Fragenkatalog zum Hauptdiplom Informatik - Kurs Verteilte Systeme

1. KE1 - Grundlagen verteilter Systeme
    1. Was unterscheidet verteilte Systeme von herkömmlichen Rechnern?
       Verteilte Systeme bestehen aus mehreren Rechnern ohne gemeinsamen Speicher, die dem Benutzer wie ein virtuelles Einprozessorsystem erscheinen (Transparenz)
    2. Was genau versteht man in diesem Zusammenhang unter 'Transparenz' und welche Arten der Transparenz gibt es?
       Ein System, daß seine Verteiltheit vollständig vor dem Benutzer verbirgt und wie ein Einprozessorsystem erscheint, heißt transparent da die Implementierung für den Benutzer durchsichtig - im Sinne von unsichtbar - ist. Man unterscheidet:
        - Ortstransparenz
          Die Benutzer wissen nicht, wo sich bestimmte Betriebsmittel befinden.
        - Migrationstransparenz
          Die Betriebsmittel können migrieren, d.h. sich an einen anderen Ort begeben, ohne daß sich ihr Name ändert.
        - Replikationstransparenz
          Die Benutzer wissen nicht wieviele Kopien existieren.
        - Nebenläufigkeitstransparenz
          Mehrere Benutzer können automatisch Betriebsmittel gemeinsam benutzen.
        - Parallelitätstransparenz
          Aktivitäten können ohne das Wissen des Benutzers parallel stattfinden.
          Beispiel: Programmierer muß sich nicht um die Aufteilung des Kontrollflusses auf mehrere Prozessoren kümmern, weil das System dies 'von selbst' erkennt.
    3. Was ist der Unterschied zwischen einem Multiprozessor- und einem Multicomputersystem?
       Ein Multiprozessorsystem besitzt für alle Prozessoren einen gemeinsamen Arbeitsspeicher, in einem Multicomputersystem hat jeder Prozessor seinen eigenen Arbeitsspeicher.
       Man unterscheidet bus-basierte (bis 64 Prozessoren) und schalter-basierte-Multiprozessorsysteme auf der einen Seite und bus-basierte und schalter-basierte-Multicomputersysteme auf der anderen Seite.
    4. Was ist der Unterschied zwischen einem eng und einem lose gekoppelten System?
       Im allgemeinen sind Multiprozessorsysteme meistens enger gekoppelt als Multicomputersysteme da bei einem gemeinsamen Speicher die Wege zu diesem entweder kurz oder in high speed Technik (e.g. Lichtwellenleiter) ausgeführt werden müssen. Die lose gekoppelten Systeme werden meist als verteilte Systeme, die übrigen als parallele Systeme gekennzeichnet.
    5. Was versteht man unter zustandslosen und zustandsbehafteten Servern?
       Ein zustandsloser Server benötigt keine Informationen über vergangene Aktionen, jede Anfragenachricht eines Clients beschreibt den verlangten Server-Dienst vollständig. Beispiel: Dateien unter NFS benötigen kein Open und kein Close, jede Leseanforderung ist in sich abgeschlossen.
    6. Was versteht man unter einem Mikrokern?

Der klassische Ansatz der Bertriebssystemarchitektur führt zu rel. großen monolithischen Betriebssystemkernen die alle Dienste und Treiber für die Client-Prozesse bereitstellen. Soll eine Funktionalität und/oder ein neuer Gerätetreiber hinzugefügt werden, dann muß der Kern entsprechend neu gebunden werden. Bei einer Mikrokern-Architektur dagegen ist der Betriebssystemkern nur sehr klein und beinhaltet nur die allernötigsten Funktionalitäten:

- Speicherverwaltung
- Prozeßverwaltung
- Kommunikationsverwaltung
- I/O Verwaltung

Alle anderen Dienste - insbesondere der Dateidienst - werden durch Server bereitgestellt, die außerhalb des Betriebssystemkerns oder sogar auf einem anderen Rechner im Benutzermodus laufen. (Beispiele: Amoeba, Mach, Windows-NT)

7. Nennen Sie einige Ihnen bekannte verteilte Anwendungen!
   Banksysteme, Flug/Reise-Buchungssysteme, verteilte Datenbanken. Betriebssysteme Amoeba, Mach, Athena, NT(?)
8. Welche Vorteile hat ein verteiltes System gegenüber einem Großrechner?
   - Wirtschaftlichkeit
   - Geschwindigkeit
   - Verteiltheit - dadurch sind u.U. natürlichere Problemlösungen möglich
   - Zuverlässigkeit durch Redundanz
   - Skalierbarkeit
9. Welche Nachteile gibt es?
   - Komplexere Software
   - Kommunikationsprobleme sind möglich - dadurch schlechtere Performance, Verlust von Nachrichten
   - Nur schwache Schutzvorkehrungen - Ambivalenz zwischen Sicherheit und Einfacheit des Zugriffs auf Resourcen
10. Was ist ein Snoopy-Cache?
    Ein Snoopy Cache ist ein write-through-cache der außerdem noch den gemeinsamen Bus in einem Multi-Prozessor-System 'abschnüffelt'. Bei einer Schreiboperation eines anderen Prozessors auf eine Adresse im gemeinsamen Speicher, die im Snoopy-Cache gespeichert ist, wird diese entweder entfernt oder gegen den aktuellen Wert ausgetauscht. Snoopy Caches sind kohärent, d.h. sie enthalten für jede gespeicherte Adresse den jeweils aktuellen Wert.

2. KE 2 - Kommunikation
   1. Erklären Sie das OSI-Schichtenmodell (wichtig Layer 1-4)!
      Siehe Antwort zur Frage 2!
   2. Wie funktioniert der Nachrichenaustausch zwischen Rechnern grundsätzlich und welche Protokolle werden dabei eingesetzt?
      Nachrichtenaustausch mit Hilfe des OSI (Open System Interconnection) Protokollstapels. Das OSI-Schichtenmodell modelliert die Kommunikation zwischen Rechnern in 7 Schichten:
      - Applikation Layer (Anwendungsschicht)
        Sammlung diverser Protokolle für allgemeine Aktivitäten wie z.B. email (X.400) oder die X.500 Directory Services.

