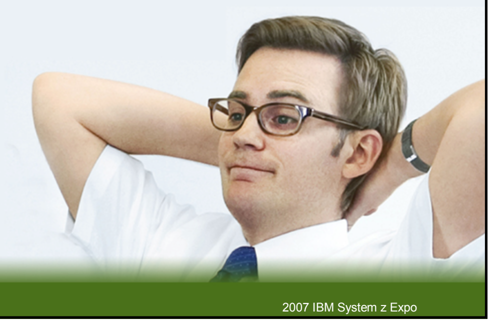
E57

Securing FTP on VSE

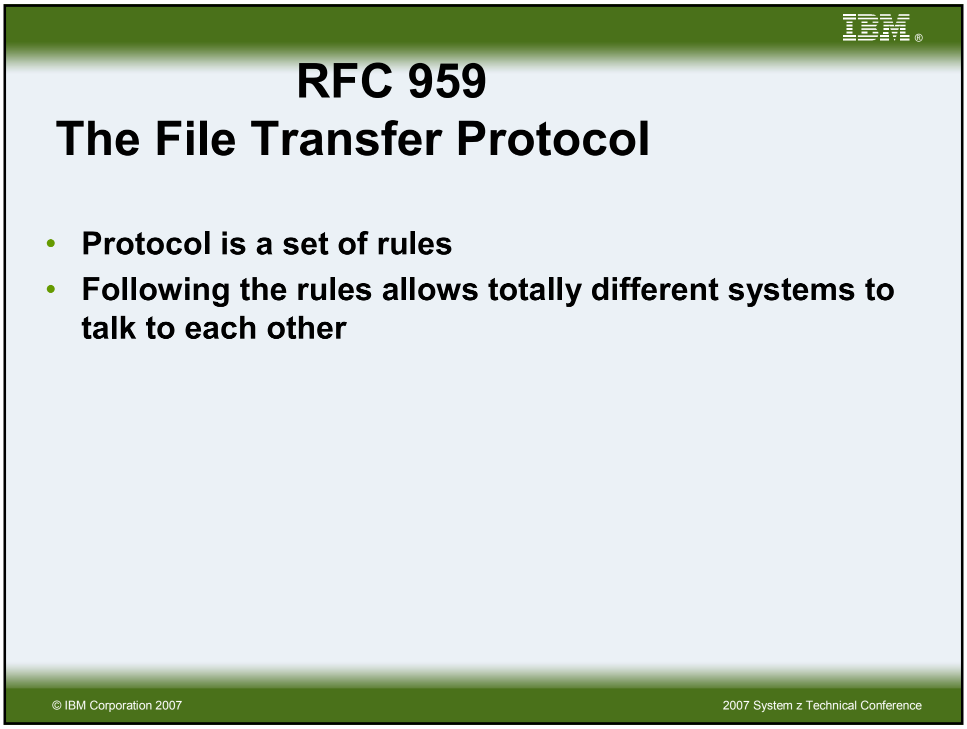
Don Stoever – don@e-vse.com



IBM System z Expo
September 17-21, 2007
San Antonio, TX

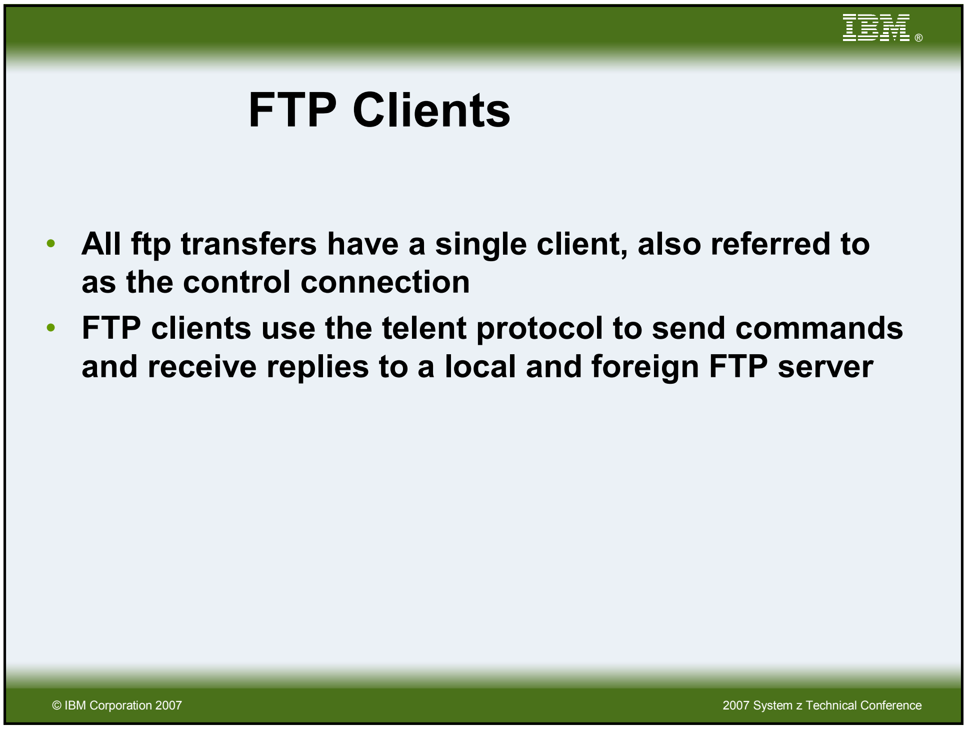
RFC 959
The File Transfer Protocol

- The objectives of FTP are:
  - to promote sharing of files and encourage use of remote computers
  - to shield a user from variations in file storage systems among hosts
  - to transfer data reliably and efficiently



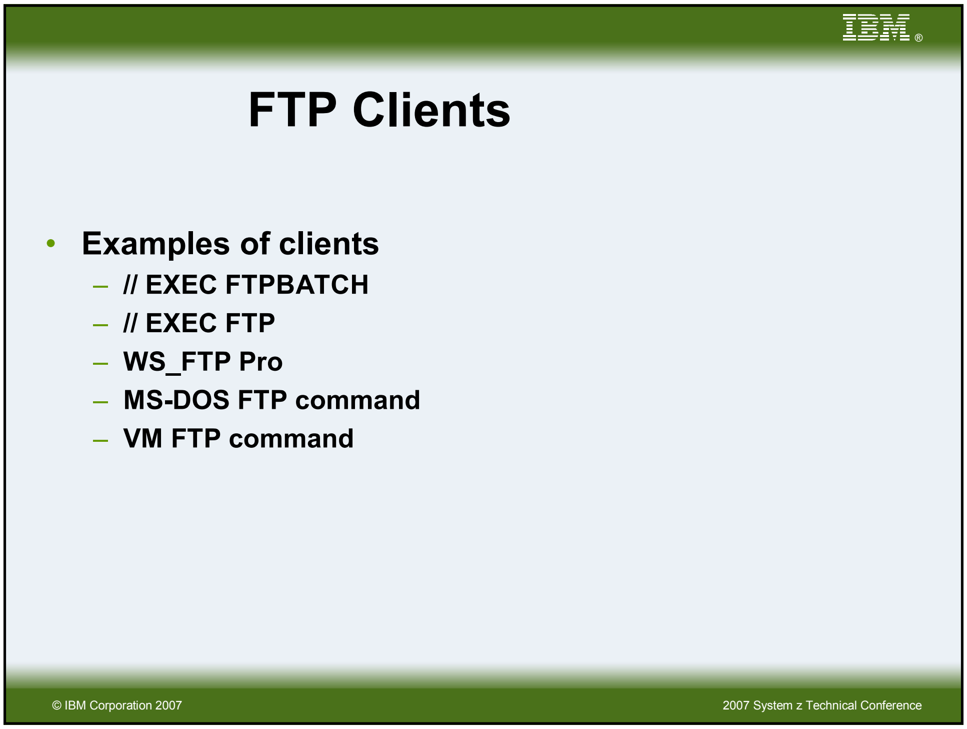
RFC 959
The File Transfer Protocol

- Protocol is a set of rules
- Following the rules allows totally different systems to talk to each other



