NORTHROP GRUMMAN

DEFINING THE FUTURE

Command and Control System Administration at Headquarters U.S. Central Command

LISA 2006

7 December 2006

Andrew Seely
Northrop Grumman Corporation

Overview

Purpose of this presentation

Introduction to "Command and Control"

The systems that make up a C2 capability

"The Global Command and Control System (GCCS)"

The task of C2 system administration

Creativity as the catalyst – some examples

The people behind the systems

Resources and references

This presentation is unclassified in its entirety

NORTHROP GRUMMAN

Introduction to Command and Control (C2)

Critical function for any large-scale operation: Military, emergency response, disaster coordination, humanitarian relief, evacuation

- Concepts of Force Projection, Force Sustainment, and Force Protection
- Mission planning for current and potential requirements
- Logistics
- Intelligence
- Readiness assessment
- Situational awareness

NORTHROP GRUMMAN

The Global Command and Control System

- **GCCS-J is a family of systems**
  - Joint Operation Planning and Execution System (JOPES)
  - Integrated Imagery and Intelligence (I3)
  - Common Operational Picture (COP)
  - Integrated C4I System Framework (ICSF)
  - Global Status of Readiness and Training (GSORTS)
  - Theater Ballistic Missile Warning and Display (TBMWD)

Who uses GCCS?

- **GCCS-J is used by the Joint military community**
- **GCCS is used by by foreign nations and coalition partners**
- **Individual services have service-specific C2 flavors**
  - GCCS-AF, Theater Battle Management Core Systems (TBMCS), Deliberate and Crisis Action Planning and Execution System (DCAPES)
  - GCCS-A, Force XXI Battle Command Brigade and Below (FBCB2)
  - GCCS-M, Force Over-the-horizon Track Coordinator (FOTC), Automated Identification System (AIS)
  - Information Operations Server (IOS), Marine Air Ground Task Force War Planning System (MAGTAF II), Command and Control Personal Computer (C2PC)
- **There is increasing GCCS-J integration with civil defense and homeland security organizations**

NORTHROP GRUMMAN

Service and System Interoperability

- **A core strength of GCCS-J is interoperability, the ability to fuse data from a wide range of diverse sources**
  - Army, Navy, Air Force, Marine, coalition forces, civil defense
  - Beacon, radar, UAV, “hand-jam”
  - Tactical, operational, and strategic level visibility
  - Interoperability with new systems and products

NORTHROP GRUMMAN

The Systems That Make Up a C2 Capability

- Pre-1996: Mainframe-based (WWMCCS)
- 1996: Introduction of client-server GCCS: Sun Sparc 1, 5, 20, E1000, Solaris 2.3
- 1999: Sun Ultra 2, 5, 10, 60, 80, E6500, Windows NT 4.0, Solaris 2.5.1
- 2001: SunFire 280r, 420r, E450, E4000, Solaris 2.5.1
- 2003: SunFire 880, 1280, Solaris 2.5.1, Windows 2000
- 2005: SunFire 240r, 440r, Solaris 8
- 2007: Solaris 10, Windows XP
- 2010: GCCS becomes Net-Enabled Command Capability (NECC)

NORTHROP GRUMMAN

Primary CENTCOM GCCS-J locations

- Headquarters in Florida, USA
- Deployed Headquarters and CFSOCC in Doha, Qatar
- CENTAF in South Carolina, USA
- CFACC Combined Air Operations Center in Al Udeid, Qatar
- ARCENT in Georgia, USA
- CLFCC in Camp Arifjan, Kuwait
- NAVCENT/CFMCC in Manama, Bahrain
- MNC-I in Baghdad, Iraq
- CJTF in Kabul, Afghanistan
- CJTF in Djibouti, Djibouti

NORTHROP GRUMMAN

CENTCOM Area of Responsibility

NORTHROP GRUMMAN

The Task of C2 System Administration

- **Mission Orientation**
  - Keeping the purpose of the system in mind at all times
- **Reliability and Redundancy**
  - Ensuring the system can be trusted and that the mission can survive the loss of a server or a site
- **Flexibility**
  - Maintaining focus and professionalism in a constantly changing environment

NORTHROP GRUMMAN

Field Conditions

- **GCCS system administration is primarily, but not always, an in-garrison job**
- **Tactical conditions present challenges**
  - Inconsistent power and HVAC
  - Unreliable communications
  - Old equipment, no spares
  - Environment: Dust, moisture, scorpions, mortars, TCN staff, time zones
  - Rapid personnel rotation, leadership who feel they must “make their mark”
  - Lack of sysadmin resources

NORTHROP GRUMMAN

Sysad challenges

- Glacial rate of change in systems means living with obsolescence
- Tight, centralized configuration management means using only the tools you are given
- Stove-pipe applications mean many programs fighting over common system resources
- Government Off The Shelf (GOTS) applications mean industry isn't going to be much help
- Yet a wide range of expertise is needed at a moment's notice: You have to be sharp and learn fast