- Presentation Layer (Darstellungsschicht)
  Interpretation der Semantik der übertragenen Bits. Vereinfacht die Kommunikation zwischen Maschinen mit unterschiedlichen Datenrepräsentationen. (ASCII -> EBCDIC oder big-/little Endianformat)
- Session Layer (Sitzungsschicht, Kommunikationssteuerungsschicht)
  Verbesserte Version der Transportschicht. Stellt Dialogkontrolle zur Verfügung, überwacht welcher Teilnehmer gerade kommuniziert und bietet Synchronisationsmöglichkeiten an. Wird nur selten unterstützt!
- Transportation Layer (Transportschicht)
  Teilt den Datenstrom in Segmente (Daten(Layer 5-7) -> Segmente(Layer 4) -> Packet(Layer 3) -> Frame(Layer 2)) auf und sorgt empfängerseitig dafür, daß die Segmente in der richtige Reihenfolge angeordnet werden (bei IP ist die Reihenfolge des Eintreffens beim Empfänger nicht vorgegeben - bei X.25 gibt es eine definierte Reihenfolge).
  Das offizielle ISO-Transportprotokoll gibt es in den Varianten TP0 bis TP4 die jeweils Unterschiede in der Fehlerbehandlung und der Möglichkeit mehrere Transportverbindungen über eine X.25 Verbindung zu schicken, aufweisen.
  TCP aus dem TCP/IP Protokollstack ist TP4 ähnlich.
  Im Gegensatz zur Vermittlungsschicht, die Verbindungen zwischen Endsystemen bereitstellt, liefert die Transportschicht Verbindungen zwischen in verschiedenen Endsystemen aktiven, miteinander kommunizierenden Anwendungsprozessen
- Network Layer (Vermittlungsschicht)
  Hauptaufgabe: Routing in WANs mit dem Ziel die kürzesten Packet(! Packet ist die Dateneinheit dieser Schicht)laufzeiten zu erreichen. Zwei Network-Layer Protokolle werden benutzt:
  - X.25: verbindungsorientiertes Protokoll, beste Route wird während des Verbindungsaufbaus bestimmt.
  - IP: verbindungsloses Protokoll. Jedes IP-Paket wird unabhängig von allen anderen verschickt und geroutet.
- Data Link Layer (Sicherungsschicht)
  Sendet Bits in Frames (Rahmen) und überprüft, ob diese korrekt empfangen wurden. Jeder Rahmen wird dazu markiert (Anfang/Ende) und mit einer Prüfsumme versehen (CRC). Falls ein Frame fehlerhaft empfangen und nicht korrigiert werden kann, wird der Frame nochmals angefordert
- Physical Layer (Bitübertragungsschicht)
  Bestimmt wie die 0- und 1-Bits übertragen werden. Welche Spannung für 0 und welche Spannung für 1 benutzt wird. Und ob die Übertragung gleichzeitig und mit welcher Geschwindigkeit möglich ist. Das Protokoll der phys. Schicht behandelt die Standardisierung der elektrischen, mechanischen und signalisierenden Schnittstellen, so daß falls ein 0-Bit gesendet wird auch ein 0-Bit empfangen wird und umgekehrt (Beispiel: RS-232C Standard)

3. Fragen zum OSI-Schichtenmodell:
    - Data-Link-Layer: Wie kann der Empfänger Bitfehler erkennen und ggf beheben (Fehlererkennung in Schicht 2)?
    - Durch Berechnung eines Parity-Bits.
    - Durch Anwendung eines Hamming-Codes. Damit können mit einem Hamming Abstand k, k-1 Bit-Fehler erkannt und (k-1)/2 Fehler behoben werden.
    - CRC (Cyclic Redundancy Code): Dabei wird die zu sendende Bitfolge als binäres Polynom der Länge d betrachtet. Das Polynom der zu sendenden Bits wird zunächst mit der höchsten Potenz des Generatorpolynoms multipliziert (Left Shift um grad(C)) und dann durch das Generatorpolynom dividiert. Der sich dabei ergebende Rest wird zum linksgeshifteten Datenpolynom addiert und gesendet. Auf der Empfängerseite wird die gesendete Bitfolge durch Division mit dem Codierungspolynom auf Fehlerfreiheit geprüft. Verbleibt ein Rest, dann ist ein Übertragungsfehler aufgetreten.
      Bei Verwendung eines Generatorpolynoms vom Grad k werden Fehlerbündel der Länge k (Burstfehler) mit Sicherheit erkannt. Als Generatorpolynome verwendet man sogenannte irreduzible (=unzerlegbare P. -> 'Primpolynom') Polynome, d.h. es gibt keine reelen Nullstellen. Übliche Generatorprobleme sind $X^{16}+X^{12}+X^5+1$ (CCITT) oder $X^{16}+X^{15}+X^2+1$ (CRC-16). Auch zyklische Verschiebungen von Codewörtern sind gültige Codewörter! (math. Grundlage: Polynom Restklassen-Ringe)
      CRC Codierung kann leicht in Hardware mit Hilfe rückgekoppelter Schieberegister und EOR Gattern realisiert werden (rekursive Potenzrestgewinnung mittels Horner Schema).
    - Erklären Sie die Unterschiede zwischen verbindungsorientierten und verbindungslosem Datenaustausch! Wie unterscheiden sich die Datenpakete?
      Beim verbindungsorientierten Datenaustausch beginnt jede Kommunikation mit der Verbindungs-Anfrage des Senders an den Empfänger. Falls die Verbindung akzeptiert wird, erhält der Anfrager eine Verbindungsbezeichnung die dann für weitere Anfragen benutzt wird. Die Route der Nachrichten wird ebenfalls während der Verbindungsaufnahme festgelegt.
      Beim verbindungslosen Protokoll werden alle Nachrichten unabhängig gesendet, geroutet und empfangen. Im Kopf der Datenpakete muß die Reihenfolge vermerkt werden, da diese wegen des unterschiedlichen Routings empfängerseitig nicht identisch sein muß mit der Sendereihenfolge.
    - Es sind viele Nachrichten unterwegs - wie unterscheidet man die zugehörigen Pakete?
      Dieses Phänomen/Problem tritt beim IP-Protokoll auf, man begegnet ihm durch Numerierung der Datenpakete.
    - Was tut die Netzwerkschicht?
      Routing im WAN, verbindunglos, verbindungsorientiert (siehe Antwort Frage 2)
    - Was macht die Transportschicht?
      Siehe 2). Kommunikation zwischen Prozessen.