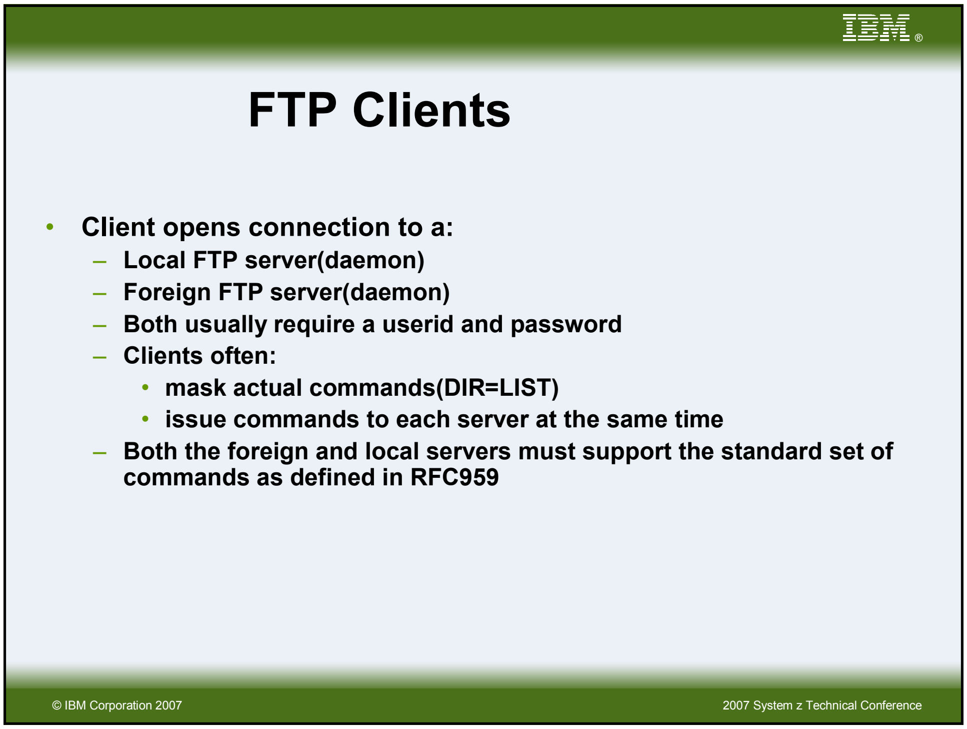
FTP Clients

- All ftp transfers have a single client, also referred to as the control connection
- FTP clients use the telent protocol to send commands and receive replies to a local and foreign FTP server



FTP Clients

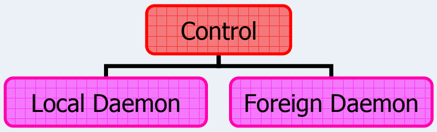
- Examples of clients
  - // EXEC FTPBATCH
  - // EXEC FTP
  - WS_FTP Pro
  - MS-DOS FTP command
  - VM FTP command



FTP Clients

- Client opens connection to a:
  - Local FTP server(daemon)
  - Foreign FTP server(daemon)
  - Both usually require a userid and password
  - Clients often:
    - mask actual commands(DIR=LIST)
    - issue commands to each server at the same time
  - Both the foreign and local servers must support the standard set of commands as defined in RFC959

FTP Protocol



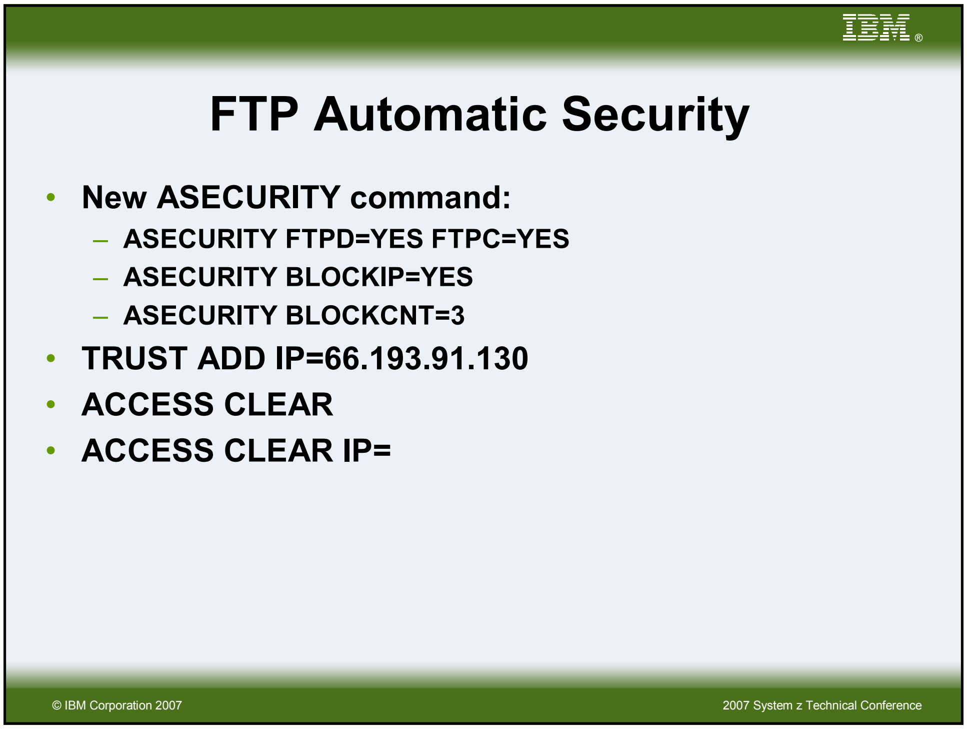
Control

Local Daemon

Foreign Daemon

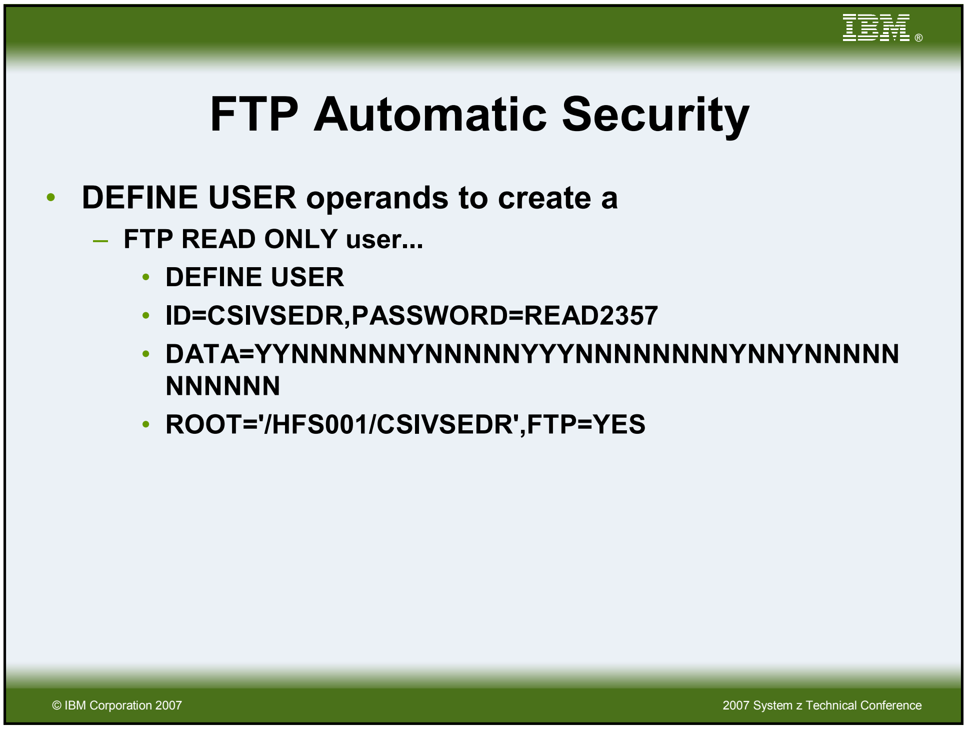
FTP Automatic Security

- No Need to code a security exit!!!
  - SECURITY ON
    - AUTO=ON
    - BATCH=ON
    - MODE=FAIL LOGGING=FAIL