NORTHROP GRUMMAN

Creativity as the catalyst – some examples

Examples of sysadmin challenges:

- **Using truss to find the lockfile on a JRE**
- **Moving a ufsdump tape across an ocean**
- **Why is my system crashing? Analysis of memory, CPU, and I/O**

NORTHROP GRUMMAN

Using truss – the JRE

- Scenario: The Joint Range Extension (JRE) is a black-box that showed up in our rack with no documentation. The application stopped working – it wouldn't launch when the icon was clicked – and the functional team who had bought it was preparing to crate and ship for warranty service. Solaris 8 on what appears to be a proprietary hardware setup. Application looks (looked, when it ran) like Java. The original subscription to help desk support had expired and the peer JRE, with supposedly qualified staff who might be able to help us, is in Qatar.

NORTHROP GRUMMAN

Using Truss -- Troubleshooting steps

- Find the executable represented by the icon and run from the command line
- Same behavior, but additional output that was hidden by the GUI environment tells us that a log file is being created and, most importantly, *where*
- Inspect the log file (and discover that the application uses half a dozen log files for a variety of things), but the logs all report what appears to be normal behavior
- Inspect the file system around and above the logs and find a wad of JAR files and a few somethings.conf; inspect the .conf files and try changing a few parameters, with no improvement (though we made it much worse several times)

NORTHROP GRUMMAN

Using Truss – Reduced to lying

There's no documentation, the logs claim everything is working correctly, the configuration is apparently sane, but the system still doesn't work. What next?

Lie.

Called the help desk for the JRE and pretended I forgot my JRE account number.

Help desk tech explained how to run the application in debug mode. With more verbosity than before, the application failed in exactly the same way. The help desk was stumped and asked if we have a backup system.

We didn't.

NORTHROP GRUMMAN

Using Truss – Setup

The launch script runs a variety of programs that make up the application, so we doctored the launch script:

/path/to/sub_program_one

becomes…

truss –a –e –f –o /tmp/sub_program_one.truss \
/path/to/sub_program_one

We ran the application again and it failed, again, but now we had a variety of truss files to parse

NORTHROP GRUMMAN

Using Truss -- Secret Lockfile Discovery!

- Several eye-crossing hours later we discovered a "short read" error on a file in the application's "bin" directory
- The creation date on the file was the last time we ran the application, and the file was zero bytes
- *Removed the file and re-ran the application, and it works! The lockfile is recreated in the bin directory with six bytes of non-ASCII data*
- Apparently the developer chose to not only poorly document a lockfile that gets created in a bin directory, but decided to use the lockfile to store some sort of state value

NORTHROP GRUMMAN

The Tape Gets Lost in the Mail

- Scenario: Purpose-built Trusted Solaris 8 box in Tampa, sister box in Qatar. Accreditation of the system is incredibly tight; build and configuration must be done by a special team and takes up to two weeks. To save time and money, the plan is to build the system in Tampa, take a level 0 dump, and mail the tape via secure channels to Qatar while the team flies. Because nothing ever goes wrong (and because it's a huge hassle to hand-carry classified material), the team does not take original software to do a full build. The team arrives on-site – and the tape never shows up. (It finally showed up a month later!) Comms between Tampa and Qatar are very shaky and will not support FTP of large files.

NORTHROP GRUMMAN

Lost tape: Brainstorming

1. Mail another tape? (Tick-tock)
2. Make a copy of the tape and put someone on a plane today? ($$!)
3. Cancel the mission and re-schedule it? ($$$!)
4. FTP the contents of the tape?
5. ???

Lost tape: Attempts to FTP

First we need to get the files off the tape. The system is all on one file partition, tape created as level 0 ufsdump of that partition.

First make a backup of the backup, then make a file out of the tape:

dd if=/dev/rmt/0 of=/path/to/classifiedsystem.ufsdump

Kick off the ftp of classifiedsystem.ufsdump, about 3 gig, to the remote site – several hours later we find that the connection is hung; repeated attempts fail and a whole day is lost

NORTHROP GRUMMAN

Lost tape: What about small files?

- Large files don't seem to FTP well, but smaller files appear to transfer fine
- We don't have any special tools but we do have Perl, so we built a tool to break our big ufsdump into little bits

