Internet Engineering

241-461

Robert Elz

kre@munnari.OZ.AU
kre@fivedots.coe.psu.ac.th

http://fivedots.coe.psu.ac.th/~kre

Security Issues

- Why we need Security
- What is required
- Mechanisms for digital security
- Encryption/Decryption and algorithms
- Symmetric (shared) key vs Public Key
- Use of Security for
  - The Web (HTTP)
  - E-Mail

Why Security?

- Can use web to look for information
  - Information available to everyone
  - Nothing secret
  - No real security issues
    - just protect data from modification
- Can also use web to buy things
  - When buying, must pay
  - How?
    - Credit card is common solution
    - Supply number to web server (product supplier)
    - Supplier sends invoice to credit card company
    - They pay supplier, bill you
- Other methods exist
  - All require buyer to send something to server
  - Something server can use to obtain payment

Why Security? (2)

- The critical step
  - The only network step
  - I send credit card number to web server
- What if someone is watching the packets?
  - They see credit card numbers
    - Easy - credit card numbers are easy to recognise
  - They collect the numbers
    - Use them to buy whatever they want
    - The bill comes to me!
  - Not Nice!!
- Need protection
  - Prevent packets being observed
  - That is, secrecy
    - One of the types of security

Confidentiality

- Confidentiality
    - Privacy
    - Secrecy
  - Not a new requirement with networks
  - Spies have needed this forever
    - Send secret message back to headquarters
  - Spy & HQ have code book
    - Spy encodes message
    - Transmits encoded message
    - HQ receives message & decodes it
  - Anyone else receiving message
    - It is just nonsense (need code book to decode)
  - Same basic method used today
    - Details differ (No code books any more)
    - Both systems have a secret
      - usually called key
    - Use key to encrypt message before sending
    - And to decrypt messages received

Other Security Requirements

- Confidentiality easy to understand
  - some data must remain secret
- But not the only issue
- Integrity
  - Protection from modification
  - Even if data not secret
- Authentication
  - Knowing identity of other end-point
    - (host / user / organisation ...)
- Non-repudiation
  - Other party cannot deny transaction later
  - Aspect of authentication

Encryption and Requirements

- Using encryption
  - Confidentiality
    - obvious - only meaningless data visible
  - Integrity
    - if data modified
      - will decrypt to junk
      - attacker does not know key
        - cannot create correct encoding
  - Authentication
    - Message correctly encrptyed
    - Sender knew key
    - Only you and I know key
    - I did not send the message
    - So I know it must have really come from you
  - Non-repudiation
    - A problem ...
    - I know message came from you
      - But I cannot prove it
    - I might have sent the message

Security Algorithms

- Many algorithms exist to encrypt data
  - Most have the properties
    - secret shared key exists
      - shared ==> both ends know it
    - algorithm uses data & key
      - produces scrambled data (encrypted)
    - algorithm given encrypted data & key
      - produces original data
- Terminology
  - plaintext The un-encrypted data
  - ciphertext The encrypted data
- Properties of algorithms
    - Shared (or Symmetric) key algorithms
  - Very hard to go from ciphertext to plaintext
    - without having key
  - Extremely hard to discover key
    - even with both plaintext & ciphertext
    - even with chosen plaintext & ciphertext

Some Symmetric Key Algorithms

- DES
  - Data Encryption Standard
    - US Govt standard - in the past
    - 56 bit key
    - Considered too weak by today's standards
- 3DES
  - Run DES algorithm (approximately) 3 times
    - Forwards, backwards, forwards again
    - Using different keys
- AES
  - Advanced Encryption Standard
    - 128 bit keys (or more)
    - Much more secure
- Blowfish
    - 40 to 448 bit key
- Cast128
    - 40 to 128 bit key

Web Security (Reprise)

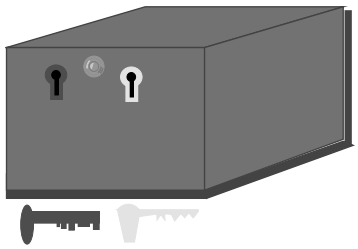
- Now we have method to send secret data
  - We can use that to send credit card numbers
  - And any other secret data
- The problem:
  - How do we create the key?
    - Easy: someone picks a big random number
  - How to we get key to our partner?
    - Cannot just send over net
    - It must be kept secret
    - We cannot use encryption...
      - Or not this kind anyway
- Key Exchange Protocols
  - Protocols exist for key exchange
  - That is
    - exchanging keys securely
  - Not covered in this course

