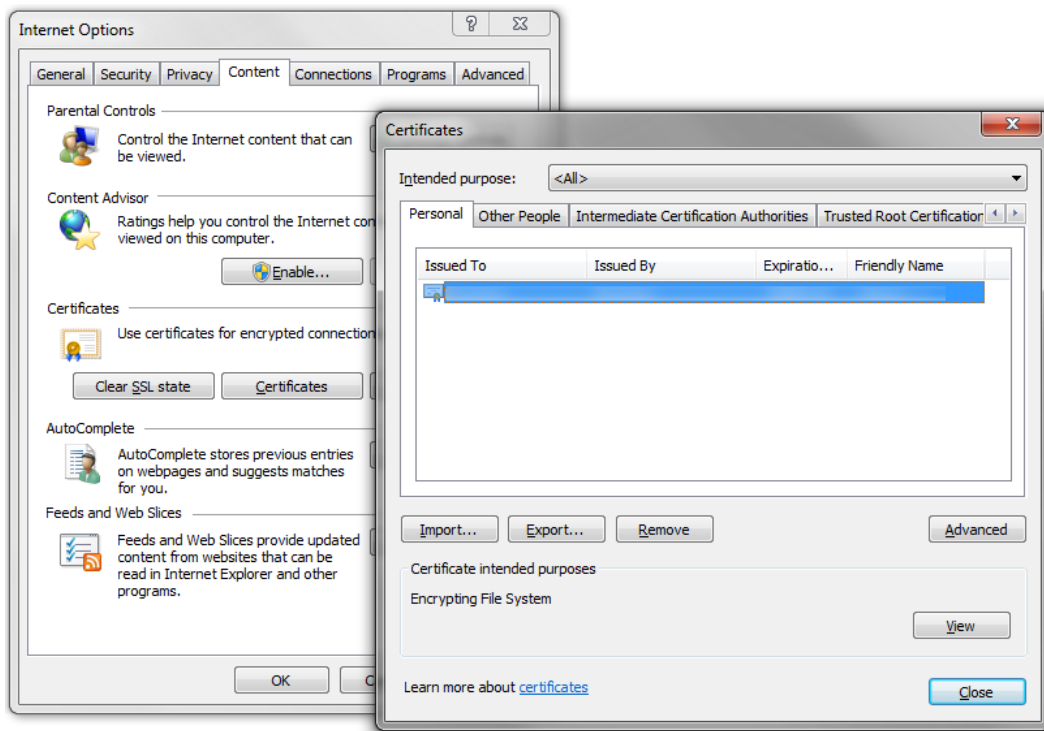




Εικόνα 5-13 – Google Chrome - Εξαγωγή ψηφιακού πιστοποιητικού, ολοκλήρωση διαδικασίας

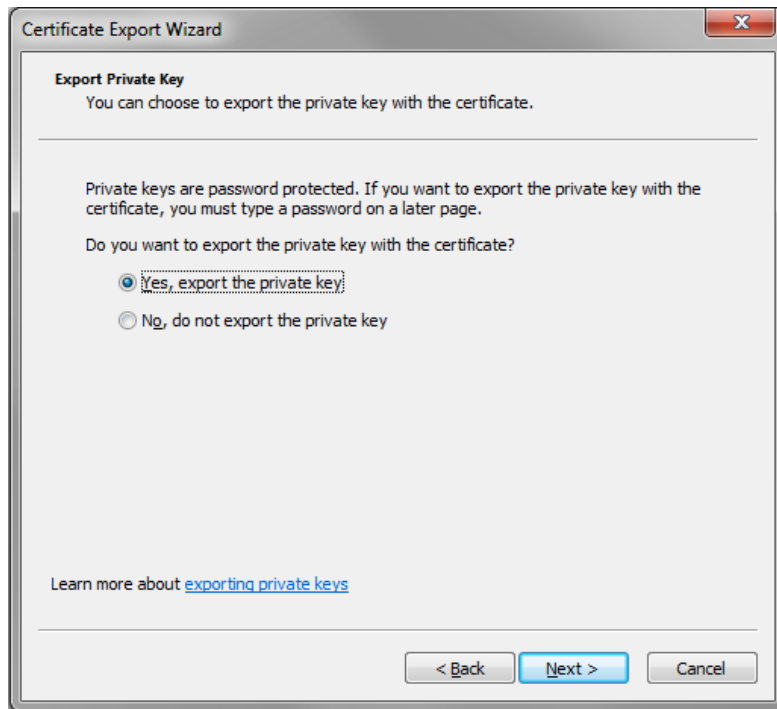
#### 5.3.3.1. Εξαγωγή ψηφιακού πιστοποιητικού χρήστη από τον Microsoft Internet Explorer

Ανοίγουμε το παράθυρο ρυθμίσεων από την επιλογή **Internet Options** στο menu **Tools** του Internet Explorer. Επιλέγουμε το tab **Content** και πατάμε το κουμπί **Certificates** για να δούμε τη λίστα εγκατεστημένων ψηφιακών πιστοποιητικών (Εικόνα 5-14). Προσέχουμε ώστε να είναι επιλεγμένο το tab **Personal** για να δούμε τα προσωπικά μας πιστοποιητικά.



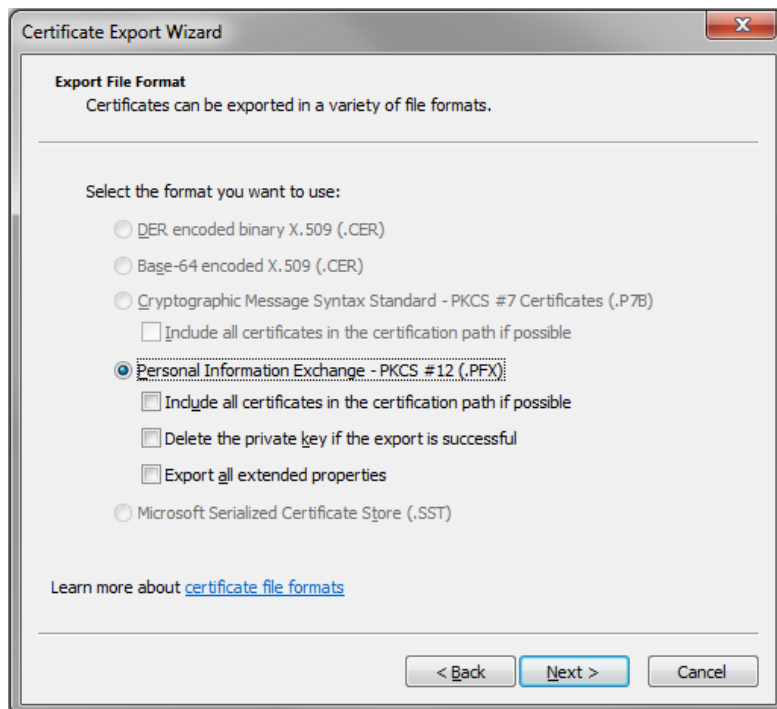
**Εικόνα 5-14 - Internet Explorer - Εξαγωγή ψηφιακού πιστοποιητικού**

Από τη λίστα εγκατεστημένων πιστοποιητικών επιλέγουμε το προσωπικό μας πιστοποιητικό για το HellasGrid. Σε περίπτωση που υπάρχουν πιστοποιητικά προηγούμενων ετών από το HellasGrid επιλέγουμε αυτό που λήγει αργότερα (στήλη *Expiration Date*). Και πατάμε το κουμπί **Export**. Στη συνέχεια πατάμε **Next** μέχρι να φτάσουμε στην Εικόνα 5-15 όπου επιλέγουμε **Yes, export the private key** και πατάμε **Next**.



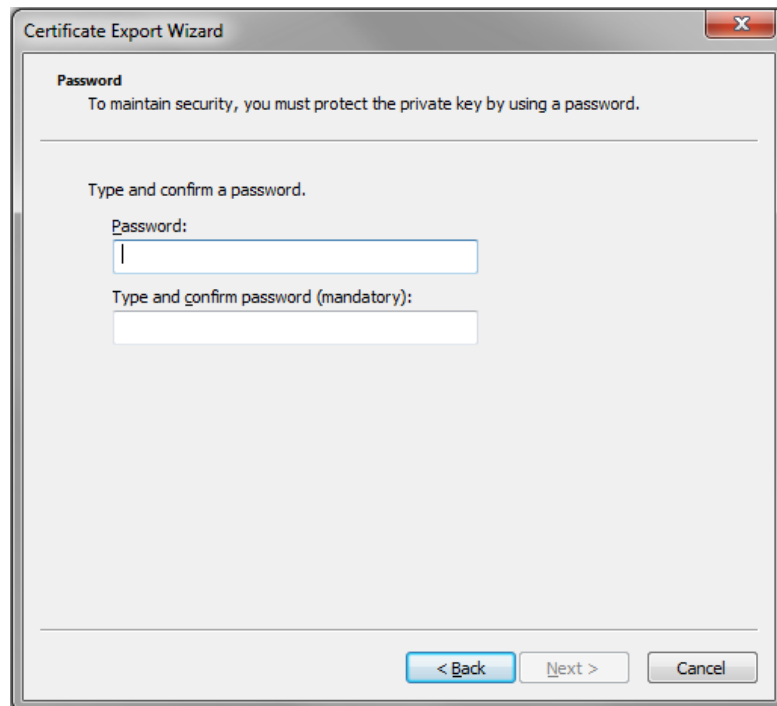
Εικόνα 5-15 - Internet Explorer - Εξαγωγή ψηφιακού πιστοποιητικού

Στο επόμενο βήμα επιλέγουμε το είδος αρχείου στο οποίο θα γίνει η εξαγωγή του πιστοποιητικού. Εδώ επιλέγουμε Personal Information Exchange – PKCS #12 (.PFX) όπως φαίνεται στην Εικόνα 5-16 και πατάμε **Next**.



Εικόνα 5-16 - Internet Explorer - Εξαγωγή ψηφιακού πιστοποιητικού, επιλογή τύπου αρχείου

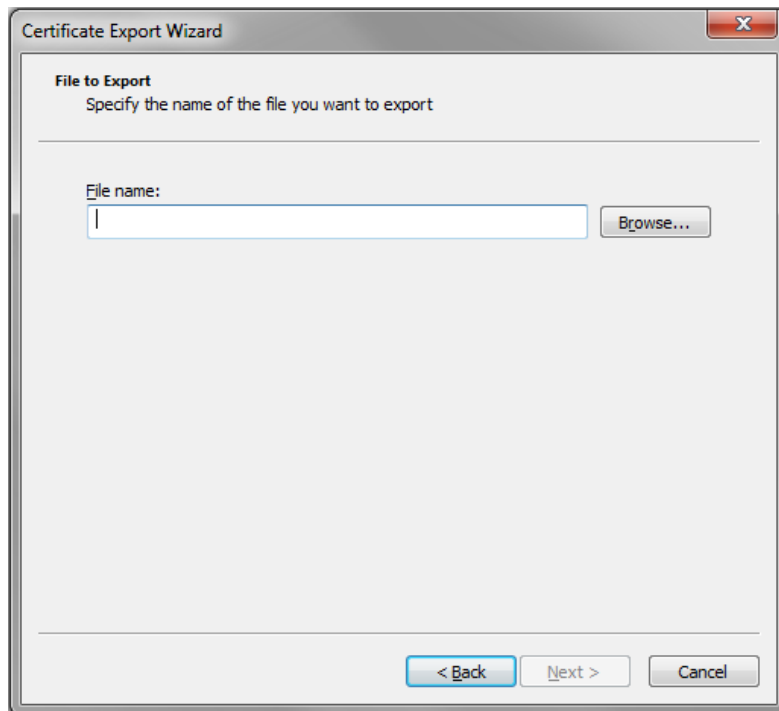
Το επόμενο βήμα παρουσιάζεται στην Εικόνα 5-17 όπου το ιδιωτικό κλειδί του προσωπικού πιστοποιητικού πρέπει να προστατευθεί με ένα κωδικό. Ο κωδικός πρέπει να εισαχθεί δύο φορές για επιβεβαίωση και προτείνεται να περιέχει αρκετούς χαρακτήρες τόσο αλφαριθμητικούς όσο και συμβόλων για αυξημένη ασφάλεια. Αφού εισάγουμε τον κωδικό πατάμε ξανά **Next**.



The image shows a Windows dialog box titled "Certificate Export Wizard". The main heading is "Password". Below it, the text reads: "To maintain security, you must protect the private key by using a password." There is a horizontal line separating this text from the input area. Below the line, it says "Type and confirm a password." There are two input fields: the first is labeled "Password:" and the second is labeled "Type and confirm password (mandatory):". At the bottom of the dialog, there are three buttons: "< Back", "Next >", and "Cancel".

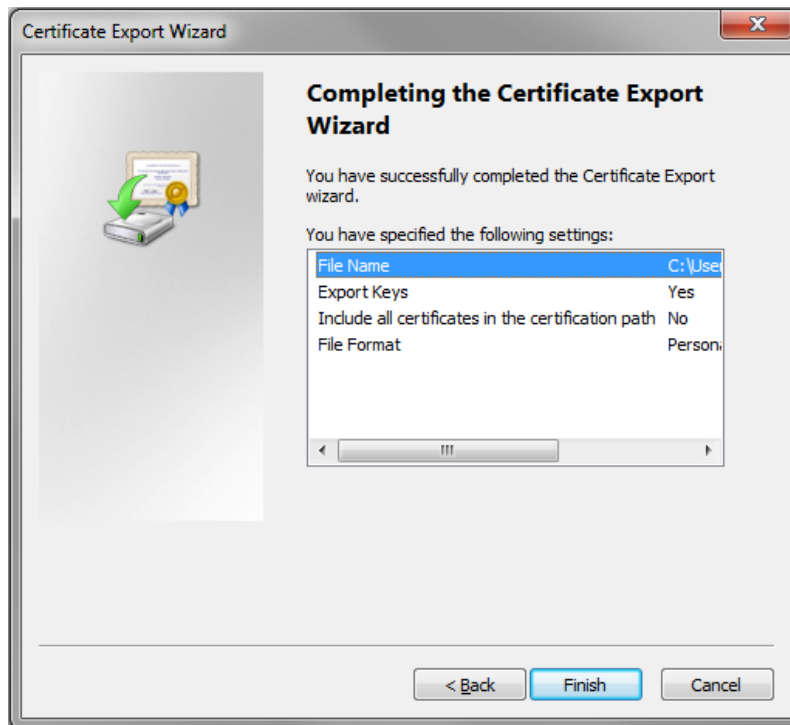
**Εικόνα 5-17 - Internet Explorer - Εξαγωγή ψηφιακού πιστοποιητικού, επιλογή κωδικού προστασίας ιδιωτικού κλειδιού**

Στη συνέχεια επιλέγουμε το όνομα του αρχείου στο οποίο θα αποθηκευτεί το ψηφιακό πιστοποιητικό και πατάμε **Next** (Εικόνα 5-18).



**Εικόνα 5-18 - Internet Explorer - Εξαγωγή ψηφιακού πιστοποιητικού, επιλογή ονόματος αρχείου**

Το τελευταίο βήμα της διαδικασίας παρουσιάζεται στην Εικόνα 5-19 όπου εμφανίζεται μια σύνοψη των επιλογών μας. Εάν οι επιλογές είναι έγκυρες πατάμε ***Finish*** για να ολοκληρωθεί η εξαγωγή του πιστοποιητικού.



Εικόνα 5-19 - Internet Explorer - Εξαγωγή ψηφιακού πιστοποιητικού, ολοκλήρωση διαδικασίας

#### 5.3.4. Μεταφορά αρχείου ψηφιακού πιστοποιητικού χρήστη στο UI

Αφού ολοκληρωθεί η εξαγωγή του πιστοποιητικού από τον Browser, είναι απαραίτητη η μεταφορά του στο φάκελο του χρήστη στο User Interface. Η μεταφορά αυτή γίνεται μέσω SCP με την εφαρμογή PSCP (κεφάλαιο 5.2.1). Συγκεκριμένα εάν υποθέσουμε ότι η εξαγωγή του πιστοποιητικού χρήστη έχει γίνει στο αρχείο **c:\cert.p12**, ο χρήστης έχει λογαριασμό στο UI ui01.ariagni.hellasgrid.gr και το όνομα του λογαριασμού χρήστη είναι **mkats**, η σύνταξη της εντολής **pscp** είναι η εξής:

```
pscp c:\cert.p12 mkats@ui01.ariagni.hellasgrid.gr:/home/mkats
```

#### 5.3.5. Μετατροπή ψηφιακού πιστοποιητικού σε μορφή .pem

Παρόλο που τα ψηφιακά πιστοποιητικά εξάγονται από τους Web Browsers σε μορφή PKCS #12, το UI αναγνωρίζει μόνο ψηφιακά πιστοποιητικά .pem. Για το λόγο αυτό η μεταφορά του πιστοποιητικού ενός χρήστη στο UI δεν αρκεί. Είναι απαραίτητη και η μετατροπή του σε μορφή .pem.