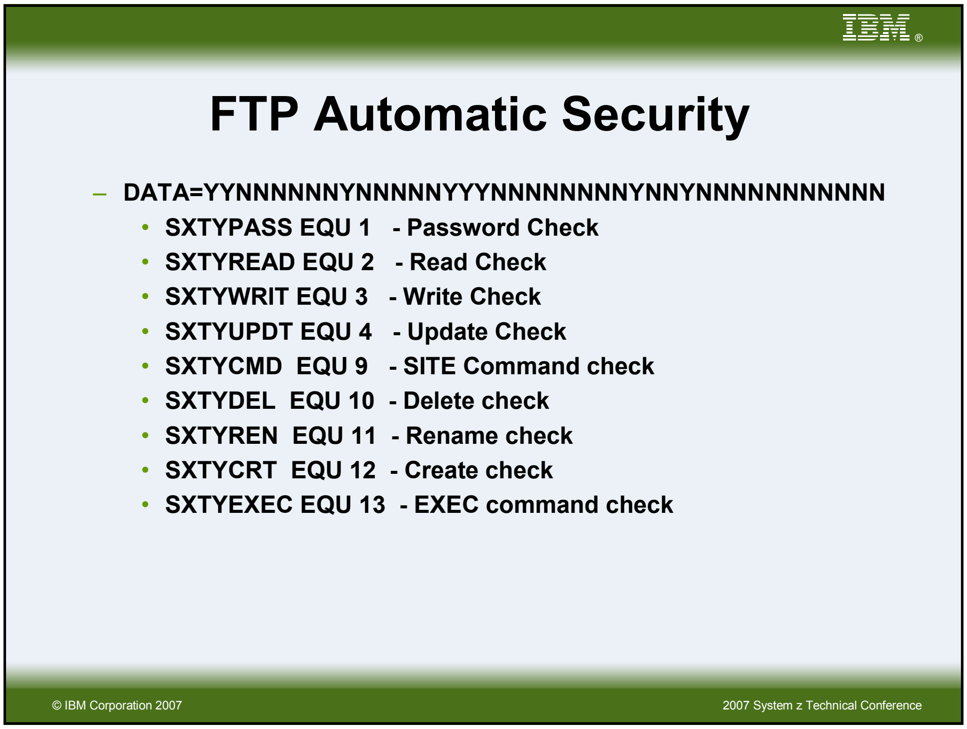
FTP Automatic Security

- New ASECURITY command:
  - ASECURITY FTPD=YES FTPC=YES
  - ASECURITY BLOCKIP=YES
  - ASECURITY BLOCKCNT=3
- TRUST ADD IP=66.193.91.130
- ACCESS CLEAR
- ACCESS CLEAR IP=



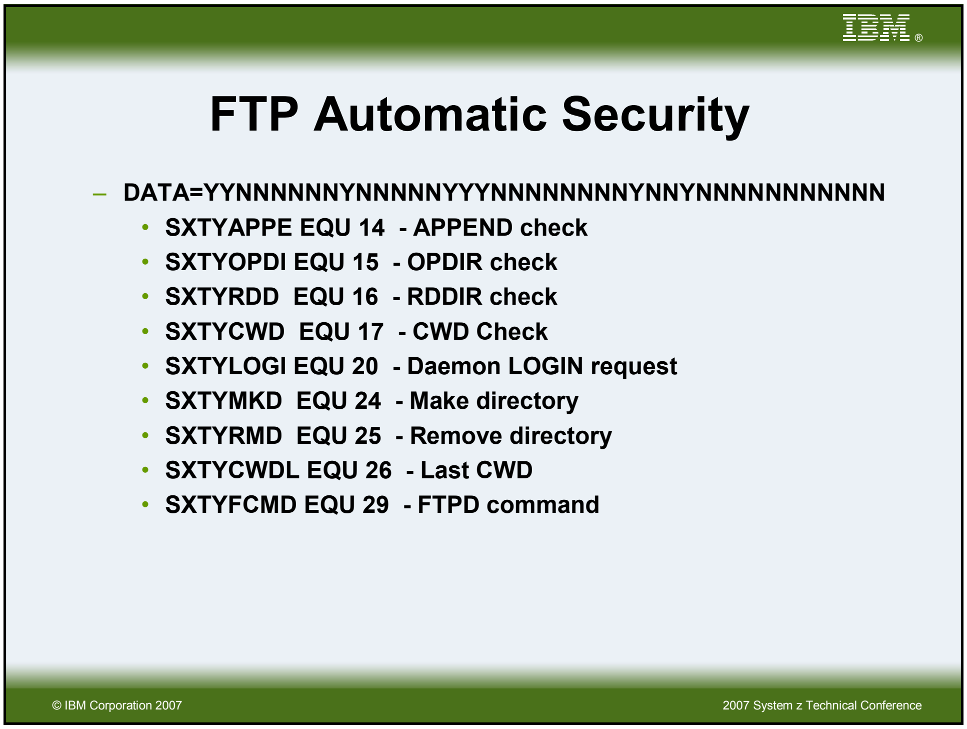
FTP Automatic Security

- DEFINE USER operands to create a
  - FTP READ ONLY user...
    - DEFINE USER
    - ID=CSIVSEDR,PASSWORD=READ2357
    - DATA=YYNNNNNNYNNNNNYYYNNNNNNNNYNNYNNNNN NNNNNN
    - ROOT='/HFS001/CSIVSEDR',FTP=YES



FTP Automatic Security

- DATA=YYNNNNNNYNNNNNYYYNNNNNNNNYNNYNNNNNNNNNNN
  - SXTYPASS EQU 1   - Password Check
  - SXTYREAD EQU 2   - Read Check
  - SXTYWRIT EQU 3   - Write Check
  - SXTYUPDT EQU 4   - Update Check
  - SXTYCMD EQU 9   - SITE Command check
  - SXTYDEL EQU 10 - Delete check
  - SXTYREN EQU 11 - Rename check
  - SXTYCRT EQU 12 - Create check
  - SXTYEXEC EQU 13 - EXEC command check