4. Wie erkennt man beim Hamming-Code welche Bits gestört sind?
   Bei der im Kurs beschriebenen Besicherung mit bitpositionsbezogenen Paritybits und der Annahme, daß jeweils nur ein Bit falsch ist, folgt, daß man die verfälschte Bitposition durch Addition der Positionsnummern derjenigen Paritybits deren Paritybedingung verletzt ist erhält.
   Zum Berechnen der Paritybits schreibt man alle d Datenbits eines Rahmens so auf daß an den Stellen $2^0,2^1,2^2,...,2^{p-1}$,wobei p=é $\log_2$dù ist, die Prüfbits stehen. Die Werte der Paritybits ergeben sich durch das Abzählen der übereinstimmenden Einsen in den Dualdarstellungen der Bitpositionsnummern und des Datenrahmens. Bei der Fehlererkennung - unter der Annahme, daß nur ein Bit gestört ist - gibt es zwei Möglichkeiten:
   - Datenbit gestört
     Der Empfänger addiert die Nummern der Prüfbiits deren Paritybedingung verletzt ist; das Ergebnis ist die Position des gestörten Datenbits.
   - Paritybit gestört
     Hier ergibt sich die Korrektur direkt aus den Datenbits, indem man die Berechnung der Paritybits einfach empfängerseitig wiederholt. Die Summe der gestörten Paritybits weist wieder auf die Nummer des gestörten Datenbits - nämlich auf sich selbst!
     Allgemein (nach Kurs Rechnernetze) werden bei einem Hamming Abstand von k k-1 Fehler erkannt und ⌊ (k-1)/2 Fehler behoben, indem man empfängerseitig dasjenige Codewort wählt, das den kleinsten Hammingabstand zum Fehlerpolynom hat. Die beschriebene Hammingcodierung hat einen Hammingabstand von mindestens 3, denn es können (k-1)/2=1 Fehler behoben und damit auch k-1=2 Fehler erkannt werden.
5. Was versteht man unter dem 'Sliding Window Protokoll'?
   Hierbei handelt es sich um ein verbindungsloses Protokoll zur Datenübertragung im Datalink-Layer bei dem Bestätigungen Huckepack (Piggyback) mit den auszutauschenden Frames verschickt werden. (Siehe EA2 Aufg.4)
6. Die Bestätigung der von A gesendeten i - Ahead Frames erfolgt durch die Station B implizit (Piggyback) durch das Versenden eines Frames der Nummer i in die entgegengesetzte Richtung.
7. Station A darf nur Rahmen senden, deren Nummern im Intervall [$LNAck_A$,$LNRec_A$+Ahead-1] dem Sending Window liegen.
8. Station B braucht nur Rahmen zu empfangen, deren Nummern im Intervall [$LNRec_B$,$LNRec_B$+2xAhead-1] dem Receiving Window liegen.
   Statt unendliche Arrays für Sending und Receiving Windows zu benutzen, genügt es, zyklische Arrays der Länge 4xAhead zu verwalten.
   $LNAck_{A:}$= niedrigste Rahmennummer die noch nicht bestätigt ist.
   $LNRec_{A:}$= niedrigste Nummer eines noch nicht bei A eingegangenen Rahmens.
   Frage: Kann der Abstand zwischen $LNRec_A$ und $LNAck_A$ höchstens Ahead betragen? Ja! Es gilt nach Lemma 2.4: $LNRec_A$ - Ahead <= $LNAck_A$.
9. Kann beim Sliding-Window-Protokoll eines der Sendefenster leer werden? Ja aber nur eines von beiden! Begründung siehe Musterlösung EA2 96/97.
10. Was versteht man unter dem Client/Server Modell?

Darunter versteht man die Konstruktion eines Betriebssystems aus einer Menge kooperierender Prozesse (=Servern) die für die Benutzer (=Clients) Dienste bereitstellen. Ein gegebener Rechner kann nur einen Prozeß, mehrere Clients, mehrere Server oder eine Mischung aus beiden ausführen.

11. Wie funktioniert die Kommunikation beim C/S
    Einfaches und im Gegensatz zu OSI (im LAN) schnelles, verbindungsloses Anfrage/Antwort-Protokoll mit nur drei Übertragungsschichten.
12. Nennen Sie Beispiele von Servern!
    Fileserver, Datenbankserver, X-Server, Programm(Applikations)server
13. Welche Schichten und Protokolle werden beim C/S im LAN benötigt?
    - Bitübertragungsschicht (durch Hardware realisiert)
    - Verbindungsschicht (durch Hardware realisiert)
    - Anfrage/Antwort Protokoll (verbindungslos)
      Schicht 3+4 werden wegen fehlendem Routing und wegen fehlender Verbindung nicht benötigt

3. KE 3 - RPC, Kommunikation in Gruppen, Sicherheit
    1. Wie funktioniert RPC (remote procedure call)? Welche Probleme gibt es insbesondere bei der Parameterübergabe?
       Aufrufreihenfolge beim RPC: Client (Aufruf der Prozedur)-> Clientstub(Verpacken in Nachricht) -> Kern des Client OS (Versenden der Nachricht) -> Kern des Server OS (Empfang der Nachricht) -> Serverstub (Auspacken der Nachricht) -> Server (Ausführen der Prozedur) und zurück.
       Bei der Parameterübergabe gibt es im Gegensatz zu lokalen Prozeduren wegen fehlendem gemeinsamen Speichers kein Call-by-Reference sondern nur einen Copy-Restore Mechanismus. Der Copy-Restore Mechanismus kann jedoch mit dem Call-by-reference Kombiniert werden, wenn man dem Server-Stub einen zeiger auf das kopierte Feld übergibt, bzw dem Client einen Zeiger auf das Ergebnisfeld. Trotzdem gibt es keine Lösung für komplexere (Structs, Graphen, Bäume) Datenstrukturen.
       Weitere Probleme gibt es durch unterschiedliche Datenrepräsentationen auf den beteiligten Rechnern (Lösung: Konvertierung).
    2. Wie werden RPC Prozeduren generiert?
       Mit Hilfe der Spezifikationen der Server-Unterprogramme und der Codierungsregeln für Nachrichten können spezielle Stub-Übersetzer Bibliothekseinträge für Stubs erzeugen.
    3. Wie erfolgt die Fehlerbehandlung bei RPC?
       Es gibt mehrere Fehlermöglichkeiten
       - Client kann Server nicht lokalisieren
         Behandlung: Exception (=Ausnahmebehandlung)
       - Anfragenachricht an Server geht verloren
         Behandlung: Timeout abwarten, dann Exception auslösen
       - Antwortnachricht von Server geht verloren
         Behandlung: Schwierig! Bei idempotenten Anfragen kein Problem: einfach die Anfrage so oft wiederholen wie es nötig ist (bis eine Antwort erhalten wird).
         Bei nicht idempotenten Anfragen: Sequenznummern für Anfragen vergeben, so daß der Server erkennen kann, ob es sich um die Original-Anfrage oder um eine Wiederholung handelt.