Η διαδικασία αυτή μπορεί να ολοκληρωθεί στο περιβάλλον γραμμής εντολών του UI μέσω της υποδομής OpenSSL. Έστω ότι το ψηφιακό πιστοποιητικό βρίσκεται στο φάκελο του χρήστη mkats με όνομα αρχείου cert.p12 και ο χρήστης mkats είναι συνδεδεμένος μέσω SSH στη γραμμή εντολών του UI. Οι εντολές που θα χρησιμοποιηθούν για τη μετατροπή του πιστοποιητικού σε μορφή .pem και την αντιγραφή του μέσα στον υποκατάλογο **.globus/** όπου πρέπει να βρίσκεται, είναι οι εξής:

```
openssl pkcs12 -nocerts -in cert.p12 -out /home/mkats/.globus/userkey.pem
```

Για δημιουργία του ιδιωτικού κλειδιού του χρήστη σε μορφή .pem και:

```
openssl pkcs12 -clcerts -nokeys -in cert.p12 -out /home/mkats/.globus/usercert.pem
```

Για δημιουργία του πιστοποιητικού χρήστη σε μορφή .pem.

Κατά τη μετατροπή τόσο του ιδιωτικού κλειδιού όσο και του πιστοποιητικού θα ζητηθεί από το χρήστη να εισάγει τον κωδικό που χρησιμοποίησε για να εξάγει το πιστοποιητικό του από τον Web Browser. Το μήνυμα που θα εμφανιστεί θα είναι το εξής:

```
Enter import password:
```

Ο χρήστης θα πρέπει να εισάγει τον κωδικό του προσέχοντας να είναι ακριβώς ίδιος με αυτόν που χρησιμοποίησε κατά την εξαγωγή του πιστοποιητικού. Σημειώστε ότι οι χαρακτήρες του κωδικού δεν θα εμφανίζονται στην οθόνη κατά την εισαγωγή τους για λόγους ασφάλειας.

Στην περίπτωση του ιδιωτικού κλειδιού, εάν ο χρήστης εισάγει σωστά τον κωδικό, θα εμφανιστεί το μήνυμα:

```
Enter PEM pass phrase:
```

Εδώ ζητείται από τον χρήστη να εισάγει ένα νέο κωδικό ως τελευταίο μέτρο ασφάλειας για την προστασία του ιδιωτικού του κλειδιού. Προτείνεται να χρησιμοποιηθεί ένας δύσκολος κωδικός με αρκετούς χαρακτήρες τόσο αλφαριθμητικούς όσο και συμβόλων. Ο κωδικός θα ζητηθεί μια δεύτερη επιπλέον φορά για επιβεβαίωση με το παρακάτω μήνυμα:

```
Verifying - Enter PEM pass phrase:
```

Εάν η διαδικασία ολοκληρωθεί επιτυχώς θα εμφανιστεί το μήνυμα:

```
MAC verified OK
```

Εάν η επιβεβαίωση δεν ταιριάζει, τότε η διαδικασία δεν θα ολοκληρωθεί και θα εμφανιστεί το μήνυμα:

```
Verify failure
```

Στην περίπτωση αυτή η διαδικασία θα πρέπει να επαναληφθεί από την αρχή.

### 5.3.6. Ολοκλήρωση εγκατάστασης πιστοποιητικού χρήστη στο UI

Αφού το πιστοποιητικό και το κλειδί χρήστη σε μορφή `.pem`, μεταφερθούν στον κατάλογο `.globus/` στο φάκελο του χρήστη του UI, είναι απαραίτητο να αποκτήσουν τα απαραίτητα προνόμια ανάγνωσης του λειτουργικού συστήματος. Αυτό γίνεται με τις παρακάτω εντολές στη γραμμή εντολών του UI:

```
chmod 444 ~/.globus/usercert.pem
```

για το πιστοποιητικό του χρήστη και:

```
chmod 400 ~/.globus/userkey.pem
```

για το ιδιωτικό κλειδί του χρήστη.

### 5.3.7. Πληροφορίες εγκατεστημένου πιστοποιητικού χρήστη

Εάν η διαδικασία εισαγωγής του πιστοποιητικού χρήστη έχει ολοκληρωθεί επιτυχώς, ο χρήστης μπορεί να δει πληροφορίες σχετικά με το ψηφιακό πιστοποιητικό του με την εντολή ***grid-cert-info***.



## 6. ΧΡΗΣΗ ΥΠΗΡΕΣΙΩΝ ΠΙΣΤΟΠΟΙΗΣΗΣ ΤΑΥΤΟΤΗΤΑΣ ΚΑΙ ΕΞΟΥΣΙΟΔΟΤΗΣΗΣ ΠΡΟΣΒΑΣΗΣ ΣΤΟ HELLASGRID

### 6.1. Πληρεξούσια πιστοποιητικά Grid (Grid Proxy Certificates)

---

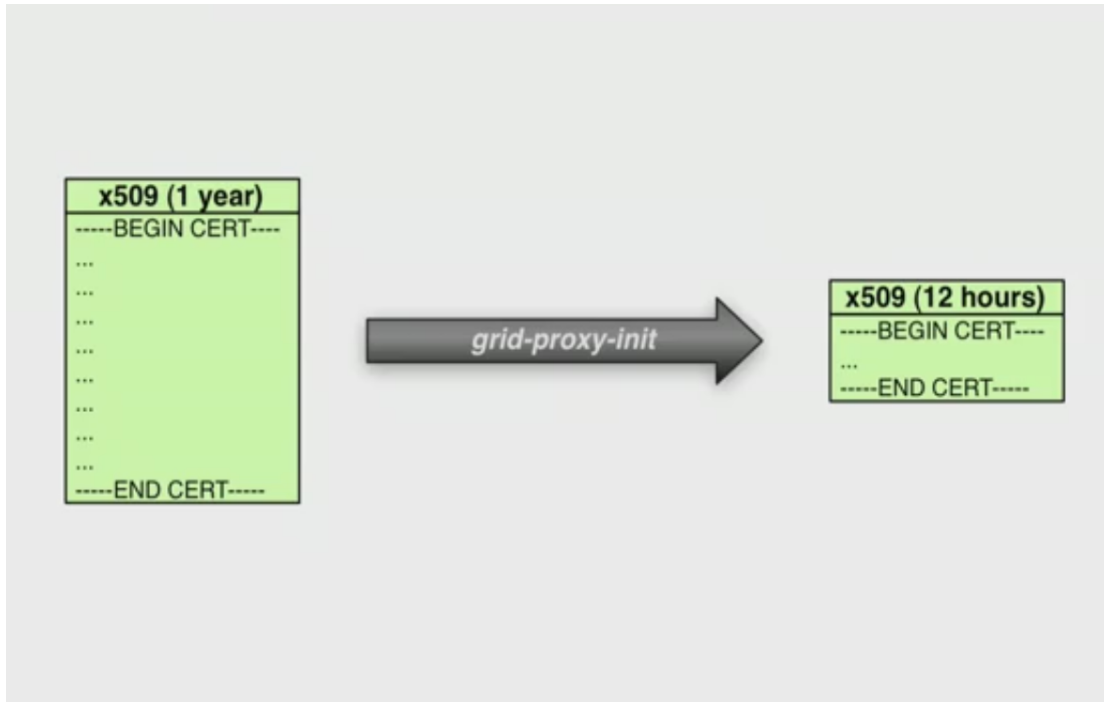
Όπως αναφέρθηκε προηγουμένως στο **κεφάλαιο 4.2** οι χρήστες του Grid χρησιμοποιούν πιστοποιητικά x509 για να πιστοποιήσουν την ταυτότητά τους και να επικοινωνήσουν ασφαλώς με τις υπηρεσίες της υποδομής. Τα πιστοποιητικά που χρησιμοποιούνται σε αυτή την περίπτωση δεν πρέπει να συγχέονται με τα πιστοποιητικά χρήστη που εκδίδονται από την Αρχή Πιστοποίησης και έχουν διάρκεια ισχύος ενός έτους. Πρόκειται για πιστοποιητικά που ονομάζονται **πληρεξούσια** (proxy certificates) τα οποία δημιουργούνται χρησιμοποιώντας τα στοιχεία των πιστοποιητικών χρηστών, έχουν όμως πολύ μικρή διάρκεια ισχύος (12 ώρες).

#### 6.1.1. Δημιουργία Proxy Certificate

Τα proxy certificates μπορούν να δημιουργηθούν μέσω της γραμμής εντολών του UI χρησιμοποιώντας την εντολή **grid-proxy-init**. Κατά την εκτέλεση της εντολής θα ζητηθεί ο κωδικός προστασίας του ιδιωτικού κλειδιού του χρήστη (ορίστηκε από τον χρήστη κατά τη μετατροπή του πιστοποιητικού του στο κεφάλαιο 5.3.5).

```
[root@localhost ~]# grid-proxy-init
Your identity: /C=GR/O=HellasGrid/OU=teiher.gr/CN=Emmanouil Katsifarakis
Enter GRID pass phrase for this identity: _
```

Εικόνα 6-1- Δημιουργία proxy certificate από τη γραμμή εντολών



Εικόνα 6-2 – Σχηματική αναπαράσταση δημιουργίας Grid Proxy Certificate μέσω της εντολής `grid-proxy-init`

### 6.1.2. Πληροφορίες Proxy Certificate

Οι χρήστες μπορούν να δουν πληροφορίες για το τρέχον proxy certificate τους μέσω της γραμμής εντολών του UI με την εντολή **`grid-proxy-info`**. Η εντολή αυτή παρέχει στο χρήστη γενικές πληροφορίες σχετικά με την έκδοση και τον τύπο του πιστοποιητικού, καθώς και τη διαδρομή στο δίσκο (path) όπου βρίσκεται το προσωρινό αρχείο του πιστοποιητικού. Η τελευταία γραμμή (timeleft) ενημερώνει τον χρήστη τη χρονική διάρκεια για την οποία το πιστοποιητικό θα παραμείνει έγκυρο. Μετά το πέρας της διάρκειας αυτής, το πιστοποιητικό δεν είναι πλέον έγκυρο και ο χρήστης θα πρέπει να εκδώσει νέο με την εντολή **`grid-proxy-init`** (κεφάλαιο 6.1.1).

```
[root@localhost ~]# grid-proxy-info
subject : /C=GR/O=HellasGrid/OU=teiher.gr/CN=Emmanouil Katsifarakis/CN=441074165
issuer  : /C=GR/O=HellasGrid/OU=teiher.gr/CN=Emmanouil Katsifarakis
identity : /C=GR/O=HellasGrid/OU=teiher.gr/CN=Emmanouil Katsifarakis
type    : Proxy draft (pre-RFC) compliant impersonation proxy
strength : 512 bits
path    : /tmp/x509up_u0
timeleft : 11:37:58
```

Εικόνα 6-3 - Πληροφορίες τρέχοντος proxy certificate

### 6.1.3. Ακύρωση Proxy Certificate

Μετά τη δημιουργία του, ένα proxy certificate μπορεί να ακυρωθεί από τον χρήστη που το δημιούργησε μέσω της εντολής **grid-proxy-destroy**.

## 6.2. Πληρεξούσια πιστοποιητικά VOMS (VOMS Proxy Certificates)

---

Παρόλο που η δημιουργία ενός proxy certificate πιστοποιεί την ταυτότητα του χρήστη στην υποδομή, δεν είναι αρκετή για την υποβολή εργασιών (job submission) στο Grid. Για να μπορέσει ο χρήστης να υποβάλλει jobs, πρέπει να αποκτήσει διαπιστευτήρια πρόσβασης μέσω της υπηρεσίας VOMS τα οποία είναι ένα proxy certificate VOMS με τη μορφή επεκτάσεων VOMS (VOMS extensions) που προστίθενται στο τέλος του grid proxy certificate και έχουν την ίδια χρονική διάρκεια εγκυρότητας (12 ώρες). Με αυτό το «πακέτο» των δυο πληρεξούσιων πιστοποιητικών, ο χρήστης έχει πλέον πιστοποιήσει τη ταυτότητά του στην υποδομή και είναι εξουσιοδοτημένος να έχει πρόσβαση στις υπηρεσίες που παρέχονται από το VO του οποίου είναι μέλος.

### 6.2.1. Δημιουργία VOMS Proxy Certificate

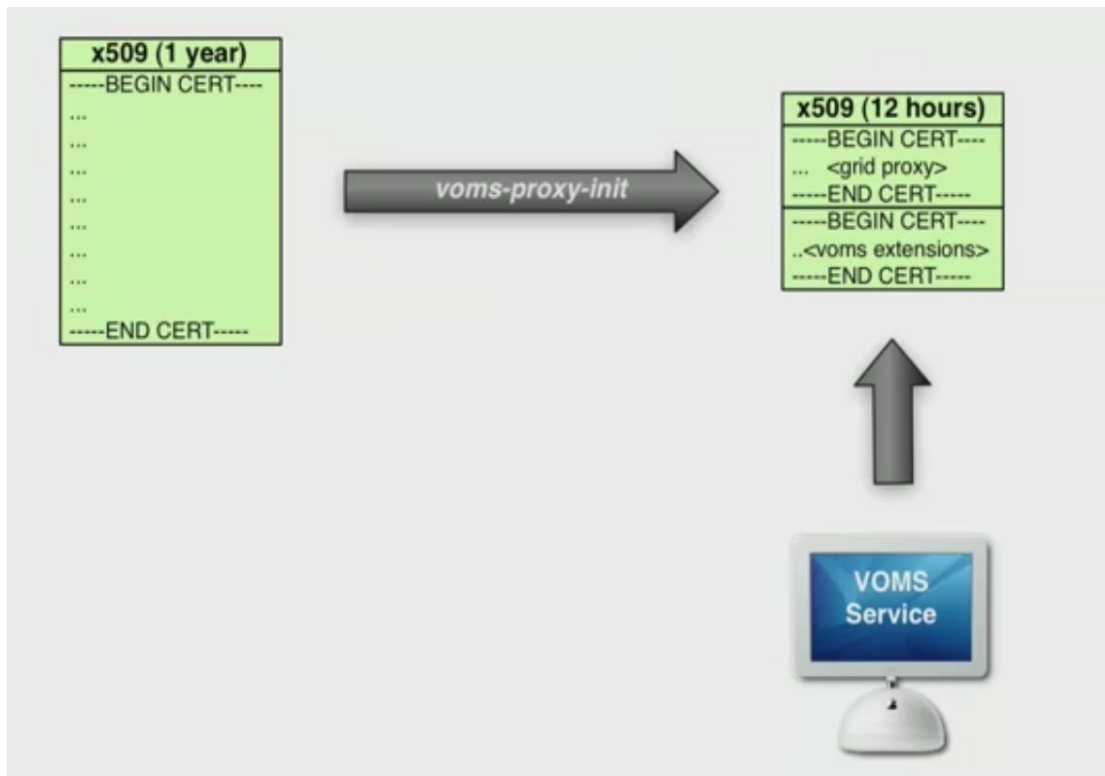
Τα voms proxy certificates μπορούν να δημιουργηθούν μέσω της γραμμής εντολών του UI χρησιμοποιώντας την εντολή **voms-proxy-init**. Η σύνταξη της εντολής έχει ως εξής:

```
voms-proxy-init -vo <όνομα VO>
```

Για παράδειγμα, ένας χρήστης που ανήκει στο VO Νοτιοανατολικής Ευρώπης (SEE VO) θα χρησιμοποιούσε την εντολή με την παρακάτω σύνταξη:

```
voms-proxy-init -vo see
```

Κατά την εκτέλεση της εντολής θα ζητηθεί ο κωδικός προστασίας του ιδιωτικού κλειδιού του χρήστη (ορίστηκε από τον χρήστη κατά τη μετατροπή του πιστοποιητικού του στο κεφάλαιο 5.3.5).



Εικόνα 6-4 - Δημιουργία «πακέτου» πιστοποιητικών από την υπηρεσία VOMS με την εντολή *voms-proxy-init*.

