ITUG SUMMIT 06

15-18 OCTOBER 2006
San Jose, California USA

Denial of Service

And what can be done about it

Carl Weber
GreenHouse Software & Consulting
18. October 2006

P-1-U, Marriott Salon 1, 10:30 – 11:30
ITUG, San Jose

Brief Intro

- 1978 start as an analyst with Tandem, Germany
- 1979 first cryptographic program on \DUES
  (causing trouble with the US ...)
- 1985 specialization in SAFEGUARD & Security
- 1988 made PGP available on \DORT
  (causing massive trouble with the US ...)
- 1989 start of system evaluations (NCSC, GISA)
- 1993 successful end of evaluations (C2, F2/F7, Q3)

Brief Intro

- 1994 start of GreenHouse as an Alliance Partner
- NSAA member

In other words:

Since 28+ years on the best platform available!

Why is Tandem chosen?

- Availability
  - Hardware (mirrored volumes, lock step CPUs)
  - Software (process pairs)
  - in short: NonStop (hardware only?)
- Scalability
  - auto load balancing (e.g. PATHWAY)
- Security
  - GUARDIAN (pretty good basics)
  - SAFEGUARD (additional granularity and features)

Why is Tandem chosen?

- Expandability
  add of CPU cycles, disk space, comm lines, etc.
- Integrity
  software (basically TMF & RDF) and hardware (CRC)
- Reliability
  ... just: Tandem!

- ... and years ago for many years:
  Best service in the world!

What is Security?

- Confidentiality of data and services
  - GUARDIAN
    pretty basic, but may be sufficient
  - SAFEGUARD
    NOT more security, but better granularity, and more functionality
  - Cryptographic services
    Atalla products, cryptographic procedures, etc.

What is Security?

- Availability of data and services
  - NonStop
  - 99,99 999 % ...

Can we relax?

I’m afraid – but: No!

Confidentiality

- Strong (Dutch) mechanism

Confidentiality

- Strong (German) mechanism

Confidentiality

- Needed mechanism

Confidentiality

You need a strong **active** mechanism

AND

You have to make the **correct use of it**!

You have to **invest** into it!

Confidentiality

- ... needs permanent attention
- ... is a dynamic process
- ... is an investment
- No risk – no fun ... ? - No costs – no security... !!!

Availability

- Resources
  - Hardware (mirrored volumes, lock step CPUs, etc.)
    we talk about 99,99 999 % uptime ...
    (... nice marketing ...)

  - Software (process pairs, TMF, RDF, etc.)
    Interestingly enough: NO promises for uptime here!
    (did you count the number of IPMs, needing a cold load?)

Availability

- Hardware
  - not easy to manipulate by the normal system user
  - assumed to be OK - or NOT OK: A 'digital' decision
  - 1st computer axiom: Hardware goes wrong, ...!

Availability

- Software
  - easy (!) to manipulate by any interactive system user
  - can possibly be manipulated even by an application only user
  - 1st computer axiom: ... and Software IS wrong!

Availability

Availability does not mean,
that a system is up and running
when the CPU is eating CPU cycles,
but that the system is doing
what is is intended to do!

Security & Availability

Access Setting	Access Result	Meaning
Grant	Grant	Wonderful

Security & Availability

Access Setting	Access Result	Meaning
Grant	Grant	Wonderful
Deny	Deny	Wow! That‘s Security!!

Security & Availability

Access Setting	Access Result	Meaning
Grant	Grant	Wonderful
Deny	Deny	Wow! That‘s Security!!
Deny	Grant	Oooops ...: Security Breach

Security & Availability

Access Setting	Access Result	Meaning
Grant	Grant	Wonderful
Deny	Deny	Wow! That‘s Security!!
Deny	Grant	Oooops ...: Security Breach
Grant	Deny	DoS

Why is DoS possible?

- GUARDIAN is an 'online' operating system
  - in contrast to batch
  - based on original development goals from 1974
- No user based resource measures and (re-)actions
  - no accounting data automatically fed back into the OS

Security people have a good heart,
but a sick mind.

(And customers sometimes just do it wrong ...)

Attacks

Two types of DoS attacks

- Wrong error handling
  - the 'normal' DoS 'attack'
- Pure Denial of Service (DoS) attack
  - by intention from the in- AND outside (watch out!)