- Server fällt aus nachdem er die Anfrage erhalten hat
  Behandlung:
  - Mindestens-Einmal-Semantik. Warten bis der Server neu gestartet wird oder bis der Client an einen neuen Server gebunden wird. Dann wird ein neuer Versuch unternommen. RPC mindestens einmal, möglicherweise aber auch öfter.
  - Höchstens-Einmal-Semantik. Es wird sofort aufgegeben und dem Client das Scheitern der Anfrage mitgeteilt. RPC höchstens einmal, möglicherweise aber auch keinmal.
  - Nichts wird garantiert. Der RPC wird keinmal oder viele Male ausgeführt.
- Client fällt aus nachdem er Anfrage geschickt hat
  Task auf einem Server die von einem ausgefallenen Client stammt nennt man Waise (orphan). Vier Möglichkeiten zur Behandlung von Waisen:
  - Ausrottung. Clients protokollieren ihre RPC's in einer Datei auf einem stabilen Hintergrundspeicher. Nach einem Neustart wird diese Datei überprüft und etwaige Waisen werden terminiert.
  - Reinkarnation. Jeder Client unterrichtet seine Partner-Rechner über einen Neustart indem er den Beginn einer neuen Epoche ankündigt. Alle Rechner brechen daraufhin alle älteren (=frühere Epochen) RPC-Berechnungen ab. Sollten trotzdem noch Waisen überlebt haben und Antworten zurückschicken können diese an der nicht mehr aktuellen Epoche erkannt und abgebrochen werden.
  - Sanfte Reinkarnation. Jeder Rechner überprüft beim Eintreffen einer Nachricht über eine neue Epoche ob er RPC's ausführt. Falls ja, wird versucht den Besitzer zu finden. Falls es keinen Besitzer mehr gibt, wird die Berechnung abgebrochen.
  - Verfallszeitpunkte (expiration). Jede RPC erhält eine feste Zeitdauer T zur Bearbeitung. Falls die Bearbeitung dann noch nicht abgeschlossen ist, muß explizit ein neues Zeitintervall T angefordert werden. Fall keine neue Anforderung kommt, handelt es sich nach dem Ablauf von T um eine Waise und der Prozess wird abgebrochen.

4. Wie unterscheidet sich C/S von RPC?
   C/S ist I/O-orientiert (Anfrage/Antwort), RPC besitzt den höheren Abstraktionsgrad, da es (nahezu) transparente Prozeduraufrufe bereitstellt. RPC ist auf Situationen beschränkt, in denen ein einzelner Client mit einem Server kommuniziert.
5. Was versteht man unter Gruppenkommunikation?
   Eine Gruppe ist eine Menge von Prozessen, die auf eine vom System oder vom Benutzer festgelegte Art und Weise miteinander zusammenarbeiten. Eine wichtige Eigenschaft aller Gruppen ist, daß wenn eine Nachricht an die Gruppe gesendet wird, alle Mitglieder der Gruppe diese Nachricht empfangen (Eins-Zu-Viele Kommunikation). Eine besondere Technik der Gruppenkommunikation in Netzwerken ist das Multicasting: Jeder Gruppe wird eine Multicasting Adresse zugewiesen. Beim Senden an diese Multicasting Adresse wird die Nachricht

automatisch allen Stationen mit dieser Multicast-Adresse geschickt. Wo dieser Mechanismus nicht zur Verfügung steht, gibt es häufig den Broadcast. Dabei wird die Nachricht an jede Station geschickt. Diese müssen dann selbst entscheiden, ob die Nachricht für sie relevant ist oder nicht.
Eine weitere Möglichkeit ist die Prädikatenadressierung bei der sich diejenigen Stationen angesprochen fühlen, die eine bestimmte Eigenschaft haben (z.B. eine bestimmte Menge freien Speichers)
Man unterscheidet offene und geschlossene Gruppen im Hinblick auf die Außenkommunikation, sowie demokratische und hierarchische Gruppen mit Blick auf die innere Strukturierung.

6. Atomarität der Gruppenkommunikation
   Eine Charakteristik der Gruppenkommunikation ist die Alles-oder-Nichts-Eigenschaft. Diese Eigenschaft heißt auch Atomarität oder atomarer Broadcast und besagt, daß eine an eine Gruppe gesendete Nachricht entweder alle Mitglieder erreicht oder keinen.
7. Problem der Reihenfolge von Nachrichten
   Falls zwei Mitglieder einer Gruppe in der kein Multicast und kein Broadcast zur Verfügung stehen, an alle anderen Mitglieder der Gruppe eine Nachricht senden wollen, dann müssen die Nachrichten der Reihe nach verschickt werden. Der Zeitpunkt des Eintreffens der Nachrichten ist dann nicht vorherbestimmbar, da der Zugang zum LAN ebenfalls nichtdeterministisch ist. Die Empfänger bekommen also u.U. kritische Nachrichten in nicht vorhersagbarer und in unterschiedlicher Reihenfolge. Um dem zu begegnen, wird eine globale Zeitordnung implementiert; oder zumindest, da das nicht einfach zu realisieren ist, eine konstistente Zeitordnung. Es wird dadurch sichergestellt, daß Nachrichten von allen Empfängern in der gleichen Reihenfolge registriert werden.
8. Beschreiben Sie das Sicherheitsteilsystem Kerberos aus dem System Athena (MIT)!
   Kerberos ist ein Server im verteilten System Athena. Kerberos ist für die Sicherheit zuständig. Der Server, auf dem Kerberos ausgeführt wird, muß physisch gegen Einbruch gesichert sein. Kerberos speichert alle Passwörter des Systems, die Workstations der Benutzer sind ungesichert und enthalten keine Passwort-Verzeichnisse.
9. Was geschieht beim Login unter Kerberos?
   Zunächst erfolgt die Eingabe des Benutzernamens. Dieser wird vom Login-Programm an Kerberos weitergegeben. Kerberos holt das Benutzer-Passwort aus seiner Datenbank und bildet damit den User-Key K-B sowie den Session-Key B-TGS. Mit dem User-Key verschlüsselt Kerberos nun den Session-Key als auch das mit dem Schlüssel K-TGS des Ticket-Granting-Servers verschlüsselte Ticket-Granting-Ticket (TGT) nachdem B-TGS in das TGT eingesetzt wurde und schickt es an die Workstation. Der Benutzer hat nun an der Workstation sein Passwort eingegeben. In der Workstation wird daraus ebenfalls der User-Key K-B abgeleitet. Nur wenn das Passwort korrekt war, können die von Kerbereos gesendeten Pakete entschlüsselt werden, weil nur damit K-B(B-TGS) und K-B(K-TGS(TGT)) in der Workstation zu entschlüsseln sind.
10. Wie können Nachrichten bei der Übertragung geschützt werden?
    - Private Key Verfahren: DES (symetrisches Verfahren)
      Data Encryption Standard (Details siehe unten)

