Chapter 63
FTP and HTTP

- **The Art of Serving Files and Content.**

“And bring me a hard copy of the Internet so I can do some serious surfing.”

-- Scott Adams

DeepSec Vienna 2007
7 Layers of Insecurity

Copyright Information

- Michael Kafka, René Pfeiffer, Sebastian Mayer
  C.a.T. Consulting and Trainings, Vienna, Austria

Chapter 63
FTP and HTTP

- Agenda
  - FTP Overview
  - HTTP Overview
  - HTTP Attacks
  - WebDAV
  - Tunneling

File Transfer Protocol (FTP)

- **Historical File Transfers.**

FTP Overview

- Born with RFC 114 in 1971, older than TCP
- FTP uses three modes
  - active FTP, passive FTP and
  - extended passive FTP
- FTP uses two TCP connections
  - Command channel 21/TCP
  - Data channel 20/TCP, dynamic/TCP

FTP Properties

- FTP transmissions are unencrypted
  - Important for logins/passwords
- Anonymous mode for serving data
- Higher latency because of commands
- No integrity check of transmitted files
- FTP adds complexity to filters

Anonymous FTP

- Anonymous FTP is used frequently
- No login, no password
  - Any login will do
  - Polite users use e-mail as password
- Writable anon FTP servers are dangerous
  - Abuse as file deposit possible
  - Fix: anonymous ≠ writable

File eXchange Protocol (FXP)

Client C → Server A

C→A : Connect
C→A : PASV
A→C : 227 Entering Passive Mode.
A1,A2,A3,A4,a1,a2

C→A : STOR

Client C → Server B

C→B : Connect

C→B : PORT A1,A2,A3,A4,a1,a2
B→C : OK
C→B : RETR

B→A : Connect to Server A, port a

DeepSec Vienna 2007
7 Layers of Insecurity

Trivial File Transfer Protocol (TFTP)

- **Adding Triviality may create Insecurity.**

DeepSec Vienna 2007
7 Layers of Insecurity

TFTP Overview

- TFTP uses 69/UDP for data transmission
- TFTP uses no encryption
- TFTP reads/writes files
  - No commands
  - No directory listings
  - No authentication
- TFTP supplies own transport and session
  - Single file transfers (one at a time)
  - Lock-step mode

DeepSec Vienna 2007
7 Layers of Insecurity

TFTP Security

- TFTP server often carry boot information
  - Initial configurations
  - Boot images
- TFTP don't support directory listings
  - Scanning nevertheless possible
  - Brute-force file requests
- Implementation bugs
  - Long file names, packet floods, …
- Risk: medium Impact: medium

HyperText Transfer Protocol

- **Let's just Call It The Web.**

DeepSec Vienna 2007
7 Layers of Insecurity

HTTP Overview

- Stateless request/response protocol
  - Request type
  - Header information
  - Data (request/response body)
- Response carries status code
- HTTP assumes reliable transport
  - TCP/IP is common
  - Can be used with other protocols

DeepSec Vienna 2007
7 Layers of Insecurity

Common HTTP Methods

- HEAD
- GET
- POST
- PUT
- DELETE
- TRACE
- OPTIONS
- CONNECT

HTTP Authentication

- HTTP offers “built-in” authentication
- Basic access authentication
  - Server asks for password to realm
  - Client answers with BASE64 “encryption”
- Digest access authentication
  - Server asks for challenge
  - Client must create a suitable response
  - Challenge/response uses MD5 hashes

DeepSec Vienna 2007
7 Layers of Insecurity

Breaking Authentication

- Basic authentication
  - BASE64 is next to no challenge
  - Brute-force passwords, sniffing
- Digest authentication
  - Brute-force with sniffed nonce/response
  - MITM attack with brute-force
- Risk: medium
- Impact: medium/high

HTTP Response Splitting

- Trick web application with %0a and %0d
- Target redirect scripts
  - Insert CR and LF (%0a, %0d)
  - Add a second HTTP header
  - Overwrite first header information
- User sees different web page
- Caches might store different content
- Risk: medium Impact: medium

HTTP Request Smuggling

- Create a HTTP request, send it to proxy
- Embed second HTTP request in header
- Check if intermediate system works correctly
  - SunOne Proxy 3.6 (SP4) doesn't
  - FW-1/FP4-R55W β doesn't
- Fw/IPS/IDS evasion
- Cache poisoning
- Risk: medium
- Impact: medium

WebDAV

- Web-based Distributed Authoring and Versioning.

WebDAV Overview

- WebDAV enables reading & writing to server
  - Used for collaborative editing
  - Other extensions exist
- WebDAV adds methods to HTTP
  - PROPFIND, MKCOL, COPY, LOCK, …
- WebDAV clients use permissions of server
  - Important for WebDAV-enabled “shares”
  - Shares mustn't overlap with web space

WebDAV Attacks

- mod_dav keeps request bodies in memory
  - Possible memory exhaustion
  - LimitXMLRequestBody directive
- PROPFIND causes directory walks
  - Memory/CPU exhaustion
  - Disallow ∞ depth requests
- WebDAV allows file sharing
  - Disk space exhaustion

WebDAV Network Traffic

- Protect WebDAV access
  - by authentication such as digest auth
  - by encryption such as SSL/TLS
  - by limiting exposed directories on server
  - by limiting HTTP methods
- Inspect HTTP headers
  - Use layer 7 proxy
  - Monitor & analyse for anomalies

HTTP Tunneling

- **Hyper Text Tunneling Protocol.**

Firewall Enhancement Protocol (FEP)

Internet Transparency via the end-to-end architecture of the Internet has allowed vast innovation of new technologies and services [1]. However, recent developments in Firewall technology have altered this model and have been shown to inhibit innovation. We propose the Firewall Enhancement Protocol (FEP) to allow innovation, without violating the security model of a Firewall. With no cooperation from a firewall operator, the FEP allows ANY application to traverse a Firewall. Our methodology is to layer any application layer Transmission Control Protocol/User Datagram Protocol (TCP/UDP) packets over the HyperText Transfer Protocol (HTTP) protocol, since HTTP packets are typically able to transit Firewalls.

-- RFC 3093, 1rst April 2001

HTTP Tunneling

- HTTP is very proxy-friendly
- Many protocols use it as tunnel
  - CONNECT method provides TCP/IP
  - HTTP transports all things binary
- Ideal for piercing firewalls
- Works through proxies
- Mediator web server connects from end-point

HTTP Tunnel Tools

- GNU httptunnel
  - htc/hts, work without web server
- JHTTPtunnel
  - Java implementation
  - Think mobile clients
- HTTPtunnel - in PHP/Perl

DeepSec Vienna 2007
7 Layers of Insecurity

Chapter 62
FTP and HTTP

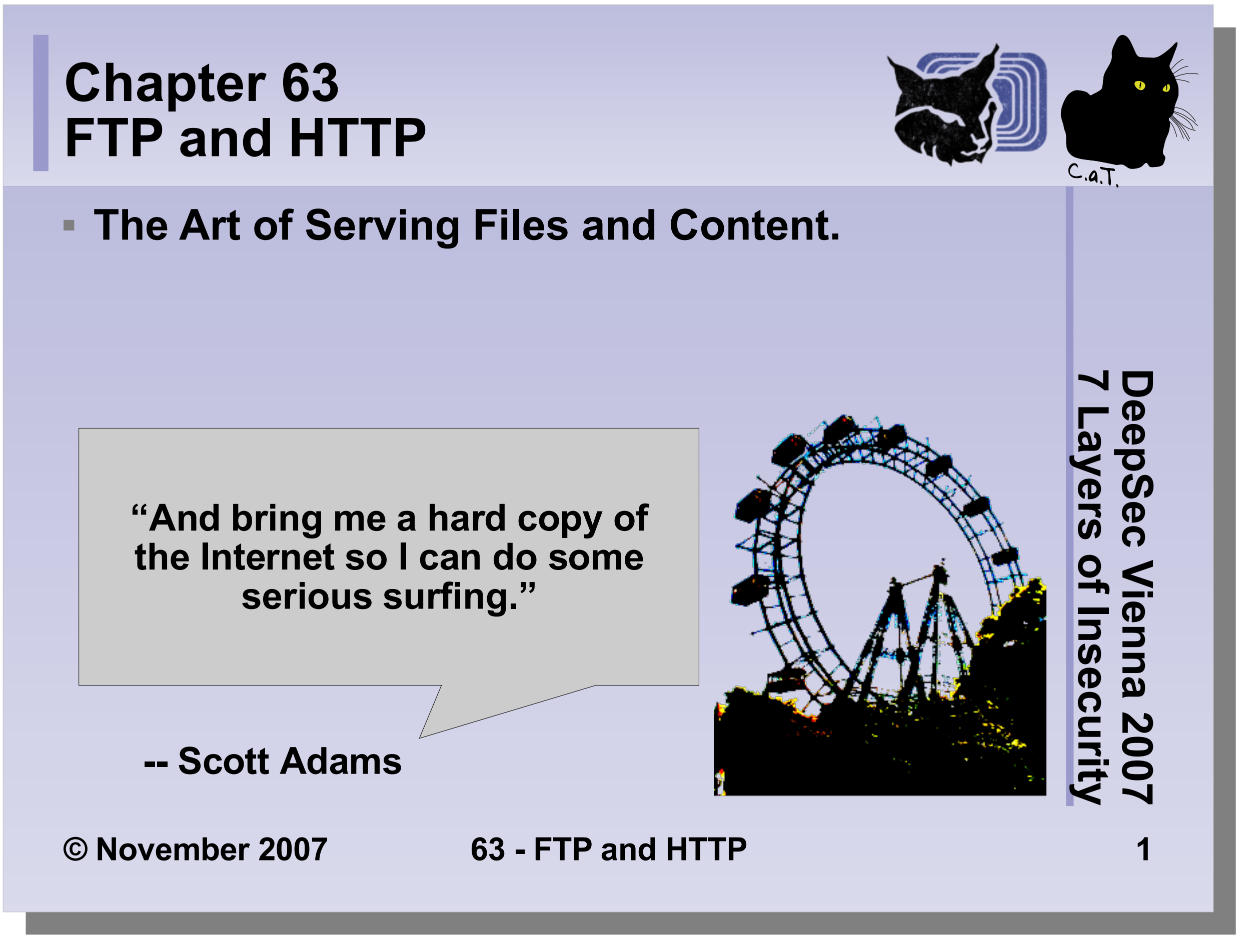
- **Summary**
  - **Use FTP servers with caution.**
  - **TFTP servers are a critical resource.**
  - **HTTP is a data delivery service.**
  - **Use authentication and encryption if required.**
  - **Take care of proxies and tunnels.**

DeepSec Vienna 2007
7 Layers of Insecurity

Thank You

- Questions?

DeepSec Vienna 2007
7 Layers of Insecurity



Chapter 63
FTP and HTTP

- The Art of Serving Files and Content.

> “And bring me a hard copy of the Internet so I can do some serious surfing.”

-- Scott Adams

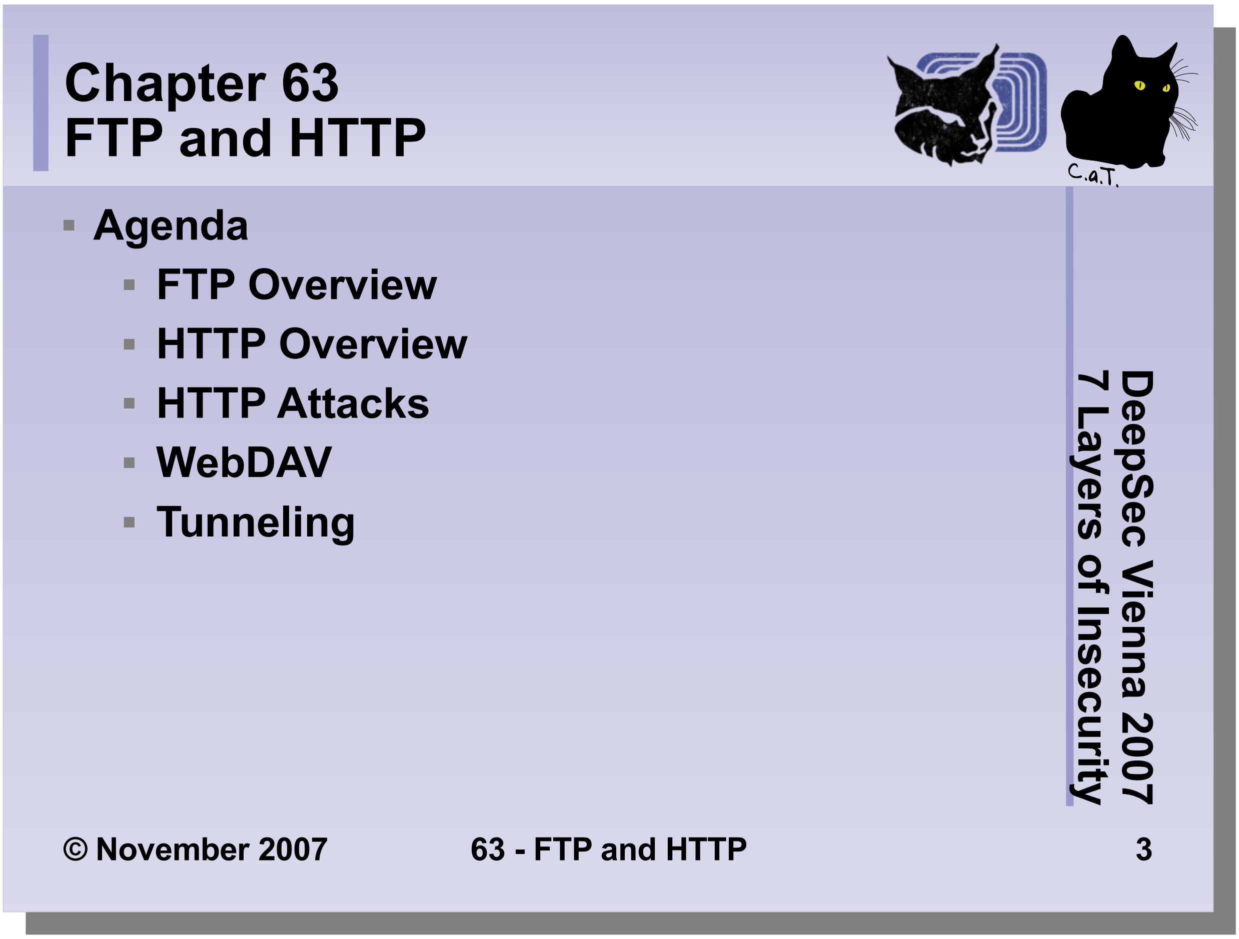
DeepSec Vienna 2007
7 Layers of Insecurity

C.a.T.