### 6.2.2. Πληροφορίες VOMS Proxy Certificate

Οι χρήστες μπορούν να δουν πληροφορίες για το τρέχον *voms proxy certificate* τους μέσω της γραμμής εντολών του UI με την εντολή ***voms-proxy-info -all***. Το πρώτο μέρος των αποτελεσμάτων της εντολής ενημερώνουν το χρήστη για το τρέχον Grid Proxy Certificate του, ενώ το δεύτερο μέρος αναφέρεται στα VOMS Extensions του Certificate.

### 6.2.3. Ακύρωση VOMS Proxy Certificate

Μετά τη δημιουργία του, ένα *voms proxy certificate* μπορεί να ακυρωθεί από τον χρήστη που το δημιούργησε μέσω της εντολής ***voms-proxy-destroy***.

## 6.3. Υπηρεσία MyProxy

---

Όπως αναφέρεται παραπάνω, τα proxy certificates έχουν διάρκεια 12 ωρών. Αυτός το χρονικό διάστημα φαίνεται αρκετό για μικρά jobs που ολοκληρώνονται μέσα σε μερικές ώρες, αλλά μπορεί να αποδειχθεί υπερβολικά σύντομος για πιο απαιτητικές εφαρμογές που πολλές φορές χρειάζονται μέρες για να ολοκληρώσουν τις εργασίες τους.

Η υπηρεσία MyProxy αναλαμβάνει να εξυπηρετήσει τέτοιου είδους χρονοβόρες εργασίες παρέχοντας έναν αυτόματο μηχανισμό «ανανέωσης» των proxy certificates παρατείνοντας έτσι το χρόνο που μπορεί να εκτελείται ένα job στις 7 ημέρες.

### 6.3.1. Χρήση υπηρεσίας MyProxy

Οι χρήστες μπορούν να χρησιμοποιήσουν την υπηρεσία MyProxy εκτελώντας την εντολή **myproxy-init** στη γραμμή εντολών του UI η οποία αναλαμβάνει να μεταφέρει τα πιστοποιητικά ταυτοποίησης του χρήστη στο server της υπηρεσίας και να ξεκινήσει τη διαδικασία ανανέωσης των proxy πιστοποιητικών. Η προτεινόμενη απλοποιημένη σύνταξη που είναι επαρκής για τις περισσότερες περιπτώσεις είναι η εξής:

```
myproxy-init -d -n
```

### 6.3.2. Πληροφορίες χρήσης υπηρεσίας MyProxy

Οι χρήστες μπορούν να δουν πληροφορίες σχετικά με την κατάσταση της υπηρεσίας MyProxy μέσω της εντολής **myproxy-info -d**

### 6.3.3. Διακοπή υπηρεσίας MyProxy

Οι χρήστες μπορούν να σταματήσουν να χρησιμοποιούν την υπηρεσία MyProxy με την εντολή **myproxy-destroy -d**.

## 7. ΥΠΟΒΟΛΗ ΕΡΓΑΣΙΩΝ ΣΤΟ HELLASGRID

### 7.1. Η υποβολή εργασιών σε βήματα

---

Η υποβολή εργασιών στο Grid πραγματοποιείται μέσω της γραμμής εντολών του UI, χρησιμοποιώντας το σετ εντολών του WMS.

Παρακάτω αναλύεται η διαδικασία υποβολής ενός job στα συνηθέστερα βήματα:

1. Απόκτηση προσωρινού proxy certificate (κατά προτίμηση με VOMS extensions).
2. Δημιουργία υποκαταλόγου με τα αρχεία του job.
3. Δημιουργία αρχείου JDL περιγραφής των χαρακτηριστικών του job (δείτε κεφάλαιο 7.2).
4. Υποβολή του job στο WMS.
5. Έλεγχος κατάστασης job.
6. Λήψη των αποτελεσμάτων του Job όταν ολοκληρωθεί (φαίνεται στον έλεγχο κατάστασης).

Στα επόμενα κεφάλαια τα παραπάνω βήματα περιγράφονται αναλυτικότερα. Το κεφάλαιο 7.3.6 περιέχει ένα ολοκληρωμένο παράδειγμα υποβολής εργασίας στο HellasGrid.

### 7.2. Αρχεία Περιγραφής Εργασιών JDL

---

Τα jobs που υποβάλλονται στο Grid συνοδεύονται πάντα από μια περιγραφή που προσδιορίζει τα χαρακτηριστικά τους καθώς και τις απαιτήσεις που έχουν σε πόρους της υποδομής. Η περιγραφή αυτή δημιουργείται στη Γλώσσα Περιγραφής Εργασιών (Job Description Language – JDL) και αποθηκεύεται σε ένα αρχείο .jdl που συνοδεύει κάθε job. Για να γίνει υποβολή ενός job στο grid, πρέπει να υποβληθεί το .jdl αρχείο του job στο WMS.

Τα αρχεία .jdl είναι στην πραγματικότητα αρχεία κειμένου που μπορούν να ανοιχτούν σε οποιαδήποτε εφαρμογή επεξεργασίας κειμένου. Η γλώσσα JDL είναι μια γλώσσα υψηλού επιπέδου (high-level) προσανατολισμένη προς το χρήστη (user-oriented) για την περιγραφή jobs. Αν και οι προδιαγραφές της JDL περιέχουν πλήθος χαρακτηριστικών το WMS λαμβάνει υπόψη του μόνο ένα συγκεκριμένο υποσύνολο αυτών.

Παρακάτω γίνεται μια σύντομη αναφορά στα πιο συχνά χρησιμοποιούμενα κομμάτια της JDL για υποβολή εργασιών στο HellasGrid. Μια λεπτομερής ανάλυση της JDL μπορεί να βρεθεί στη διεύθυνση:

<https://edms.cern.ch/file/555796/1/EGEE-JRA1-TEC-555796-JDL-Attributes-v0-8.pdf>

**ΣΗΜΕΙΩΣΗ:**

Οι αλφαριθμητικές τιμές (strings) που δέχονται τα χαρακτηριστικά της JDL πρέπει να περικλείονται μέσα σε `"`. Για παράδειγμα: `Executable="grep"`.

**ΣΗΜΕΙΩΣΗ:**

Κάθε χαρακτηριστικό της JDL θα πρέπει στο τέλος του να έχει το χαρακτήρα του ελληνικού ερωτηματικού και μετά από αυτόν δεν θα πρέπει να ακολουθούν κενά ή tabs.

### 7.2.1. Executable

Το χαρακτηριστικό Executable είναι **ΑΠΑΡΑΙΤΗΤΟ** στο JDL αρχείο ενός job και αντιπροσωπεύει το όνομα του εκτελέσιμου αρχείου του job. Το εκτελέσιμο αρχείο μπορεί να περιλαμβάνεται στα αρχεία του job ή να είναι ακόμα και μια εντολή του συστήματος.

Εάν για παράδειγμα θέλαμε να εκτελέσουμε την εντολή `hostname` στο CE που επιστρέφει το όνομα δικτύου ενός υπολογιστή, θα χρησιμοποιούσαμε την παρακάτω σύνταξη:

```
Executable="hostname";
```

**ΣΗΜΕΙΩΣΗ:**

Σε περίπτωση που το εκτελέσιμο αρχείο αποτελεί εντολή του συστήματος ή δεν συμπεριλαμβάνεται στα αρχεία του job, ο χρήστης πρέπει να είναι απόλυτα σίγουρος ότι το αρχείο υπάρχει στο CE όπου θα εκτελεστεί το job. Επιπλέον θα πρέπει να συμπεριλάβει την απόλυτη διαδρομή αρχείου προς το εκτελέσιμο, όπως είναι αυτή ορισμένη στο CE όπως φαίνεται και στο παρακάτω παράδειγμα (εάν η διαδρομή δεν συμπεριλαμβάνεται στο Path του Linux στο συγκεκριμένο CE):

```
Executable = "usr/local/java/j2sdk1.4.0_01/bin/java";
```

### 7.2.2. JobType

Το χαρακτηριστικό JobType προσδιορίζει τον τύπο του job που θα υποβληθεί στο Grid. Για τα απλά jobs που υποβάλλονται στην υποδομή και δεν απαιτούν παρέμβαση από τον χρήστη, λαμβάνει την τιμή "Normal". Για παράλληλα jobs που χρησιμοποιούν τη διεπαφή MPI (Message Passing Interface), λαμβάνει την τιμή "MPICH" (καθώς το Grid χρησιμοποιεί την MPICH υλοποίηση του MPI). Υπάρχουν επίσης οι λιγότερο χρησιμοποιούμενες επιλογές Partitionable, Checkpointable και Parametric,

### 7.2.3. Arguments

Το χαρακτηριστικό Arguments προσδιορίζει τα ορίσματα με τα οποία θα εκτελεστεί το εκτελέσιμο αρχείο που προσδιορίζεται στο χαρακτηριστικό Executable (κεφάλαιο 7.2.1). Αν για παράδειγμα θέλαμε το job μας να εκτελέσει την εντολή `ls -al` θα χρησιμοποιούσαμε την παρακάτω σύνταξη των ορισμάτων Executable και Arguments:

```
Executable="ls";  
Arguments="-al";
```

### 7.2.4. StdInput

Το χαρακτηριστικό StdInput δέχεται ως όρισμα ένα αρχείο που παρέχεται στο εκτελέσιμο του job ως η πρότυπη είσοδος του (standard input).

Παράδειγμα: StdInput="myJobInput";



**ΣΗΜΕΙΩΣΗ:**

Όπως και στο χαρακτηριστικό Executable, ο χρήστης πρέπει να είναι σίγουρος ότι το αρχείο περιέχεται στο CE και να φροντίσει να παρέχει την ακριβή διαδρομή του αρχείου εάν αυτή δεν συμπεριλαμβάνεται στο Path του Linux του CE.

### 7.2.5. StdOutput

Το χαρακτηριστικό StdOutput δέχεται ως όρισμα ένα αρχείο που παρέχεται στο εκτελέσιμο του job ως η πρότυπη έξοδος του (standard output).

Παράδειγμα: StdOutput="myJobOutput";

**ΣΗΜΕΙΩΣΗ:**

Όπως και στο χαρακτηριστικό Executable, ο χρήστης πρέπει να είναι σίγουρος ότι το αρχείο περιέχεται στο CE και να φροντίσει να παρέχει την ακριβή διαδρομή του αρχείου εάν αυτή δεν συμπεριλαμβάνεται στο Path του Linux του CE.

### 7.2.6. StdError

Το χαρακτηριστικό StdError δέχεται ως όρισμα ένα αρχείο που παρέχεται στο εκτελέσιμο του job ως η πρότυπη έξοδος σφαλμάτων του (standard error).

Παράδειγμα: StdError="myJobError";

**ΣΗΜΕΙΩΣΗ:**

Όπως και στο χαρακτηριστικό Executable, ο χρήστης πρέπει να είναι σίγουρος ότι το αρχείο περιέχεται στο CE και να φροντίσει να παρέχει την ακριβή διαδρομή του αρχείου εάν αυτή δεν συμπεριλαμβάνεται στο Path του Linux του CE.

### 7.2.7. InputSandbox

Το χαρακτηριστικό InputSandbox δέχεται ως όρισμα μια αλφαριθμητική τιμή (string) ή μια λίστα από αλφαριθμητικές τιμές που αντιπροσωπεύουν αρχεία τα οποία είναι απαραίτητα για την

εκτέλεση του job και θα πρέπει να μεταφερθούν στα Worker Nodes πριν την εκκίνηση της εκτέλεσης.

Τα αρχεία του InputSandbox θα πρέπει να βρίσκονται είτε στο UI από το οποίο γίνεται η υποβολή του job, είτε σε κάποιον προσβάσιμο gridFTP Server.

Εάν για παράδειγμα θέλαμε να μεταφέρουμε το εκτελέσιμο αρχείο ενός job που βρίσκεται στο UI, στον υποκατάλογο /jobs/ με όνομα myJob, καθώς και το αρχείο της πρότυπης εισόδου του, που βρίσκεται στο UI, στον υποκατάλογο /jobs/input/ με όνομα myJobStdInput, η σύνταξη του χαρακτηριστικού InputSandbox θα ήταν ως εξής:

```
InputSandbox = {"/jobs/myJob", "/jobs/input/myJobStdInput"};
```

Όπως φαίνεται στο παράδειγμα, πρέπει να προσδιορίσουμε τη διαδρομή των αρχείων στο δίσκο του UI ώστε να είναι δυνατή η μεταφορά τους στο WMS και από εκεί στα WNs.

#### **ΣΗΜΕΙΩΣΗ:**

Στο χαρακτηριστικό InputSandbox επιτρέπονται ειδικοί χαρακτήρες wildcards (π.χ. \* ? κλπ.) για επιλογή πολλαπλών αρχείων.

Εάν για παράδειγμα θέλαμε να μεταφέρουμε πολλαπλά αρχεία δεδομένων της εφαρμογής του job μας, τα οποία βρίσκονται μέσα στον υποκατάλογο /jobs/data/ η σύνταξη του χαρακτηριστικού InputSandbox του προηγούμενου παραδείγματος θα γινόταν ως εξής:

```
InputSandbox = {"/jobs/myJob", "/jobs/input/myJobStdInput", "/jobs/data/*"};
```

### **7.2.8. OutputSandbox**

Το χαρακτηριστικό OutputSandbox δέχεται ως όρισμα μια αλφαριθμητική τιμή (string) ή μια λίστα από αλφαριθμητικές τιμές που αντιπροσωπεύουν αρχεία τα οποία θα μεταφερθούν από τα WNs πίσω στο UI του χρήστη ή σε κάποιο gridFTP server όταν το job ολοκληρωθεί.

Εάν για παράδειγμα θέλουμε να μεταφέρουμε το αρχείο myJobStdOutput που περιέχει την πρότυπη έξοδο του εκτελέσιμου του job και το αρχείο myJobStdError που περιέχει την πρότυπε

έξοδο σφαλμάτων του εκτελέσιμου του job, η σύνταξη του χαρακτηριστικού OutputSandbox θα ήταν ως εξής:

```
OutputSandbox={"myJobStdOutput","myJobStdError"};
```

**ΣΗΜΕΙΩΣΗ:**

Στο χαρακτηριστικό OutputSandbox επιτρέπονται ειδικοί χαρακτήρες wildcards (π.χ. \* ? κλπ.) για επιλογή πολλαπλών αρχείων.

Εάν για παράδειγμα θέλαμε να μεταφέρουμε πολλαπλά αρχεία με αποτελέσματα της εφαρμογής του job μας, τα οποία βρίσκονται μέσα στον υποκατάλογο output/ που βρίσκεται στον ίδιο υποκατάλογο με το εκτελέσιμο της εφαρμογής στο WN, η σύνταξη του χαρακτηριστικού OutputSandbox του προηγούμενου παραδείγματος θα γινόταν ως εξής:

```
OutputSandbox={"myJobStdOutput","myJobStdError","output/*"};
```

### 7.2.9. CPUNumber

Το χαρακτηριστικό CPUNumber δέχεται ως όρισμα έναν ακέραιο αριθμό μεγαλύτερο του 1, που προσδιορίζει τον αριθμό των CPU που χρειάζεται το job για να εκτελεστεί. Αυτό το χαρακτηριστικό χρησιμοποιείται με jobs που χρησιμοποιούν MPI.

Παρόλο που το WMS θα κατανέμει τον προσδιορισμένο αριθμό των CPU στο συγκεκριμένο job, ο τρόπος που οι CPUs αυτές θα χρησιμοποιηθούν εξαρτάται αποκλειστικά και μόνο από την εφαρμογή που εκτελεί το job.

Παράδειγμα, για δέσμευση 5 CPU η σύνταξη του χαρακτηριστικού είναι ως εξής:

```
CPUNumber=5;
```

**ΠΡΟΣΟΧΗ:**

Το χαρακτηριστικό CPUNumber δέχεται ως όρισμα έναν ακέραιο αριθμό ο οποίος δεν περικλείεται σε "", σε αντίθεση με τα προηγούμενα χαρακτηριστικά που δέχονται ως ορίσματα αλφαριθμητικές τιμές.

### 7.2.10. Requirements

Το χαρακτηριστικό Requirements επιτρέπει στους χρήστες να θέσουν περιορισμούς στους υπολογιστικούς πόρους που θα εκτελέσουν το Job. Είναι δυνατός ο προσδιορισμός τόσο του υλικού όσο και του λογισμικού που είναι απαραίτητα για την εκτέλεση του Job μέσω ετικετών χαρακτηριστικών (Attribute Tags). Στο κεφάλαιο 9 υπάρχουν περισσότερες πληροφορίες για τα διαθέσιμα Attribute Tags.