- Public Key Verfahren: RSA (asymetrisches Verfahren)
  Rivest, Shamir, Adleman (MIT 1978)

11. Können die Schlüssel umgekehrt angewendet werden, zuerst Entschlüsselungs- und dann Verschlüsselungsschlüssel? Ja!
12. Wozu kann man das benutzen? Zum 'Unterschreiben' da nur der Absender den geheimen Schlüssel kennt. Absender A: $V_B(E_A(K),A)$ Empfänger B: $E_A(K)$,A sieht das die Nachricht von A ist und wendet den öffentlichen Schlüssel V_A darauf an: $V_A(E_A(K))=K$.
13. Wozu dient das Passwort
    Zur Authentifikation bzw. zur Generierung des User-Keys K-B.
14. Wie wird unter Athena auf Fileresourcen zugegriffen?
    Wenn ein Benutzer auf eine Resource zugreifen will benötigt er ein Ticket für den entsprechenden Server. Er schickt dem TGS dazu
    - Bitte um Ausstellung des entsprechenden Tickets.
    - Das mit dem Schlüssel von TGS verschlüsselte Ticket-Granting-Ticket K-TGS(TGT)
    - Den mit dem Session-Key B-TGS verschlüsselten Authentisierer B-TGS(Aut)

    Aut steht für Namen, Internet-Adresse des Benutzers sowie einen Zeitstempel. TGS entschlüsselt nun K-TGS(TGT) und findet hier im TGT die Identität des Benutzers und den Session-Key B-TGS den Kerberos beim Login generiert hat. Damit kann nun auch Aut entschlüsselt werden. Stimmen die beiden Benutzeridentitäten überein, so ist hierdurch der Benutzer authentisiert und erhält das gewünschte Ticket. Dieses enthält insbesondere den Schlüssel B-Server.
15. Was versteht man unter TGS und TGT?
    Ticket-Granting-Service und Ticket-Granting Ticket (=Ticket für den Dienst TGS)
16. Wie wird der Server gegenüber dem Client authentifiziert?
    Der Server F verschlüsselt mit Hilfe des Session-Keys B-F eine beiden Seiten bekannte Information z.B. einen Zeitstempel und schickt diese Information an den Client. Dieser kann mit Hilfe von B-F die Sendung entschlüsseln und die Information überprüfen. An der Übereinstimmung sieht der Client, daß F den Session Key B-F kennt und diese Kenntnis kann F nur aus dem Ticket haben das der Client ihm geschickt hat. Das Ticket war aber mit K-F verschlüsselt und K-F kann außer Kerberos nur der echte Server F kennen! Lücke: Erste Kommunikation des Benutzers mit Kerberos beim Login, Gefahr der Passwort-Ausspähung durch Dummy-Login-Programm->Gegenmaßnahme: Workstation neu booten.
17. Wie kann man vermeiden, daß Passwörter übers Netz verschickt werden?
    Mit Hilfe eines Sicherheitsservers wie z.B. dem Server Kerberos im System Athena. Das Passwort ist hierbei nur dem Kerberos und dem Benutzer bekannt. Mit Hilfe eines DES Verschlüsselungsverfahrens wird aus dem Passwort auf beiden Seiten ein Schlüssel berechnet der dann übers Netz verschickt wird. Nur wenn Kerberos und die Workstation an der die Anmeldung erfolgt, den gleichen Schlüssel berechnen, gelingt der Login, d.h. nur wenn beide Passworte gleich waren!
18. Beschreiben Sie DES!
    DES bedeutet Data-Encryption-Standard, es handelt sich um ein symetrisches Chiffrierverfahren.. Die Chiffrierung von 8Byte Klartext liefert 8Byte Geheimtext. Die Chiffrierung wird durch eine 56Bit langen Schlüssel

(key) parametrisiert. Dieser Schlüssel muß Sender und Empfänger bekannt sein. Die Chiffrierung verläuft folgendermaßen:

- Die 64 Bit (=8 Byte) des Eingabewortes werden einer (bijektiven) Permutation P unterworfen.
- Teilung des Eingabewortes in zwei gleichgroße Halbwörter L und R.
- Nun wird 16mal das alte rechte Teilwort zum neuen linken Teilwort während das neue rechte Teilwort durch R:=L⊕ f(R,K(key,i)), wobei K_i=K(key,i) ist (⊕ = EOR) aus den beiden alten Teilwörten und dem Schlüssel key bestimmt wird. Die eigentliche Verschlüsselungsfunktion f vervielfacht zunächst bestimmte Bits in R wodurch ein 48 Bit langes Wort R' entsteht. Dann wird R'⊕ K_i berechnet. Das resultierende 48 Bit Wort wird nun in 8 Teilworte zerlegt und durch jeweils eine S-Box in ein 4-Bit Ausgabe Wort umgewandelt. Das erste und das letzte der 6 Eingabe Bits entscheiden über die Substitution. Auf die zusammengesetzte Ausgabe aller S-Boxen (32 Bit) wird nochmals eine feste Permutation Q angewendet.
  Insgesamt ist DES:=P^{-1}• h_{16}• g• h_{15}• g• ....• h_2• g• h_1• P.
  mit h_i : (R,L) ® (R,LÅ f(R,K_i)), für i=1,...,16
  und g: (R,L) ® (L,R)
- Das chiffrierte Wort wird der Umkehrung der Permutation P, also P^{-1} unterworfen

Die Entschlüsselung erfolgt mit Hilfe des gleichen Algorithmus mit dem Unterschied daß diesmal die Schlüssel K_1...K_{16} in der umgekehrten Reihenfolge angewendet werden.