FTP Automatic Security

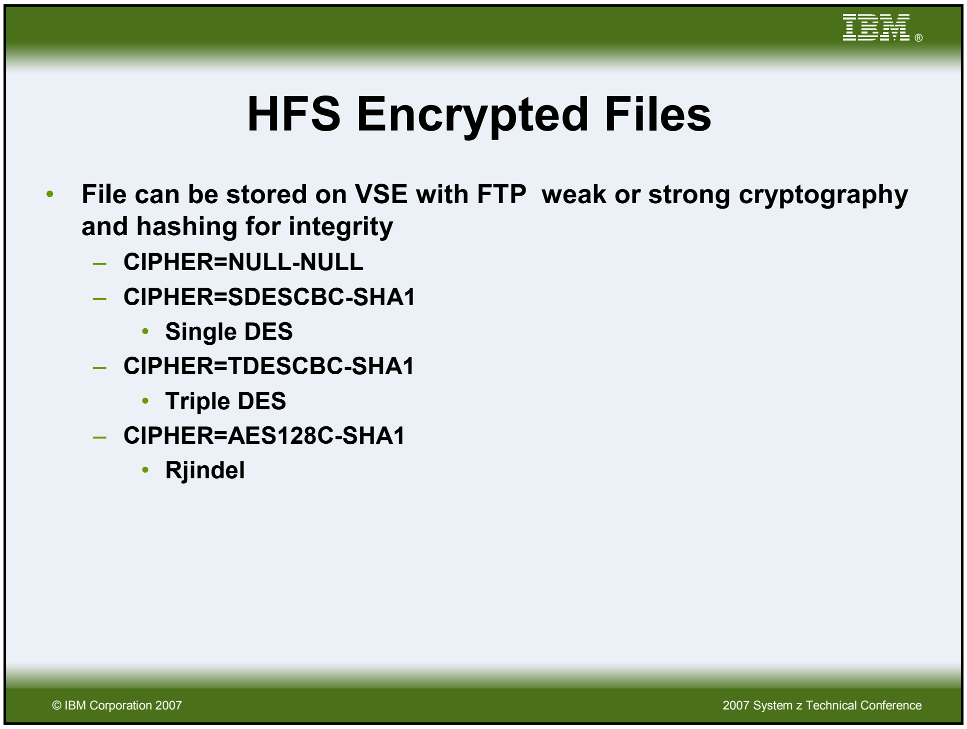
- DATA=YYNNNNNNYNNNNNYYYNNNNNNNNYNNYNNNNNNNNNNN
  - SXTYAPPE EQU 14 - APPEND check
  - SXTYOPDI EQU 15 - OPDIR check
  - SXTYRDD EQU 16 - RDDIR check
  - SXTYCWD EQU 17 - CWD Check
  - SXTYLOGI EQU 20 - Daemon LOGIN request
  - SXTYMKD EQU 24 - Make directory
  - SXTYRMD EQU 25 - Remove directory
  - SXTYCWDL EQU 26 - Last CWD
  - SXTYFCMD EQU 29 - FTPD command

FTP Automatic Security

- Suppose you want to stop any new ftp sessions from being established on VSE
- Simply issue a:
- ASECURITY FTPD=NO
  - No 220-welcome to VSE msg will be sent out to anyone connecting into VSE on the ftp port(usually 21)

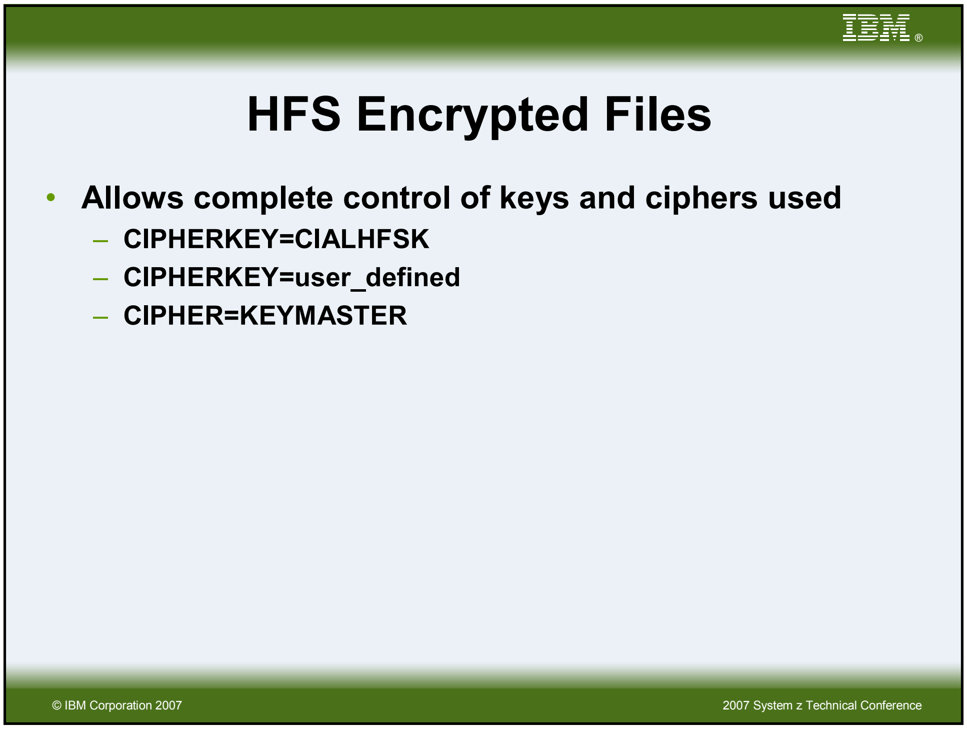
HFS Encrypted Files

- File can be stored on VSE with FTP encrypted!!!
  - Simply use the DEFINE FILE command
  - DEFINE FILE
    - DLBL=HFSTST,
    - PUBLIC=HFSTST
    - TYPE=HFS,RECFM=S,LRECL=4096
    - CIPHER=SDESCBCSHA1
    - CIPHERKEY=CIALHFSK



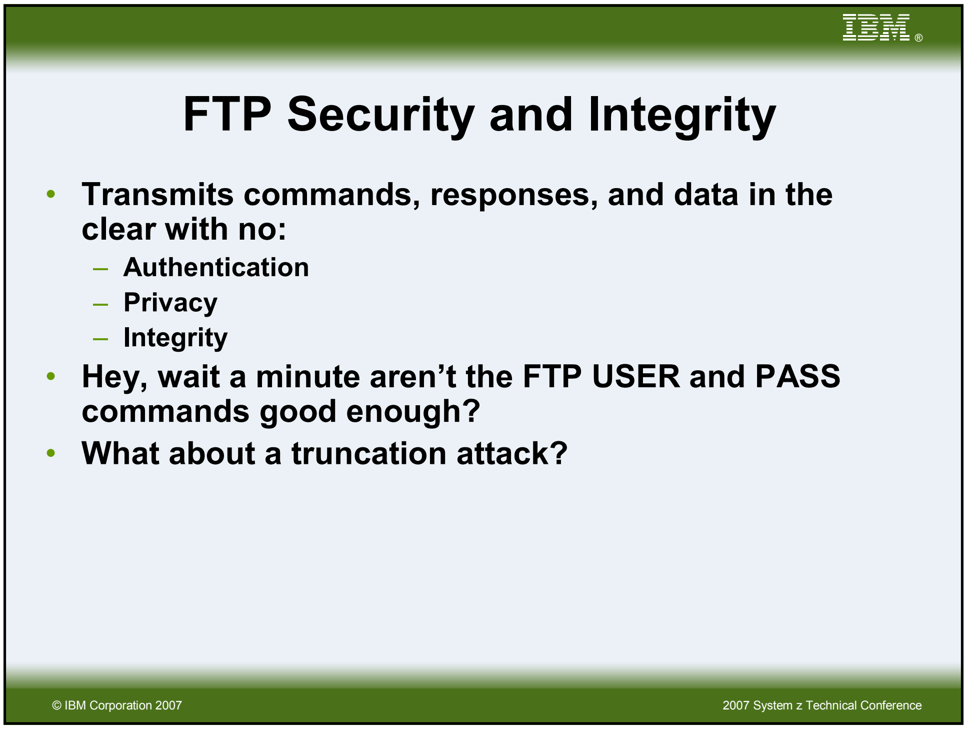
HFS Encrypted Files

- File can be stored on VSE with FTP weak or strong cryptography and hashing for integrity
  - CIPHER=NULL-NULL
  - CIPHER=SDESCBC-SHA1
    - Single DES
  - CIPHER=TDESCBC-SHA1
    - Triple DES
  - CIPHER=AES128C-SHA1
    - Rjindel