## 7.3. ΔΙΑΧΕΙΡΙΣΗ ΕΡΓΑΣΙΩΝ ΣΤΟ GRID

---

Η διαχείριση εργασιών στο Grid πραγματοποιείται μέσω της γραμμής εντολών, με το σετ εντολών του WMS. Στα επόμενα κεφάλαια παρουσιάζονται με παραδείγματα οι σημαντικότερες εντολές διαχείρισης jobs στο HellasGrid. Επιπλέον στο κεφάλαιο 7.3.6 υπάρχει ένα ολοκληρωμένο παράδειγμα υποβολής και διαχείρισης ενός απλουστευμένου job.

### 7.3.1. Έλεγχος .jdl αρχείου και εμφάνιση διαθέσιμων CE

Καλό είναι πριν την υποβολή του job στην υποδομή, να εκτελείται η εντολή ***glite-wms-job-list-match*** που ελέγχει την εγκυρότητα του .JDL αρχείου και εμφανίζει μια λίστα με τα διαθέσιμα CE που ταιριάζουν με την περιγραφή που περιέχει.

Η σύνταξη της εντολής είναι η εξής:

```
glite-wms-job-list-match -a <αρχείο .jdl>
```

Αν για παράδειγμα το όνομα του αρχείου .jdl του job είναι myJob.jdl η σύνταξη της εντολής γίνεται:

```
glite-wms-job-list-match -a myJob.jdl
```

Παρακάτω φαίνεται ενδεικτικά μια λίστα από CE που επιστρέφει η εντολή.

```
Connecting to the service https://wms03.egee-see.org:7443/glite_wms_wmproxy_server
-----
COMPUTING ELEMENT IDs LIST
The following CE(s) matching your job requirements have been found:

*CEId*
- ce01.afroditi.hellasgrid.gr:2119/jobmanager-pbs-hgdemo
- ce01.ariagni.hellasgrid.gr:2119/jobmanager-pbs-hgdemo
- ce01.athena.hellasgrid.gr:2119/jobmanager-pbs-hgdemo
- ce01.grid.auth.gr:2119/jobmanager-pbs-hgdemo
- ce01.isabella.grnet.gr:2119/jobmanager-pbs-hgdemo
- ce01.kallisto.hellasgrid.gr:2119/jobmanager-pbs-hgdemo
- ce01.marie.hellasgrid.gr:2119/jobmanager-pbs-hgdemo
- ce02.athena.hellasgrid.gr:2119/jobmanager-pbs-hgdemo
- ce02.marie.hellasgrid.gr:2119/jobmanager-pbs-hgdemo
- cream-ce01.marie.hellasgrid.gr:8443/cream-pbs-hgdemo
-----
```

Εικόνα 7-1 - Εκτέλεση της εντολής `glite-wms-job-list-match`

### 7.3.2. Υποβολή job στο WMS

Η υποβολή μιας εργασίας στο WMS προϋποθέτει ότι ο χρήστης έχει δημιουργήσει το JDL αρχείο της εφαρμογής που περιγράφει την εργασία του (κεφάλαιο 7.2) και έχει αποκτήσει προσωρινό proxy certificate από την υποδομή (κεφάλαια 6.1, 6.2).

Η εντολή υποβολής του .jdl αρχείου στο WMS και άρα του job στην υποδομή, είναι η **glite-wms-job-submit**.

Η εντολή επιστρέφει ένα job identifier δηλαδή ένα μοναδικό αναγνωριστικό στοιχείο το οποίο χαρακτηρίζει το job και είναι απαραίτητο για κάθε είδους μελλοντική αλληλεπίδραση με αυτό (π.χ. έλεγχος κατάστασης, λήψη αποτελεσμάτων μετά την ολοκλήρωση, ακύρωση κ.α.).

Για να μην είναι απαραίτητο ο χρήστης να θυμάται το job id, η εντολή υποστηρίζει την αποθήκευσή του σε αρχείο με την παράμετρο -o ακολουθούμενη από το όνομα του αρχείου.

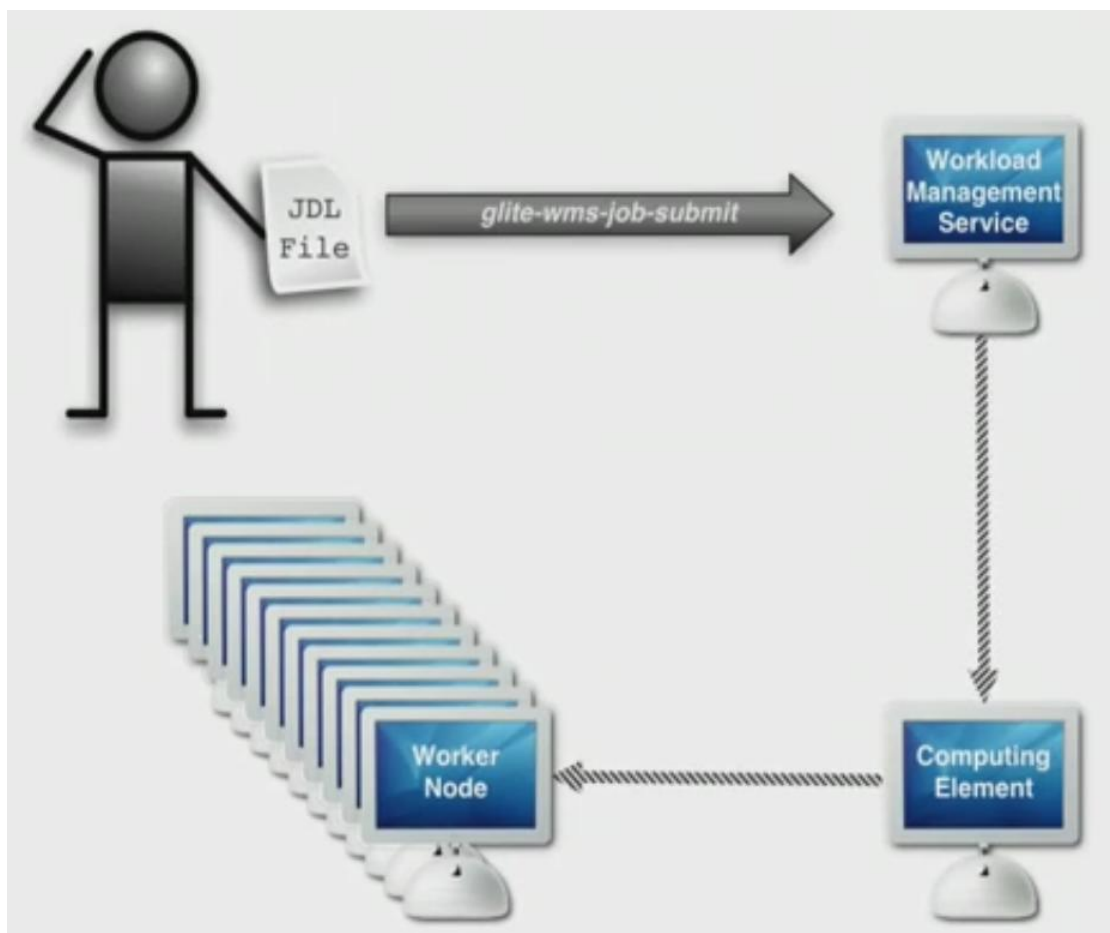
Επιπλέον είναι χρήσιμο να εκτελούμε την εντολή με την παράμετρο -a που φροντίζει να προωθήσει αυτόματα το proxy certificate του χρήστη χωρίς δική του παρέμβαση (automatic delegation).

Η τελευταία παράμετρος της εντολής πρέπει να είναι πάντα το όνομα του .jdl αρχείου του job.

Παρακάτω φαίνεται ένα συγκεντρωτικό παράδειγμα σύνταξης με αυτά που αναφέρονται παραπάνω για ένα job με αρχείο .jdl το myJob.jdl και αποθήκευση του job id στο αρχείο job.id:

```
glite-wms-job-submit -a -o job.id myJob.jdl
```

Στην ουσία αυτό που συμβαίνει με την εκτέλεση της παραπάνω εντολής, είναι ότι το αρχείο .jdl καθώς και τα αρχεία που προσδιορίζει ως απαραίτητα για το job μεταφέρονται στο WMS. Το WMS στη συνέχεια, τα προωθεί στο καταλληλότερο CE το οποίο με τη σειρά του τα προωθεί στα WNs όπου και θα γίνει η εκτέλεση του job.



Εικόνα 7-2 - glite-wms-job-submit

Παρακάτω φαίνεται το αποτέλεσμα της εκτέλεσης της εντολής.

```
glite-wms-job-submit -a -o job.id myjob.jdl
```

```
Connecting to the service https://wms03.egee-see.org:7443/glite_wms_wmproxy_server
```

```
===== glite-wms-job-submit Success=====
```

```
The job has been successfully submitted to the WMPProxy
Your job identifier is:
https://lb01.egee-see.org:9000/mbne480_2834fq3Tce9bBp
The job identifier has been saved in the following file:
/home/mkats/job.ids
=====
```

### 7.3.3. Έλεγχος κατάστασης jobs

Οι χρήστες μπορούν να ελέγχουν την τρέχουσα κατάσταση των jobs που έχουν υποβάλλει στην υποδομή με την εντολή **glite-wms-job-status**.

Η εντολή περιμένει από τον χρήστη να παρέχει το job id του job ως τελευταία παράμετρο ή το όνομα ενός αρχείου στο οποίο είναι αποθηκευμένα ένα ή περισσότερα job ids με την παράμετρο -i.

Η απλούστερη σύνταξη της για job ids αποθηκευμένα σε ένα αρχείο με όνομα job.id είναι ως εξής:

```
glite-wms-job-status -i job.id
```

Κατά την εκτέλεση της παραπάνω εντολής επιστρέφεται η κατάσταση του job που βρίσκεται αποθηκευμένη στο WMS. Ακόμα και αν ο χρήστης δεν έχει εκτελέσει την παραπάνω εντολή, το WMS ενημερώνεται αυτόματα σε προκαθορισμένα χρονικά διαστήματα από το CE για την πρόοδο των jobs που έχουν υποβληθεί στην υποδομή και δεν έχουν ακυρωθεί ή ολοκληρωθεί.

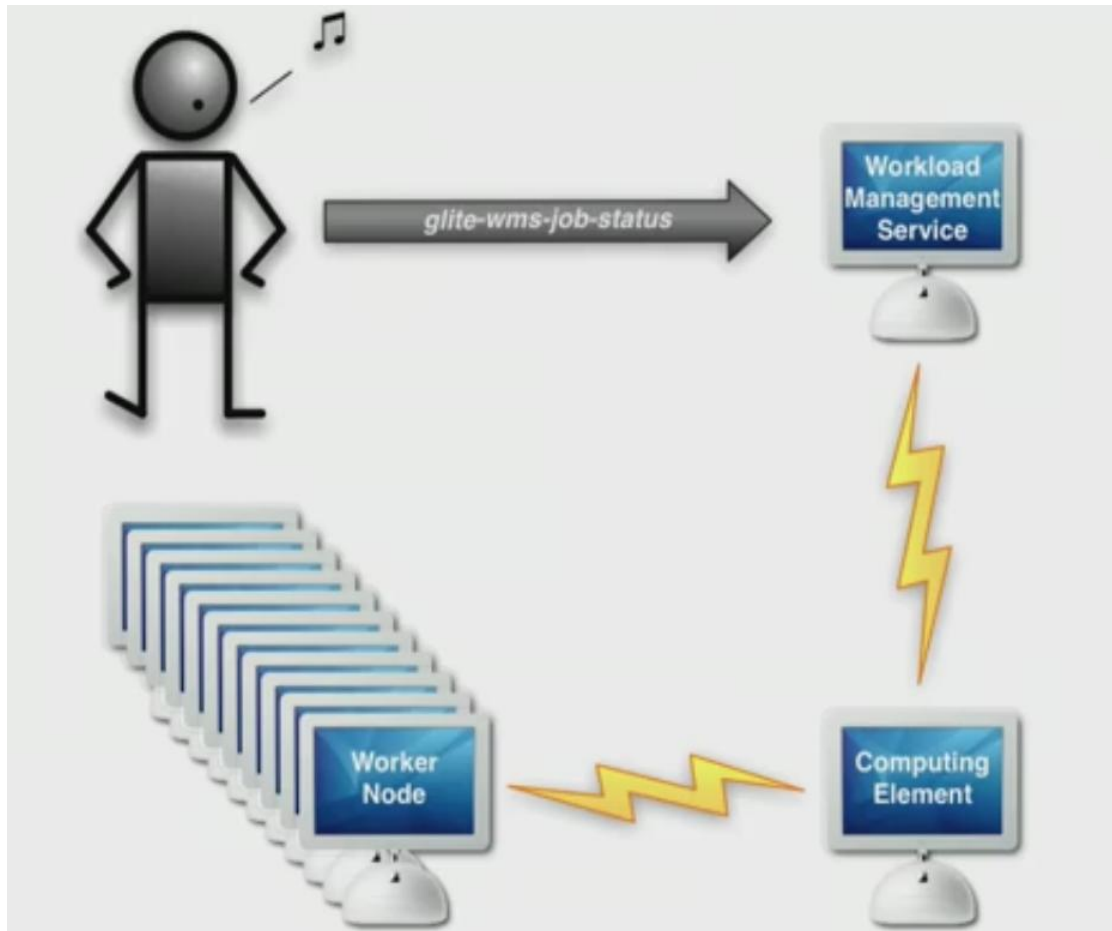
Παρακάτω φαίνεται το αποτέλεσμα της εκτέλεσης της παραπάνω εντολής.

```
glite-wms-job-status -i jobids.txt
```

```
*****
BOOKKEEPING INFORMATION:
Status info for the Job : https://lb01.egee-see.org:9000/mbne480_2834fq3Tce9bBp
Current Status:      Scheduled
Status Reason:      Job successfully submitted to Globus
Destination:        kalkan1.ulakbim.gov.tr:2119/jobmanager-lcgpbs-see
Submitted:          Tue 2 Nov 27 20:45:02 2010 EEST
*****
```

Στη γραμμή Current Status φαίνεται η τρέχουσα κατάσταση του job. Όταν το job ολοκληρωθεί το Current Status θα γίνει Done (Success) και ο χρήστης θα είναι σε θέση να λάβει τα αποτελέσματα του job με

την εντολή ***glite-wms-job-output*** (κεφάλαιο 7.3.4). Στην περίπτωση που εμφανίζεται παραπάνω, το job δεν έχει αρχίσει ακόμα να εκτελείται αλλά βρίσκεται σε αναμονή μέχρι να υπάρχουν ελεύθεροι πόροι στο σύστημα.

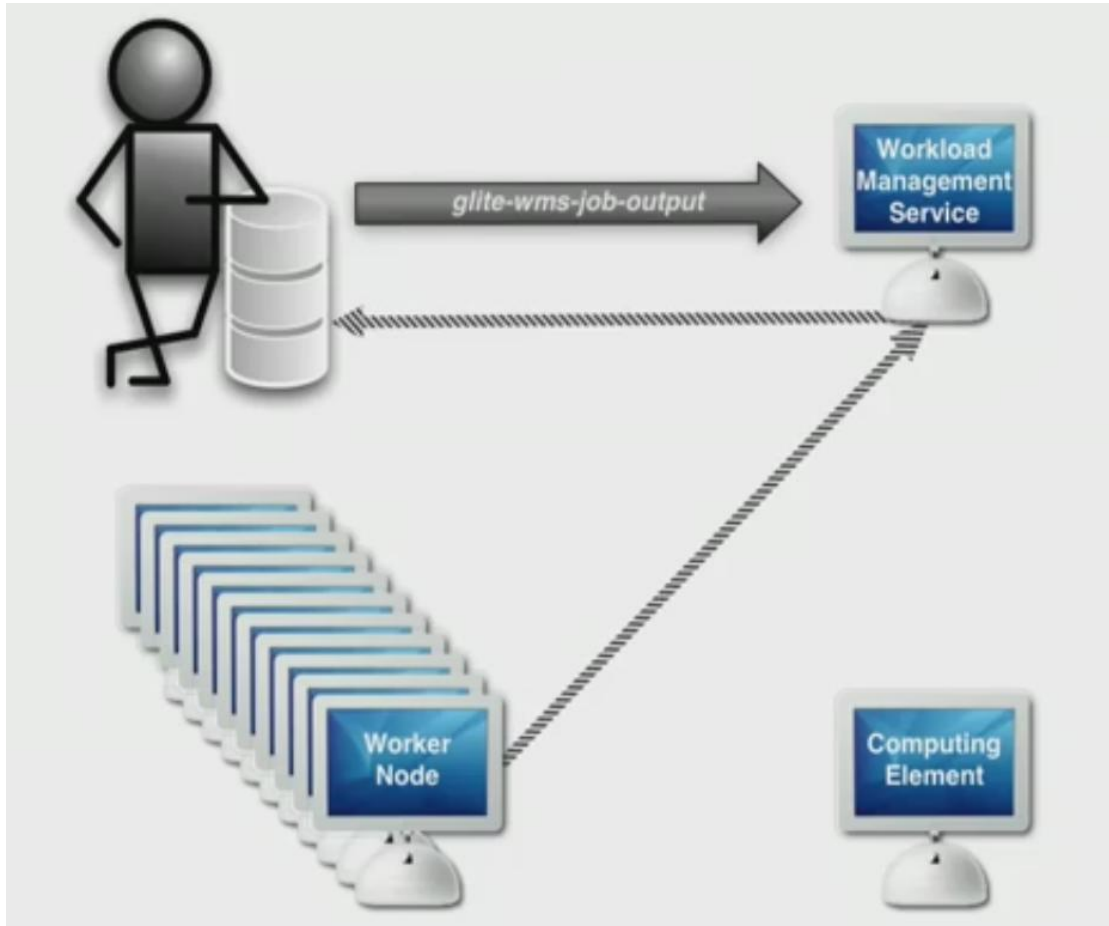


Εικόνα 7-3 - Ενημέρωση κατάστασης job με την εντολή `glite-wms-job-status`



### 7.3.4. Λήψη αποτελεσμάτων από ολοκληρωμένα jobs

Όταν ένα job ολοκληρωθεί, τα αποτελέσματά του μεταφέρονται απευθείας στο WMS από τα WNs.