19. Beschreiben Sie RSA!
    Es handelt sich um ein Public-Key-Verfahren nach Diffie-Hellmann.
    - E• V(K)=K (also E(V(K))=K. Es gilt aber auch: V(E(K))=K! Dieser Sachverhalt wird zum elektronischen Unterschreiben genutzt.
    - V(K) kann effizient aus K berechnet werden.
    - E(G), mit G=V(K), kann ohne Kenntnis einer geheimen Zusatzinformation Z nicht effizient berechnet werden. (Die Geheiminformation Z ist φ (n). Und wegen φ (n)= φ (p*q)= φ (p)* φ (q)=(p-1)*(q-1) ist dies äguivalent zur Kenntnis der Faktorisierung p*q.
      Verfahrensklasse heißt "Einbahnfunktion mit Falltür" (wobei z die Falltür ist). Zur Vorgehensweise:
      Für die Verschlüsselung V wird eine Zahl n (Anm.: Für RSA werden Primzahlen mit > 155 Stellen =512 Bit empfohlen) die Produkt zweier Primzahlen≠ 2 ist und eine Zahl e, die zu φ (n) teilerfremd ist, gewählt. (φ (n) ist die Anzahl der zu n teilerfremden Zahlen). Diese beiden Zahlen sind öffentlich bekannt. Der zu verschlüsselnde Klartext wird auf einfache Weise in ein Folge von Ganzzahlen k_1, k_2,.. zwischen 0 und n-1 gebracht. Für jedes k_i berechnet man zur Verschlüsselung dann V(k_i) = k_i^e mod n.
      Für die Entschlüsselung bestimmt man ein multiplikatives Inverses von e mod φ (n), also eine Zahl d mit e× d=m× φ (n) +1 für eine geeignete Zahl m. Dann gilt für jede Geheimzahl g_i=V(k_i):
      E(g_i)= g_i^d mod n (Entschlüsselung)

$E(g_i) = k_i^{ed} \bmod n$
$E(g_i) = k\varphi^{(n)m+1} \bmod n$
$E(g_i) = k_i$

4. KE 4 - Verteilte Algorithmen
    1. Welche Eigenschaften haben Transaktionen?
        - Serialisierbarkeit
          Nebenläufige Transaktionen interferieren nicht.
        - Atomarität
          Eine Transaktion wird entweder ganz oder gar nicht ausgeführt.
        - Permanenz
          Das Ergebnis einer ordnungsgemäß ausgeführten Transaktion sind festgeschrieben.
    2. Wie wird mit Deadlocks in verteilten System umgegangen?
        - Vogel-Strauß-Algorithmus, d.h. man tut einfach nichts
        - Erkennung (Deadlocks werden also nicht ausgeschlossen, sondern erkannt und es wird versucht die Ursache zu beheben). Die Erkennung erfolgt z.B. durch den Aufbau und die Prüfung auf Zyklusfreiheit von Wartet-Auf-Graphen.
        - Vermeidung durch die Auswahl von Sperrstrategien bei denen konzeptionell keine Deadlocks auftreten können.(Welche?) Zeitstempelverfahren
        - Verhinderung durch sorgfältige protokollierte Betriebsmittelzuweisung an Prozesse. (Strategien zur Verhinderung von Deadlocks werden in vert. Systemen wegen der auftretenden Schwierigkeiten nicht benutzt, die Betriebsmittelanforderungen jedes Prozesses müßten im Voraus bekannt sein!)
    3. Logische und Pysikalische Uhren
       Logische Uhren sind systeminterne Zeitmesser, für die gilt, das alle Zeitmesser im System eine miteinander konsistente Systemzeit haben, d.h. auf die absolute Genauigkeit kommt es nicht an (wichtiger Algo zum Abgleich logischer Uhren ist der Algorithmus von Lamport).
        - Algorithmus von Lamport
            - falls a innerhalb desselben Prozesses vor b eintritt gilt C(a)<C(b)
            - falls a dem Senden und b dem Empfangen einer Nachricht entspricht, gilt C(a)<C(b)
            - für alle Ereignisse a und b gilt C(a)<>C(b)

          Im Gegensatz dazu sind physikalische Uhren um die absolute Genauigkeit (oder eine bekannte Ungenauigkeit) der von Ihnen gehalten Zeit bemüht.
            - Algorithmus von Cristian
              Zeit-Anfrage durch die Clients bei einem zentralen Zeit-Server mit zertifizierter Zeit in regelmäßigen Abständen d / 2r , wobei d der maximale Gangunterschied zwischen zwei Uhren ist und r die maximale Abweichung einer Uhr ist.
            - Berkley Algorithmus
              Der Zeit-Server (bzw. Daemon) fragt die Clients nach ihrer momentanen Zeit, bildet aus den Antworten den Durchschnitt und teilt den Clients mit, ob sie ihre Uhr