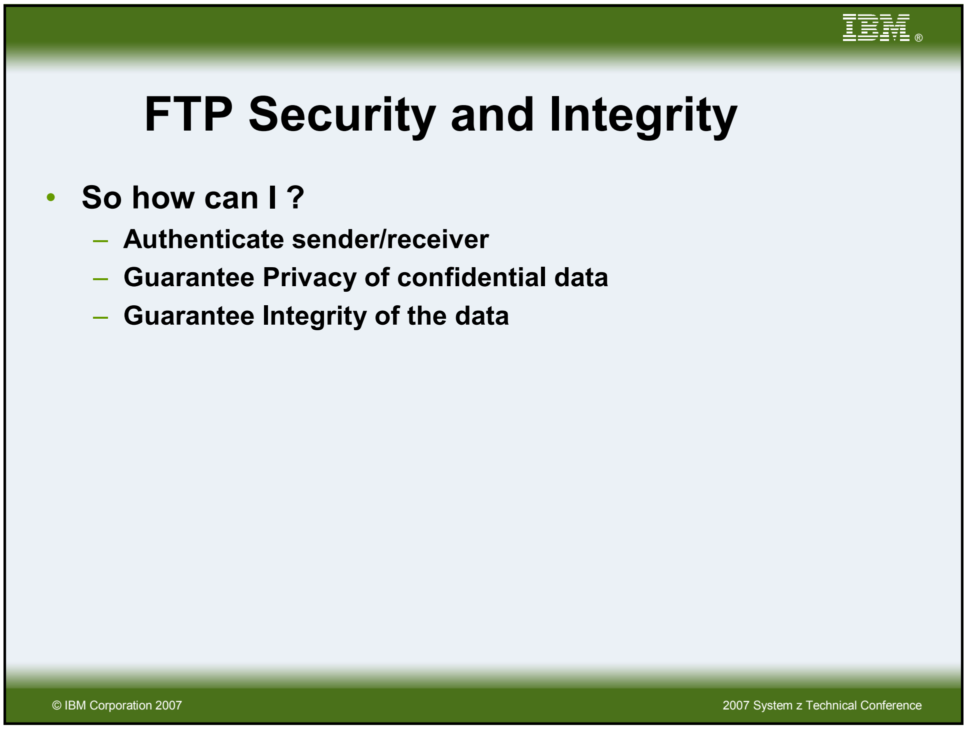
HFS Encrypted Files

- Allows complete control of keys and ciphers used
  - CIPHERKEY=CIALHFSK
  - CIPHERKEY=user_defined
  - CIPHER=KEYMASTER



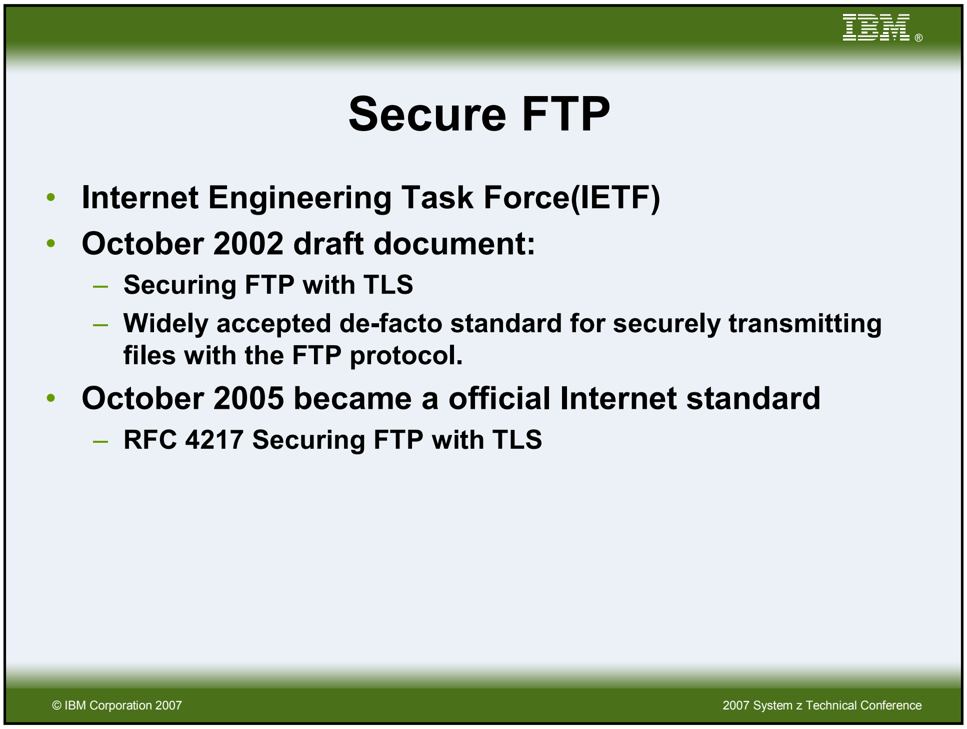
FTP Security and Integrity

- Transmits commands, responses, and data in the clear with no:
  - Authentication
  - Privacy
  - Integrity
- Hey, wait a minute aren’t the FTP USER and PASS commands good enough?
- What about a truncation attack?



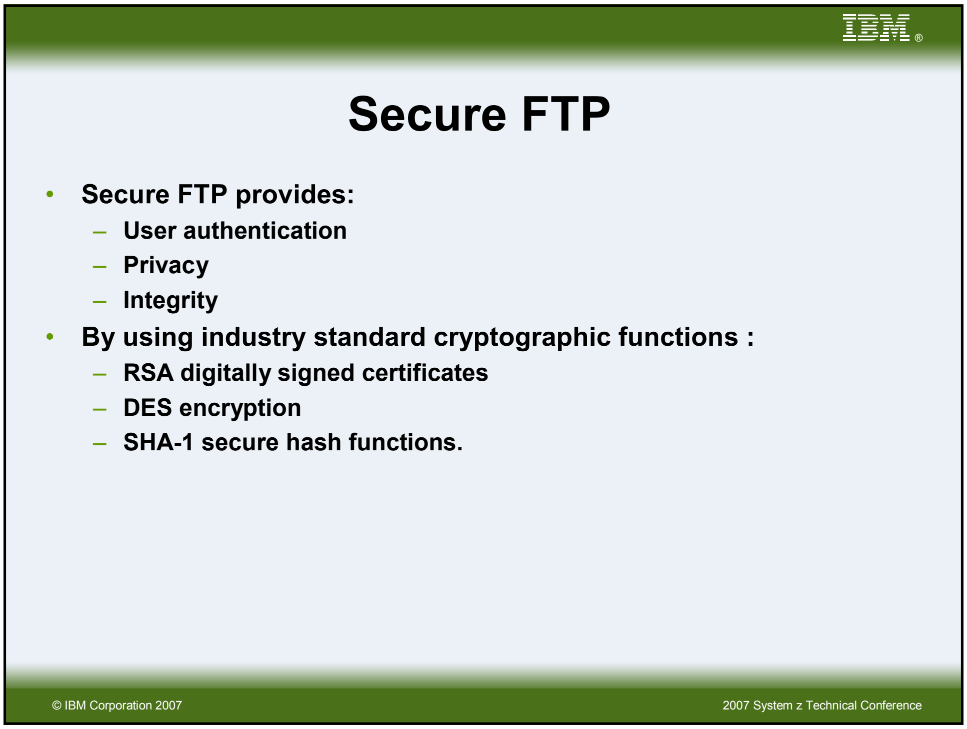
FTP Security and Integrity

- So how can I ?
  - Authenticate sender/receiver
  - Guarantee Privacy of confidential data
  - Guarantee Integrity of the data