Εικόνα 7-4 – Μεταφορά αποτελεσμάτων ολοκληρωμένου job

Ο χρήστης μπορεί να λάβει τα αποτελέσματα ολοκληρωμένων jobs με την εντολή ***glite-wms-job-output***.

Η εντολή δέχεται ως τελευταίο όρισμα ένα ή περισσότερα job ids των οποίων τα αποτελέσματα θα μεταφέρει στο UI από το WMS, αλλά είναι ευκολότερο να παίρνει τα job ids από αρχείο με την παράμετρο **-i** ακολουθούμενη από το όνομα του αρχείου.

Τα αποτελέσματα αποθηκεύονται σε ένα προσωρινό υποκατάλογο από την εντολή, εκτός αν ο χρήστης προσδιορίσει ένα διαφορετικό υποκατάλογο με την παράμετρο **--dir** ακολουθούμενη από τη διαδρομή του υποκαταλόγου.

Παρακάτω φαίνεται ένα παράδειγμα σύνταξης όπου το job id βρίσκεται στο αρχείο job.id (όπου αποθηκεύτηκε από την εντολή *glite-wms-job-submit* στο κεφάλαιο 7.3.2) και τα αποτελέσματα του job θα αποθηκευτούν σε ένα νέο υποκατάλογο job-output μέσα στο φάκελο του χρήστη mkats.

```
glite-wms-job-output -i job.id -dir /home/mkats/job-output
```

και το αποτέλεσμα της εκτέλεσης της εντολής:

```
Connecting to the service https://195.251.53.227:7443/glite_wms_wmproxy_server
=====
JOB GET OUTPUT OUTCOME
Output sandbox files for the job:
https://lb01.egee-see.org:9000/mbne480_2834fq3Tce9bBp
have been successfully retrieved and stored in the directory:
/home/mkats/job-output
=====
```

### 7.3.5. Ακύρωση job

Οι χρήστες έχουν τη δυνατότητα να ακυρώσουν jobs τα οποία δεν έχουν ακόμα ολοκληρωθεί με την εντολή ***glite-wms-job-cancel***.

Η εντολή δέχεται ως τελευταίο όρισμα μια σειρά από job ids αλλά μπορεί να δεχτεί και αυτή τα job ids από αρχείο, χρησιμοποιώντας την παράμετρο *-i* ακολουθούμενη από το όνομα του αρχείου.

Παράδειγμα ακύρωσης job του οποίου το job id βρίσκεται στο αρχείο job.id:

```
glite-wms-job-cancel -i job.id
```

### 7.3.6. Απλουστευμένο παράδειγμα υποβολής και διαχείρισης ενός job

Στο κεφάλαιο αυτό θα επιχειρήσουμε την υποβολή ενός απλουστευμένου job στο Grid, το οποίο θα εκτελεί την εντολή ***hostname*** του linux, θα παρακολουθήσουμε την κατάσταση εκτέλεσής του και θα λάβουμε τα αποτελέσματά του.

Η εντολή ***hostname*** με όρισμα *-f*, επιστρέφει ολόκληρο το hostname του υπολογιστή στον οποίο εκτελείται. Εάν δημιουργήσουμε ένα job που απλά εκτελεί την εντολή ***hostname***,

αυτό που θα λάβουμε ως αποτέλεσμα μετά την ολοκλήρωση του job, είναι ουσιαστικά τα hostnames των WNs στα οποία εκτελέστηκε το job μας.

Εάν επιχειρήσουμε να εκτελέσουμε την εντολή με το όρισμα -f, σε έναν υπολογιστή που δεν έχει κάποιο συγκεκριμένο hostname λόγω του δικτύου στο οποίο βρίσκεται (π.χ. ένας υπολογιστής που συνδέεται στο Internet μέσω DSL σύνδεσης) θα λάβουμε ένα αποτέλεσμα παρόμοιο με το παρακάτω:

```
# hostname -f
localhost.localdomain
```

Έστω τώρα ότι ξεκινάμε τη δημιουργία του job του παραδείγματος. Καλό θα είναι να αποθηκεύσουμε όλα τα αρχεία σχετικά με το job μας σε ένα ξεχωριστό υποκατάλογο στο φάκελο χρήστη μας στο UI.

Μεταφερόμαστε στον αρχικό φάκελο του χρήστη μας:

```
cd ~
```

Δημιουργούμε τον υποκατάλογο του job και μεταφερόμαστε σε αυτόν:

```
mkdir testJob
cd testJob
```

Δημιουργούμε το .jdl αρχείο του job μας, το οποίο θα ονομάσουμε testJob.jdl σε ένα text editor. Για το παράδειγμα θα χρησιμοποιήσουμε τον vi καθώς συναντάται σε όλες σχεδόν τις διανομές linux. Ξεκινάμε τον vi με παράμετρο το όνομα του αρχείου που θέλουμε να δημιουργήσουμε:

```
vi testJob.jdl
```

Αφού ξεκινήσει ο vi, πατάμε το πλήκτρο i για να ξεκινήσουμε να επεξεργαζόμαστε το αρχείο και γράφουμε τις παρακάτω γραμμές:

```
Executable = "/bin/hostname";
Arguments = "-f";
StdOutput = "std.out";
StdError = "std.err";
OutputSandbox = {"std.err", "std.out"};
```

Στη συνέχεια πατάμε το πλήκτρο Escape, το πλήκτρο : και τέλος το πλήκτρο x και Enter για αποθήκευση του αρχείου.

Το παραπάνω `.jdl` αρχείο προσδιορίζει ότι το εκτελέσιμο αρχείο του `job` είναι το `/bin/hostname` καθώς η εντολή `hostname` βρίσκεται στον υποκατάλογο `/bin` σε όλες τις διανομές του `linux`. Επιπλέον καθώς η εντολή θα εκτελεστεί με το όρισμα `-f`, το προσδιορίζουμε στην παράμετρο `Arguments`.

Στη συνέχεια προσδιορίζουμε το αρχείο πρότυπης εξόδου (`std.out`) και το αρχείο πρότυπης εξόδου σφαλμάτων (`std.err`). Το αποτέλεσμα του `job`, δηλαδή το `hostname` του `WN` που θα αναλάβει την εκτέλεση θα επιστραφεί στο αρχείο `std.out`, ενώ σε περίπτωση που παρουσιαστούν σφάλματα (τα οποία είναι όμως καθορισμένο να εμφανίζονται στην πρότυπη έξοδο σφαλμάτων από το εκτελέσιμο που τα παρουσίασε), θα επιστραφούν στο αρχείο `std.err`.

Η τελευταία γραμμή του αρχείου, η παράμετρος `OutputSandbox` προσδιορίζει τα ονόματα των αρχείων που θέλουμε να επιστραφούν από το `job` – στην προκειμένη περίπτωση το αρχείο σφαλμάτων και το αρχείο εξόδου.

Αφού έχουμε δημιουργήσει το αρχείο `.jdl` είναι ώρα να αποκτήσουμε ένα προσωρινό `proxy certificate` από το `VOMS` του `VO` `see` με την παρακάτω εντολή:

```
voms-proxy-init -voms see
```

Έχοντας αποκτήσει πλέον το προσωρινό `proxy certificate` εκτελούμε την παρακάτω εντολή η οποία θα ελέγξει την εγκυρότητα του `.jdl` αρχείου που ετοιμάσαμε και θα επιστρέψει μια λίστα με διαθέσιμα `CEs`:

```
glite-wms-job-list-match -a testJob.jdl
```

#### **ΣΗΜΕΙΩΣΗ:**

Σε περίπτωση που παρουσιαστεί κάποιο σφάλμα κατά την εκτέλεση της παραπάνω εντολής, θα πρέπει να σιγουρευτούμε ότι το αρχείο `.jdl` που δημιουργήσαμε είναι έγκυρο και απόλυτα ίδιο με το παράδειγμα.

Εάν το `.jdl` αρχείο είναι έγκυρο προχωράμε στην υποβολή του `job` στην υποδομή με την παρακάτω εντολή:

```
glite-wms-job-submit -o testJob.id -a testJob.jdl
```

Με την παραπάνω εντολή, το id του job που υποβάλλουμε θα αποθηκευτεί στο αρχείο testJob.id καθώς θα το χρειαστούμε παρακάτω.

Εάν η υποβολή του job ολοκληρωθεί κανονικά, μπορούμε να ελέγχουμε την κατάσταση του με την παρακάτω εντολή:

```
glite-wms-job-status -i testJob.id
```

Εδώ βλέπουμε ότι χρησιμοποιούμε πάλι το αρχείο με το id του job ώστε να μπορέσουμε να εξακριβώσουμε την τρέχουσα κατάστασή του από το WMS.

Το απλουστευμένο job που έχουμε δημιουργήσει για το παράδειγμα θα πρέπει να ολοκληρωθεί εκτός απροόπτου σφάλματος της υποδομής το πολύ μέσα σε λίγα λεπτά. Ωστόσο μπορούμε να ελέγχουμε την πρόοδό του με την προηγούμενη εντολή.

Μόλις το job ολοκληρωθεί, μπορούμε να λάβουμε τα αποτελέσματά του (τα αρχεία std.err και std.out που προσδιορίσαμε στην παράμετρο OutputSandbox του .jdl αρχείου) με την εντολή:

```
glite-wms-job-output -i testJob.id --dir ./output
```

Η παραπάνω εντολή χρησιμοποιεί πάλι το αρχείο που αποθηκεύσαμε το id του job ώστε να προσδιορίσει στο WMS το job που μας ενδιαφέρει. Επιπλέον χρησιμοποιεί την παράμετρο --dir για να προσδιορίσει τον υποκατάλογο όπου θα αποθηκευτούν τα δεδομένα του job (ο υποκατάλογος /output μέσα στον αρχικό φάκελο του χρήστη).

Πηγαίνοντας τώρα στον υποκατάλογο:

```
cd output
```

και βλέποντας τα αρχεία που περιέχει:

```
ls -l
```

βλέπουμε τα δύο αρχεία που επιστράφηκαν από την υποδομή μετά την εκτέλεση του job:

drwxr-xr-x	2	mkats	users	4096	Nov 28 12:11	.
drwxrwxrwx	20	root	root	4096	Oct 1 09:33	..
-rw-r--r--	1	mkats	users	0	Nov 28 14:02	std.err
-rw-r--r--	1	mkats	users	23	Nov 28 14:02	std.out

Εάν το job έχει ολοκληρωθεί χωρίς σφάλματα το μέγεθος του αρχείου std.err πρέπει να είναι 0. Διαβάζοντας το αρχείο std.out με την εντολή less όπως φαίνεται παρακάτω:

```
less std.out
```

Βλέπουμε το hostname του WN που έτρεξε το job μας (διαφέρει σε κάθε εκτέλεση ανάλογα με τους διαθέσιμους πόρους της υποδομής):

```
wn03.isabella.grnet.gr
```

## 7.4. MPI Jobs

---

Η Διεπαφή Ανταλλαγής Μηνυμάτων (Message Passing Interface – MPI) αποτελεί ένα πρότυπο που επιτρέπει σε διεργασίες να επικοινωνούν μεταξύ τους μέσω αποστολής και λήψης μηνυμάτων. Χρησιμοποιείται σε πληθώρα εφαρμογών και αποτελεί το de facto standard όσον αφορά παράλληλες εφαρμογές που εκτελούνται σε clusters υπολογιστών και supercomputers, όπου το κόστος πρόσβασης σε μη-τοπική μνήμη είναι υψηλό. Οι υλοποιήσεις του MPI που χρησιμοποιούνται στο EGEE είναι η MPICH-1, η MPICH-2 και η OpenMPI.

### 7.4.1. Ο μηχανισμός MPI-START

Ο μηχανισμός MPI-START είναι ο πιο εύκολος τρόπος να υποβληθούν MPI Jobs στην υποδομή. Ο μηχανισμός αυτός αναλαμβάνει να προετοιμάσει το περιβάλλον του MPI στα WNs που θα εκτελεστεί το Job.

Όταν χρησιμοποιείται ο μηχανισμός MPI-START, οι χρήστες συνήθως είναι απαραίτητο να χρησιμοποιούν ένα αρχείο εντολών (script) το οποίο ετοιμάζει τις μεταβλητές περιβάλλοντος που θα χρησιμοποιηθούν. Το αρχείο mpi-start-wrapper.sh που φαίνεται παρακάτω είναι αρκετά γενικό για να καλύπτει κάθε περίπτωση και θα ήταν καλό να μη τροποποιηθεί από τον χρήστη.

## mpi-start-wrapper.sh

```
#!/bin/bash

# Pull in the arguments.
MY_EXECUTABLE=`pwd`/$1
MPI_FLAVOR=$2

# Convert flavor to lowercase for passing to mpi-start.
MPI_FLAVOR_LOWER=`echo $MPI_FLAVOR | tr '[:upper:]' '[:lower:]'`

# Pull out the correct paths for the requested flavor.
eval MPI_PATH=`printenv MPI_${MPI_FLAVOR}_PATH`

# Ensure the prefix is correctly set. Don't rely on the defaults.
eval I2G_${MPI_FLAVOR}_PREFIX=$MPI_PATH
export I2G_${MPI_FLAVOR}_PREFIX

# Touch the executable. It exist must for the shared file system check.
# If it does not, then mpi-start may try to distribute the executable
# when it shouldn't.
touch $MY_EXECUTABLE

# Setup for mpi-start.
export I2G_MPI_APPLICATION=$MY_EXECUTABLE
export I2G_MPI_APPLICATION_ARGS=
export I2G_MPI_TYPE=$MPI_FLAVOR_LOWER
export I2G_MPI_PRE_RUN_HOOK=mpi-hooks.sh
export I2G_MPI_POST_RUN_HOOK=mpi-hooks.sh

# If these are set then you will get more debugging information.
export I2G_MPI_START_VERBOSE=1
#export I2G_MPI_START_DEBUG=1

# Invoke mpi-start.
$I2G_MPI_START
```

Εξάλλου ο χρήστης μπορεί να ορίσει τις δικές του εντολές που θα εκτελεστούν πριν και μετά την εκκίνηση του MPI εκτελέσιμου στο αρχείο mpi-hooks.sh. Οι εντολές που θα εκτελεστούν πριν την εκκίνηση του MPI εκτελέσιμου πρέπει να συμπεριλαμβάνονται στη συνάρτηση pre\_run\_hook(), ενώ αυτές που θα εκτελεστούν μετά την εκκίνηση, πρέπει να συμπεριλαμβάνονται στη συνάρτηση post\_run\_hook().

Στο παρακάτω παράδειγμα, ο χρήστης παρέχει ένα αρχείο κώδικα C που χρησιμοποιεί το MPI, το οποίο γίνεται compile στη συνάρτηση `pre_run_hook()` και στη συνέχεια εκτελείται.