DoS by Error Handling

- Example 1, file name resolution (1979):

  - Open a translation file (generic name -> physical name)
  - Position into it with the generic name
  - Read the related physical name
  - Return the physical name to calling instance

DoS by Error Handling

- Good logic, but ...
- Error not easy to find, and debug

DoS by Error Handling

- Problem: Missing Close
- Result: Exhaustive use of OCBS
- Causing volume crash every 7 – 10 days

DoS by Error Handling

Example 2, Error in $AOPR (1979):

- $o sends messages to $AOPR
- $AOPR filters messages, and
  writes important ones to a disk file, and printer
- In case the write to the disk file fails,
  an error message is sent to $o to report this problem

DoS by Error Handling

- Good idea, ...
- Not easy to find and debug

DoS by Error Handling

- Problem: Start of a Write loop
- Result: Exhaustive use of LCBs
- Causing CPU crash

DoS by Error Handling - Solution

- Code reading
  - to be performed by a group of programmers
- Real time test on a stand alone system
  - not even EXPANDed to production system
- Real error tests
  - REAL errors 43, or 45, or 48 etc.
  - NO error simulation
- Test the error logic in the error handling

DoS - from internal

- All you need is a non PRIV ID and an interactive system access, e.g. TACL

- Mainly two types of attacks:
  - Looping processes (CPU cycle eater)
  - Looping resource allocation (system table eater)

DoS - looping processes

- Looping processes
  TAL "one-liner":

```
Proc Loop Main;
  Begin
    While Priority(199) do;
  End;
```

DoS - looping processes

- Can be done by an interactive TACL user with access to a compiler, or LOAD access (FTP, IXF)
- Easily to add to the PM-Search List
  - name it according to a typo
  - FUP GIVE it to your “best” friend
- Uses all CPU cycles for quite some time, before it is put below the application PRI by GUARDIAN

DoS - looping processes

- A CPU bound loop causes a CPU hick-up for quite some time!
- In case you need a loooong hic-up – add some code to re-create the program …

- Show time

DoS - looping processes

- Prevention:
  - no compilers on - or strict control of - compilers on production systems
  - no FTP, IXF, and the like file transfer
  - use SAFEGUARD to protect all PM Search locations
  - have procedures in place to implement new software
  - use command level control for FUP GIVE/DUP/RENAME
  - use the SAFEGUARD authorization SEE to control all program start actions INDEPENDENT of ACLs
  - have a TACL at PRI 199 logged on available

DoS - looping resource allocation

- Recursive action
  - process start
  - file allocation
  - I/O resource action

DoS - looping resource allocation

- Recursive Process Start
  - by a TACL Macro
  - programmatically
- Causing exhaustive use of PCBs
- Use the SAFEGUARD authorization SEE to control process creates, e.g.
  user GHS.CARL is allowed to have 50 processes in parallel, but only one VIEWSYS, three SCF, one TAL, pTAL, AXCEL, etc.
- Show Time

DoS - looping resource allocation

- Recursive File Allocation
  - File create on $volume, best is on $SYSTEM.
    Simply do a CREATE $SYSTEM many times ...
  - By a TACL Macro
  - Remotely through EXPAND (most efficient method!)
- Causing loss of a volume
- Use a volume control program

- Show Time

DoS - looping resource allocation

- I/O action
  - Read/Write/WriteRead WITHOUT corresponding Reply between two processes (AOPR type)
- Causing massive system problems

- No real prevention possible ...

DoS - from external

- One way into the system from the outside is FTP:
  - It does NOT require any credentials to start an FTP server!
  - May flood your system with FTPSERV processes
  - Create as many FTP sessions as possible WITHOUT logging on
  - All you need is the IP address, that listens to the FTP port

DoS - from external

- Causes
  - Exhaustive use of PCBs
  - May crash a CPU
- Enhance the LISTNER to keep tack of FTP sessions
- Show Time

DoS - Counter Measures

- Code reading
- Test the error logic (= code in the error handling). I mean: TEST it on a system in real situations! Create a 'file is full' situation in reality!
- Make use of tools, which check the system on a regular basis
- Do NOT connect development, (QA) and production systems by EXPAND

DoS - Counter Measures

- Use SAFEGUARDs authorization SEE to control process creations (Process_Create_)
- Enhance LISTNER to check incoming FTP requests
- Scan all volumes regularly for over aged closed temporary files

DoS - Can it be prevented?

Not really ...

But you can limit the possibility to get seriously attacked.

Thank you for listening!

Questions?

By the way:

The Security SIG starts at 11:45 in Salon 6

I'm available to
discuss and/or present solutions.