Web Security (Reprise 2)

- Assume we have key distribution method
  - A secure key distribution method
- Is our web transaction now secure?
- How do we know who we are talking to?
- We can send data securely to partner
  - But partner can read it
    - Obviously!
  - What if partner is not who we thought?
  - The bad guy gets to read our secret data!
- Need a way to prove identity
  - Digital Certificates
    - Signed digital certificates

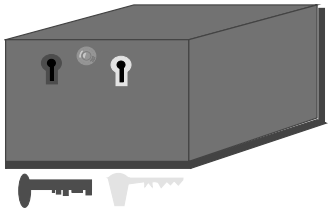
Digital Signatures

- First, a digression ...
- Suppose we had a very strong box
  - Can only be opened with a key



- The box has 2 key holes
  - And a coloured indicator (light)
- There is one key for each hole
  - Keys only fit in the correct key hole

The strong box



- When the indicator is green
  - The box is unlocked - It can be opened easily
  - The red key will lock the box and turn the indicator yellow
  - The yellow key will lock the box and turn the indicator red
- When the indicator is not green
  - The box is locked
  - Only the key shown by the indicator works
    - That key unlocks the box
    - Turns the indicator green again
- Box locked with one key
  - unlocked by the other

Using the box

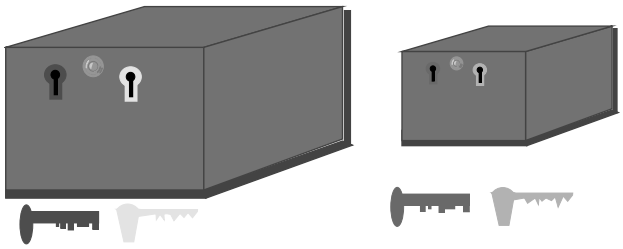
- If we make many boxes
    - All identical
  - And make many yellow keys
    - All identical
  - Only one red key
- Give a box to anyone
  - With a yellow key to lock it
- Keep the red key
  - No-one else ever sees it
- Then anyone can put things in their box
  - And lock the box with their yellow key
  - The indicator turns red
    - Only the red key opens the box
- People can send things
  - No-one else is able to look at
    - Except the owner of the red key
  - Others cannot open the box

Using the box (2)

- If we make a box (or several)
  - And make many yellow keys
  - Only one red key
- Give a yellow key to anyone
- Keep the red key
  - No-one else ever sees it
- Owner of red key
  - puts things in box
  - locks box with red key
    - indicator turns yellow
    - only yellow key can open it
  - There are many yellow keys
    - Anyone can unlock box remove contents
  - But
    - When they do, they know who locked it
    - That is, they know who sent the box
      - The owner of the one red key

Using the box (3)

- Build another similar box
  - make it smaller with different keys



- Many copies of blue key
  - Only one copy of purple key
- Owner of small purple key
  - puts stuff inside small box
  - locks box with purple key
    - indicator goes blue
  - puts small box inside big box
  - locks big box with yellow key

Using the box (4)

- Only the owner of red key
  - Can unlock the big (outer) box
- Contents delivered securely to Red-Owner
- When Red-Owner opens box
  - Small box found inside
  - Indicator is blue
  - Blue key used to open small box
    - Must have been locked by purple key
- So, contents delivered securely
  - and origin of contents known for sure
- Terminology
  - Yellow (and Blue) Key
    - Public Key
      - Given to anyone at all
  - Red (and Purple) Key
    - Private Key
      - Kept secret to owner

Public Key Encryption

- Can do the same thing
  - Using encryption algorithms (mathematics)
- Have algorithm with two keys
  - Either key used to encrypt data
  - The other key must be used to decrypt
- Give one key away to everyone
    - The Public Key
- Keep the other key secret
    - The Private Key
- Terminology
  - data encrypted by private key
    - Signed
  - Anyone can decrypt it using the public key
    - So no secrecy from this encryption
  - When they decrypt
    - They know for certain who encrypted it
  - That is the same as a signature on a document