### Ενδεικτικό αρχείο `mpi-hooks.sh`

```
#!/bin/sh

#
# This function will be called before the MPI executable is started.
# You can, for example, compile the executable itself.
#

pre_run_hook () {
    # Compile the program.
    echo "Compiling ${I2G_MPI_APPLICATION}"

    # Actually compile the program.
    cmd="mpicc ${MPI_MPICC_OPTS} -o ${I2G_MPI_APPLICATION}
${I2G_MPI_APPLICATION}.c"
    echo $cmd
    $cmd
    if [ ! $? -eq 0 ]; then
        echo "Error compiling program. Exiting..."
        exit 1
    fi

    # Everything's OK.
    echo "Successfully compiled ${I2G_MPI_APPLICATION}"

    return 0
}

#
# This function will be called before the MPI executable is finished.
# A typical case for this is to upload the results to a storage element.
#

post_run_hook () {
    echo "Executing post hook."
    echo "Finished the post hook."

    return 0
}
```



Στο αρχείο JDL του Job, θα πρέπει αυτή τη φορά να προσδιοριστεί ο αριθμός των CPU στον οποίο θα εκτελεστεί η MPI εφαρμογή μέσω του χαρακτηριστικού CPUNumber. Επιπλέον θα πρέπει να προσδιοριστούν δυο ορίσματα στο χαρακτηριστικό Arguments: το όνομα του εκτελέσιμου αρχείου της εφαρμογής και η υλοποίηση του MPI που θα χρησιμοποιηθεί (παράμετρος MPICH για MPICH-1, MPICH2 για MPICH-2 ή OPENMPI για Open MPI).

Τέλος, στο χαρακτηριστικό Requirements, θα πρέπει να συμπεριληφθεί το MPI-START καθώς και η επιθυμητή υλοποίηση MPI καθώς είναι απαραίτητη η ύπαρξή τους στο site όπου θα εκτελεστεί το Job.

Παρακάτω φαίνεται ένα παράδειγμα JDL αρχείου που εκτελεί την εφαρμογή helloworld και χρησιμοποιεί την υλοποίηση MPICH του MPI.

### Ενδεικτικό αρχείο JDL

```
JobType = "Normal";
CPUNumber = 4;
Executable = "mpi-start-wrapper.sh";
Arguments = "helloworld MPICH";
StdOutput = "std.out";
StdError = "std.err";
InputSandbox = {"mpi-start-wrapper.sh","mpi-hooks.sh","helloworld.c"};
OutputSandbox = {"std.err","std.out"};
Requirements =
  Member("MPI-START", ther.GlueHostApplicationSoftwareRunTimeEnvironment)
  && Member("MPICH", ther.GlueHostApplicationSoftwareRunTimeEnvironment);
```

Όπως φαίνεται παρακάτω τα αρχεία mpi-start-wrapper.sh, mpi-hooks.sh και helloworld.c (το όνομα του αρχείου με τον κώδικα της εφαρμογής) συμπεριλαμβάνονται όλα στο χαρακτηριστικό InputSandbox, καθώς είναι απαραίτητη η μεταφορά τους στο CE.

#### **ΣΗΜΕΙΩΣΗ:**

Εάν ο χρήστης θέλει να παρέχει κάποια ορίσματα για την εκτέλεση της εφαρμογής, μπορεί να τα συμπεριλάβει στο αρχείο mpi-start-wrapper.sh, στη γραμμή export \$I2G\_MPI\_APPLICATION\_ARGS.

## 8. ΔΙΑΧΕΙΡΙΣΗ ΔΕΔΟΜΕΝΩΝ ΣΤΟ GRID

### 8.1. Εισαγωγή

---

Η αξιόπιστη αποθήκευση και μετακίνηση δεδομένων αποτελεί βασικό χαρακτηριστικό των καταναμημένων συστημάτων. Για περιβάλλοντα Grid όπου ο όγκος δεδομένων που αποθηκεύονται και μετακινούνται είναι τεράστιος, η διαχείριση δεδομένων πρέπει να παρέχει σημαντικό βαθμό ελέγχου αλλά και ασφάλειας ώστε να μην υπάρχει απώλεια δεδομένων σε περίπτωση προβλημάτων σε κάποιον επιμέρους τομέα της υποδομής.

Ο πρόγονος του EGEE, το EDG (European Data Grid) απέδειξε ότι η επίτευξη των παραπάνω στόχων επιφέρει αρκετές δυσκολίες. Η ίδια η φύση των εργασιών που υποβάλλονταν στο Grid δημιούργησε αρκετά προβλήματα στη διαχείριση δεδομένων. Όταν για παράδειγμα μια εργασία ζητούσε δεδομένα από ένα απομακρυσμένο site του Grid, δε γνώριζε εάν το αρχείο μεταφερόταν ήδη από αίτηση μιας άλλης εργασίας. Επιπλέον οι μεταφορές αρχείων ξεκινούσαν χωρίς έλεγχο των διαθέσιμων πόρων στην προέλευση και τον προορισμό τους προκαλώντας έτσι υπερφόρτωση και επακόλουθη ανεπάρκεια των SEs.

Τέλος, τα προβλήματα απόδοσης με τους καταλόγους αρχείων του EDG απέτρεπαν κάποιες φορές την επιτυχή ανανέωση των καταλόγων. Αυτό είχε ως αποτέλεσμα αρχεία τα οποία είχαν ήδη μεταφερθεί να μεταφέρονται ξανά και ξανά, αυξάνοντας σημαντικά το φόρτο της υποδομής και προκαλώντας επιπλέον μείωση της απόδοσης.

Τα προβλήματα που αντιμετώπισε το EDG, λόγω φτωχού σχεδιασμού του middleware διαχείρισης δεδομένων, οδήγησαν σε προσπάθειες βελτιωμένων λύσεων λογισμικού για το EGEE. Οι προσπάθειες αυτές βελτίωσαν σημαντικά τη διαχείριση δεδομένων με νέες αποθηκευτικές λύσεις και νέα εργαλεία στο επίπεδο του middleware που εξασφαλίζουν ακέραιη και αποτελεσματική λειτουργία στις μεταφορές αρχείων.

## 8.2. Βασικές έννοιες

---

Όπως έχει αναφερθεί και στα πρώτα κεφάλαια, η πρωταρχική μονάδα αποθήκευσης δεδομένων στο Grid είναι, όπως και στους παραδοσιακούς υπολογιστές, το αρχείο. Μια σημαντική διαφορά που συναντάται στο Grid, όπως και στα περισσότερα καταναμημένα περιβάλλοντα, σε σχέση με τους παραδοσιακούς υπολογιστές, είναι ότι ένα αρχείο μπορεί να συναντάται ταυτόχρονα σε πολλά σημεία της υποδομής με τη μορφή αντιγράφων (replicas).

Για παράδειγμα ένας χρήστης μπορεί να δημιουργήσει αντίγραφα των αρχείων δεδομένων ενός job σε ένα SE που βρίσκεται πιο κοντά στο CE όπου θα εκτελεστεί η εργασία. Με τον τρόπο αυτό μειώνονται οι χρόνοι πρόσβασης στα δεδομένα λόγω ταχύτερης σύνδεσης δικτύου μεταξύ του CE και του SE.

Αυτό έχει ως αποτέλεσμα, οι χρήστες να μπορούν να αναφερθούν σε ένα όνομα αρχείου που βρίσκεται στο Grid με τρεις διαφορετικούς τύπους αναγνωριστικών:

- Με το Μοναδικό Αναγνωριστικό Grid (Grid Unique Identifier – GUID)
- Με το Φυσικό Όνομα Αρχείου (Physical File Name - PFN) ή αλλιώς Αποθηκευτικό URL (Storage URL - SURL).
- Με το Λογικό Όνομα Αρχείου (Logical File Name – LFN)

Παρακάτω αναλύονται περισσότερο τα διαφορετικά αναγνωριστικά αρχείων.

### 8.2.1. Αναγνωριστικά αρχείων GUID

Όταν ένας χρήστης μεταφέρει ένα αρχείο από τον τοπικό δίσκο σε ένα SE, το αρχείο αποκτά ένα μοναδικό αναγνωριστικό, το GUID. Το GUID αποτελείται από 36 χαρακτήρες με αποτέλεσμα να είναι δύσκολο για τον χρήστη να θυμάται το GUID κάθε αρχείου. Ένα παράδειγμα GUID είναι το ακόλουθο:

guid:2e5a4e12-4ca3-ab12-48ef-352a53c215d6

### 8.2.2. Αναγνωριστικά αρχείων PFN

Τα αναγνωριστικά PFN έχουν τη μορφή URL (Uniform Resource Locator) που χρησιμοποιούνται ευρέως σε πολλά ήδη δικτύων συμπεριλαμβανομένου και του Internet. Στην περίπτωση του grid τα URLs αυτά ονομάζονται Αποθηκευτικά URLs (Storage URLs – SURLs) και έχουν την παρακάτω δομή:

*sfn://<Όνομα Server>/<Σημείο Πρόσβασης>/<VO>/<Όνομα Αρχείου>*

Ένα παράδειγμα SURL είναι το εξής:

*sfn://se01.isabella.grnet.gr/storage/see/generated/2010-12-02/fileaa5314b5-f157-a2e3-f15a-526c2586e9a3*

Επιπλέον συναντάται και η μορφή TURL (Transport URL) που είναι περιέχει τα απαραίτητα στοιχεία για πρόσβαση σε ένα αρχείο που βρίσκεται σε ένα SE και έχει την ακόλουθη μορφή:

*<πρωτόκολλο>://<URI>*

Ένα παράδειγμα TURL είναι το εξής:

*gsiftp://tbed0101.cern.ch/data/dteam/doe/file1*

### 8.2.3. Αναγνωριστικά αρχείων LFN

Τα LFN αποθηκεύονται στο LFC (περισσότερα για το LFC στο κεφάλαιο 8.2.4) και αναθέτονται στα αρχεία απευθείας από τον χρήστη. Επιπλέον είναι μοναδικά από τη σκοπιά του χρήστη και παρέχουν ένα λογικό τρόπο αναφοράς σε αρχεία.

#### **ΣΗΜΕΙΩΣΗ:**

Τα LFN είναι μοναδικά **μόνο** από τη σκοπιά του κάθε χρήστη καθώς περισσότεροι χρήστες μπορεί να έχουν χρησιμοποιήσει το ίδιο LFN για να αναφερθούν ο καθένας σε διαφορετικό αρχείο (για παράδειγμα: readme.txt).

Τα LFN ακολουθούν μια ιεραρχική δομή παρόμοια με τη δομή υποκαταλόγων ενός λειτουργικού συστήματος της μορφής:

*lfn:/grid/<VO χρήστη>/<Όνομα χρήστη>/<Αρχείο>*

Για παράδειγμα το αρχείο thegrid.txt του χρήστη mkats, που είναι μέλος του VO SEE θα μπορούσε να έχει το lfn:

*lfn:/grid/see/mkats/thegrid.txt*

#### **8.2.4. Ο Λογικός Κατάλογος Αρχείων (Logical File Catalog – LFC)**

Καθώς οι ιδιαίτερες απαιτήσεις διαχείρισης δεδομένων του Grid δημιουργούν ένα σημαντικό επίπεδο πολυπλοκότητας, τα εργαλεία του middleware επιχειρούν να απλοποιήσουν τις εργασίες διαχείρισης αρχείων ώστε να μην επηρεάζεται η παραγωγικότητα των χρηστών.

Το LFC επιχειρεί να απλοποιήσει την αναφορά στα διάφορα αρχεία των χρηστών μέσω μιας βάσης δεδομένων στην οποία αποθηκεύεται μια λογική δομή υποκαταλόγων και αρχείων, παρόμοια με αυτή που συναντάται στα λειτουργικά συστήματα. Χάρη στο LFC, οι χρήστες δεν είναι απαραίτητο να θυμούνται ή να χρησιμοποιούν τα πολύπλοκα αναγνωριστικά αρχείων που δημιουργεί η υποδομή αρχείων του Grid. Αντίθετα είναι σε θέση να δημιουργήσουν υποκαταλόγους και αρχεία με απλά ονόματα, όπως θα έκαναν και στον τοπικό υπολογιστή τους και το LFC θα αναλάβει τη διαχείρισή τους και την αντιστοίχιση με τα αρχεία και τα αντίγραφα τους που υπάρχουν στα διάφορα SEs του Grid.

### **8.3. Διαχείριση αρχείων**

---

#### **8.3.1. Προαπαιτούμενα**

Για να μπορέσει ένας χρήστης να διαχειριστεί αρχεία στην υποδομή του Grid, πρέπει πρώτα να τηρούνται οι παρακάτω προϋποθέσεις:

- Ο χρήστης πρέπει να διαθέτει ένα grid proxy certificate (δείτε κεφάλαιο 6.1).
- Ο χρήστης πρέπει να έχει ορίσει κάποιες μεταβλητές στο περιβάλλον του UI που είναι απαραίτητες για τη σωστή λειτουργία των σετ εντολών διαχείρισης αρχείων (δείτε κεφάλαιο 8.3.2).

### 8.3.2. Απαραίτητες μεταβλητές περιβάλλοντος στο UI

Για τη σωστή λειτουργία των σετ εντολών διαχείρισης αρχείων, είναι απαραίτητη η ρύθμιση ορισμένων μεταβλητών περιβάλλοντος στο UI του χρήστη. Οι μεταβλητές αυτές είναι οι εξής:

- **LCG\_CATALOG\_TYPE** – προσδιορίζει τον τύπο του καταλόγου αρχείων που θα χρησιμοποιηθεί. Για τους χρήστες του HellasGrid, η μεταβλητή πρέπει να έχει την τιμή **lfc**
- **LFC\_HOST** – η διεύθυνση του LFC server του χρήστη. Για τους χρήστες του HellasGrid, ο server αυτός είναι ο **lfc.isabella.grnet.gr**
- **LCG\_GFAL\_INFOSYS** – η διεύθυνση του BDII server του χρήστη. Για τους χρήστες του HellasGrid η διεύθυνση που πρέπει να χρησιμοποιηθεί είναι: **bdii.isabella.grnet.gr:2170**
- **LCG\_GFAL\_VO** – Το VO του οποίου τα SEs θα χρησιμοποιηθούν για αποθήκευση αρχείων και στο οποίο πρέπει να είναι μέλος ο χρήστης. Οι νέοι χρήστες του HellasGrid είναι μέλη του VO Νοτιοανατολικής Ευρώπης SEE άρα η μεταβλητή πρέπει να έχει την τιμή **see**.

### 8.3.3. Ρύθμιση μεταβλητών περιβάλλοντος στο UI

Για να μην είναι απαραίτητη η ρύθμιση των μεταβλητών κάθε φορά που ο χρήστης θέλει να χρησιμοποιεί τα σετ εντολών διαχείρισης δεδομένων προτείνεται να γίνει η ρύθμισή τους στο αρχείο `.bash_profile`.

#### **ΤΟ ΑΡΧΕΙΟ `.bash_profile`:**

Το αρχείο `.bash_profile` στο Linux βρίσκεται μέσα στο φάκελό δεδομένων κάθε χρήστη και εκτελείται κάθε φορά που ο χρήστης κάνει login στο σύστημα.

Το αρχείο `.bash_profile` είναι ένα αρχείο κειμένου άρα μπορεί να γίνει επεξεργασία του σε οποιαδήποτε εφαρμογή επεξεργασίας κειμένου. Παρακάτω χρησιμοποιείται η εφαρμογή `vi` καθώς συναντάται σε όλες σχεδόν τις διανομές Linux.

Ξεκινώντας από τον αρχικό υποκατάλογο δεδομένων του χρήστη (cd ~) ανοίγουμε το αρχείο `.bash_profile` με τον `vi`. Εάν το αρχείο δεν υπάρχει ήδη θα δημιουργηθεί.

```
cd ~
vi .bash_profile
```

Πατάμε το πλήκτρο `i` για να επεξεργαστούμε το κείμενο. Εάν το αρχείο περιέχει ήδη κείμενο, προχωράμε με τα βέλη στο τέλος του, όπου προσθέτουμε τις παρακάτω γραμμές:

```
export LCG_CATALOG_TYPE=lfc
export LFC_HOST=lfc.isabella.grnet.gr
export LCG_GFAL_INFOSYS=bdi.isabella.grnet.gr:2170
export LCG_GFAL_VO=see
```

Στη συνέχεια πατάμε με τη σειρά το πλήκτρο `Escape`, το πλήκτρο `:`, το πλήκτρο `x` και τέλος το `Enter` για να αποθηκεύσουμε τις αλλαγές στο αρχείου.