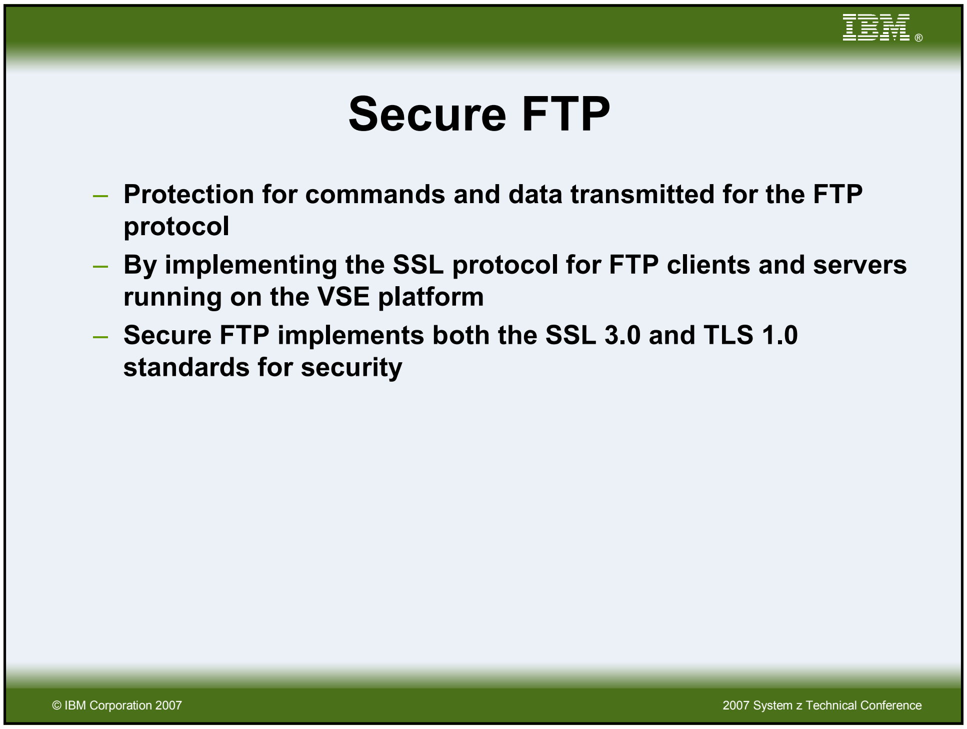
Secure FTP

- Internet Engineering Task Force(IETF)
- October 2002 draft document:
  - Securing FTP with TLS
  - Widely accepted de-facto standard for securely transmitting files with the FTP protocol.
- October 2005 became a official Internet standard
  - RFC 4217 Securing FTP with TLS



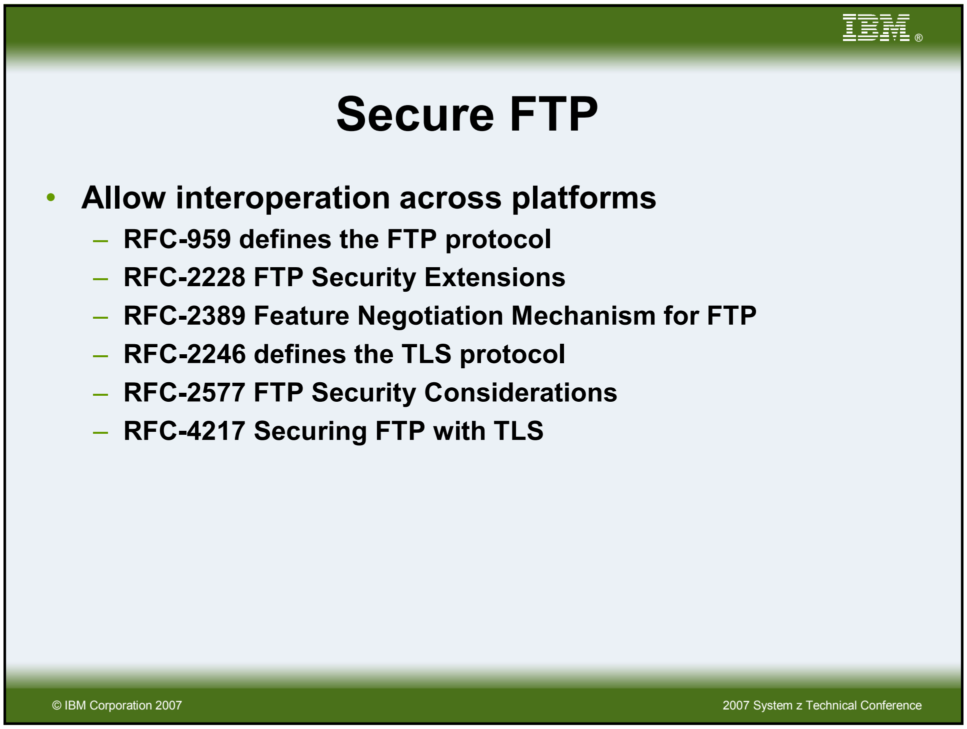
Secure FTP

- Secure FTP provides:
  - User authentication
  - Privacy
  - Integrity
- By using industry standard cryptographic functions :
  - RSA digitally signed certificates
  - DES encryption
  - SHA-1 secure hash functions.



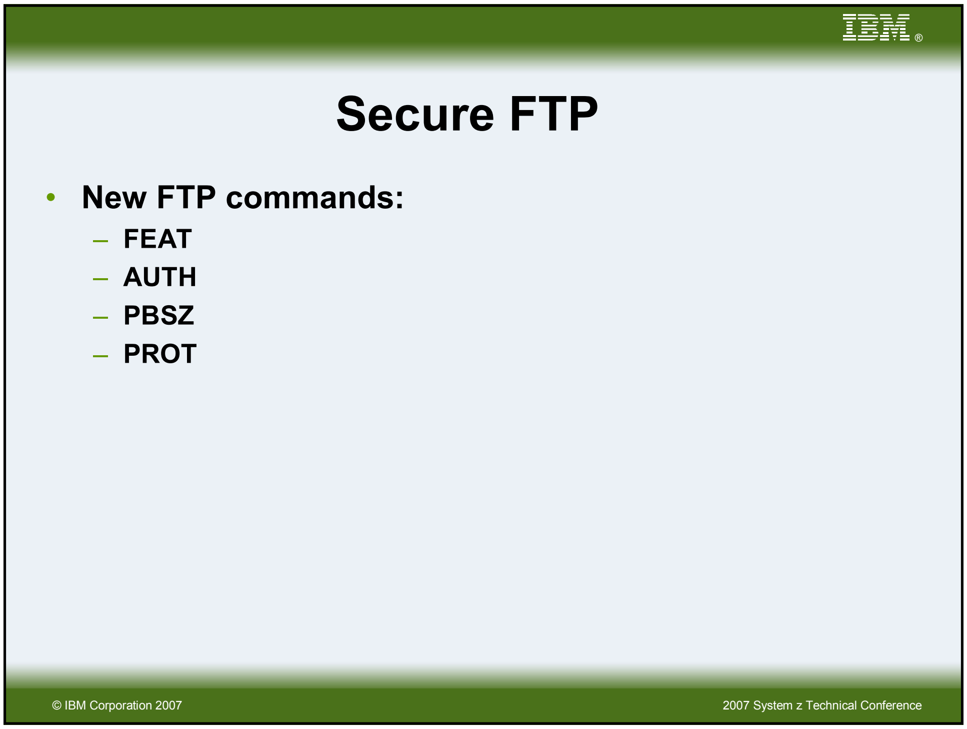
Secure FTP

- Protection for commands and data transmitted for the FTP protocol
- By implementing the SSL protocol for FTP clients and servers running on the VSE platform
- Secure FTP implements both the SSL 3.0 and TLS 1.0 standards for security