Use of Public/Private Keys

- Confidentiality
  - Encrypt using recipient's public key
- Integrity
  - Data not visible
    - Hard to modify
    - Easy to replace completely
      - public key known to all
- Authentication
  - Add signature
      - Encrypt with sender's private key
    - guarantees identity of sender
      - Only person who has that key
    - Also guarantees integrity
      - if data replaced, signature lost
- Non-repudiation
  - Only sender could have signed it
    - Even recipient does not know sender's key
  - So, if signed
    - Must have been sent by the sender

The remaining problem

- Who was the sender?
  - Data signed
    - So we know owner of private key sent it
    - But who is that?
  - How do we know who owns the private key
    - We have public key
    - Public key says
      - Aj Robert's Public key
    - How do you know that is correct?
- We need proof of identity somehow

Certificates

- Organisation (or person) creates public/private keys
- Takes public key to trusted organisation
  - Perhaps government department
  - Or another trustworthy organisation
    - Call this organisation Certificate Authority
- Proves identity to Certificate Authority
  - and proves ownership of public key
    - by using private key to encrypt some data
- Certificate Authority verifies all data
  - Then signs the public key
    - Plus owner's identity information
  - That is,
    - encrypts it all with CA's private key
- This is known as a certificate
  - Organisation takes away the signed certificate
    - Gives copies of this away instead of just public key

Certificates (2)

- CA's public key known to all
  - Widely distributed
  - If fake version appears
    - someone will recognise it is incorrect
    - advise everyone to be careful
- When anyone gets a certificate
  - They decrypt using CA's public key
  - If valid
    - That is: If decryption succeeds
  - Now know identity of owner
  - And owner's public key
- When we have a certificate
  - We can send secret message to its owner
  - We are certain who that is
    - Only that owner can read our message
    - We encrypt using their public key
  - We can authenticate their signature
    - We decrypt using their public key

Public Key Encryption

- Two common algorithms
  - RSA
    - Rivest Shamir Adleman
  - Diffie-Hellman
- RSA most common
  - very slow to operate with private key
  - average speed operating with public key
  - slow to create key pair
- D-H
  - slow to encrypt/decrypt data with either key
  - very slow to create key pair
- Major Problem
  - Both algorithms are very slow
    - We can encrypt only small amounts of data
      - Too much takes too long

Another Digression

- We have
  - Shared Key encryption
    - fast but needs shared secret
  - Public Key Encryption
    - slow but no shared secrets
- There is also
  - Cryptographic Hash Function
- Hash function
  - Take data (perhaps much data)
    - generate much smaller result
    - where result depends upon input
      - different input, different output
      - though many differnt inputs produce same output
  - Hash functions have many uses

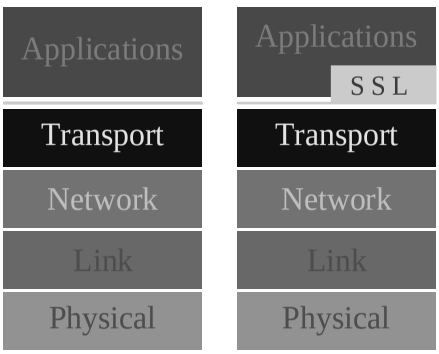
Cryptographic Hash

- Hash function with extra properties
  - Very difficult to find input
    - to generate a desired hash result
  - Very difficuly to modity input
    - and leave hash result unchanged
- Digital Signatures
  - Encrypt data using private key (very slow)
    - Too costly if data is large
    - Any data not included
      - easily changed by attacker
- Solution
  - Hash all data using crypto hash (very fast)
  - Encrypt hash result using private key (small result)
  - Attach encrypted hash result as digital signature
- Verification
  - Detach digital signature (decrypt and save)
  - Hash data - Compare result with that from signature
  - If the same: Signature verified, data unaltered

