Secure Remote Access to an Internal Web Server

Christian Gilmore, David Kormann, & Avi Rubin
AT&T Labs - Research

Internal web

- sensitive corporate data
- private employee data
- ability to change payroll data
- home phone numbers
- business plans
- manage savings plan account
- webmail, phonemail, voicemail

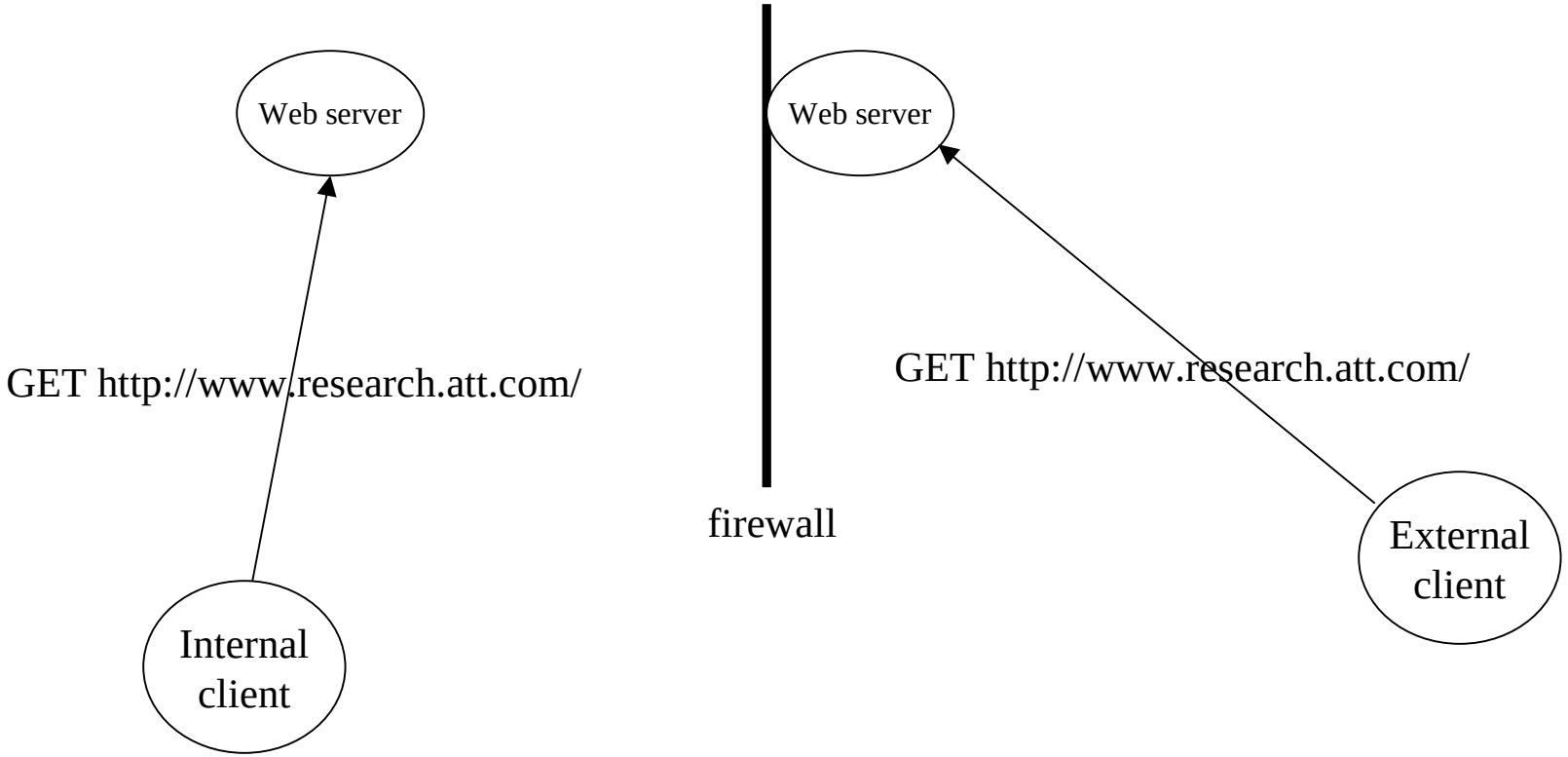
External web

- corporate information
- employment information
- investment information
- product information/purchase
- press releases
- tons of P.R.