Copyright Information

DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 2

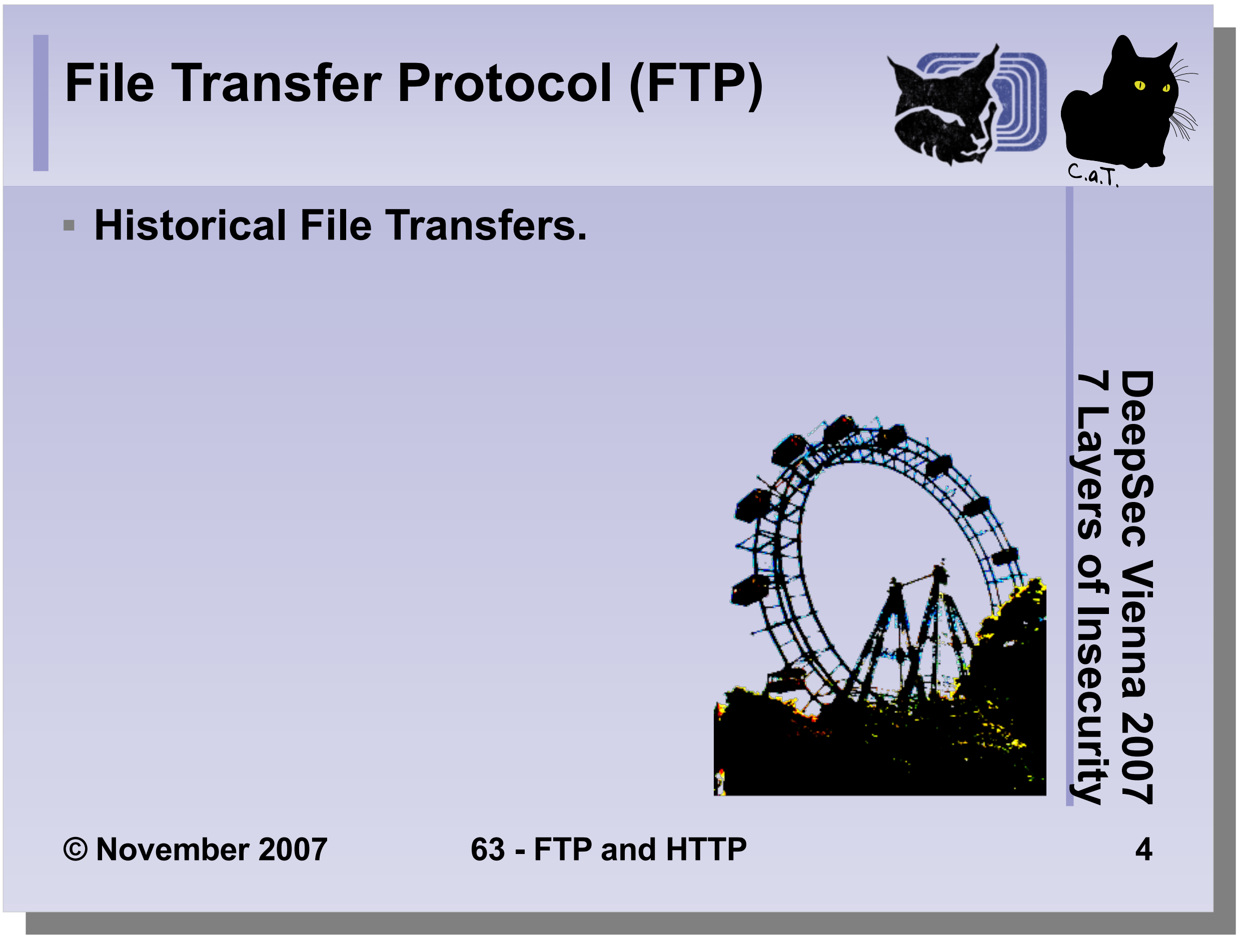


Chapter 63
FTP and HTTP

- Agenda
  - FTP Overview
  - HTTP Overview
  - HTTP Attacks
  - WebDAV
  - Tunneling

DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 3

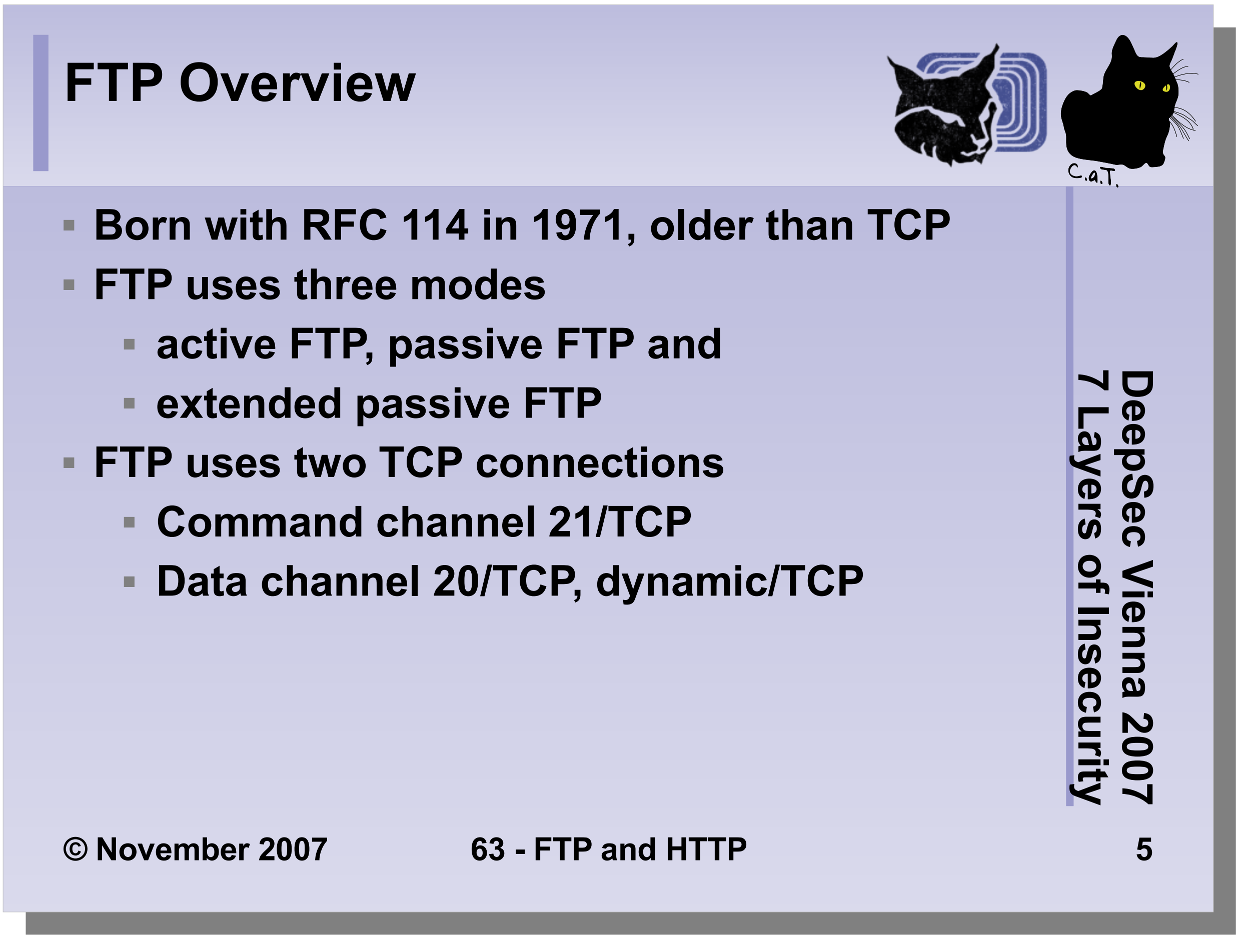


File Transfer Protocol (FTP)

- Historical File Transfers.

DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 4



FTP Overview

- Born with RFC 114 in 1971, older than TCP
- FTP uses three modes
  - active FTP, passive FTP and
  - extended passive FTP
- FTP uses two TCP connections
  - Command channel 21/TCP
  - Data channel 20/TCP, dynamic/TCP

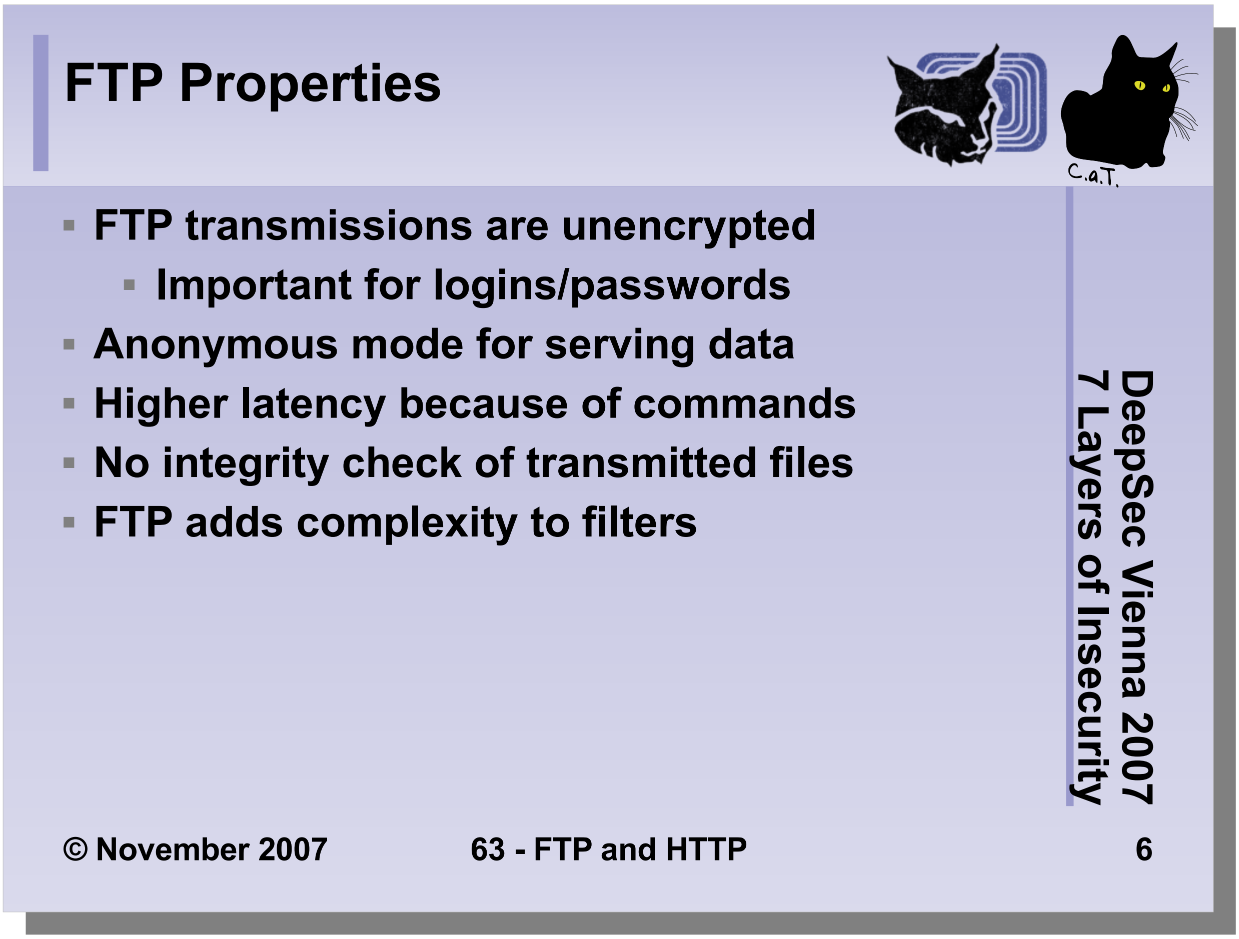
DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 5

The active mode in FTP opens a data connection from the server port 20/TCP to a random dynamic port on the client side. This breaks NAT and renders static packet filters useless. Modern packet filters analyse the FTP PORT command and create dynamic filter rules to allow active FTP data connections.

The passive mode lets the client open a TCP data connection from a dynamic port to a dynamic port on the FTP server. This solves the NAT problem on the client side, but requires a FTP-aware packet filter on the server side in order to avoid opening the whole dynamic port range for FTP data transmissions.

Extended passive FTP mode is very similar to the passive mode. The only difference is the format of the PORT announcement. The PORT command uses a string consisting of the IP address and the port to be used for the data transmission. The string look like this: *h1,h2,h3,h4,p1,p2* The first part is the decimal representation of all IP address octets. The port is calculated as $port = p1 \times 256 + p2$. In extended passive mode client only transmits the port number directly without IP address.



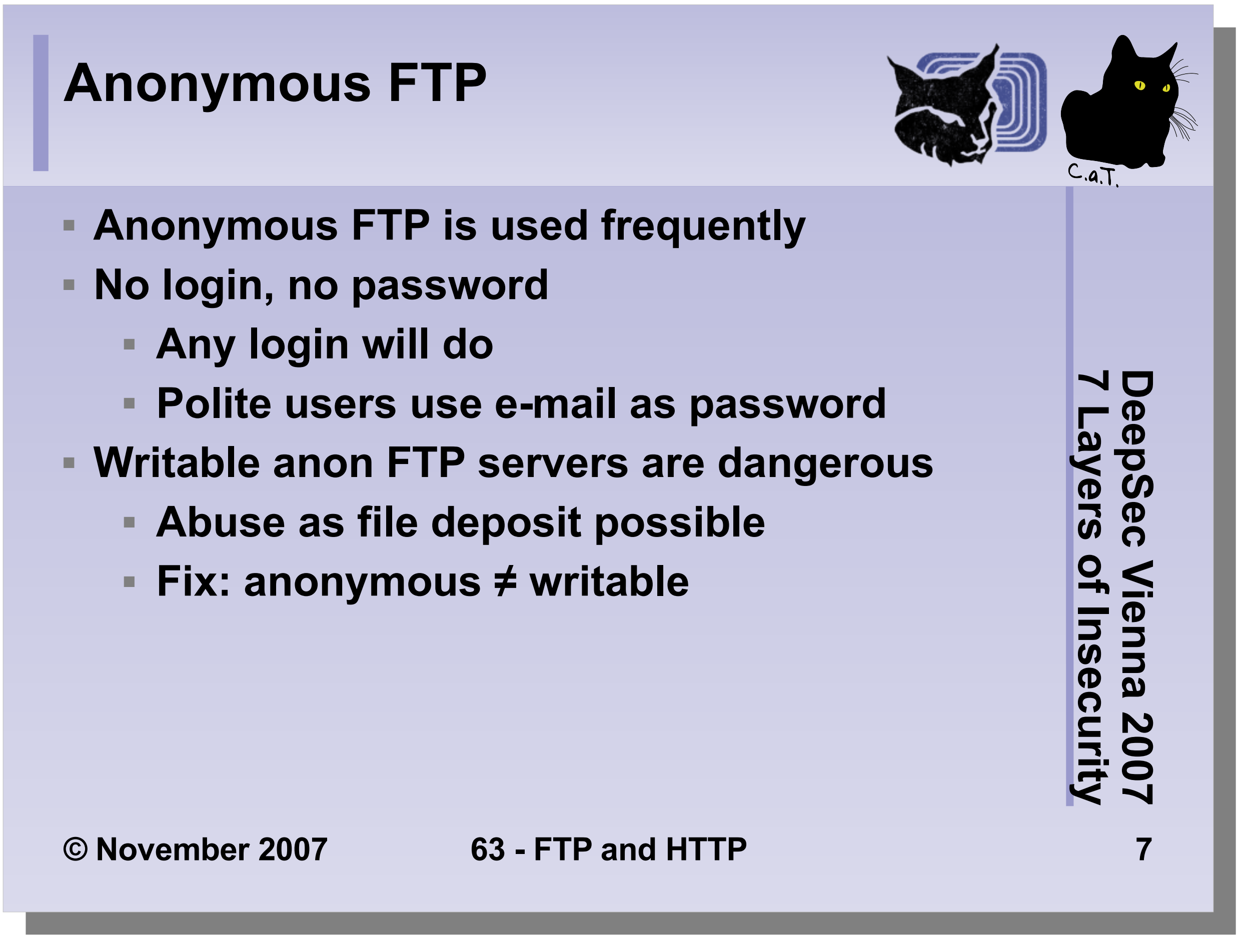
FTP Properties

C.a.T.