Back to the Web

- Secure Web pages
  - Page accompanied by a certificate
  - Certificate guarantees identity of owner
    - Page itself can be signed
- To send secure data
  - Generate key for a symmetric key protocol
  - Encrypt that using Public Key encryption
    - Using public key from certificate
  - Encrypt other data using symmetric key protocol
- Owner of certificate
    - The organisation we want to send to
  - Has private key
  - Can decrypt the key for symmetric protocol
  - Can then decrypt the other data
- Others
  - Do not have the private key
  - Cannot decrypt the message

Secure Sockets Layer



- Traditional Internet layers
- Secure Sockets Layer

SSL

  - Extends Sockets Interface
  - Implemented in Applications
    - As library routines
    - Available to any application

Network Security

- SSL is implemented at transport layer
  - or just above it
  - Also known as Transport Layer Security
- Can also have security at network layer
  - IP Security
  - Node to Node security

IPsec

- IPsec Protects all communications between two nodes
  - All transport protocols
    - Including ICMP
  - Hides transport headers
    - Port numbers protected
  - But:
    - Requires implementation in network stack
    - Key distribution is harder
      - Used much less often

Security Mechanisms

- IPsec
  - Network Level Security
  - Node to Node
- TLS
  - Transport Level Security
  - Process to Process
- For E-Mail
  - Which is appropriate?
  - Is either appropriate?
- NO
  - Want to know
    - message is from who it says
    - message is unmodified
    - message is private (perhaps)
  - Want user to user security

E-Mail Security

- IPsec for e-mail
  - Protects mail between MTA & next MTA
    - or MTA to/from MUA
  - Content of mail might be anything
    - No guarantees at all
  - Content is what is important
- TLS (SSL)
  - Almost the same as IPsec
    - But from MTA to MTA
    - rather than MTA's host to MTA's host
- Neither knows about people
  - e-mail is all about people
  - Is From: header address
    - really the authorising entity

Security Model for e-mail

- Most server security
  - client application must verify server
  - Make sure server client connected to
    - is the server it claims to be
  - Done using certificate
    - Owner of certificate has Private Key
    - Can prove ownership
    - Certificate contains identity
  - Client identity largely irrelevant
    - Server serves everyone
- Useless for e-mail
- For e-mail
  - client identity most important
    - Actual human user identity
  - Server largely irrelevant

Security Model for e-mail (2)

- Proof of identity
  - Certificates handle this
  - But "real" certificates cost a lot
    - perhaps 100,000 THB
    - every year or two
- How many people in the room have a certificate?
  - How many will ever have one?
- Costly part of certificate
  - The validation
  - Signature of Certificate Authority
    - They must investigate first
- Self-signed certificates
  - I say: "Yes, it really is me"
- Can we trust that?

Trust Model for e-mail

- No, self-signed certificate not much better
  - than the From: header address
- Sender creates both
  - Either might be fake
- But
  - Certificate can be widely published
  - Claimed to belong to me
  - If fake
    - hope someone will notice
    - expose the forgery
  - Better than nothing
- Better still
  - If I know you
  - You tell me that is your certificate
    - I can sign it for you
  - You tell other friends
    - They also sign your certificate

Trust Model for e-mail (2)

- Recipient receives mail
  - Signed by the sender
- Recipient looks up certificate
  - Sender's certificate
- Obtains Public Key
  - Can verify signature
- Now we know
  - Message was signed by certificate owner
- But, who is that?
- If sender's certificate is signed by me
  - Then I know it is correct
  - I checked before I signed it
- If sender's certificate is signed by you
  - And I trust you
  - Then I can also believe certificate
    - To same extent I trust you

Trust Model for e-mail (3)

- Can extend this
  - Certificate claims to be owner by A
    - Signed by B
  - B's certificate
    - Proves B is who he says
    - Allows us to validate signature
      - on A's certificate
    - Signed by C
      - C promises B is really B
      - And this is B's certificate
  - C's certificate
    - Same as B's Signed by D ...
- Eventually some certificate is signed by me
  - That one I certainly trust
  - Allows all certificates in chain
    - to be trusted
      - Including A's eventually
- We now know mail is really from A