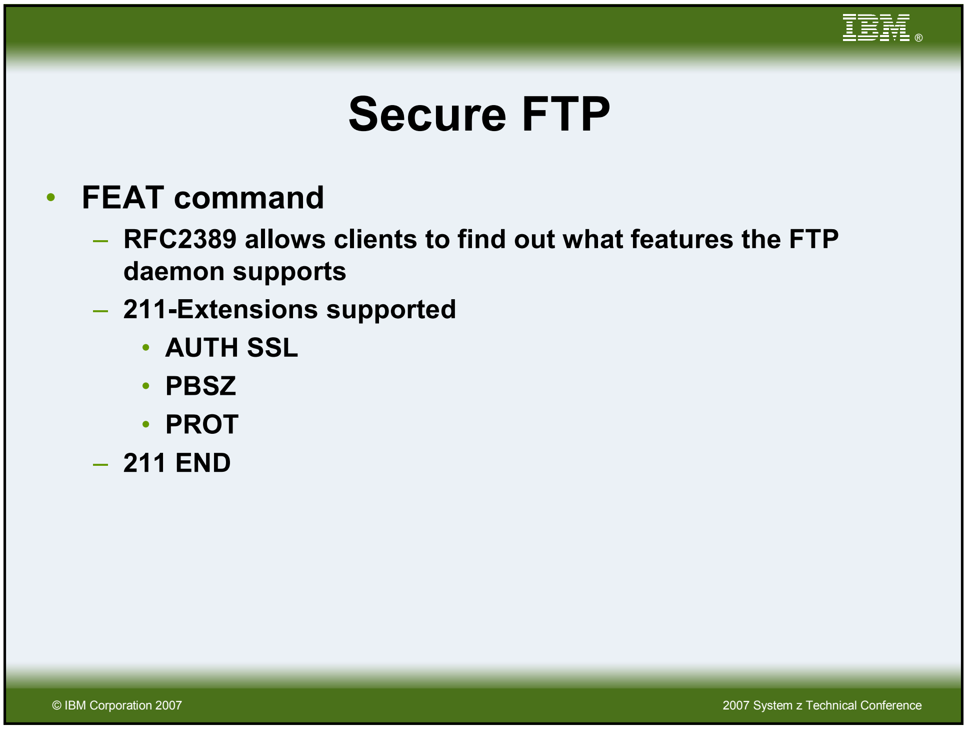
Secure FTP

- Allow interoperation across platforms
  - RFC-959 defines the FTP protocol
  - RFC-2228 FTP Security Extensions
  - RFC-2389 Feature Negotiation Mechanism for FTP
  - RFC-2246 defines the TLS protocol
  - RFC-2577 FTP Security Considerations
  - RFC-4217 Securing FTP with TLS



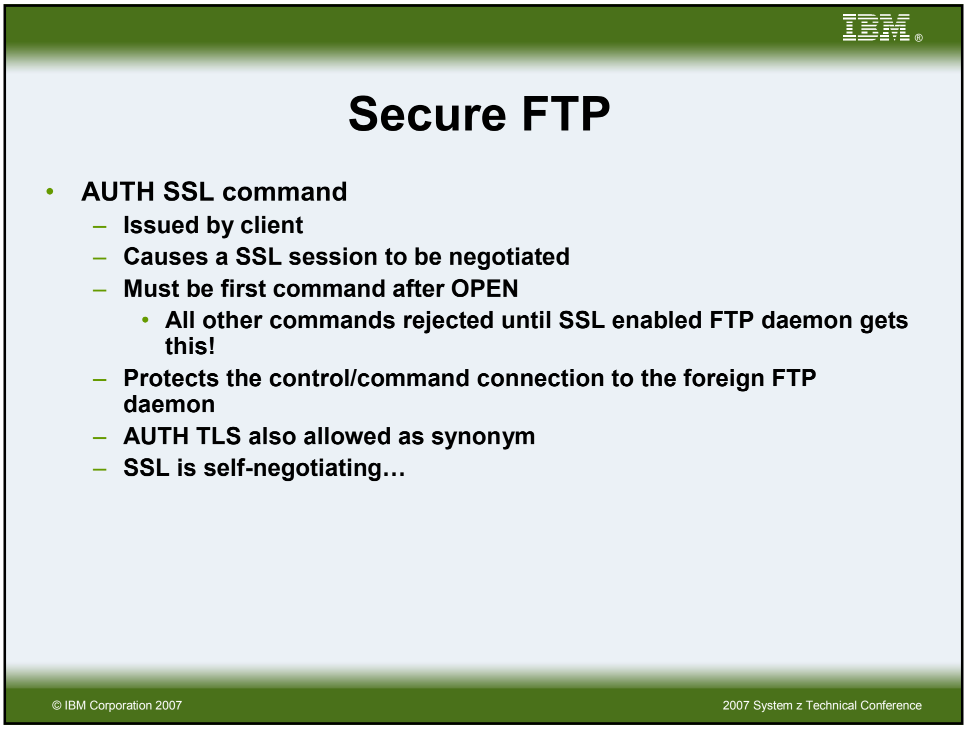
Secure FTP

- New FTP commands:
  - FEAT
  - AUTH
  - PBSZ
  - PROT



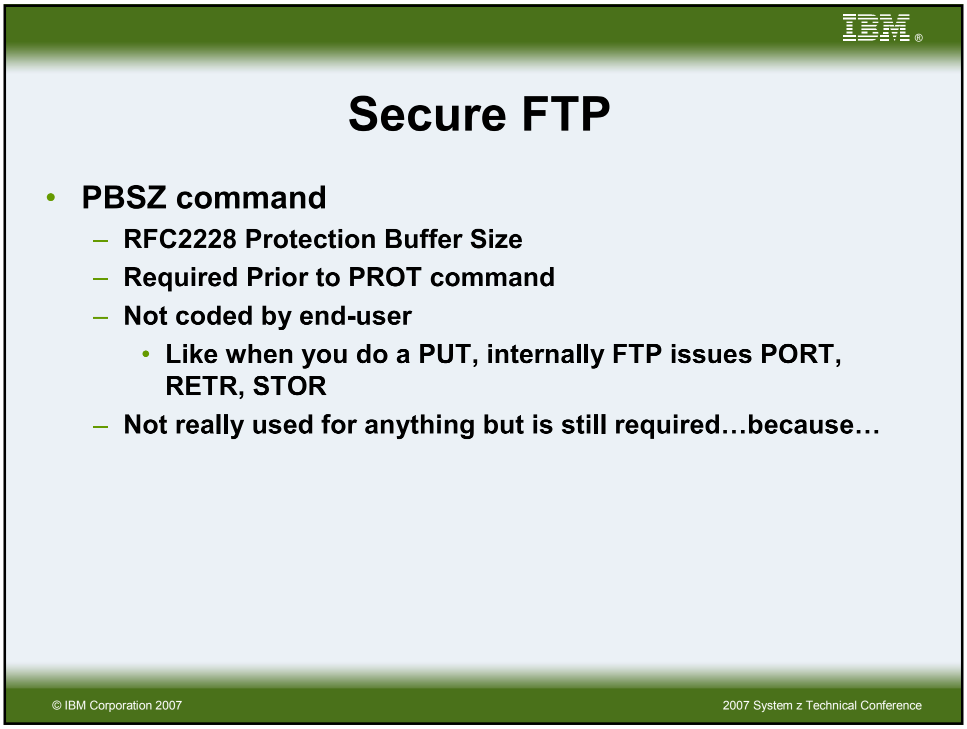
Secure FTP

- FEAT command
  - RFC2389 allows clients to find out what features the FTP daemon supports
  - 211-Extensions supported
    - AUTH SSL
    - PBSZ
    - PROT
  - 211 END