vorstellen oder verlangsamen müssen, um sich der Durchschnittszeit anzugleichen. (eignet sich für Systeme ohne zertifizierte Zeitquelle).
        - Durchschnittsbildende Algorithmen
          Jeder Rechner schickt allen anderen in festen Abständen 'seine' Zeit. Aus den empfangenen Zeiten wird - ev. unter Streichung des höchsten und des niedrigsten Wertes - der Duchschnitt gebildet und als eigene Zeit genommen.
    - Mehrere externe Zeitquellen
      Eigentlich kein Algorithmus, sondern der Versuch mit Hilfe verschiedener zertifizierter Zeitquellen das Zeitintervall in das die UTC gerade fällt, mit höchstmöglicher Genauigkeit zu bestimmen (für hohe Genauigkeitsansprüche).
4. Wechselseitiger Ausschluß
    - Zentraler Algorithmus
      Ausschluß wird mit Hilfe eines zentralen Koordinators erreicht (ähnlich einem Monitor). Nachteile: Koordinator kann ausfallen, Koordinator ist Engpaß.
    - Verteilter Algorithmus
      Hierbei handelt es sich um ein Zeitstempelverfahren. Die Anfrage zum Benutzen eines kritischen Abschnitts wird an alle anderen Proezesse geschickt. Falls ein Prozess diese Betriebsmittel gerade benutzt schweigt er. Falls er kein Interesse daran hat sendet er 'oK'. Falls er es auch benutzten will, entscheidet der Zeitstempel, wer die älteren Rechte hat gewinnt. Die Verlierer werden jeweils in die Warteschlange des Gewinners eingereiht und werden beim Verlassen des kritischen Abschnitts benachrichtigt.
    - Token-Ring-Algorithmus
      Nur wer im Besitz des zyklisch rotierenden Tokens ist, hat das Recht, einen kritschen Bereich zu betreten. Problem: Tokenverlust kann schlecht erkannt werden, 'Egoismus' eines Prozesses blockiert alle anderen.
5. Wellenalgorithmen
   Bei diesem verteilten Algorithmus geht es darum, daß ein Prozessor als Initiator allen anderen eine Nachricht schickt und um Bestätigung bittet, die Bestätigung kann aber jeder angesprochene Prozessor erst dann geben, wenn er von allen anderen (mit Ausnahme des Initiators) eine Bestätigung hat. Dieses allgemeine Problem tritt an vielen Stellen auf, so zum Beispiel bei der durchschnittbildenden Uhrensynchronisation, beim verteilten Algorithmus zum wechselseitigen Ausschluß und beim Bully-Algorithmus.
    - Echo-Algorithmus
      Hierbei sendet einer der Prozessoren als Initiator eine Nachricht mit der Bitte um Bestätigung an alle anderen Prozessoren die mit ihm direkt verbunden sind. Die übrigen Prozessoren senden diese Nachricht an alle ihre Nachbarn mit Ausnahme des Initiators bzw. desjenigen von dem sie die Nachricht zuerst empfangen haben ('Vater'). Wenn Sie von allen Nachbarn eine Bestätigung haben, senden sie ihrerseits eine Bestätigung an den Initiator/an den Vater. Dieser Algorithmus ist eng verwandt mit den Graphenbesuchsalgorithmen - es entsteht ein spannender Baum

des Netzwerks.
- Phasen-Algorithmus
  Dabei muß der Initiator nicht im voraus feststehen; vielmehr kann eine beliebige Gruppe von Prozessoren den Algorithmus spontan starten. Alle Prozessoren haben am Ende eine Entscheidung getroffen. Funktioniert auch für Netze mit gerichteten Kanälen, es muß nur jeder Prozessor von jedem anderen erreichbar sein. Der Durchmesser des Netzes muß jedem Prozessor bekannt sein. (Ein Netz hat den Durchmesser D, wenn jeder Prozessor p jeden Prozessor q über eine Strecke mit höchstens D Kanälen erreichen kann.
  Die Nachbarn eines Prozessors p werden in zwei Gruppen geteilt; Ein_p und Aus_p. Jeder Prozessor der ein erstes Token empfängt, schickt jedem Aus-Nachbarn ein Token. Danach wird erst wieder ein Token an alle Aus-Nachbarn geschickt, wenn von allen Ein-Nachbarn ein Token empfangen wurde. Das geht solange, bis jeder Prozessor jedem Aus-Nachbarn genau D Token geschickt hat.

6. Wahlalgorithmen zur Auswahl eines Koordinators
   - Bully-Algorithmus
     Falls kein Koordinator mehr antwortet, sendet ein Prozeß P an alle Prozesse mit höherer Prozess Nr. eine Nachricht, meldet sich niemand, dann gewinnt p und ist der neue Koordinator.
   - Ringalgorithmus
     Auswahl erfolgt nicht durch Broadcast, sondern durch zyklisches Versenden der Wahlbotschaft.

5. KE 5 - Prozesse und Prozessoren in verteilten Systemen
   1. Was versteht man unter einem leichtgewichtigen Prozess und was sind die Vorteile?
      Ein leichtgewichtiger Prozeß wird als Thread bezeichnet. Der Vorteil ist, das wenn ein Thread einen blockierenden Systemaufruf ausführt, nicht alle Threads des Prozesses (=des gemeinsamen Adressraumes) mit blockiert sind, sondern weiterarbeiten können. Zwei eigenständige Prozesse können dies u.U. nicht leisten, da sie keinen gemeinsamen Adressraum/Cache haben.
   2. Was ist der Unterschied zwischen einem Prozeß mit mehreren Threads und mehreren Prozessen mit nur einem Thread?
      Ein Prozess mit mehreren Threads besitzt in einem Adressraum mehrere Kontrollflüsse und für jeden Thread einen Stack. Bei einem Prozeß mit nur einem Thread gibt es auch nur einen Stack und einen Programmzeiger.
   3. Organisationsmodelle für Prozessoren in verteilten Systemen
      Workstation- und Prozessor-Pool.
   4. Algorithmen zur Prozessorzuteilung
      Bei einer Menge von Prozessoren wird ein Algorithmus benötigt, der Prozesse an Prozessoren bindet. Diese Algorithmen können deterministisch oder heuristisch, zentral oder verteilt, optimal oder suboptimal und sender- oder empfänger-initiiert sein. Z.B.:
      - Graphentheoretischer, deterministischer Algorithmus
        Prozessor- und Speicheranforderungen müssen bekannt sein. Das System wird dann als gewichteter Graph dargestellt, in dem in jedem Knoten ein Prozeß steht und jede Kante einen Nachrichtenfluss zwischen zwei Prozessen symbolisiert. Die

Kanten sind mit den Kommunikationskosten zwischen den Prozessen gewichtet. Der Graph wird dann so in prozessorbezogene Teilgraphen zerlegt das die Kommunikationskosten zwischen den Teilgraphen minimal werden.