#### **ΣΗΜΕΙΩΣΗ:**

Στο Linux τα αρχεία που το όνομά τους έχει στην αρχή το χαρακτήρα της τελείας `.` (όπως το `.bash_profile`) θεωρούνται «κρυφά» και δεν εμφανίζονται απευθείας με την εντολή `ls`.

Για να μπορέσετε να δείτε τα κρυφά αρχεία σε ένα φάκελο του Linux χρησιμοποιείτε την παράμετρο `-a` της εντολής `ls`.

Παράδειγμα: `ls -a`

### **8.3.4. Τα σετ εντολών διαχείρισης αρχείων**

Υπάρχουν δυο σετ εντολών διαχείρισης αρχείων στο Grid. Ένα σετ με πρόθεμα εντολών `lcg-` και ένα με πρόθεμα `lfc-`. Το γεγονός αυτό προκαλεί πολλές φορές σύγχυση στους νέους χρήστες που μπερδεύουν τη σημασία των εντολών. Στην πραγματικότητα τα δυο αυτά σετ έχουν διαφορετικούς ρόλους, όπως περιγράφεται παρακάτω:

- **Εντολές με πρόθεμα `lcg-`**

Αυτό το σετ εντολών περιέχει εντολές που επιδρούν απευθείας επάνω στα δεδομένα. Επιπλέον, κάποιες από αυτές επηρεάζουν και το LFC όπως για παράδειγμα η **`lcg-cr`** που μεταφέρει ένα

αρχείο στο SE και παράλληλα το καταχωρεί στο LFC με ένα συγκεκριμένο όνομα.

- **Εντολές με πρόθεμα lfc-**

Αυτές οι εντολές επιδρούν μόνο πάνω στο LFC και δεν επηρεάζουν τα δεδομένα. Παράδειγμα: η εντολή **lfc-mkdir** δημιουργεί ένα νέο υποκατάλογο στο LFC.

### 8.3.5. Λίστες αρχείων και υποκαταλόγων

Για κάθε υποστηριζόμενο VO, υπάρχει ένας ξεχωριστός αρχικός υποκατάλογος κάτω από τον υποκατάλογο /grid/. Η εντολή **lfc-ls** επιτρέπει στους χρήστες να δουν τα περιεχόμενα κάθε υποκαταλόγου. Καθώς η εντολή είναι παρόμοια με την εντολή **ls** του linux, πολλές από τις παραμέτρους της **ls** υπάρχουν ίδιες και στην **lfc-ls**. Στο παρακάτω παράδειγμα βλέπουμε τα περιεχόμενα του υποκαταλόγου /grid/see:

```
lfc-ls -l /grid/lsgrid/
-rw-rw-r-- 1 26397 26000      2412768502 Jul 20 2009 100406.tar.gz
drwxrwxr-x 0 26413 26000          0 Jun 01 2010 100s_18mm
drwxrwxr-x 0 26413 26000          0 Jun 02 2010 100s_4mm
-rw-rw-r-- 1 26123 26000      2787 Feb 09 2007 150k LAN.rpad
-rw-rw-r-- 1 26123 26000      2787 Feb 09 2007 150k.rpad
drwxrwxr-x 0 26413 26000          0 Jun 01 2010 150s_18mm
drwxrwxr-x 0 26413 26000          0 Jun 02 2010 150s_4mm
drwxrwxr-x 0 26413 26000          0 Jun 07 2010 1s_4mm
drwxrwxr-x 0 26413 26000          0 Jun 01 2010 1s_plug_18mm
drwxrwxr-x 0 26413 26000          0 May 31 2010 1source_4mm
.....
.....
.....
```

### 8.3.6. Δημιουργία νέου υποκαταλόγου

Η εντολή που χρησιμοποιείται για τη δημιουργία νέων υποκαταλόγων είναι η **lfc-mkdir**. Η εντολή είναι παρόμοια με την **mkdir** του linux. Για να μπορέσει ένας χρήστης να δημιουργήσει ένα νέο υποκατάλογο σε ένα ήδη υπάρχον υποκατάλογο, πρέπει να έχει δικαίωμα εγγραφής στον υποκατάλογο αυτό. Για να δημιουργήσετε ένα υποκατάλογο με όνομα test μέσα στον υποκατάλογο του χρήστη σας στο VO SEE η σύνταξη της εντολής θα ήταν η εξής:

```
lfc-mkdir /grid/see/<όνομα χρήστη>/test
```



Παράδειγμα, για τον χρήστη mkats:

```
lfc-mkdir /grid/see/mkats/test
```

### 8.3.7. Αποθήκευση αρχείου σε ένα SE

Η εντολή που χρησιμοποιείται για μεταφορά αρχείων από τον τοπικό υπολογιστή σε ένα SE είναι η **lcg-cr**. Το cr σημαίνει copy-register (αντιγραφή – καταχώρηση) καθώς η εντολή μεταφέρει το αρχείο στο SE και στη συνέχεια το καταχωρεί στο LFC με ένα λογικό όνομα.

Βασική παράμετρος της εντολής είναι η παράμετρος --vo <ονομασία VO> που προσδιορίζει την ονομασία του VO στο οποίο ανήκει ο χρήστης. Εξάλλου, η παράμετρος -l <lfn αρχείου> προσδιορίζει το λογικό όνομα που θα αποκτήσει το αρχείο στο lfc (χρησιμοποιούμε το lfn του αρχείου). Τέλος, η παράμετρος -d <διεύθυνση SE> προσδιορίζει το SE στο οποίο θα αποθηκευτεί το αρχείο.

Στο παρακάτω παράδειγμα αντιγράφουμε το αρχείο test.txt από το φάκελο του χρήστη μας στο UI, στο SE se01.kallisto.hellasgrid.gr και το καταχωρούμε στο LFC στη διαδρομή /grid/see/mkats/test.txt.

```
lcg-cr --vo see -l lfn:/grid/see/mkats/test.txt -d se01.kallisto.hellasgrid.gr \
file://$HOME/test.txt
```

Αφού ολοκληρωθεί η εκτέλεση της εντολής, επιστρέφεται το guid του αρχείου που αντιγράφηκε, το οποίο θα είναι ίδιο και για τα replicas του αρχείου αυτού στο Grid.

Μπορούμε να ελέγξουμε ότι η αντιγραφή ολοκληρώθηκε επιτυχώς με την εντολή **lcg-cr** (κεφάλαιο 8.3.5).

### 8.3.8. Δημιουργία αντιγράφων (replication) αρχείων

Κάποιες φορές μπορεί το ίδιο αρχείο να υπάρχει σε περισσότερα από ένα SEs για διάφορους λόγους όπως ασφάλεια ή γρηγορότερη δικτυακή πρόσβαση. Τα αντίγραφα του αρχείου ονομάζονται replicas και η διαδικασία δημιουργίας τους, replication.

Για να δούμε όλα τα replicas ενός αρχείου χρησιμοποιούμε την εντολή **lcg-lr** με παράμετρο το lfn του αρχείου. Το παρακάτω παράδειγμα επιστρέφει τα replicas του αρχείου που αποθηκεύσαμε στο SE στο κεφάλαιο 8.3.7:

```
lcg-cr lfn:/grid/see/mkats/test.txt
```

Για να δημιουργήσουμε επιπλέον replicas ενός αρχείου σε διαφορετικά SEs χρησιμοποιούμε την εντολή **lcg-rep**. Η παράμετρος `--vo <ονομασία VO>` προσδιορίζει το VO το οποίο παρέχει πρόσβαση στο SE και στο οποίο ο χρήστης είναι μέλος. Εξάλλου η παράμετρος `-d <διεύθυνση SE>` προσδιορίζει το SE στο οποίο θα δημιουργήσουμε το νέο replica του αρχείου. Τέλος προσδιορίζουμε το lfn του αρχείου που μας ενδιαφέρει.

Στο παρακάτω παράδειγμα κάνουμε replicate το αρχείο lfn:/grid/see/test.txt που αποθηκεύσαμε στο παράδειγμα του κεφαλαίου 8.3.7 στο SE se01.kallisto.hellasgrid.gr.

```
lcg-rep --vo see -d se01.isabella.gnet.gr lfn:/grid/see/mkats/test.txt
```

### 8.3.9. Επιστροφή αρχείου από το Grid στο UI

Για να μεταφέρουμε ένα αρχείο πίσω στον τοπικό υπολογιστή, από το SE που το έχουμε αντιγράψει, χρησιμοποιούμε την εντολή **lcg-cp**. Η πρώτη παράμετρος που χρησιμοποιούμε είναι η `--vo <ονομασία VO>` με την οποία προσδιορίζουμε το VO το οποίο έχει πρόσβαση στο SE που μας ενδιαφέρει και του οποίου είμαστε μέλος. Στη συνέχεια προσδιορίζουμε πρώτα το lfn του αρχείου που θέλουμε να μεταφέρουμε στο UI και στη συνέχεια τη διαδρομή στο δίσκο του UI που θέλουμε να το αποθηκεύσουμε.

Στο παρακάτω παράδειγμα μεταφέρουμε το αρχείο lfn:/grid/see/test.txt που αποθηκεύσαμε στο παράδειγμα του κεφαλαίου 8.3.7, στο φάκελο χρήστη μας στο UI, με όνομα test2.txt.

```
lcg-cp --vo see lfn:/grid/see/mkats/test.txt file://$HOME/test2.txt
```

### 8.3.10. Διαγραφή αρχείων και υποκαταλόγων

Η εντολή που χρησιμοποιείται για διαγραφή αρχείων και υποκαταλόγων από την υποδομή, είναι η **lcg-del**. Για να γίνει διαγραφή ενός μόνο replica του αρχείου χρησιμοποιούμε τη παρακάτω σύνταξη:

```
lcg-del <SURL αρχείου στο SE>
```

Για να βρούμε το SURL σε ένα SE, του αρχείου που μας ενδιαφέρει, χρησιμοποιούμε την εντολή **lcg-lr** (κεφάλαιο 8.3.8).

Για να διαγράψουμε ένα αρχείο από όλα τα SEs και το LFC, χρησιμοποιούμε την παράμετρο **-a** <LFN αρχείου>. Εάν για παράδειγμα θέλαμε να σβήσουμε το αρχείο που αποθηκεύσαμε στο παράδειγμα του κεφαλαίου 8.3.7, θα χρησιμοποιούσαμε την εξής σύνταξη:

```
lcg-del -a lfg:/grid/see/mkats/test.txt
```

Για να διαγράψουμε ένα υποκατάλογο από το LFC, χρησιμοποιούμε την εντολή **lfc-rm** με την παράμετρο **-r** και το όνομα του υποκαταλόγου. Για παράδειγμα, για να σβήσουμε τον υποκατάλογο **test** που δημιουργήσαμε στο κεφάλαιο 8.3.6 στη διαδρομή **/grid/see/mkats**, θα χρησιμοποιούσαμε την παρακάτω σύνταξη:

```
lfc-rm -r /grid/see/mkats/test
```

#### **ΣΗΜΕΙΩΣΗ:**

Για να μπορέσουμε να σβήσουμε ένα υποκατάλογο, πρέπει να είναι άδειος.

## **9. ΥΠΗΡΕΣΙΑ ΠΛΗΡΟΦΟΡΙΩΝ**

### **9.1. Εισαγωγή**

Το Grid αποτελείται από sites που διαθέτουν από διαφορετικές συνθέσεις υλικού και έχουν διαφορετικά πακέτα λογισμικού. Εξάλλου οι εργασίες των χρηστών έχουν συνήθως συγκεκριμένες απαιτήσεις σε υλικό και σε πακέτα λογισμικού. Η υπηρεσία πληροφοριών του Grid επιτρέπει την αναζήτηση των απαραίτητων πόρων για την εκτέλεση των εργασιών και πληροφορεί τους χρήστες για τους πόρους που έχουν στη διάθεσή τους ως μέλη συγκεκριμένων VO.

Η Υπηρεσία βασίζεται στο Berkeley Database Information Index (BDII), μια βάση δεδομένων LDAP που ανανεώνεται από εξωτερικές διαδικασίες. Η ανανέωση γίνεται μέσω ανταλλαγών μηνυμάτων μορφής LDIF από πολλές εξωτερικές πηγές. Τα μηνύματα LDIF συγχωνεύονται, συγκρίνονται με τα περιεχόμενα της βάσης

δεδομένων και δημιουργείται ένα αρχείο LDIF με τις διαφορές. Αυτό το αρχείο χρησιμοποιείται για την ανανέωση της βάσης δεδομένων.

## 9.2. Αναζήτηση γενικών πληροφοριών πόρων

Η αναζήτηση πληροφοριών σχετικές με τους πόρους του Grid στους οποίους έχει πρόσβαση ένα VO γίνεται με την εντολή **lcg-infosites**. Η παράμετρος --vo <ονομασία VO> θα πρέπει να συμπεριλαμβάνεται σε κάθε εκτέλεση της εντολής ώστε να προσδιορίζεται το VO του οποίου οι πόροι θα επιστραφούν.

### 9.2.1. Αναζήτηση πληροφοριών για υπολογιστικούς πόρους

Η **lcg-infosites** επιστρέφει πληροφορίες σχετικά με τα διαθέσιμα CE σε ένα VO προσθέτοντας την παράμετρο ce. Εάν για παράδειγμα θέλουμε να δούμε τα διαθέσιμα CEs του VO Νοτιοανατολικής Ευρώπης (SEE VO) θα χρησιμοποιήσουμε την παρακάτω σύνταξη:

```
lcg-infosites --vo see ce
```

Η εντολή θα μας επιστρέψει μια λίστα με τα διαθέσιμα CEs, και πληροφορίες σχετικά με το κάθε ένα από αυτά (συνολικός αριθμός CPU, αριθμός διαθέσιμων CPU, αριθμός συνολικών Jobs, αριθμός Jobs που εκτελούνται, αριθμός Jobs που βρίσκονται σε αναμονή) όπως φαίνεται παρακάτω:

#CPU	Free	Total Jobs	Running	Waiting	Computing	Element
184	132	1	1	0		ce.ngcc.acad.bg:2119/j...
10	2	5	5	0		ce02.grid.acad.bg:2119/ ...
296	171	0	0	0		tau-ce.hep.tau.ac.il:2119/ ...
6	6	0	0	0		ce02.marie.hellasgrid.gr:2119/.....
.....						

### 9.2.2. Αναζήτηση πληροφοριών για αποθηκευτικούς πόρους

Η **lcg-infosites** επιστρέφει πληροφορίες σχετικά με τα διαθέσιμα SE σε ένα VO προσθέτοντας την παράμετρο se. Εάν για παράδειγμα θέλουμε να δούμε τα διαθέσιμα SEs του VO Νοτιοανατολικής Ευρώπης (SEE VO) θα χρησιμοποιήσουμε την παρακάτω σύνταξη:

```
lcg-infosites --vo see se
```

Η εντολή θα μας επιστρέψει μια λίστα με τα διαθέσιμα SEs, και πληροφορίες σχετικά με το κάθε ένα από αυτά (διαθέσιμος αποθηκευτικός χώρος σε kb, δεσμευμένος αποθηκευτικός χώρος κ.α.) όπως φαίνεται παρακάτω:

Avail Space(Kb)	Used Space(Kb)	Type	SEs
1	1	n.a	tau-se.hep.tau.ac.il
1	1	n.a	tau-se.hep.tau.ac.il
555110498	145763352	n.a	grid002.ics.forth.gr
555110498	145763352	n.a	grid002.ics.forth.gr
665607866	7924326726	n.a	se01.athena.hellasgrid.gr
450353487	41798001	n.a	testbed002.grid.ici.ro

### 9.2.3. Αναζήτηση του κοντινότερου SE για κάθε CE

Για να δούμε το κοντινότερο SE για κάθε διαθέσιμο CE χρησιμοποιούμε την εντολή **lcg-infosites** με την παράμετρο `closeSE`. Στην περίπτωση που ενδιαφερόμαστε για τους πόρους του SEE VO η σύνταξη φαίνεται παρακάτω:

```
lcg-infosites -vo see closeSE
```

#### ΠΡΟΣΟΧΗ:

Οι παράμετροι των εντολών κάνουν διάκριση πεζών και κεφαλαίων χαρακτήρων. Το SE στην παράμετρο `closeSE` γράφεται με κεφαλαίους χαρακτήρες.