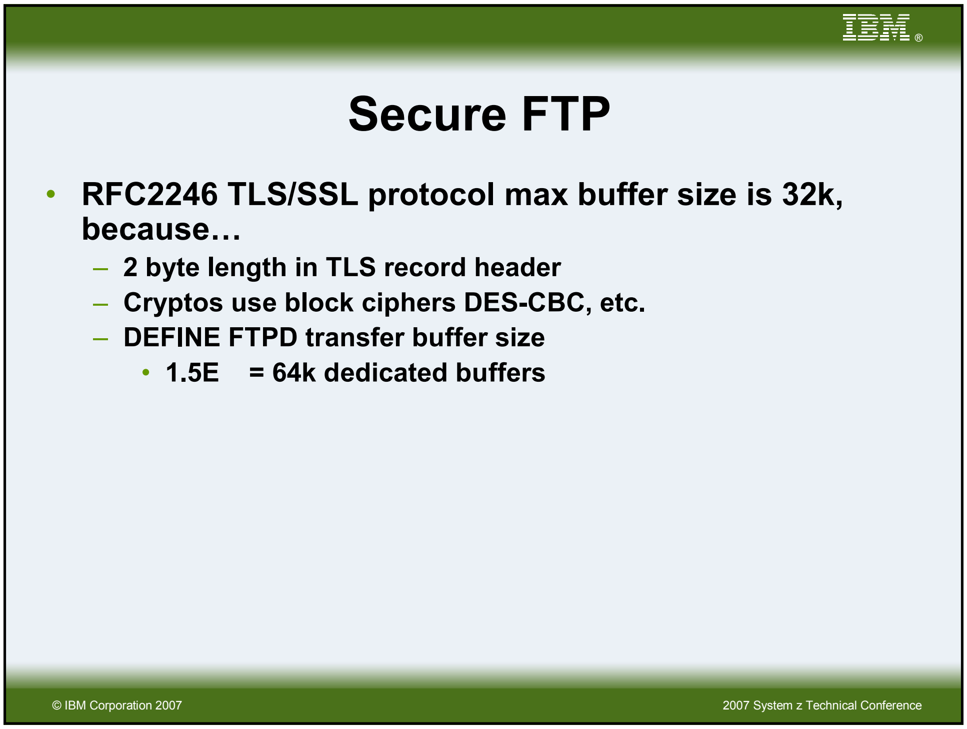
Secure FTP

- AUTH SSL command
  - Issued by client
  - Causes a SSL session to be negotiated
  - Must be first command after OPEN
    - All other commands rejected until SSL enabled FTP daemon gets this!
  - Protects the control/command connection to the foreign FTP daemon
  - AUTH TLS also allowed as synonym
  - SSL is self-negotiating…



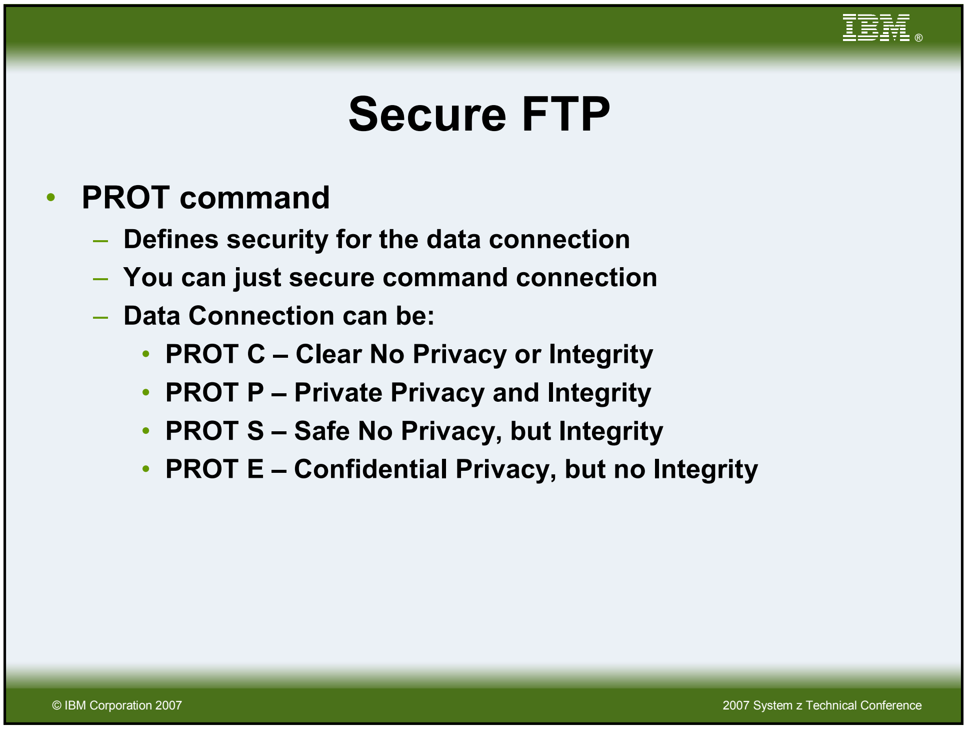
Secure FTP

- PBSZ command
  - RFC2228 Protection Buffer Size
  - Required Prior to PROT command
  - Not coded by end-user
    - Like when you do a PUT, internally FTP issues PORT, RETR, STOR
  - Not really used for anything but is still required…because…



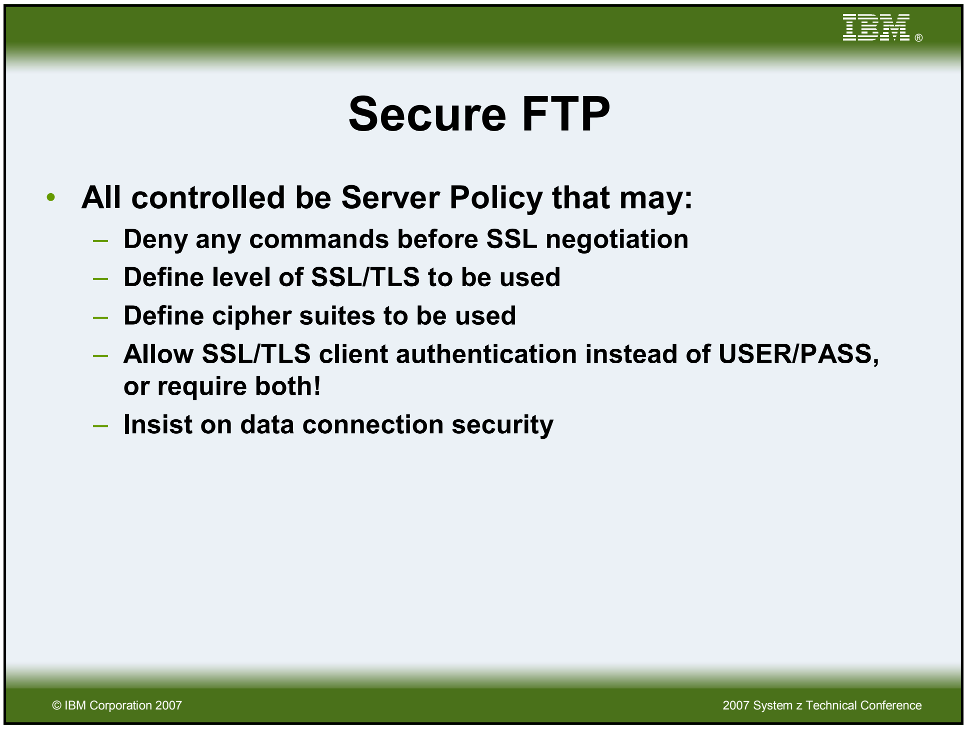
Secure FTP

- RFC2246 TLS/SSL protocol max buffer size is 32k, because…
  - 2 byte length in TLS record header
  - Cryptos use block ciphers DES-CBC, etc.
  - DEFINE FTPD transfer buffer size
    - 1.5E = 64k dedicated buffers



Secure FTP

- PROT command
  - Defines security for the data connection
  - You can just secure command connection
  - Data Connection can be:
    - PROT C – Clear No Privacy or Integrity
    - PROT P – Private Privacy and Integrity
    - PROT S – Safe No Privacy, but Integrity
    - PROT E – Confidential Privacy, but no Integrity



Secure FTP

- All controlled be Server Policy that may:
  - Deny any commands before SSL negotiation
  - Define level of SSL/TLS to be used
  - Define cipher suites to be used
  - Allow SSL/TLS client authentication instead of USER/PASS, or require both!
  - Insist on data connection security

Questions?