At AT&T Labs - Research

- packet-based firewall
- no access to internal web from outside
- www.research.att.com = akalice (inside)
  akpublic (outside)
- no safe way for users outside to access inside web
- plethora of useful stuff inside
  - home phone numbers
  - business plans
  - payroll/benefit selections

Different view inside and out



Without absent

- use securenet key to telnet inside
  - use lynx to access internal web from inside machine
- drawbacks
  - sensitive data travels in clear to remote site
  - No support for snazzy browser features
    - no graphical user interface
    - no java/javascript/ActiveX
    - no multimedia
    - no helper apps
    - not the **real** web experience

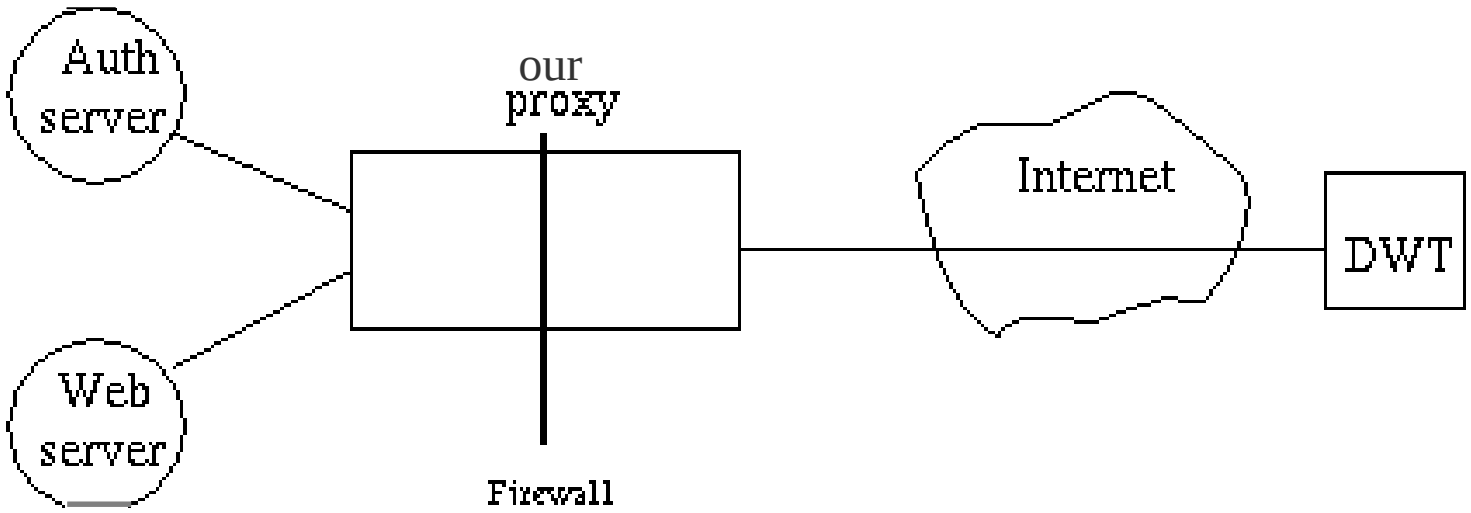
Assumptions

- user has legitimate access to internal web site
- user is at a dumb web terminal (DWT)
- DWT is SSL enabled
- user may not be able to change proxy settings
- path between DWT and home site is hostile
- no changes allowed to infrastructure
  - no open, reserved port in firewall
  - no change to web server

Why not use VPN?