E-Mail security encoding

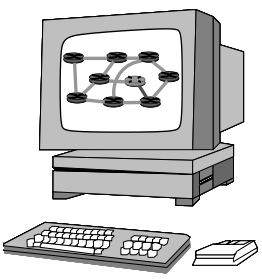
- SMIME
  - Secure MIME
- PGP
  - Pretty Good Privacy
    - (Pretty here means "Fairly" or "Quite")
- Message is signed
  - Signature appended
  - Private encapsulation method
    - PGP pre-dates use of MIME
- Signature covers hash of message content
  - stops cutting signature from one message
  - pasting it into another
- Message is optionally encrypted
  - Encrypted using recipient's public key
  - Only recipient can decrypt it

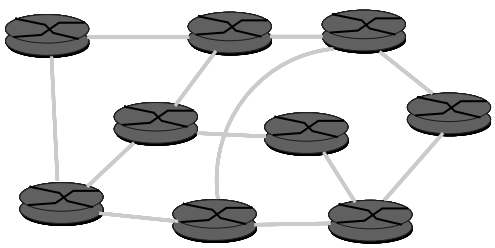
Contents

Network Management

- Large Network
  - Many Routers
  - Many Links
- Is it all working?
  - How do we know?
  - Wait for user to complain?
- Not good enough
  - Need to monitor operations
  - Need to correct problems
    - When problems detected
  - Even better
    - Predict problems in advance
    - Fix before they occur
- This is Network Management
  - Sometimes includes Network Configuration

Network Management





- Management Station
  - Display Network Information
    - Graphical or otherwise
  - Indicate Problems
    - Colour, Flashing, Pop-Up, ...
  - Allow investigation

Network Problems

- Problems to watch for
  - Crashed Routers
  - Faulty Links
    - Link Broken
    - High error rate
  - Routing Problems
  - Network Congestion
  - (and more)
- Data collection
  - In-Band
    - Uses network
      - Good, provided network works
      - Cheap
  - Out-of-Band
    - Uses other links
    - Dial up
      - Expensive
      - Works with broken net

Contents

- Intro to Network Management
- Collecting Statistics
- Management Information Base
- Simple Network Management Protocol
- Host Configuration
- DHCP
- IPv6

Predicting Faults

- Can
  - Wait for problem
  - Fix
  - Repeat
- Not good
  - Between problem & fix
    - network not operating
- Better
  - Watch for coming problem
    - Error rate increasing
    - Router approaching 100% CPU
    - Link usage approaching capacity
  - Correct before problem serious
    - Fix link errors
    - Add capacity

Network Capacity Monitoring

- When
  - Usage of link increasing
  - Seems to need more bandwidth
    - Expensive
- But, do we really?
  - Is it end of semester traffic
    - Does increase happen each year?
  - What is causing increase?
    - Maybe reduce demand
    - instead of increasing capacity
- To answer
  - Need to know
    - What kind of traffic
    - From where to where
    - Historical usage patterns
- Network Monitoring

Contents

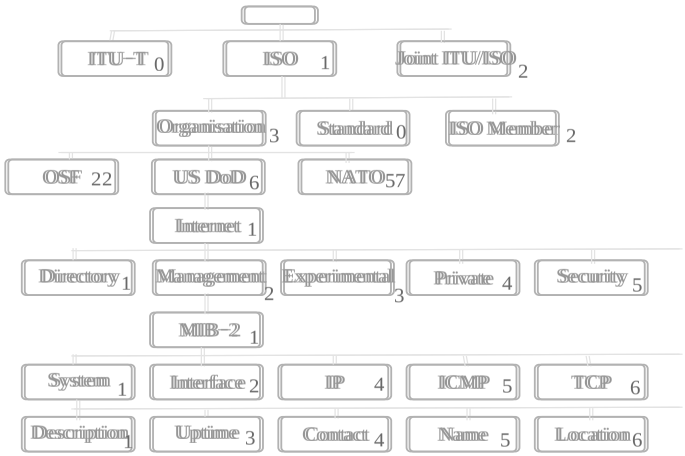
Collecting Data

- Management Information
  - Collected in Database
- Data in Database: Management Information
  - Management Information Base
- Tree Structured Database
  - Data at leaves of tree
  - Branches group related information
  - Higher levels (towards root)
    - Definition authority
- All information is named
  - Necessary to fetch over network
- All information is typed
  - So know what data represents
    - Numbers, Strings, Status. ...