Η εντολή επιστρέφει μια λίστα με CEs και τα κοντινά τους SEs όταν αυτά υπάρχουν όπως φαίνεται παρακάτω:

Name of the CE: ce301.intercol.edu:2119/jobmanager-lcgpbs-see

Name of the CE: cream01.athena.hellasgrid.gr:8443/cream-pbs-see  
se01.athena.hellasgrid.gr

Name of the CE: cream02.athena.hellasgrid.gr:8443/cream-pbs-see  
se01.athena.hellasgrid.gr

Name of the CE: ce01.athena.hellasgrid.gr:2119/jobmanager-pbs-see  
se01.athena.hellasgrid.gr

Name of the CE: ce02.athena.hellasgrid.gr:2119/jobmanager-pbs-see

se01.athena.hellasgrid.gr

Name of the CE: vm004.one.ypepth.grnet.gr:8443/cream-pbs-see

Αρχικά επιστρέφεται το η διεύθυνση του CE και στις γραμμές που ακολουθούν οι διευθύνσεις των SEs. Είναι πιθανό κάποια CEs να μη διαθέτουν κοντινά SEs, όπως το πρώτο CE (ce301.intercol.edu) στην παραπάνω λίστα.

#### 9.2.4. Αναζήτηση των τοπικών εξυπηρετητών LFC

Η εντολή **lcg-infosites** επιστρέφει τη λίστα των τοπικών εξυπηρετητών LFC με την παράμετρο lfcLocal. Για να δούμε τους τοπικούς εξυπηρετητές LFC για το SEE VO θα χρησιμοποιήσουμε την παρακάτω σύνταξη:

```
lcg-infosites --vo see lfcLocal
```

Η εντολή επιστρέφει τις διευθύνσεις των εξυπηρετητών LFC όπως φαίνεται παρακάτω:

```
lfc.isabella.grnet.gr  
lfc.ipb.ac.rs  
lfc.grid.auth.gr
```

### 9.3. Αναζήτηση πληροφοριών πόρων βάση κριτηρίων

---

Η εντολή **lcg-infosites** που αναφέρεται στο κεφάλαιο 9.2 επιστρέφει πληροφορίες σχετικά με όλους τους πόρους ενός VO του Grid. Αντίθετα η εντολή **lcg-info** επιστρέφει πληροφορίες μόνο για πόρους που πληρούν χαρακτηριστικά που ορίζονται από το χρήστη.

#### 9.3.1. Χαρακτηριστικά (attributes)

Κάθε πόρος της υποδομής διαθέτει συγκεκριμένα «χαρακτηριστικά» (attributes) τα οποία εξαρτώνται από τη σύνθεσή του υλικού, το διαθέσιμο λογισμικό και τις ρυθμίσεις του.

Τα υποστηριζόμενα από την υποδομή χαρακτηριστικά, έχουν απλοποιημένα ονόματα τα οποία μπορούν να χρησιμοποιούν οι χρήστες με την εντολή **lcg-info**. Η εντολή επιστρέφει τα απλοποιημένα ονόματα όλων των διαθέσιμων χαρακτηριστικών με την παράμετρο `--list-attrs` όπως φαίνεται στο παρακάτω παράδειγμα:

```

lcg-info --list-attrs
Attribute name      Glue object class  Glue attribute name
WorstRespTime      GlueCE             GlueCEStateWorstResponseTime
CEAppDir            GlueCE             GlueCEInfoApplicationDir
TotalCPUs           GlueCE             GlueCEInfoTotalCPUs
MaxRunningJobs     GlueCE             GlueCEPolicyMaxRunningJobs
CE                  GlueCE             GlueCEUniqueID
WaitingJobs         GlueCE             GlueCEStateWaitingJobs
MaxCPUTime          GlueCE             GlueCEPolicyMaxCPUTime
LRMSVersion         GlueCE             GlueCEInfoLRMSVersion
MaxTotalJobs        GlueCE             GlueCEPolicyMaxTotalJobs
CEStatus            GlueCE             GlueCEStateStatus
LRMS                GlueCE             GlueCEInfoLRMSType
CEVOs               GlueCE             GlueCEAccessControlBaseRule
AssignedJobSlots    GlueCE             GlueCEPolicyAssignedJobSlots
FreeCPUs            GlueCE             GlueCEStateFreeCPUs
RunningJobs         GlueCE             GlueCEStateRunningJobs
EstRespTime         GlueCE             GlueCEStateEstimatedResponseTime
FreeJobSlots        GlueCE             GlueCEStateFreeJobSlots
Cluster             GlueCE             GlueCEInfoHostName
TotalJobs           GlueCE             GlueCEStateTotalJobs
Priority             GlueCE             GlueCEPolicyPriority
.....

```

### 9.3.2. Αναζήτηση CE βάση κριτηρίων

Η εντολή **lcg-info** μπορεί να θέσει πολύ συγκεκριμένα ερωτήματα (queries) στο σύστημα πληροφοριών του Grid, σχετικά με τα χαρακτηριστικά των πόρων προς αναζήτηση.

Η παράμετρος `--vo <Όνομασία VO>` προσδιορίζει το VO σπου οποίου τους πόρους θα γίνει αναζήτηση. Εξάλλου, η παράμετρος `-query` συνθέτει ένα ερώτημα (query) προς το σύστημα πληροφοριών που περιορίζει τα αποτελέσματα βάση προδιαγραφών που πρέπει να πληρούν ενώ η παράμετρος `--list-ce` προσδιορίζει ότι θα επιστραφούν μόνο CEs που ικανοποιούν το query. Τέλος η παράμετρος `--attrs` προσδιορίζει τα attributes (κεφάλαιο **Error! Reference source not found.**) που επιθυμεί ο χρήστης να εμφανιστούν για κάθε επιστρεφόμενο αποτέλεσμα.

Παρακάτω δίνεται ένα παράδειγμα όπου γίνεται αναζήτηση των CEs του VO SEE που διαθέτουν επεξεργαστή της οικογένειας Athlon και χρησιμοποιούν λειτουργικό σύστημα Scientific Linux. Επιπλέον για τα συγκεκριμένα CEs, ζητάμε να εμφανιστούν τα χαρακτηριστικά RunningJobs (αριθμός Jobs που εκτελούνται) και FreeCPUs (αριθμός ελεύθερων CPU).

```
lcg-info --vo see --list-ce --query 'Processor=*Athlon*,OS=*Scientific*' \
--attrs 'RunningJobs,FreeCPUs'

- CE: ce02.marie.hellasgrid.gr:2119/jobmanager-pbs-see
- RunningJobs      0
- FreeCPUs         6
```

### 9.3.3. Εμφάνιση διαθέσιμων πακέτων λογισμικού

Η εντολή **lcg-info** χρησιμοποιείται πολύ συχνά για να δούμε εάν ορισμένα πακέτα λογισμικού είναι διαθέσιμα στα CEs ενός συγκεκριμένου VO ώστε να γνωρίζουμε εάν η εφαρμογή που θέλουμε να εκτελέσουμε ως Job στο Grid θα εκτελεστεί επιτυχώς. Τα πακέτα λογισμικού βρίσκονται στο attribute Tag (περισσότερα για τα attributes στο κεφάλαιο **Error! Reference source not found.**).

Συνεπώς, για να εμφανίσουμε τα εγκατεστημένα πακέτα λογισμικού στα CEs του VO SEE θα χρησιμοποιήσουμε την παρακάτω σύνταξη:

```
lcg-info --vo see --list-ce --attrs Tag

- CE: node001.grid.auth.gr:2119/jobmanager-pbs-see

- Tag          LCG-2
                LCG-2_1_0
                LCG-2_1_1
                LCG-2_2_0
                LCG-2_3_0
                LCG-2_3_1
                LCG-2_4_0
                LCG-2_5_0
                LCG-2_6_0
                LCG-2_7_0
                GLITE-3_0_0
                GLITE-3_0_1
                GLITE-3_0_2
                GLITE-3_1_0
                R-GMA
                SI00MeanPerCPU=630
                SF00MeanPerCPU=545
                MPICH
```



RASTER3D  
MOLSCRIPT-1.0.2  
VO-argo-prod-corsika-v6.7.20-SL3-i386-gccany  
VO-atlas-offline-14.0.0-i686-slc4-gcc34-opt  
VO-atlas-offline-14.1.0-i686-slc4-gcc34-opt  
VO-atlas-offline-14.2.0-i686-slc4-gcc34-opt  
VO-atlas-offline-14.2.10-i686-slc4-gcc34-opt  
VO-atlas-production-12.0.31  
VO-atlas-production-12.0.6  
VO-atlas-production-12.0.7  
VO-atlas-production-13.0.30  
VO-atlas-production-13.0.30.4

## ΕΠΕΞΗΓΗΣΗ ΣΥΝΤΟΜΟΓΡΑΦΙΩΝ

<b>API</b>	Application Programming Interface
<b>BDII</b>	Berkeley Database Information Index
<b>CA</b>	Certification Authority
<b>CE</b>	Computing Element
<b>CPU</b>	Central Processing Unit
<b>DSL</b>	Digital Subscriber Line
<b>D-SEE</b>	Distributed SEE
<b>EDG</b>	EU-DataGrid
<b>EGEE</b>	Enabling Grids for E-science and industry in Europe
<b>FLOPs</b>	Floating Point Operations
<b>GFAL</b>	Grid File Access Library
<b>GEANT</b>	The pan-European Gigabit Research Network
<b>GSI</b>	Grid Security Infrastructure
<b>GUID</b>	Grid Unique Identifier
<b>IS</b>	Information Service
<b>JDL</b>	Job Description Language
<b>LFC</b>	Logical File Catalog
<b>LFN</b>	Logical File Name
<b>LCG</b>	LHC Computing Grid
<b>LDAP</b>	Lightweight Directory Access Protocol
<b>MPI</b>	Message Passing Interface
<b>MPICH</b>	MPI Chameleon
<b>MPPs</b>	Massively Parallel Processors
<b>PFN</b>	Physical File Name
<b>RA</b>	Registration Authority

<b>RC</b>	Resource Center
<b>ROC</b>	Regional Operations Center
<b>SA</b>	Specific Service Activities
<b>SCP</b>	Secure Copy
<b>SE</b>	Storage Element
<b>SEE</b>	South Eastern Europe
<b>SFN</b>	Site File Name
<b>SRM</b>	Storage Resource Management
<b>SSH</b>	Secure Shell
<b>SSL</b>	Secure Sockets Layer
<b>SURL</b>	Storage Uniform Resource Locator
<b>UI</b>	User Interface
<b>URI</b>	Uniform Resource Identifier
<b>URL</b>	Uniform Resource Locator
<b>VO</b>	Virtual Organization
<b>VOMS</b>	Virtual Organization Membership Service
<b>WMS</b>	Workload Management System
<b>WN</b>	Worker Node

## 10. ΕΥΡΕΤΗΡΙΟ ΕΝΤΟΛΩΝ

*chmod*, 41

*glite-wms-job-list-match*, 51, 52

*glite-wms-job-output*, 54, 56

*glite-wms-job-status*, 54, 55

*glite-wms-job-submit*, 52, 53, 57

*grid-cert-info*, 41

*grid-proxy-destroy*, 43, 45

*grid-proxy-info*, 43, 44

*grid-proxy-init*, 42, 43, 44

*lcg-cp*, 68, 69

*lcg-cr*, 66, 67, 68

*lcg-del*, 69

*lcg-info*, 72, 73, 74

*lcg-infosites*, 70, 71, 72

*lcg-lr*, 68, 69

*lcg-rep*, 68

*lfc-ls*, 66

*lfc-mkdir*, 66, 67

*lfc-rm*, 69

*myproxy-destroy*, 45

*myproxy-info*, 45

*myproxy-init*, 45

*openssl*, 40

*pscp*, 24, 39

*voms-proxy-init*, 44

## 11. ΕΥΡΕΤΗΡΙΟ ΕΙΚΟΝΩΝ

Οργανωτική δομή SEE ROC.....	14
Πιστοποιητικό HellasGrid CA εγκατεστημένο σε Mozilla Firefox .....	22
Ρυθμίσεις για σύνδεση στο UI ui01.ariagni.hellasgrid.gr μέσω PuTTY SSH.....	25
Mozilla Firefox - Μενού ψηφιακών πιστοποιητικών .....	28
Mozilla Firefox - Λίστα ψηφιακών πιστοποιητικών .....	28
Mozilla Firefox - Εξαγωγή ψηφιακού πιστοποιητικού .....	29
Mozilla Firefox - Προστασία ιδιωτικού κλειδιού με κωδικό .....	30
Google Chrome - Εξαγωγή ψηφιακού πιστοποιητικού.....	30
Google Chrome - Λίστα ψηφιακών πιστοποιητικών .....	31
Google Chrome - Εξαγωγή ψηφιακού πιστοποιητικού.....	32
Google Chrome - Εξαγωγή ψηφιακού πιστοποιητικού, επιλογή τύπου αρχείου .....	33
Google Chrome - Εξαγωγή ψηφιακού πιστοποιητικού, επιλογή ονόματος αρχείου .....	35
Google Chrome - Εξαγωγή ψηφιακού πιστοποιητικού, ολοκλήρωση διαδικασίας .....	36
Internet Explorer - Εξαγωγή ψηφιακού πιστοποιητικού .....	37
Internet Explorer - Εξαγωγή ψηφιακού πιστοποιητικού .....	38
Internet Explorer - Εξαγωγή ψηφιακού πιστοποιητικού, επιλογή τύπου αρχείου.....	38
Internet Explorer - Εξαγωγή ψηφιακού πιστοποιητικού, επιλογή κωδικού προστασίας ιδιωτικού κλειδιού.....	39
Internet Explorer - Εξαγωγή ψηφιακού πιστοποιητικού, επιλογή ονόματος αρχείου .....	40
Internet Explorer - Εξαγωγή ψηφιακού πιστοποιητικού, ολοκλήρωση διαδικασίας .....	41
Δημιουργία proxy certificate από τη γραμμή εντολών .....	44
Σχηματική αναπαράσταση δημιουργίας Grid Proxy Certificate μέσω της εντολής grid-proxy-init .....	45
Πληροφορίες τρέχοντος proxy certificate .....	45
Δημιουργία «πακέτου» πιστοποιητικών από την υπηρεσία VOMS με την εντολή voms-proxy-init. ....	47
Εκτέλεση της εντολής glite-wms-job-list-match .....	56
glite-wms-job-submit.....	57
Ενημέρωση κατάστασης job με την εντολή glite-wms-job-status...	59
Μεταφορά αποτελεσμάτων ολοκληρωμένου job .....	60

## 12. ΒΙΒΛΙΟΓΡΑΦΙΑ

1. Fran Berman, Geoffrey Fox, Tony Hey, «GRID COMPUTING: MAKING THE GLOBAL INFRASTRUCTURE A REALITY», John Wiley & Sons, 2003
2. Ananth Grama, Anshul Gupta, George Karypis, Vipin Kumar, «Introduction to Parallel Computing, Second Edition», Addison-Wesley, 2003
3. Ian Foster, «What Is The Grid? A Three Point Checklist», (<http://dlib.cs.odu.edu/WhatIsTheGrid.pdf>)
4. Wikipedia: Grid Computing ([http://en.wikipedia.org/wiki/Grid\\_computing](http://en.wikipedia.org/wiki/Grid_computing))
5. Toolbox for IT: Globus Toolkit ([http://it.toolbox.com/wiki/index.php/Globus\\_Toolkit](http://it.toolbox.com/wiki/index.php/Globus_Toolkit))
6. The Globus Alliance Website (<http://www.globus.org>)
7. The Grid Technology Cookbook (<http://hv3.phys.lsu.edu:8000/cookbook/gtcb/index.php>)
8. GridCafe Website (<http://www.gridcafe.org>)
9. EGEE Project Website (<http://project.eu-egee.org/index.php?id=118>)
10. EΔET Grnet Website (<http://www.grnet.gr>)
11. Hellas Grid Website (<http://www.hellasgrid.gr>)
12. Hellas Grid/GOC Wiki (<http://wiki.hellasgrid.gr>)
13. EGEE SEE ROC Wiki (<http://wiki.egee-see.org>)

14. dCache.ORG - Using x509 Certificates  
(<http://www.dcache.org/manuals/Book-1.9.12/config/cf-gplazma-certificates.shtml>)
15. Job Description Language Attributes Specification  
(<https://edms.cern.ch/file/555796/1/EGEE-JRA1-TEC-555796-JDL-Attributes-v0-8.pdf>)
16. Italian Grid Infrastructure Wiki  
(<http://wiki.italiangrid.org/>)
17. Storage and Data Management in EGEE  
([http://www.gridpp.ac.uk/papers/storage\\_and\\_dm.pdf](http://www.gridpp.ac.uk/papers/storage_and_dm.pdf))
18. SARA wiki – Using the Grid  
([https://grid.sara.nl/wiki/index.php/Using\\_the\\_Grid](https://grid.sara.nl/wiki/index.php/Using_the_Grid))