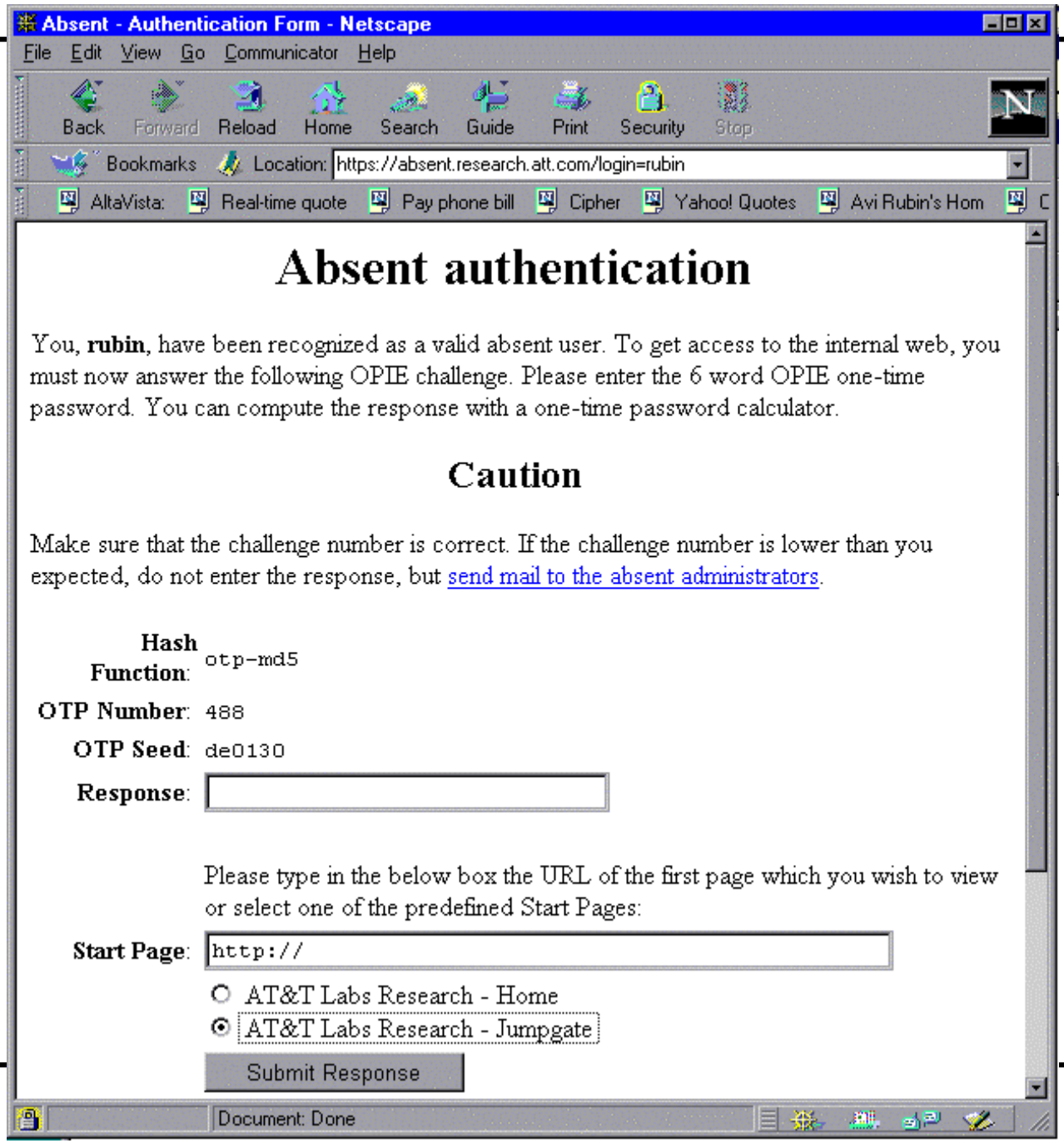
- We assume a dumb terminal
  - no client smarts
    - don’t have to update all clients
  - finer grained control of accesses
- Cost of VPN
  - cost of system
  - cost of administration
- Security
  - few systems w/complete source code released
- Practical consideration
  - too much hassle to get sys admins to install VPN

Absent Architecture



User authentication

- use one-time password scheme
  - we chose OPIE (S/KEY) based on hash chaining
- before leaving, user initializes password *pw*
- Authentication consists of a challenge and a response over SSL connection
- Server verifies response


Absent - Authentication Form - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Guide Print Security Stop
Bookmarks Location: https://absent.research.att.com/login=rubin
AltaVista: Real-time quote Pay phone bill Cipher Yahoo! Quotes Avi Rubin's Hom
Absent authentication
You, rubin, have been recognized as a valid absent user. To get access to the internal web, you must now answer the following OPIE challenge. Please enter the 6 word OPIE one-time password. You can compute the response with a one-time password calculator.
Caution
Make sure that the challenge number is correct. If the challenge number is lower than you expected, do not enter the response, but send mail to the absent administrators.
Hash Function: otp-md5
OTP Number: 488
OTP Seed: de0130
Response:
Please type in the below box the URL of the first page which you wish to view or select one of the predefined Start Pages:
Start Page: http://
AT&T Labs Research - Home
AT&T Labs Research - Jumpgate
Submit Response
Document: Done