- Up-Down-Algorithmus (Zentraler A.)
  Dieser Algorithmus arbeitet mit einem zentralen Koordinator. Dieser verwaltet eine Benutzungstabelle, die für jeden Rechner ein Punktekonto enthält das mit Null initialisiert wird. Läßt ein Rechner einen Prozeß Remote laufen, dann werden dafür pro Zeiteinheit eine gewisse Anzahl von Strafpunkten auf seinem Konto aufsummiert. Führt ein Rechner Prozesse für andere aus dann erhält er Punkte und kann dabei auch positive Punktestände erreichen. Nach Ausführung des oder der Remote-Prozesse wird der Zählerstand wieder sukkzessive mit jedem Zeittakt abgebaut. Vorteil: Stellt jedem Benutzer einen fairen Anteil an der Gesamt-Rechenleistung zur Verfügung.
- Hierarchischer Algorithmus (MICROS)
  Dabei wird der Anzatz des Up-Down-Algorithmus (Koordinator) übernommen mit dem Unterschied, daß ein Koordinator immer nur für eine begrenzte Anzahl von Rechnern zuständig ist. Dies führt zu einer baumartigen Hierarchie mit 'Managern' in den Ästen und Arbeitern in den Blättern dieser Baumstruktur.
- Verteilter, heuristischer Algorithmus nach Eager, Lazowska, Zahorjan
  Der Rechner auf dem ein neuer Prozess erzeugt werden soll befragt einen zufällig ausgewählten anderen Rechner ob die Last dort unterhalb einer bestimmten Schwelle liegt. Falls ja, wird der Prozeß dorthin verschoben andernfalls werden weitere Rechner befragt. Kann kein 'Dummer' gefunden werden, wird der Prozeß lokal ausgeführt.
- Auktionsalgorithmus
  Dies ist eine ökonomisch orientierte Strategie. Jeder Prozessor bietet beispielsweise in einer öffentlichen Datei seine 'Preise' für einen bestimmten Dienst an. Aufgrund des Angebotes geben die Rechner die einen Dienst in Anspruch nehmen möchten Gebote ab, die über oder unter dem Angebotspreis liegen können. Die Prozessoren sammeln die Gebote, wählen das höchste Gebot aus und bringen diesen Prozeß zur Ausführung. Der Angebotspreis wird durch den zugeschlagenen Preis ersetzt.

6. KE 6 - Verteilte Dateisysteme
   1. Dateien und Verzeichnisse in verteilten Dateisystemen
      Zwei Typen von Dateidiensten werden unterschieden:
      - Auf- und Ablademodell (Upload/Download)
        Es wird immer die gesamte Datei vom Server zu Client und umgekehrt übertragen. Es gibt nur zwei Operation: Upload und Download. Alle weiteren Datei-Operationen werden mit Hilfe lokaler Funktionen vorgenommen.
      - Modell des entfernten Zugriffs (Remote Access)
        Hierbei bietet der Dateidienst die ganze Palette von Zugriffsoperationen die auch von Einprozessorsystemen geboten werden an.

Der Verzeichnisdienst stellt Operationen zum Erzeugen und Löschen von Verzeichnissen, zum Benennen und Umbenennen und zum Verschieben von Dateien und Verzeichnissen bereit. Der Verzeichnisdienst definiert ein Alphabet und eine Syntax zur Komposition von Datei und Verzeichnisnamen.
In Verteilten System gibt es drei Methoden des Benennens von Dateien und Verzeichnissen:

- Rechner- und Pfadbenennung (z.B UNIX remsh oder rcp Kommando)
- Mounten auf lokale Dateihierarchie (z.B. NFS)
- Ein aus der Sicht aller Rechner gemeinsamer Namensraum.

2. Semantiken der gemeinsamen Nutzung von Dateien
   - UNIX-Semantik
     Jede Operation auf einer Datei ist unmittelbar sichtbar für alle Prozesse.
   - Sitzungssemantik
     Die Änderungen sind für andere Prozesse solange unsichtbar, bis die Datei geschlossen wird.
   - Unveränderliche Dateien
     Veränderungen sind nicht möglich; gemeinsame Benutzung und Replikation werden dadurch vereinfacht.
   - Transaktionen
     Alle Änderungen haben die Alles-oder-Nichts-Eigenschaft
3. Mögliche Probleme bei der Zwischenspeicherung von Dateien
   Cache Inkonsistenzen sind möglich beim Lesen und nachfolgendem Modifizieren durch zwei Klienten gleichzeitig. Abhilfen:
   - Write-through Cache
     Funktioniert, beeinflußt aber das Schreibaufkommen nicht mehr positiv (Bei jeder Änderung ein Schreibzugriff)
   - Verzögertes Schreiben
     Höhere Leistungsfähigkeit, aber zwischen den Schreibvorgängen sind Mehrdeutigkeiten möglich
   - Schreiben beim Schließen
     Ist im wesentlichen die Sitzungssemantik
   - Zentrale Kontrolle
     Besitzt die UNIX-Semantik ist aber weder robust noch leicht skalierbar.
4. Replikation von Dateien
   Gründe für die Replikation:
   - Erhöhung der Zuverlässigkeit durch Redundanz.
   - Gestatten von Dateizugriffen wenn einer der Dateiserver nicht verfügbar ist.
   - Verteilen der Arbeitslast auf mehrere Server.

   Protokolle zur Replikation:
   - Replikation der ersten Kopie
     Eine Kopie ist die Master-Kopie, alle anderen werden nachgezogen.
   - Abstimmung (voting)
     Hierbei wird jeweils ein Lese- und Schreibquorum von allen Servern erhoben, bevor eine replizierte Datei gelesen oder

geschrieben werden darf. Nur wenn entsprechende Mehrheiten zustanden kommen wird die Erlaubnis gewährt. Dies ist insbesondere beim Schreiben einer replizierten Datei wichtig, weiß man doch nicht ob diese Datei die jüngste Version ist.

- Abstimmung mit Schatten
  Variante des vorhergehenden die sicherstellt das auch nach Ausfall eines Teils der Server immer noch ein Schreib-Quorum zustande kommt, indem Schattenserver, die sich nur an den Abstimmungen beteiligen aber keine eigenen Dateien besitzen, erzeugt werden.

5. Beispiel für ein verteiltes Dateisystem: Das Andrew-Dateisystem (AFS)
6. Entwicklungstendenzen