Standard MIB

- Tree similar to DNS tree
  - Nodes represent management information
  - Written with root at left
- Root at top of tree
  - Major organisational nodes under root
  - Delegations down tree
    - to organisations that define data
    - Standards Bodies
      - IETF, ISO
    - Companies
      - For their equipment
- Naming uses numbers
  - Primary name source
    - No arguments
    - No meaning
- String names added
  - User-Friendly naming

Standard MIB (2)



- MIB Variables
  - .1.3.6.1.2.1.1.5

Object

Ide

  - .iso.org.dod.internet.mgmt.mib2.system.sysName

MIB Variables

- Simple Variables
  - sysName
  - sysUptime
- Tables
  - Multiple copies of variable
  - One for each X
    - For some X
  - eg:
    - Interface type
    - One for each interface

	1 Index	2 Descr	3 Type	4 Mtu	5 Speed	6 PhysAddress	7 AdminStatus	...
1	1	Text	Ether	1500			up	
2	2	Text	Loop	30000			up	
3	3	Text	FDDI	8400			down	
4	4	Text	Token Ring	1500			up	
5	5	Text	PPP	1500			testing	

MIB Variables (2)

- Variable Name
  - Object Identifier + Instance Identifier
- Simple Variables
  - Instance Identifier is "0"
- Tables
  - Instance Identifier is table index
    - Can be several elements
- Eg:
  - sysName .1.3.6.1.2.1.1.5.0
  - ifNumber .1.3.6.1.2.1.2.1.0
  - ifType .1.3.6.1.2.1.2.2.1.3.3
  - ifDescr .1.3.6.1.2.1.2.2.1.2.5

Contents

SNMP

- Simple Network Management Protocol
  - Not very simple
    - Never mind...
- Manager (client)
  - The Management Station
    - And usually the human
- Agent (server)
  - System that can access a MIB
    - MIB exists locally
- Manager uses SNMP to access agent
  - Fetch Variable(s)
  - Set Variable(s)

SNMP Operations

- 3 major operations
  - more exist, used less
  - GET
    - Send Instance Name of Variable
    - Receive Name & Value of Variable
      - Operates on multiple variables
      - As many as fit in packet
  - SET
    - Send Instance Name of Variable
    - Send Desired Value for Variable
      - Agent alters variable
      - if permitted
    - Receive new value of Variable
      - Altered if successful
  - GETNEXT
    - Send Object Identifier
      - Any Object Identifier
    - Receive
      - Name of next variable instance
      - Value of that variable

Powerful GetNext

- SNMP people always say
  - The powerful GetNext operator
  - When referring to GETNEXT
- Very useful operator
  - Usually used as:
    - Start with known Object Identifier
      - Make first request
    - Use ObjectId from result
      - As ObjectId for next GETNEXT request
- eg: start with
    - ifType .1.3.6.1.2.1.2.2.1.3
  - receive
    - ifType .1.3.6.1.2.1.2.2.1.3.1
  - then
    - ifType .1.3.6.1.2.1.2.2.1.3.2
    - ifType .1.3.6.1.2.1.2.2.1.3.3
  - later
    - ifMtu .1.3.6.1.2.1.2.2.1.4.1

MIB Enumeration

- Agent not required
  - to implement full MIB
  - Just parts meaningful to Agent
    - In groups
      - If part of group implemented
      - All must be
      - cannot just ignore "hard" variables
- Problem for Manager
  - How to know what parts of MIB supported?
- Answer
  - The powerful GetNext operator
- Start with .1, Use GETNEXT
    - Obtain first implemented variable
- Keep using GETNEXT
  - With variable from result
  - Until Name Error result is returned
- Manager now has complete list of supported variables

SNMP Operations

- GET
  - access to (possibly dynamic) memory
- SET
  - modifications to memory
- Agent implements MIB
  - Whatever the MIB wants to happen
  - Agent can make occur
- "Network" management can do anything
  - that computers are able to achieve
- MIB definitions exist
  - for all kinds of exotic tasks
  - standardises & proprietry

Contents

- Intro to Network Management
- Collecting Statistics
- Management Information Base
- Simple Network Management Protocol
- Host Configuration
- DHCP
- IPv6