One-time passwords

- Once a password is used, it is useless in the future.
- Any OTP > n, should not be derivable from passwords *1* through *n*.
- Authentication server must be able to verify that OTP is correct.
- Avoid storing large databases of OTP for each user on auth. server
- Must have option to use on untrusted machine or terminal

OPIE

- OTP's derived from one secret
- No secrets on server
- Mechanism for use with untrusted host or dumb terminal
- Cheap, and easy to administer
- Requires *secure* initialization phase
- Based on one-way hash function

One-way Hash Functions

- One-way hash function

  A function, f, where f(x) = y such that

  - Given y, it is infeasible to compute x
  - Given x and y, it is infeasible to find an x' such that $x \neq x'$ and f(x') = y.
  - y has a fixed length

- E.g. Md5
  - output is always 128 bits
  - publicly available (source code)

OPIE

- Initialization - on secure machine
  - user enters password, pw and n
  - User computes:

    $pw_n = f(f(f(...f(pw))))...)$ n times

    where f is a one-way function
  - User sends pw_n to server
  - Server stores pw_n

Opieinit (cont.)

Client #1 *Server*

pw_0 = user password

$pw_1 = f(pw_0)$

$pw_2 = f(pw_1)$

$pw_3 = f(pw_2)$

$pw_4 = f(pw_3)$

...

$pw_n = f(pw_{n-1})$ -----> client #1,$pw_n = f(pw_{n-1})$

OPIE (cont.)

- To authenticate
  - Server knows $f^n(pw)$
  - Client known pw

  Client -> Server : “I wish to authenticate”

  Server -> Client : n

  Client computes $f^{n-1}(pw)$

  Client -> Server : $f^{n-1}(pw)$

  Server computes $f(f^{n-1}(pw))$

Example OPIE one-time passwords

464: DAN MAP FAIR CLAN HOVE BOO
465: TOP JAM CULT MOLT LAWN SEEN
466: SLID RODE JIG SLUG HUE COIN
467: SWAG IT AMES ELI WAST TIP
468: TIP SMOG EGAN MAP VIEW AJAR
469: EEL STAG SKIT AID DONE SLY
470: SKI APT BAND KIND BAD AD
471: BOB FREY HIDE FUSS GARY LAP
472: FIRE HUCK MIND DUE REEL KUDO
473: AGO AWRY WIT HAY BULK RAW
474: TIM KNOT KEY HASH FUM PAP
475: LYNN FIVE LILY JUG FARM AVON
476: COL COOT COLD FOOL NAGY MESH
477: NOON CHEN NAIL GAB SEEM GALA

MAC

- Message authentication code
- Very useful for Internet security protocols
- Efficient to compute
- function of a key and a message
- cannot find collisions
- cannot produce without the key