- FTP transmissions are unencrypted
  - Important for logins/passwords
- Anonymous mode for serving data
- Higher latency because of commands
- No integrity check of transmitted files
- FTP adds complexity to filters

DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 6

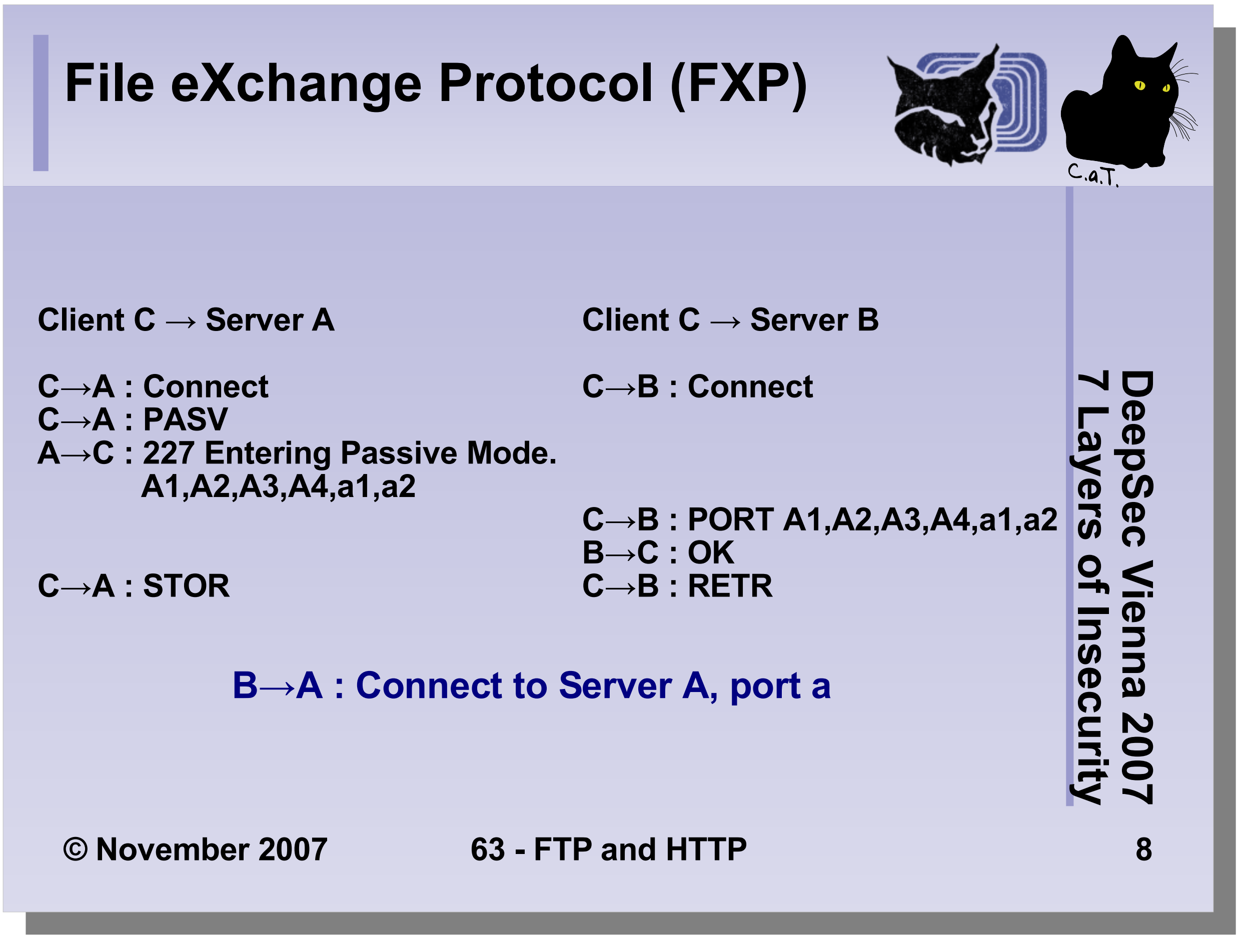


Anonymous FTP

- Anonymous FTP is used frequently
- No login, no password
  - Any login will do
  - Polite users use e-mail as password
- Writable anon FTP servers are dangerous
  - Abuse as file deposit possible
  - Fix: anonymous ≠ writable

DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 7



File eXchange Protocol (FXP)

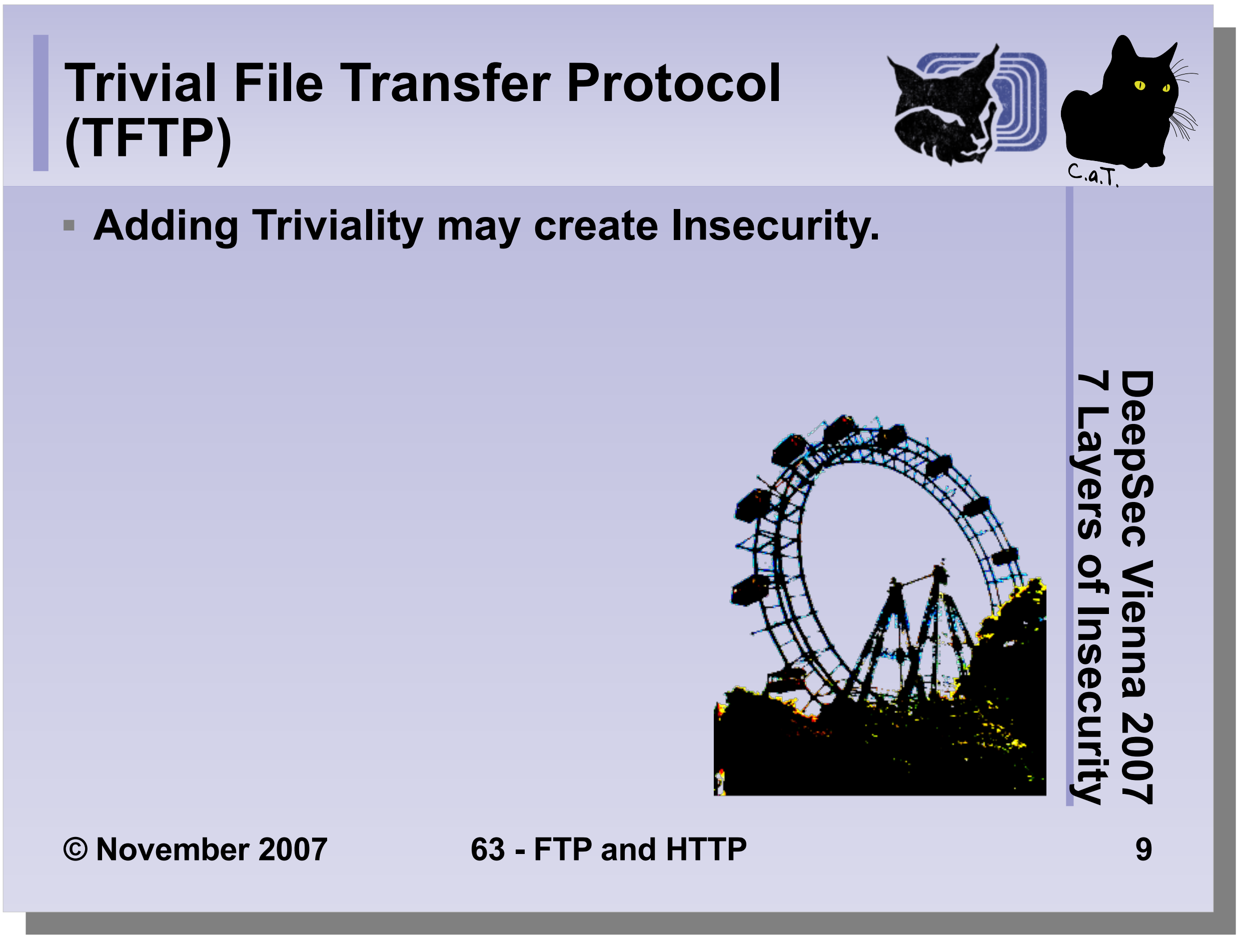
Client C → Server A	Client C → Server B
C→A : Connect	C→B : Connect
C→A : PASV	
A→C : 227 Entering Passive Mode. A1,A2,A3,A4,a1,a2	
	C→B : PORT A1,A2,A3,A4,a1,a2
	B→C : OK
C→A : STOR	C→B : RETR

B→A : Connect to Server A, port a

DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 8

FXP is an early addendum to the FTP protocol. At client can connect to two different FTP servers and links them by issuing a STOR and RETR command with port information pointing to the receiving FTP server. The original idea was to initiate file transmissions between well connected servers from a third location having limited bandwidth. A side effect of FXP is the so-called *FTP bounce attack*. A malicious client exploits the PORT commands and directs the second TCP transmission to another host. FTP servers susceptible to the bounce attack can be used to port scan victims or to act as agents for packet floods.

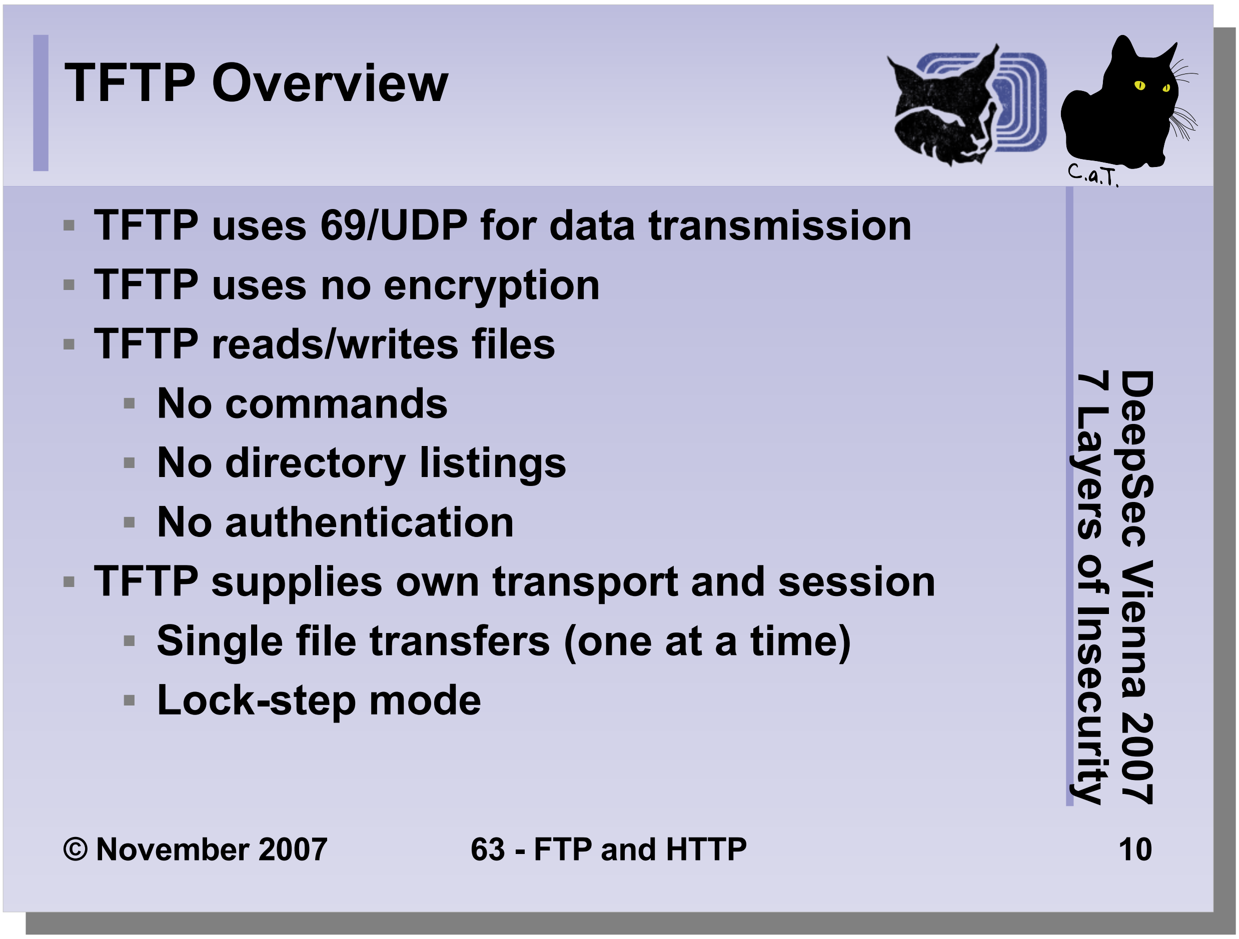


Trivial File Transfer Protocol (TFTP)

- Adding Triviality may create Insecurity.

DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 9



TFTP Overview

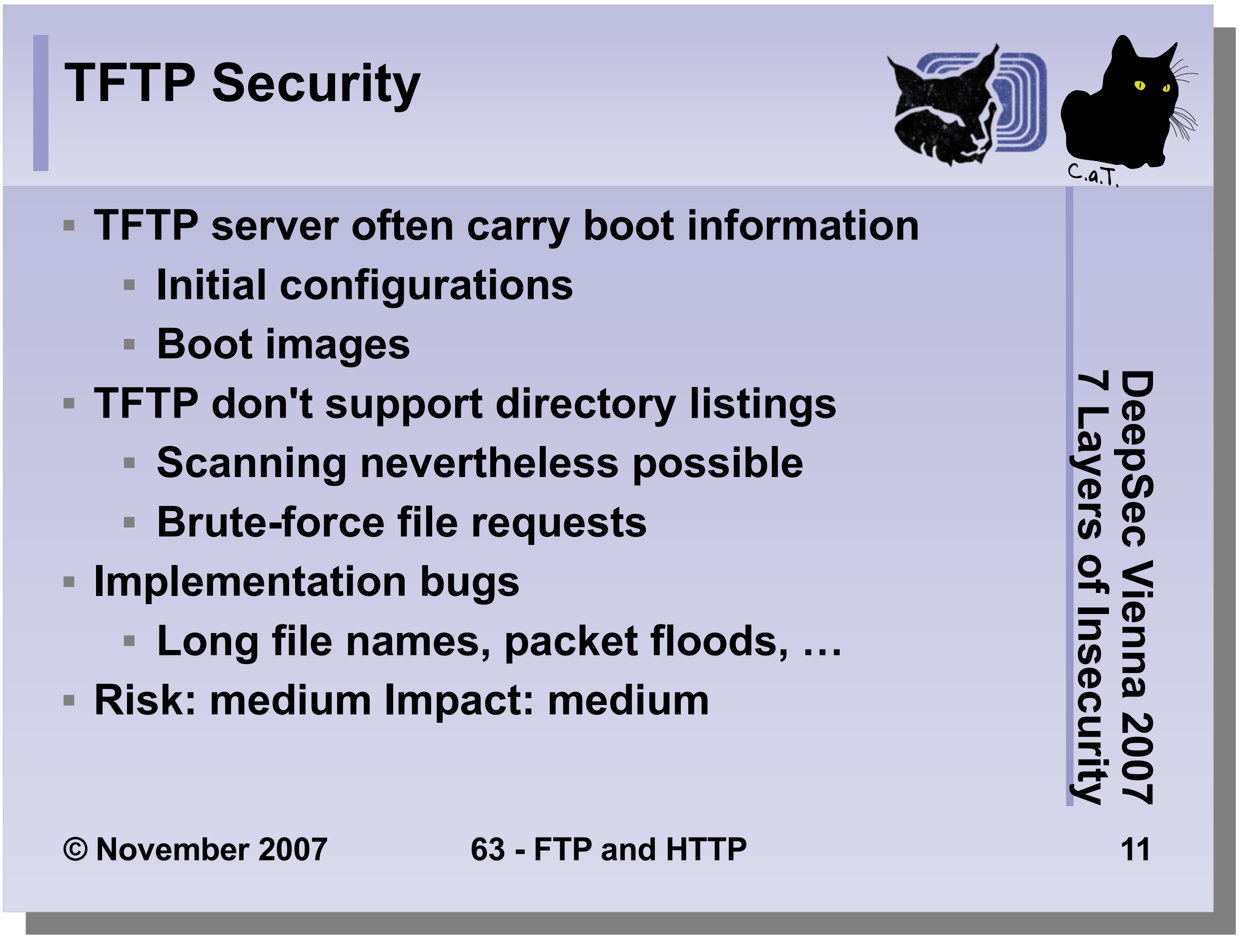
- TFTP uses 69/UDP for data transmission
- TFTP uses no encryption
- TFTP reads/writes files
  - No commands
  - No directory listings
  - No authentication
- TFTP supplies own transport and session
  - Single file transfers (one at a time)
  - Lock-step mode

DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 10

The TFTP lock-step mode ensures that only one UDP packet is “in-flight”. Every packet is acknowledged after reception. Since UDP can't do any window scaling and only one packet is kept in-transit, TFTP is quite slow over high-latency links.

TFTP is described in RFC 783, 906, 1350, 1785, 2347, 2348 and 2349.



TFTP Security

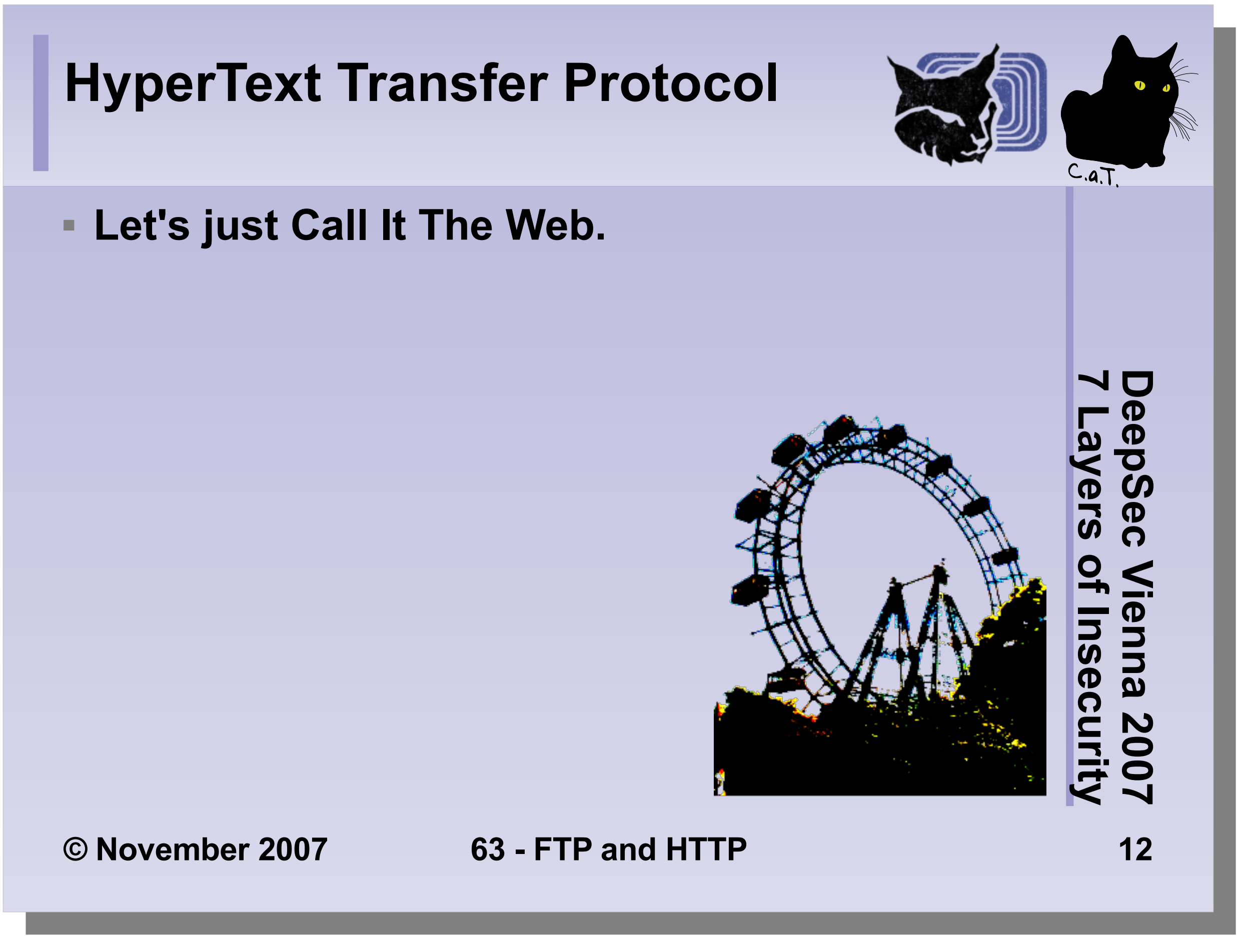
- TFTP server often carry boot information
  - Initial configurations
  - Boot images
- TFTP don't support directory listings
  - Scanning nevertheless possible
  - Brute-force file requests
- Implementation bugs
  - Long file names, packet floods, …
- Risk: medium Impact: medium

DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 11

Mitigation:

- Make sure TFTP is confined to management networks.
- Monitor TFTP access for file harvesting/guessing attacks.
- Disable write access to TFTP servers that are used to serve configurations or firmware.

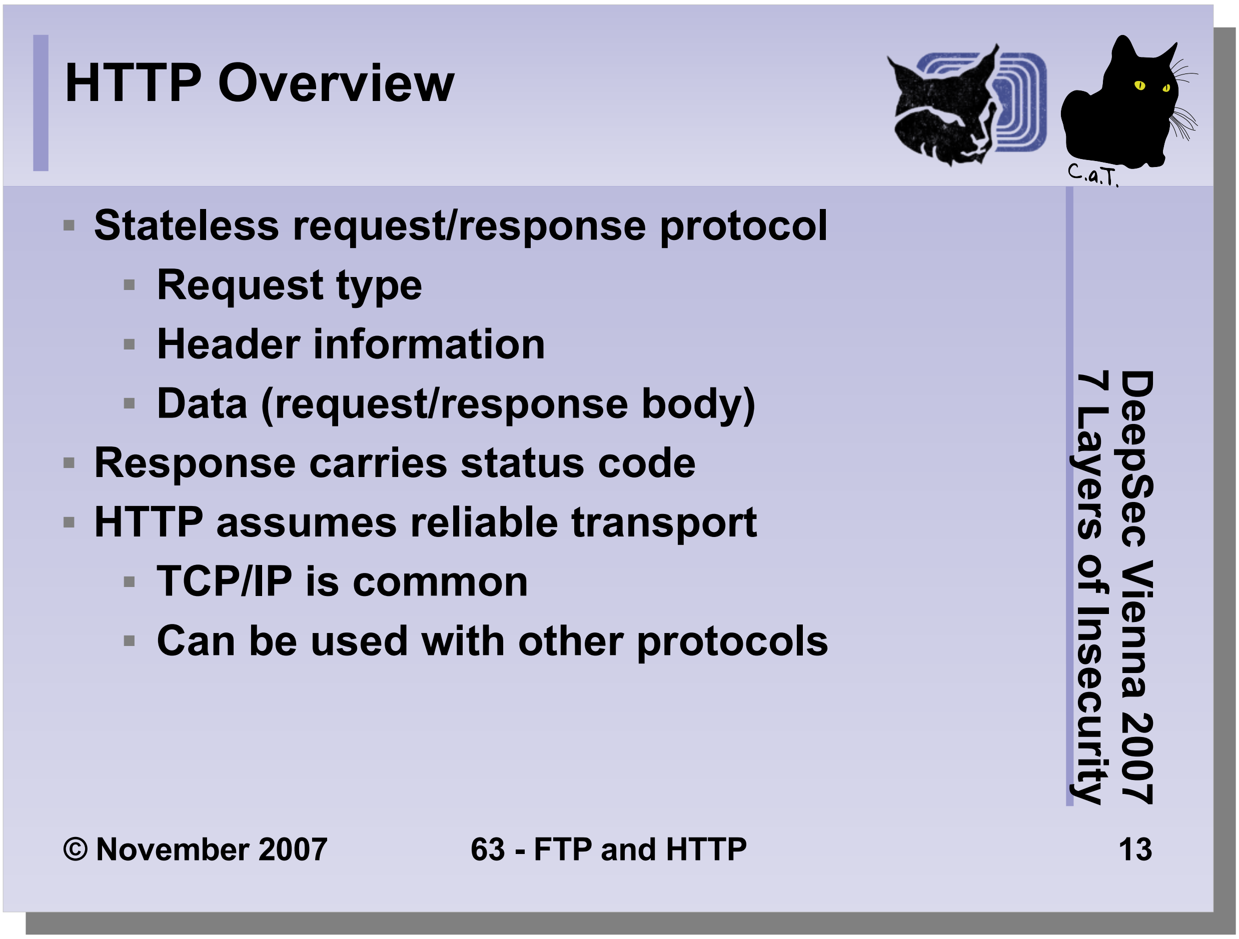


HyperText Transfer Protocol

- Let's just Call It The Web.

DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 12

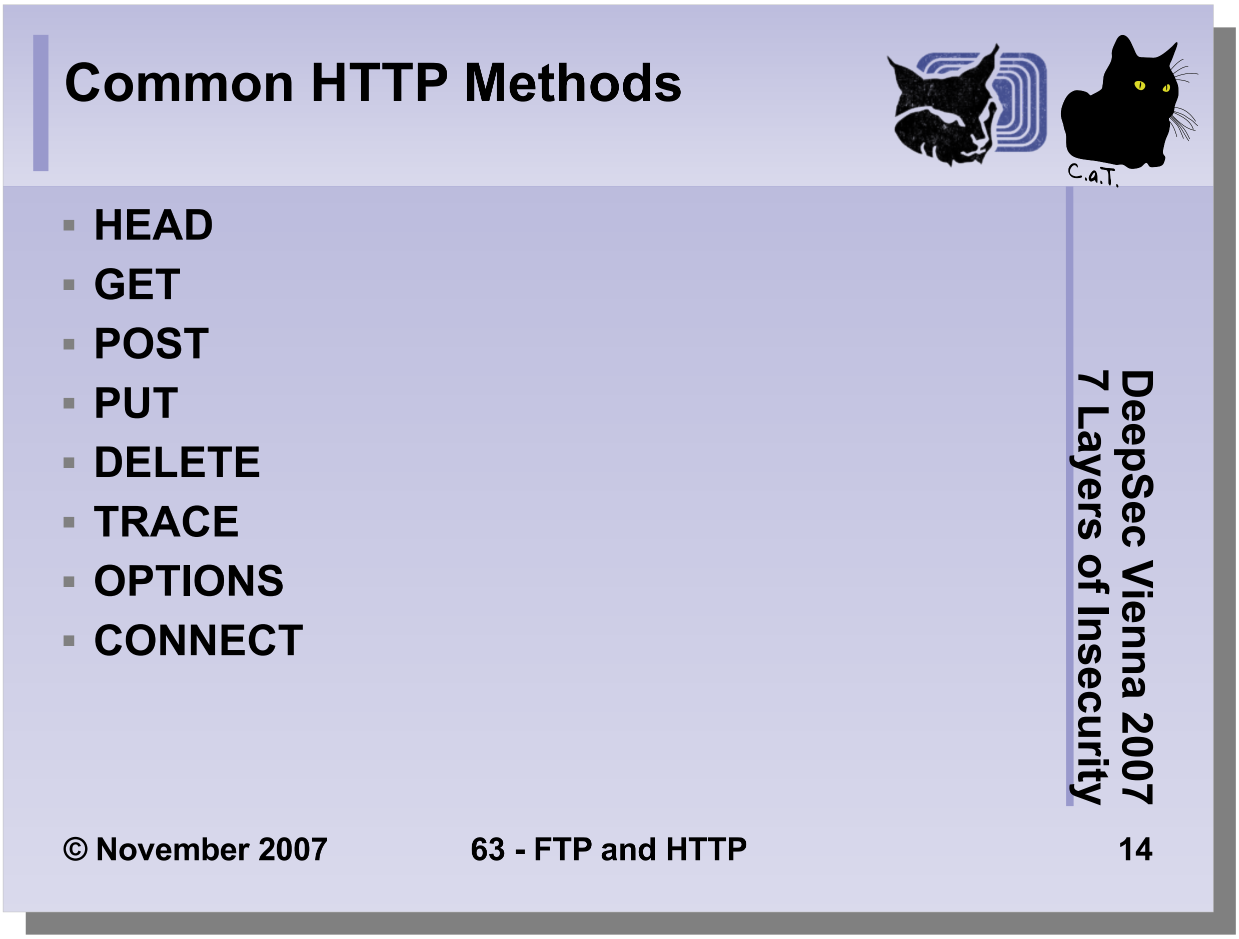


HTTP Overview

- Stateless request/response protocol
  - Request type
  - Header information
  - Data (request/response body)
- Response carries status code
- HTTP assumes reliable transport
  - TCP/IP is common
  - Can be used with other protocols

DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 13



Common HTTP Methods

- HEAD
- GET
- POST
- PUT
- DELETE
- TRACE
- OPTIONS
- CONNECT

DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 14

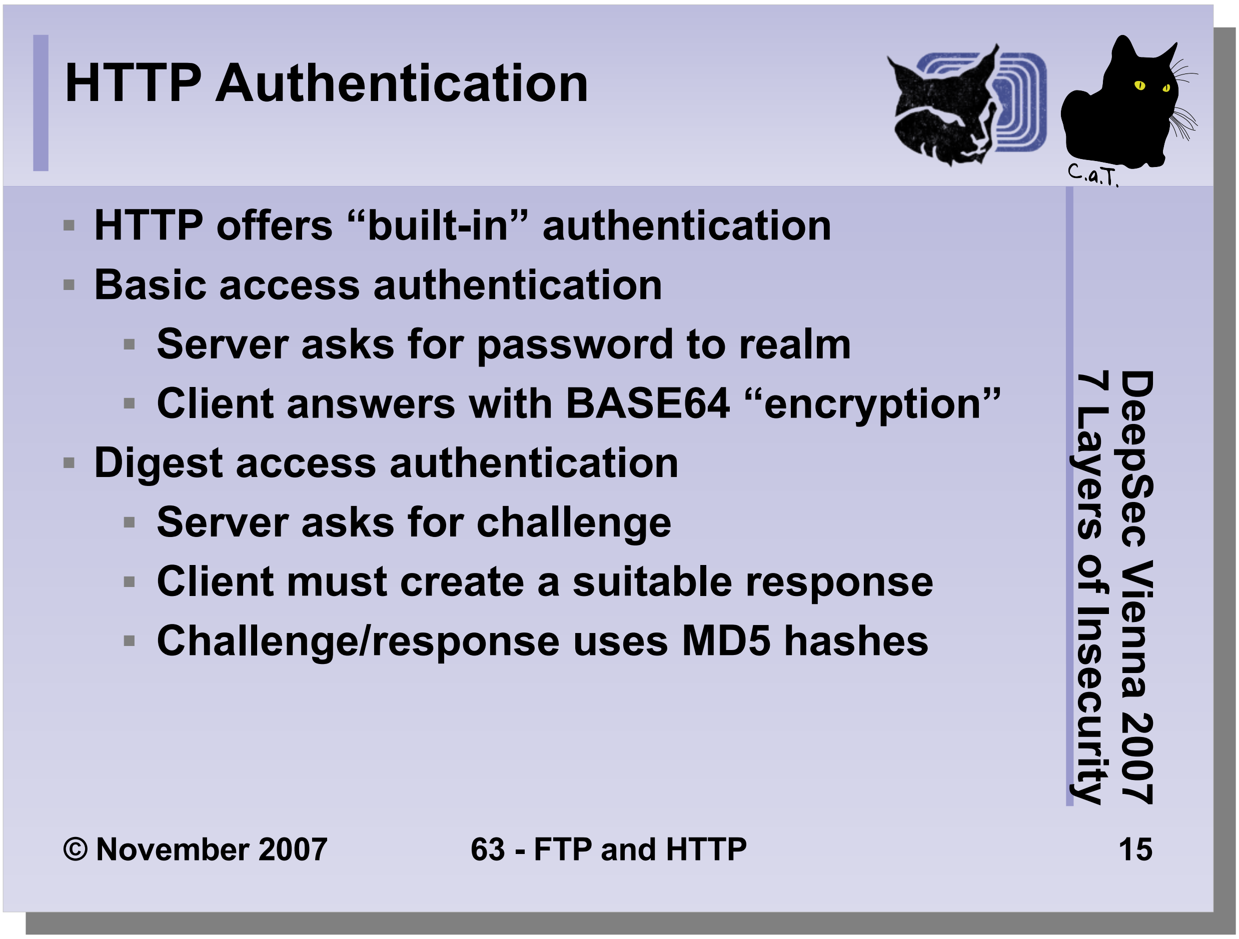
HTTP HEAD, GET and POST are the only methods a web server has to support. These are the most common methods used.

HTTP PUT allows uploads via HTTP and can be configured on the server. This method must be processed by a suitable CGI script that directs uploads. It is usually deactivated. Apache Week has an article about PUT. HTTP DELETE can be used to delete a resource in similar fashion.

HTTP CONNECT establishes a TCP connection to a specific target address. This enables an user to create a transparent TCP/IP tunnel. It is commonly used with proxies and TLS/SSL traffic.

HTTP TRACE echoes the HTTP request back to the client. This is very useful to detect intermediate network devices that alter the HTTP traffic (such as proxies for example).

HTTP OPTIONS tells the server to state its capabilities in terms of HTTP methods.



HTTP Authentication

- HTTP offers “built-in” authentication
- Basic access authentication
  - Server asks for password to realm
  - Client answers with BASE64 “encryption”
- Digest access authentication
  - Server asks for challenge
  - Client must create a suitable response
  - Challenge/response uses MD5 hashes

DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 15

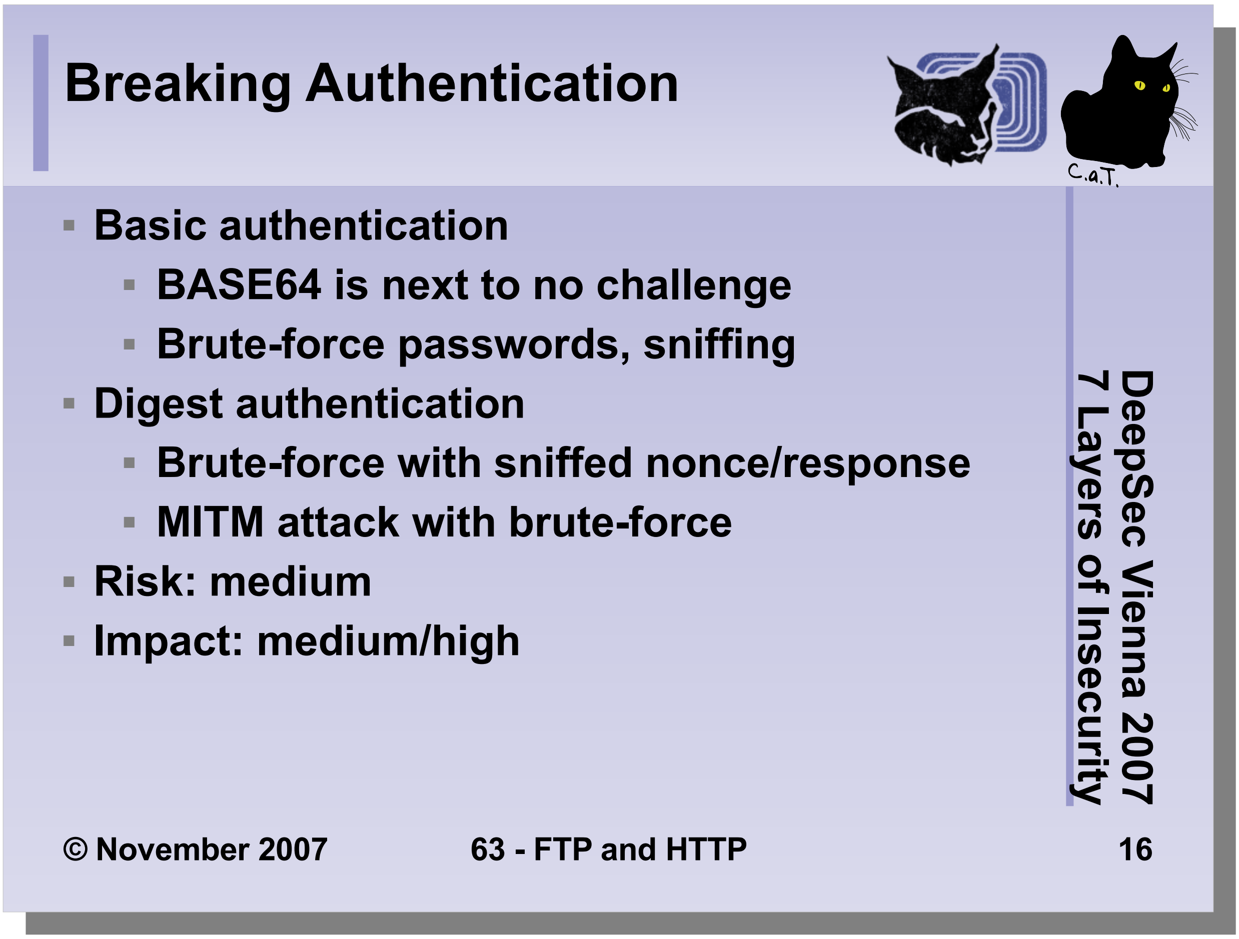
The authentication mechanisms user the *WWW-Authenticate:* header in the challenge. It is usually transmitted with a *401 Authorization Required* status code. The client uses the *Authorization:* header in order to send the response. In basic access authentication the response is the BASE64-encoded password which is practically clear text since this operation can be reversed easily.

Digest access authentication uses MD5 hashes both in challenge and response. A sample session looks like this:

```
WWW-Authenticate: Digest realm="Computer programmers never die, they just get lost in the
processing.", nonce="NyveX4I+BAA=e3daa238e35239e5d6590b76823494a9e73edfc5",
algorithm=MD5, domain="/ http://www.example.net/", qop="auth"
```

```
Authorization: Digest username="user", realm="Computer programmers never die, they just
get lost in the processing.",
nonce="NyveX4I+BAA=e3daa238e35239e5d6590b76823494a9e73edfc5", uri="/admin/",
algorithm=MD5, response="0fe14676230438afaf452d622c43f3d3", qop=auth, nc=00000001,
cnonce="0860596ee0c48d29"
```

The user name and the realm is transmitted in clear, but the other values are hashed. RFC 2617 describes how the values have to calculated. The *nonce* value protects from replay attacks. The *cnonce* value is a protection from chosen plaintext attacks and allows the client to verify the server's challenge. The *Quality Of Protection (qop)* value is optional, it is not proposed in RFC 2069, the original digest authentication method. It is nevertheless possible to attack the hashing algorithm by using brute-force methods and dictionaries.



Breaking Authentication

C.a.T.

- Basic authentication
  - BASE64 is next to no challenge
  - Brute-force passwords, sniffing
- Digest authentication
  - Brute-force with sniffed nonce/response
  - MITM attack with brute-force
- Risk: medium
- Impact: medium/high

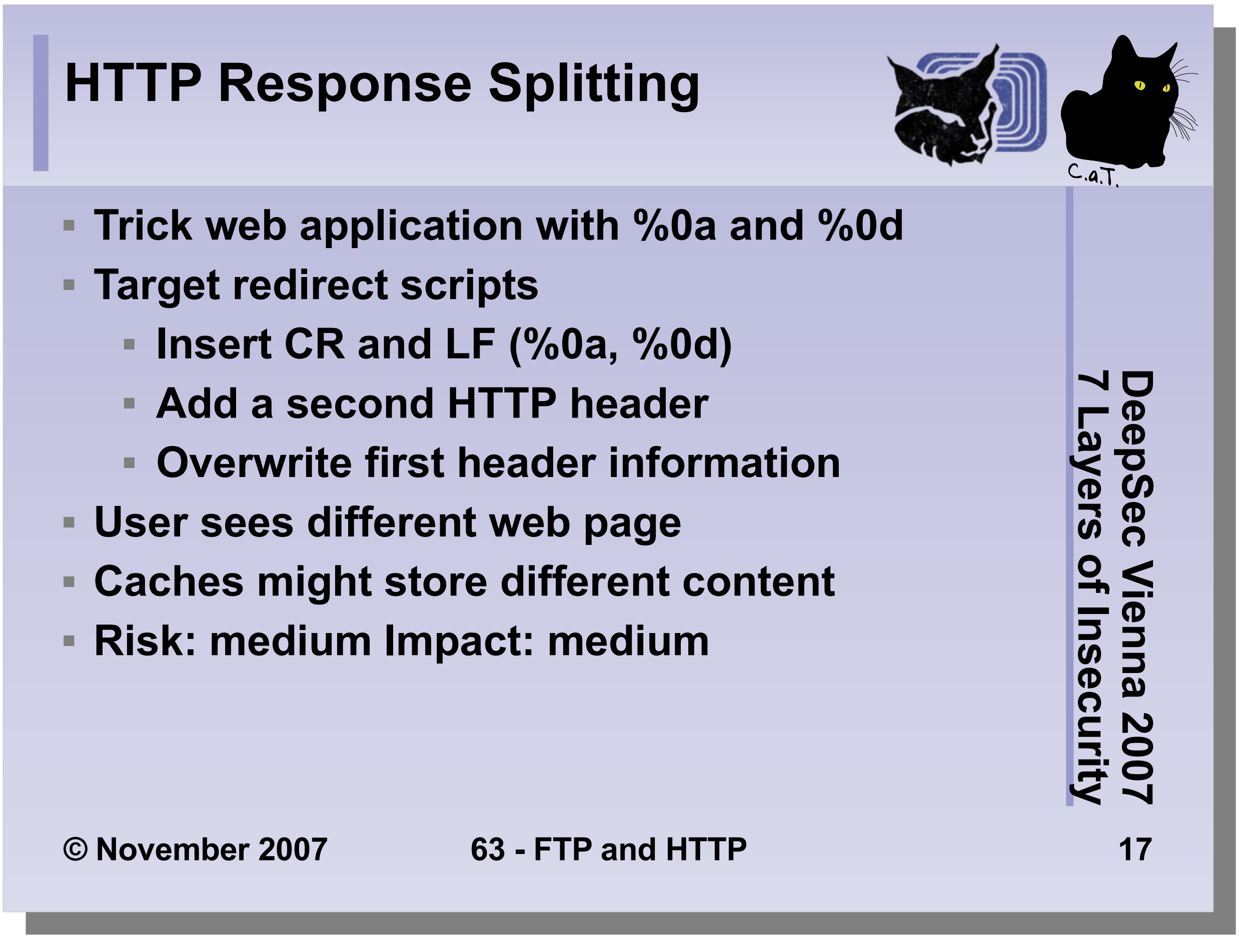
DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 16

While digest authentication is cryptographically weak, its security is sufficient for many applications. The data transmission is not protected, but the username/password pair is not transmitted in clear text. This is the main remedy against the basic authentication method (apart from the replay protection).

Mitigation:

- Avoid HTTP basic authentication.
- Rate limit HTTP digest authentication.
- Combine authentication with SSL/TLS.
- Always use HTTP authentication as one component in a multi-layered security configuration.