Host Configuration

- Minimum IP knowledge for host
  - IP address
  - Address of router on link
  - Usually
    - Subnet mask (Prefix length)
- To be useful
  - DNS resolver back end (cache) address
- IP address
  - must be correct for link
  - must not be duplicated
    - HOW?
- Router address
  - must be on same link
  - must be address of router
    - HOW?
- Original IP hosts
  - human configuration

Manual (human) Configuration

- Errors Likely
  - Most people do not know rules
  - Or simply make mistake
- In the past
  - dedicated staff
  - network specialist
    - for each computer
  - Manual configuration acceptable
    - though not desirable
- More recently
  - more computers than people
  - most users know little
    - simply expect it to work
- Need to provide automated mechanism

Early auto-configuration

- BOOTP
  - Bootstrap Protocol
- Needed by systems
  - with no human interface
  - no way to be configured
    - before booting
- Simple protocol
  - Broadcast on LAN
    - Send me my address
  - Server on LAN replies
    - Address is ...
    - Router is ...
    - Bootstrap server is ...
    - Boot File name is ...
  - Server uses MAC address of client
    - Allows it to send suitable answer
  - Answer broadcast
    - client has no address
    - cannot receive packets normally

BOOTP -> DHCP

- BOOTP too limited
  - Different systems need different data
    - netmask
    - DNS cache address
    - hostname
    - (much more possible)
- DHCP created
  - Based upon BOOTP
  - Spare space in BOOTP packet
    - used for options
  - One option is DHCP command
    - Needed for dynamic address assignment
- BOOTP
  - mac address -> IP address
    - configured for BOOTP server
  - multiple servers possible
    - all use same config file
    - all give the same answers

DHCP

- DHCP
  - mac address -> IP address
    - dynamically picked by server
  - multiple servers
    - different addresses
    - need to handle this
- Solution
  - 4 message exchange
    - Who can supply address?
      - message broadcast to everyone
    - I can: A available I can: B I can: C
      - message can be broadcast or unicast
      - unicast requires special care
    - Give me address A please
      - message broadcast to everyone
      - so servers that offered B and C see allocation
    - OK A assigned
      - message can be broadcast or unicast
      - green and purple take back offers

Re-using Addreses

- Static addresses
  - Allocated permanently
    - configured in file
    - available to system forever
- More recently
  - address shortage
  - need to conserve addresses
  - don't allocate to vanished systems
  - need to reclaim addresses
    - when no longer in use
- DHCP adds
  - Release operation
    - return address to server
  - Few clients use it
    - Must know about to be disconnected
    - often network removed first
      - or system crashes

DHCP Leases

- DHCP also adds
  - Address Lease time
    - Allocation only valid for N seconds
    - But can be renewed
      - before N seconds have passed
- Address renew
  - Give me address A please
    - sent via unicast to allocating server
  - OK A assigned
  - If address re-assignment fails
    - try again
  - If still fails
    - return to broadcast
    - search for a server again
      - repeat entire procedure
  - All done before old address lease expires
    - If lease does expire
    - Must stop using address

IPv6

- DHCP requires servers
  - servers must be configured
  - must co-operate if more than one
- Better than configuring every host
  - But not good
- IPv6 is new
  - Decided to do better
- IPv6 allows hosts to assign addresses
  - assign address to itself
- Needs network prefix
  - can be supplied by router
  - There must be a router
    - or network prefix not required

IPv6 Autoconfiguration

- Router sends Router Advertisement
  - Periodically
  - or when requested by host
    - Host can send Router Solictiation
- RA gives
  - prefix (network number)
    - validity timers
  - router address
    - if sender of RA is router
  - other information for hosts
- Host
  - takes prefix from RA (64 bit prefix)
  - MAC address
    - from LAN card (expanded to 64 bits)
  - combines using a standard rule
  - checks for duplicate address on LAN
  - assigns address to itself

IPv6 Autoconfiguration (2)

- RA repeats
  - Allows
    - updates to configurations
      - new prefix added
      - old prefix removed

network renumbered!

    - verification router remains working
    - address lifetimes to be extended
- DHCPv6 exists
  - Not needed as much
  - Can be used for address allocation
    - For closely managed networks
  - More likely used
    - to distribute other configuration information