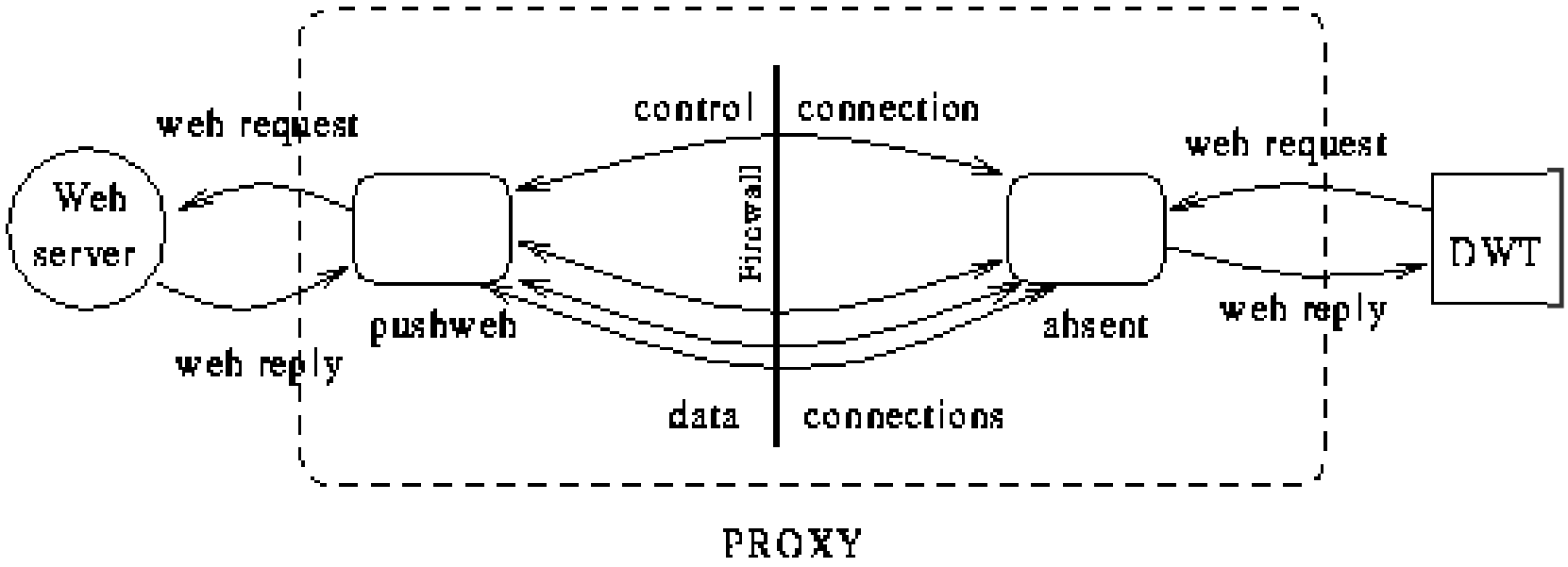
Authentication in Absent

- After user authenticates
  - random key, *k*, added to user table for each user
  - *k* is used to compute a MAC (HMAC) of each URL
  - MAC is included in rewritten URL
  - user entry expires every 20 minutes
- When URL received by proxy
  - check if user registered
  - check if key is fresh
  - verify length of URL and MAC

How absent works

- initial request from DWT
- SSL connection established (more later)
- proxy sends authentication challenge form
- user fills in response and submits
- authentication is verified
- URL request from DWT
- page served with URLs rewritten

Proxy in detail (after authentication)



Rewriting URLs

- Take

  Crowds home

http://www.research.att.com/crowds

converted to

https://absent.research.att.com/geturl=user/
2b5db86c1f6e/http://www.research.att.com/crowds

- first part is used to point DWT to absent port 443
- next: cmd=user (login, geturl, logout, OTP_resp)
- 2b represents hex of length of original URL
- 5db86c1f6e represents MAC

CGI scripts

- Take CGI program count.cgi and the URL
  XXX/http://www.research.att.com/~alice/cgi-bin/reg.cgi

which appears in a GET method form

- The value entered in form is returned in URL

XXX/http://www.research.att.com/~alice/cgi-bin/reg.cgi&name=bob

- No way server can know &name=bob in advance
- So, everything between (not including) XXX/ and & is MACed

What if absent is compromised?

- denial of service possible
- can get pushweb to open data connections
- cannot read SSL traffic
- cannot issue valid web requests
- attacker sees secret MAC key used by absent
- recovery:
  - generate new MAC key
  - probably reboot server
- no big deal, really

What if pushweb is compromised?

- consequences
  - unlimited access to internal web
  - potential to put in trojan horse server to remove authentication of future requests
  - potential to compromise other internal machines, data and services
- precaution
  - don’t run any other services on pushweb
  - proxy server runs as nobody
  - code review to avoid buffer overflow and other common problems
  - log, log, log and monitor the logs

Issues

- Other issues:
  - Cache-control: no-cache, etc.
  - randomness (randlib by Jack Lacy)
  - all sorts of networking issues (resource pooling)
- limitations:
  - policy issue: SSL over SSL
  - performance
  - scale
  - mobile code issues (embedded URLs)
  - ease of use (users hate one-time passwords)

Current Status

- Fully functional system
- In use at AT&T Labs
- Obtained release for all the code
- Code is freely available on the Internet

http://www.research.att.com/projects/absent/