HTTP Response Splitting

- Trick web application with %0a and %0d
- Target redirect scripts
  - Insert CR and LF (%0a, %0d)
  - Add a second HTTP header
  - Overwrite first header information
- User sees different web page
- Caches might store different content
- Risk: medium Impact: medium

DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 17

There is an article on the SecuriTeam™ web site with an example of response splitting:

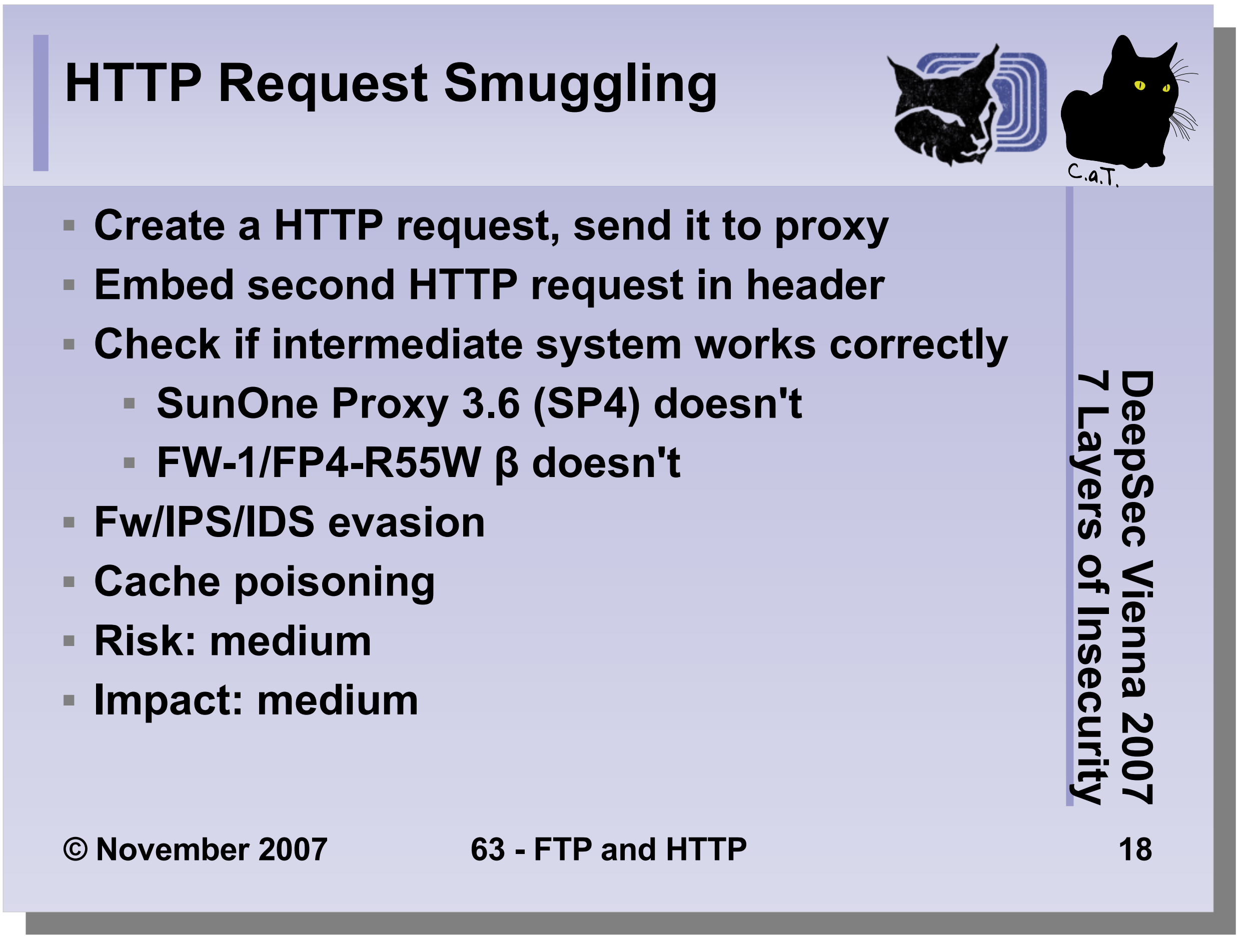
```
GET /~dcrab/redirect.php?page=%0d%0aContent-Type: text/html%0d%0aHTTP/1.1 200
OK%0d%0aContent-Type: text/html%0d%0a%0d%0a%3Chtml%3E%3Cfont
color=red%3Ehey%3C/font%3E%3C/html%3E HTTP/1.1\r\n
Host: icis.digitalparadox.org\r\n
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.7.6) Gecko/20050317
Firefox/1.0.2\r\n
Accept: text/xml,application/xml,application/xhtml
xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5\r\n
Accept-Language: en-us,en;q=0.5\r\n
Accept-Encoding: gzip,deflate\r\n
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7\r\n
Keep-Alive: 300\r\n
Connection: keep-alive\r\n
\r\n
```

Server to User 302 Found Response:

```
HTTP/1.1 302 Found                  [First standard 302 response]
Date: Tue, 12 Apr 2005 22:09:07 GMT
Server: Apache/1.3.29 (Unix) mod_ssl/2.8.16 OpenSSL/0.9.7c
Location:
Content-Type: text/html
HTTP/1.1 200 OK                     [Second new response created by attacker]
Content-Type: text/html

<html><font color=red>hey</font></html>        [Arbitrary input by user shown instead of redirect]
Keep-Alive: timeout=15, max=100
Connection: Keep-Alive
Transfer-Encoding: chunked
Content-Type: text/html

0
```



HTTP Request Smuggling

- Create a HTTP request, send it to proxy
- Embed second HTTP request in header
- Check if intermediate system works correctly
  - SunOne Proxy 3.6 (SP4) doesn't
  - FW-1/FP4-R55W β doesn't
- Fw/IPS/IDS evasion
- Cache poisoning
- Risk: medium
- Impact: medium

DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 18

The second HTTP request can be in an additional header line.

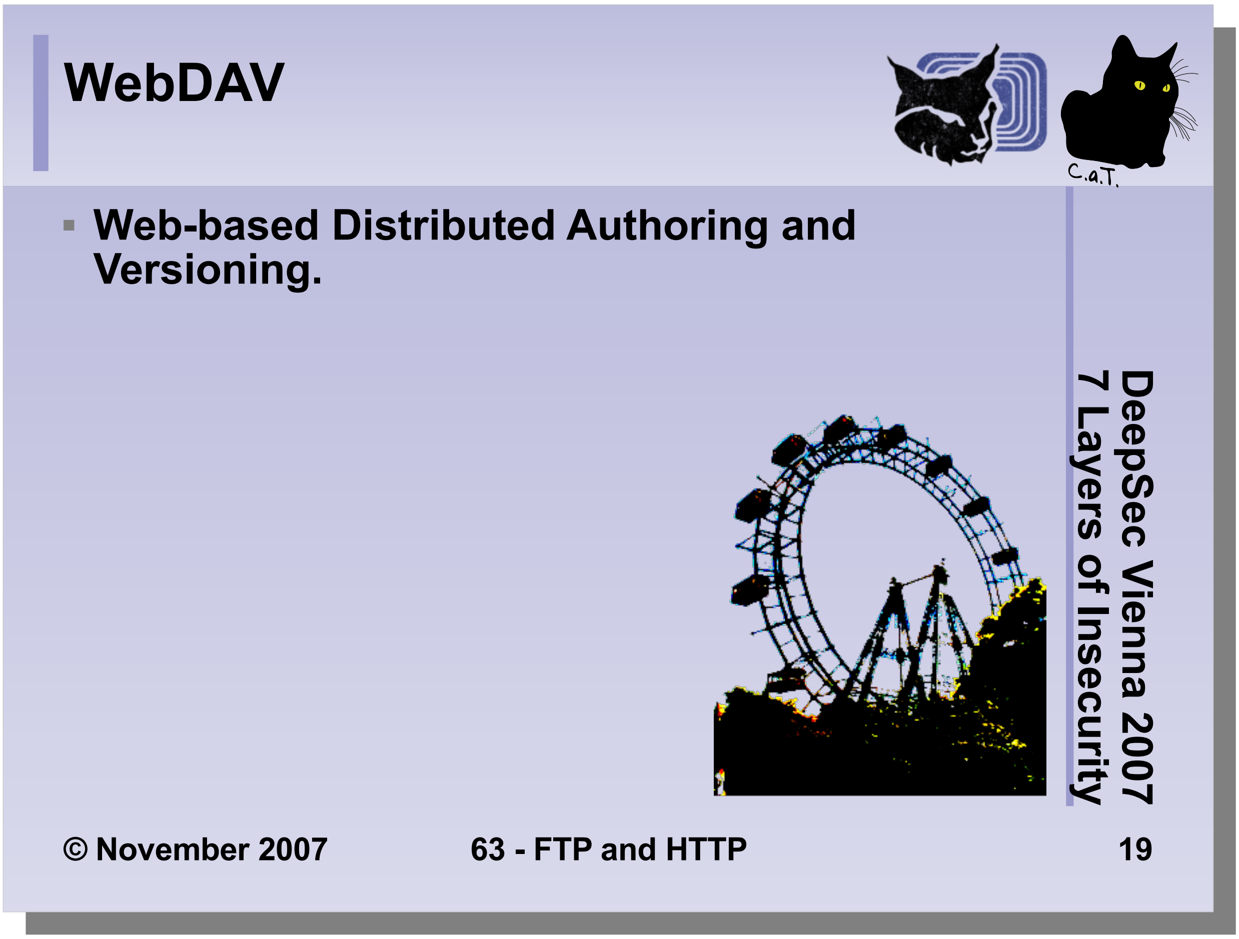
```
POST http://web.example.net/foobar.html HTTP/1.1
Host: web.example.net
Connection: Keep-Alive
Content-Type: application/x-www-form-urlencoded
Content-Length: 0
Content-Length: 44
[CRLF]
GET /poison.html HTTP/1.1
Host: web.example.net
MyHeader: [add space after the "MyHeader:", but no CRLF]
GET http://web.example.net/page_to_poison.html HTTP/1.1
Host: web.example.net
Connection: Keep-Alive
[CRLF]
```

Intermediate HTTP servers (proxies, firewalls, layer 7 filters) may have bugs in their parsers and may slip these requests through to the target server or even other web servers. There's a white paper available from Watchfire that discusses the attack vector.

Mitigation:

• Inspect HTTP headers (application firewalls may be able to do this, the Apache `mod_security` module does inspect HTTP requests and responses).

• Check implementation of intermediate proxies/filters and web servers for handling of HTTP headers.

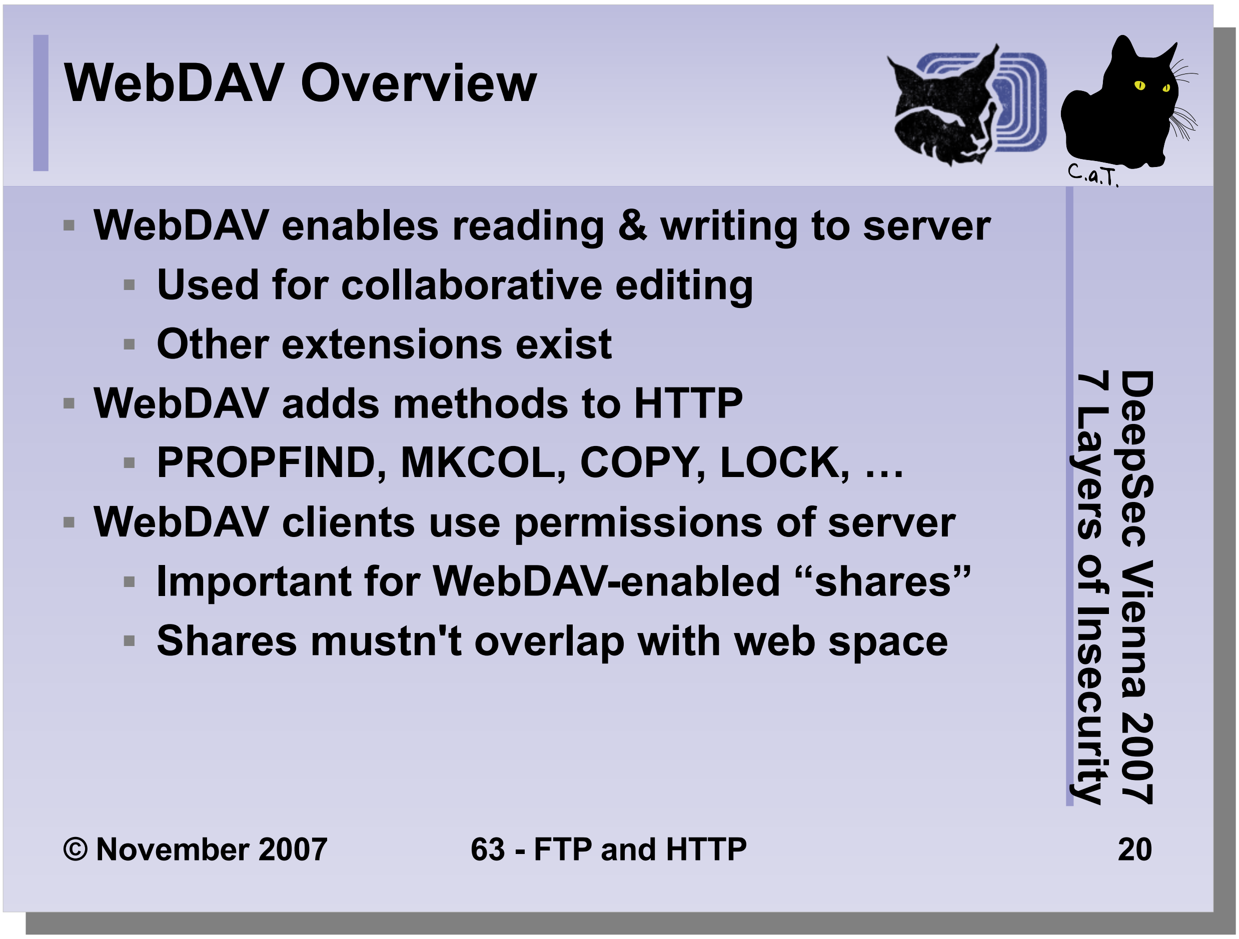


WebDAV

- **Web-based Distributed Authoring and Versioning.**

DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 19

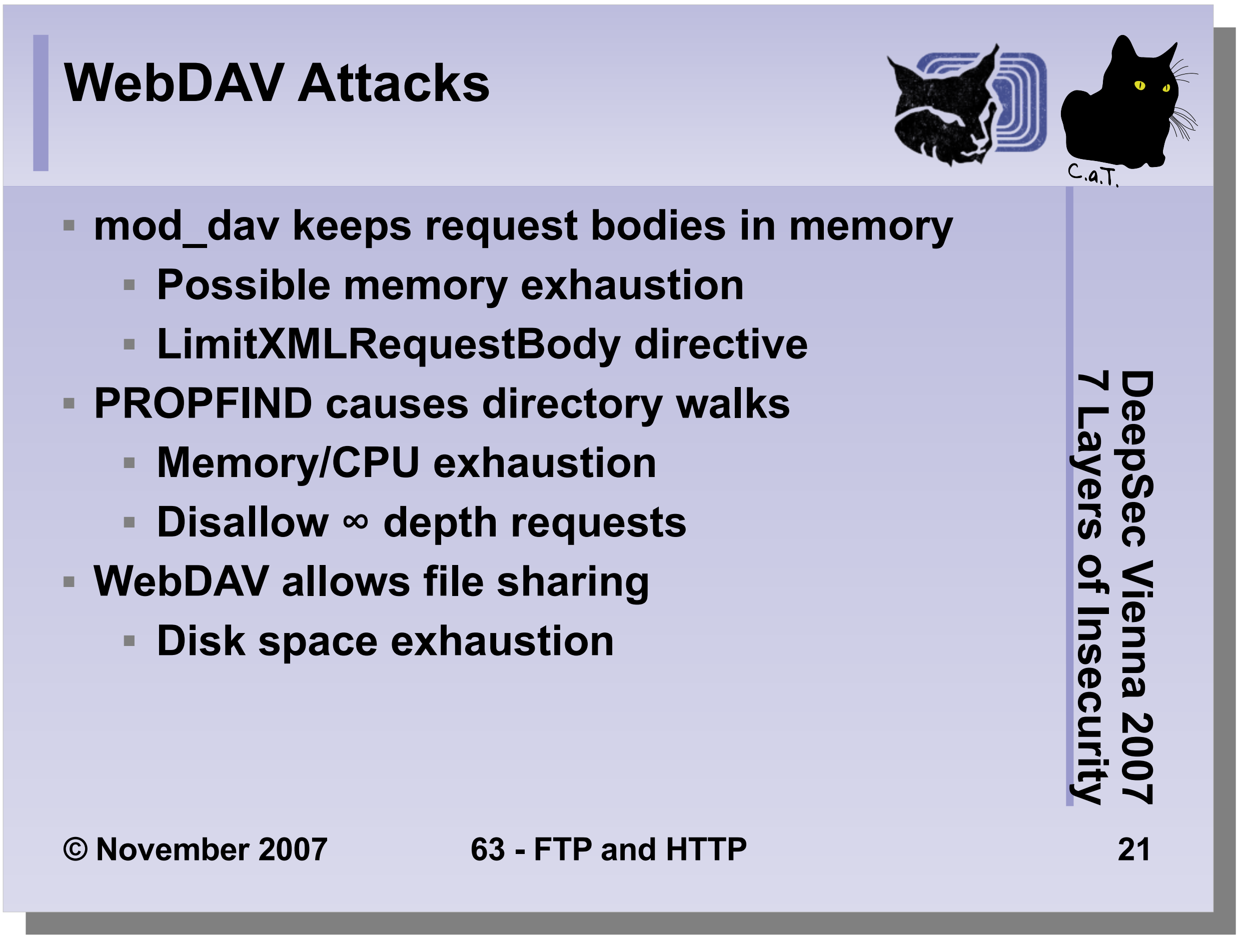


WebDAV Overview

- WebDAV enables reading & writing to server
  - Used for collaborative editing
  - Other extensions exist
- WebDAV adds methods to HTTP
  - PROPFIND, MKCOL, COPY, LOCK, …
- WebDAV clients use permissions of server
  - Important for WebDAV-enabled “shares”
  - Shares mustn't overlap with web space

DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 20



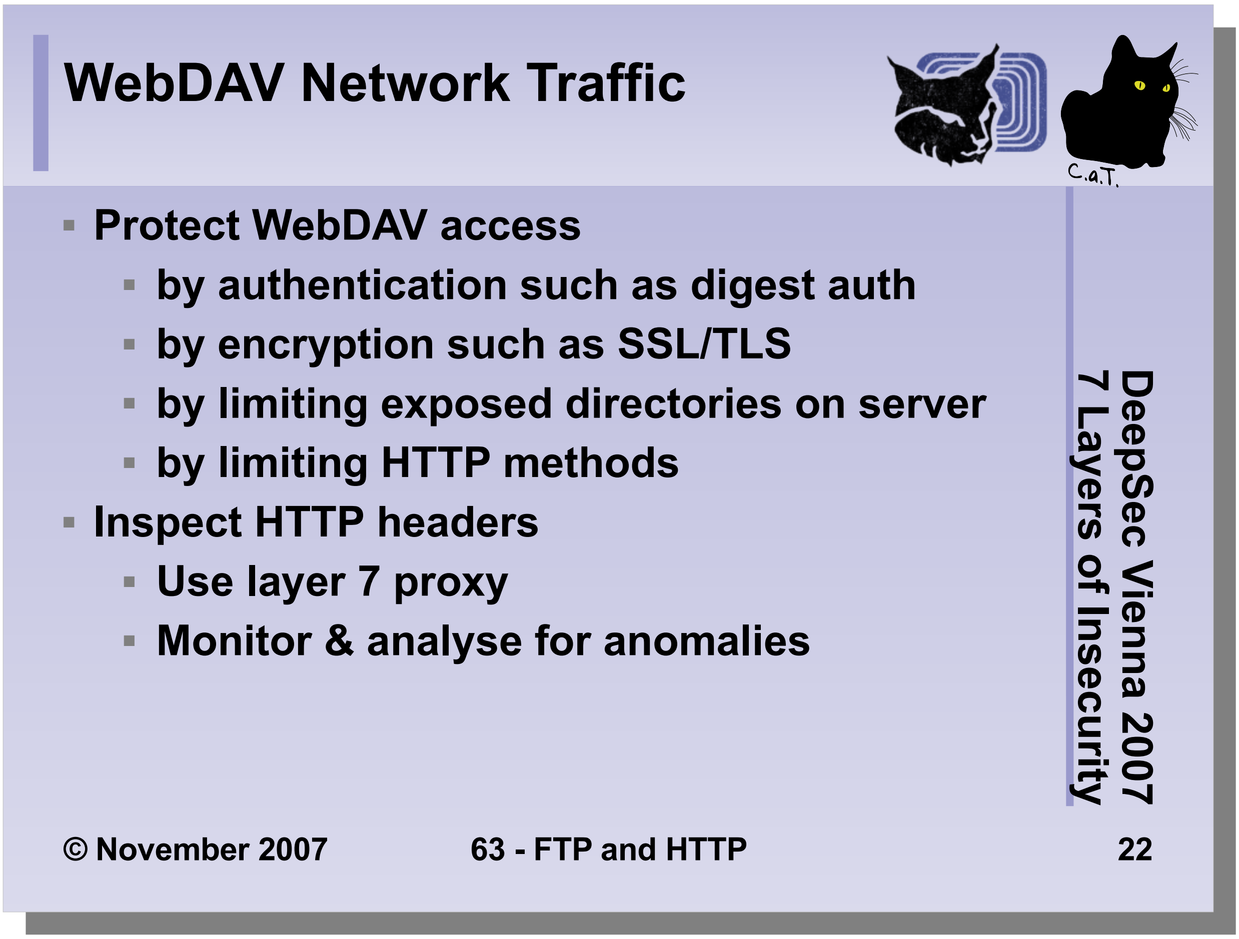
WebDAV Attacks

- mod_dav keeps request bodies in memory
  - Possible memory exhaustion
  - LimitXMLRequestBody directive
- PROPFIND causes directory walks
  - Memory/CPU exhaustion
  - Disallow ∞ depth requests
- WebDAV allows file sharing
  - Disk space exhaustion

DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 21

C.a.T.

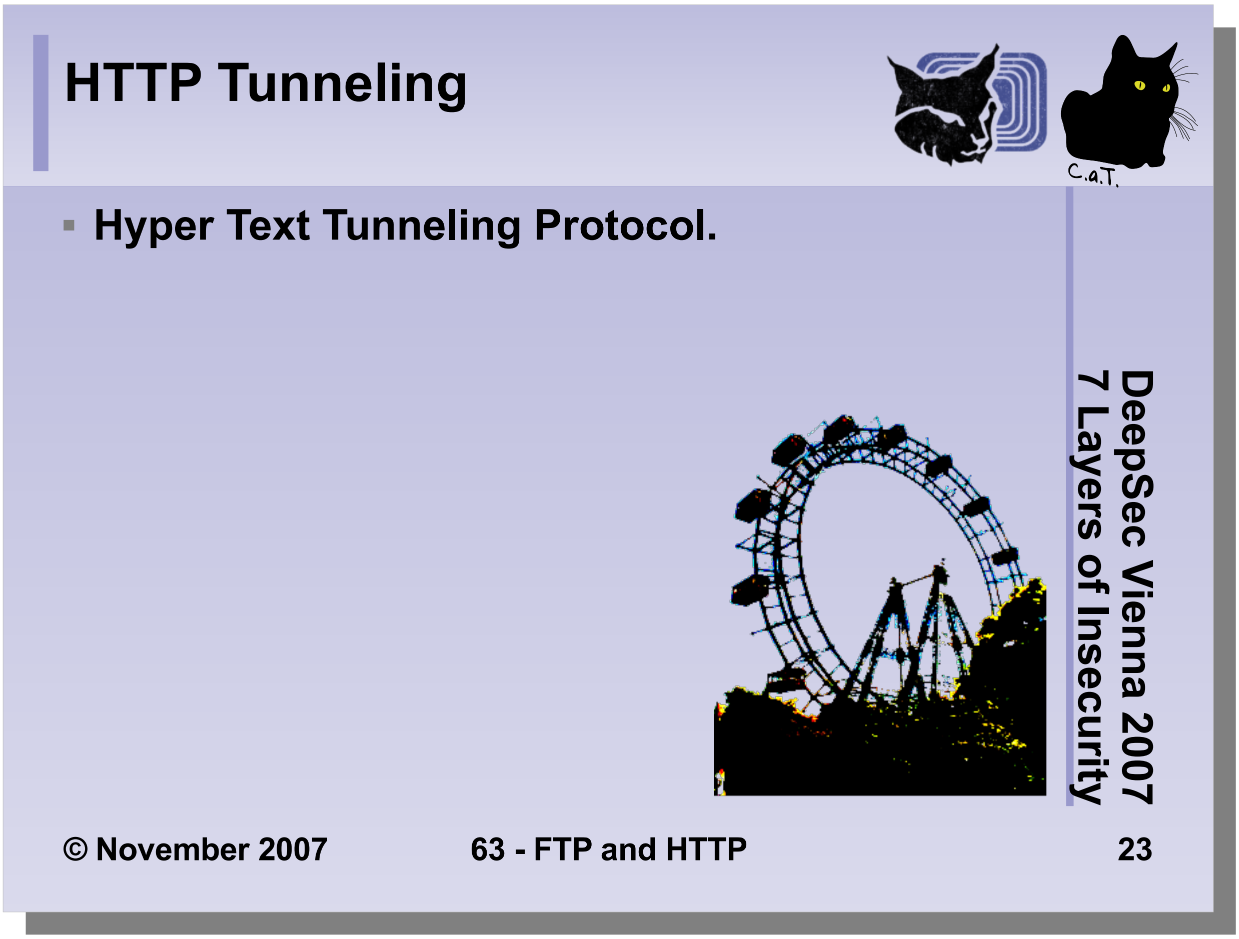


WebDAV Network Traffic

- Protect WebDAV access
  - by authentication such as digest auth
  - by encryption such as SSL/TLS
  - by limiting exposed directories on server
  - by limiting HTTP methods
- Inspect HTTP headers
  - Use layer 7 proxy
  - Monitor & analyse for anomalies

DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 22

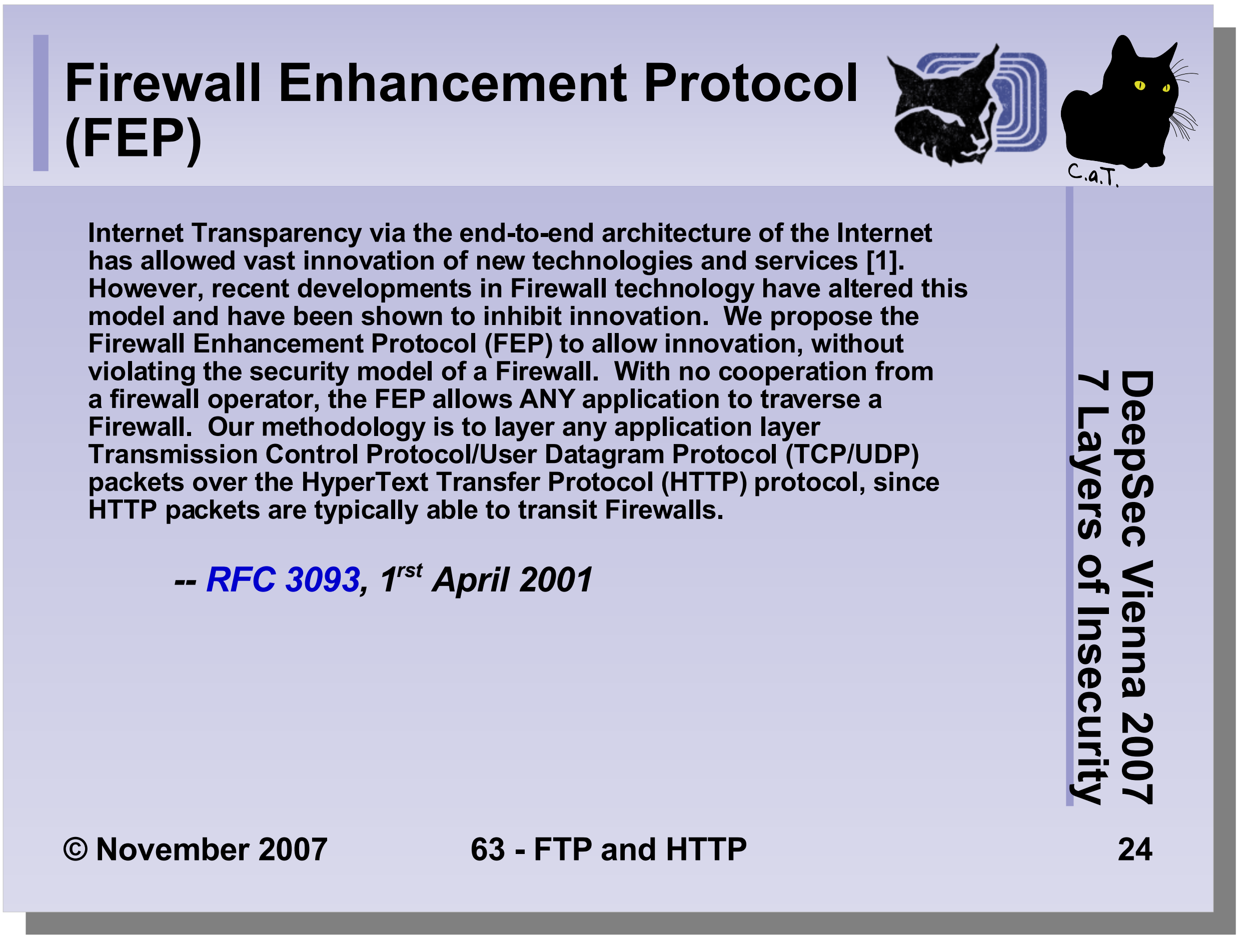


HTTP Tunneling

- Hyper Text Tunneling Protocol.

DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 23



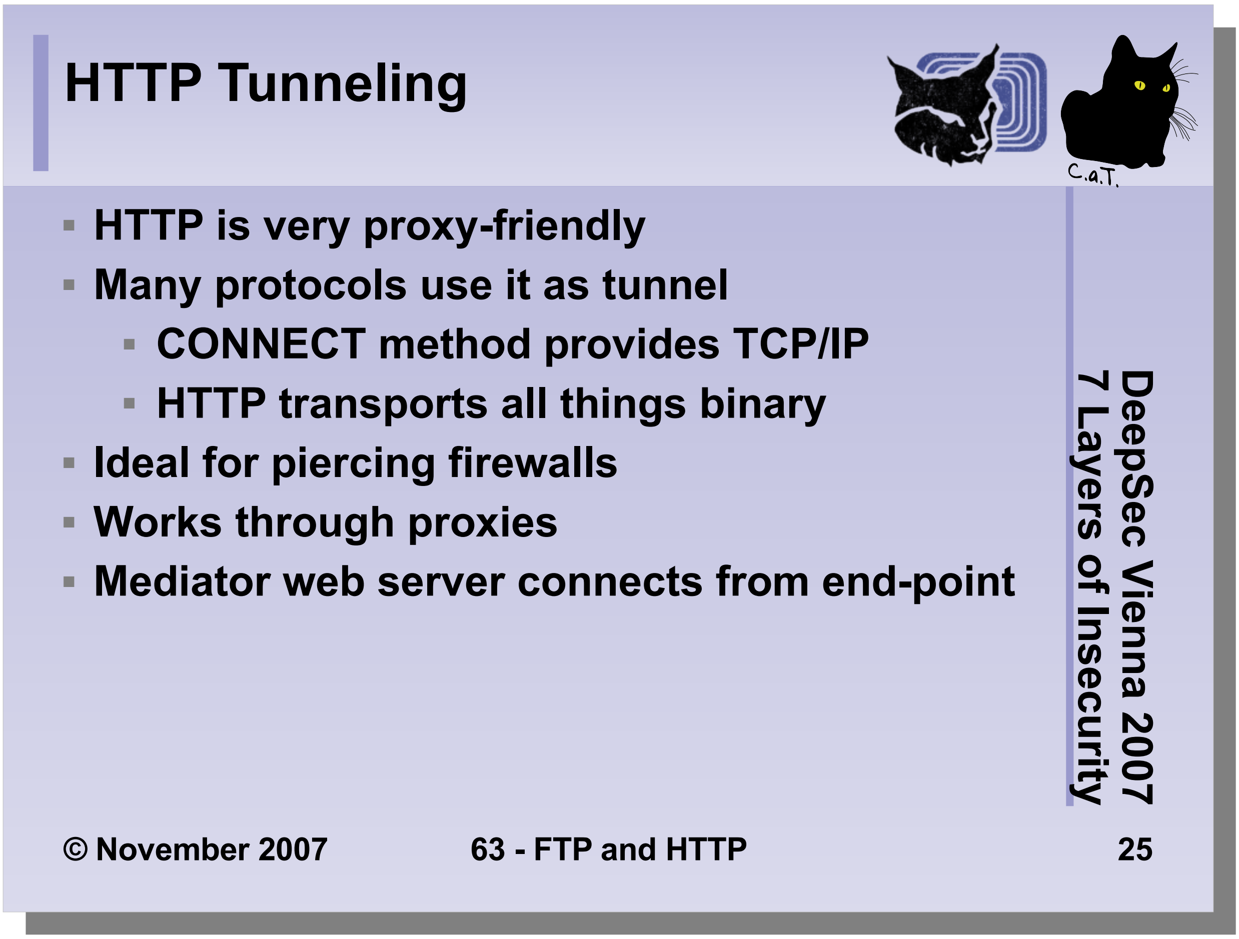
Firewall Enhancement Protocol (FEP)

Internet Transparency via the end-to-end architecture of the Internet has allowed vast innovation of new technologies and services [1]. However, recent developments in Firewall technology have altered this model and have been shown to inhibit innovation. We propose the Firewall Enhancement Protocol (FEP) to allow innovation, without violating the security model of a Firewall. With no cooperation from a firewall operator, the FEP allows ANY application to traverse a Firewall. Our methodology is to layer any application layer Transmission Control Protocol/User Datagram Protocol (TCP/UDP) packets over the HyperText Transfer Protocol (HTTP) protocol, since HTTP packets are typically able to transit Firewalls.

-- RFC 3093, 1rst April 2001

DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 24

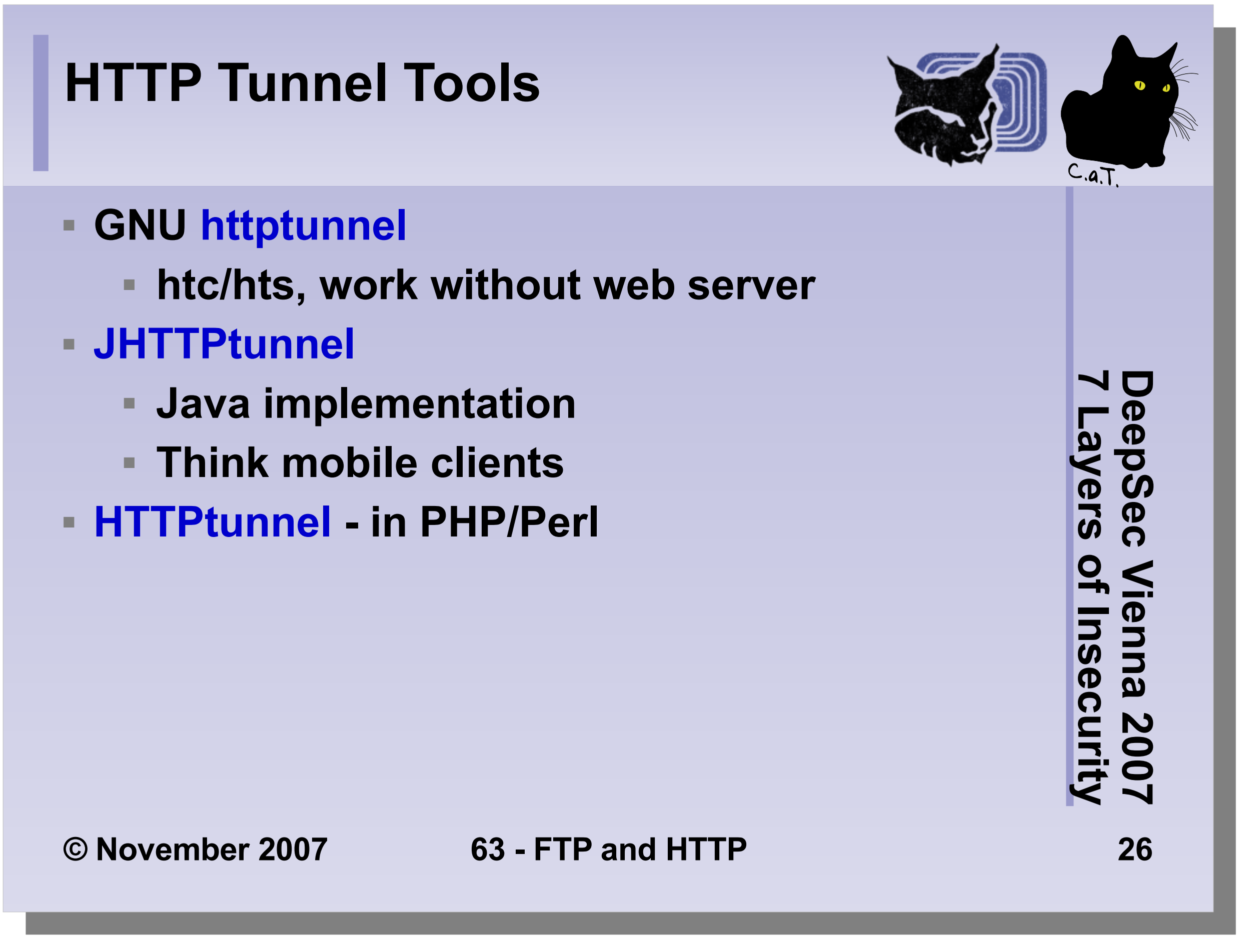


HTTP Tunneling

- HTTP is very proxy-friendly
- Many protocols use it as tunnel
  - CONNECT method provides TCP/IP
  - HTTP transports all things binary
- Ideal for piercing firewalls
- Works through proxies
- Mediator web server connects from end-point

DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 25

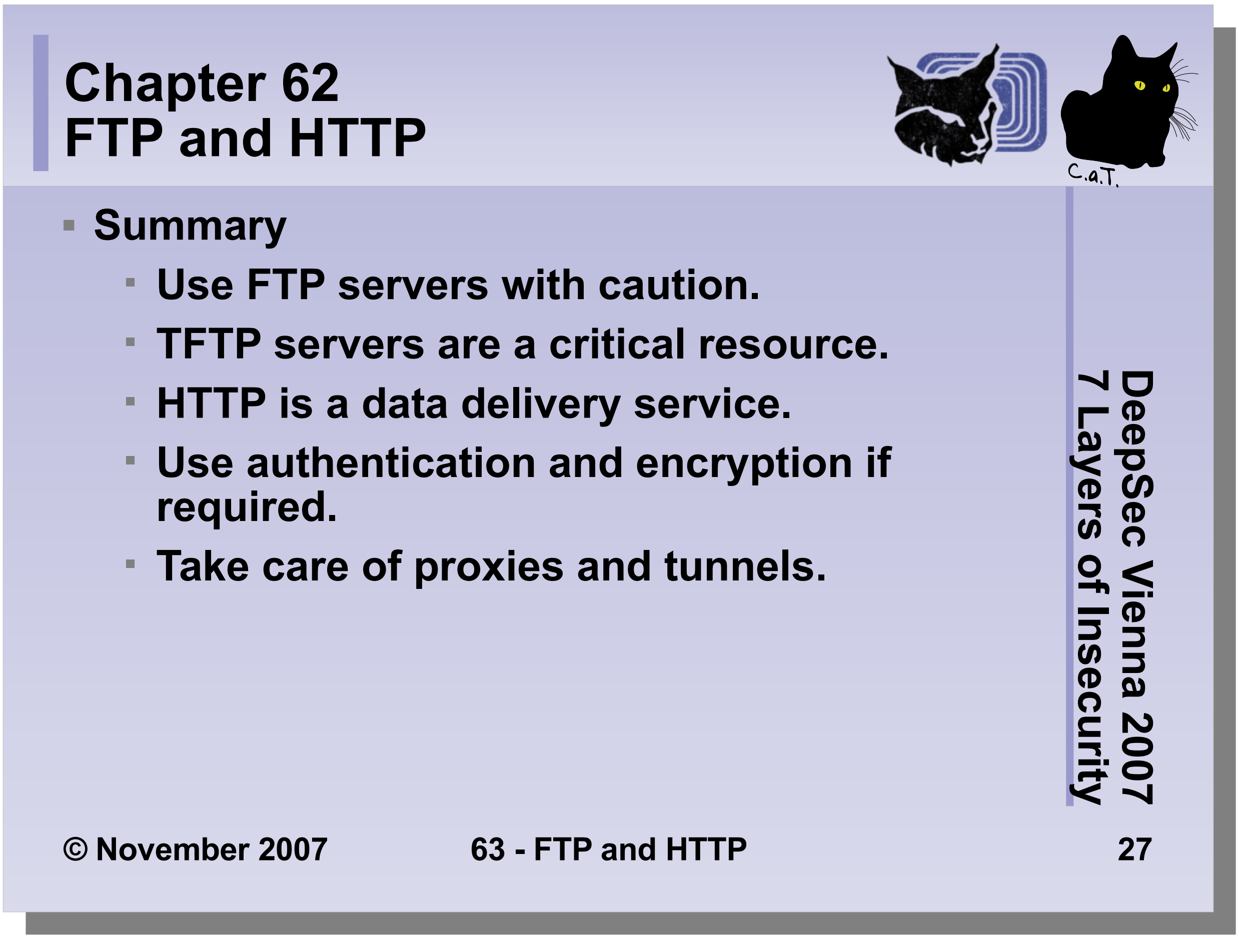


HTTP Tunnel Tools

- GNU httptunnel
  - htc/hts, work without web server
- JHTTPtunnel
  - Java implementation
  - Think mobile clients
- HTTPtunnel - in PHP/Perl

DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 26

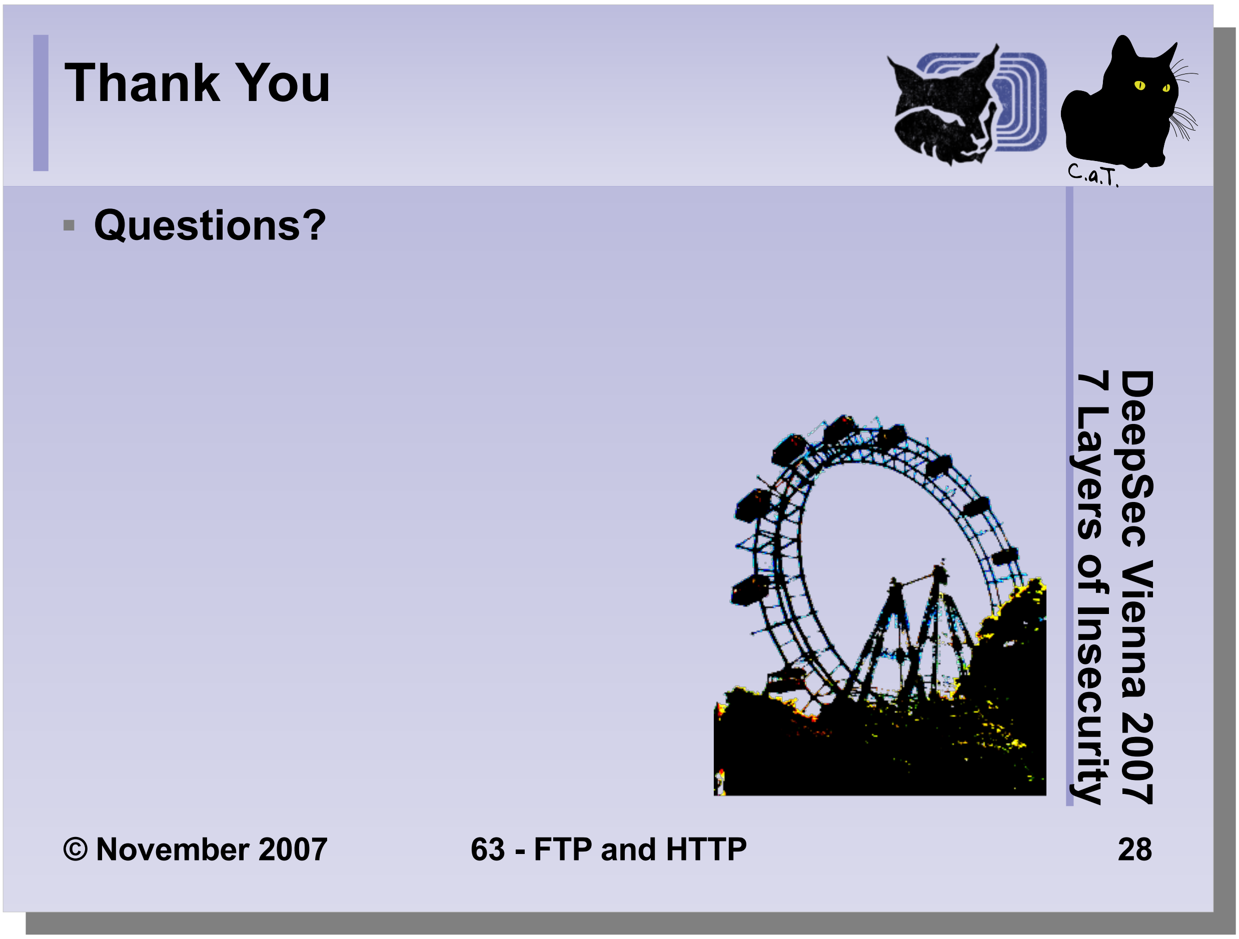


Chapter 62 FTP and HTTP

- Summary
  - Use FTP servers with caution.
  - TFTP servers are a critical resource.
  - HTTP is a data delivery service.
  - Use authentication and encryption if required.
  - Take care of proxies and tunnels.

DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 27



Thank You

- Questions?

DeepSec Vienna 2007
7 Layers of Insecurity

© November 2007 63 - FTP and HTTP 28