NORTHROP GRUMMAN

Lost tape: File chunker code

```
#!/usr/local/bin/perl
#  Verify that we have at least three command-line arguments
if (! $ARGV[2]) { print "Usage:  $0 sourcefile destinationfile maxchunksize\n"; }
else {#  Assign local variables
  $infile = $ARGV[0];  #  the input file
  $outfile = $ARGV[1];  # the stub filename for the output
  $maxfile = $ARGV[2];  # the size of the output chunks
  $chunk = 1024;  # the read buffer size
  $flag = 1;  # looping flag
  if (! -r $infile) {    print "Unable to open $infile\n"; } # Exit if we can't open the input file
  else { # Open the input file for read, exit if the open fails
    print "Opening $infile for read.\n";
    open (I, "$infile") or die "$! on open $infile\n";
    while ($flag) { # Loop until the end of the file, $flag is toggled at input file EOF in the inner loop
      # set the current output file name pre-pad with zeros for better sorting assumes no more than 99999 chunks
      $fcount++;
      if ($fcount < 10) {  $fn = "000${fcount}.$outfile";  }
      elsif ($fcount < 100) { $fn = "00${fcount}.$outfile";   }
      elsif ($fcount < 1000) {  $fn = "0${fcount}.$outfile";   }
      else { $fn = "${fcount}.$outfile";  }
      $counter = 0; # $counter keeps track of how much we've read in this loop
      print "Opening $fn for write.\n";
      open (O,">$fn") or die "$! on open $fn\n"; # open the output file for writing, exit on failure
      while ($counter < $maxfile) {# read from the input file until we have read $maxfile bytes
        if (read (I, $buff, $chunk)){
          print O $buff; $counter += $chunk; }
        else {  # we've read through the whole input file
          print O $buff; # write out any residual read
          $flag=0; # unset the $flag so we exit the outer loop
          $counter=$maxfile; # this will cause the inner loop to exit
          print "Reached end of $infile\n";
          }
        } # end of while
      close O;
      } # end of while $flag
    close I;
    } # end of infile
  print "Chunking Complete!\n";
  } # end of top if
```


NORTHROP GRUMMAN

Lost tape: De-chunker

```
#!/usr/local/bin/perl
#  Verify that we have at least two command-line arguments
if (! $ARGV[1]) {
  print "Usage:  $0 sourcefile destinationfile\n";
  }
else {
  # set local variables
  $infile = $ARGV[0];
  $outfile = $ARGV[1];
  $readsize = 1024;
  # read the files from the current directory that end with the supplied name
  opendir D, "." or die "$! On opendir";
  @files = grep /$infile$/, readdir D;
  closedir D;
  print "Opening $outfile for write\n";
  open (O, ">$outfile") or die "$! On open $outfile for write\n";
  # sort the files and read each one, writing into the $outfile
  foreach (sort @files) {
      print "Reading $_\n";
      open (I, "$_") or die "$! On open $_\n";
      while (read (I, $buf, $readsize)) {
        print O $buf;
        }
      close I;
      } # end foreach
    close O;
    print "Finished assembling $outfile\n";
    } # end else
```


NORTHROP GRUMMAN

Lost tape: Setup and execution

- Chunk the file in Tampa into about 300 files
- FTP small files in batches to a remote server in Qatar; manual inspection and re-FTP of failed files (a dozen or so)
- Reassemble the ufsdump on the remote system
- Original plan is to dd the ufsdump back onto a tape (the "teleported tape" idea)
- Even better: Added a second hard drive to the remote system, use format and newfs to make a single partition, and simply ufsrestore from file directly to this drive
- Remove the restored drive and put it in the target system in the boot drive slot
- *Power on …. Success!*

NORTHROP GRUMMAN

Crashing system

Scenario: Server runs for about four hours before full freeze-up; reboot returns to service but performance degrades steadily. Software application is new and written in Java but supporting software is rumored to be legacy C or Ada. System has lots of network I/O and lots of IPC. Common wisdom is that RAM is always the key to tuning this system and that the core process that ties all subsystems together is very old code and is not multi-threaded. The developers are scratching their heads. SunFire 240r with 4 gig RAM and 2x 1-Ghz CPU. This is the single most important server in the rack, and the General is calling. We're rebooting every two hours.

Crashing system: Problem analysis

- **Load balancing: We shift some data feeds to other systems in an attempt to ease the load and mitigate impact**
- **Increase RAM? Upgrade to a larger platform? We can do these, but what's the real problem?**
- **What tools do we have to work with? Solaris 8 02/02.**
  - vmstat
  - iostat
  - prstat
  - psradm
  - *Solaris Management Console has been disabled!*

NORTHROP GRUMMAN

Crashing system: I/O

Could the problem be a highly transactional file system with slow drives?

- Ran iostat while system was performing well and during degraded conditions
- Showed high CPU utilization, zero to 10% CPU idle time as system degrades
- Wait queue was consistently empty
- The disks are busy, but not overburdened – they're just "earning their paychecks"

NORTHROP GRUMMAN

Crashing system: Memory

Could the problem be insufficient RAM or swap?

- **vmstat reports similar CPU idle percentages in single-digits**
- **Run queue is sane, free list is stable and comparable to systems running normally**
- **Amount of VM appears to be adequate; the system isn't showing signs of thrashing**

Crashing system: Applications

- **Could the problem be some runaway process?**
- **Use prstat to look at process statistics**
- **Process statistics show some processes holding on to the CPU and never releasing it (remember, this is only a 2-way system)**
  - We experimented with using priocntl to tamp down the scheduling priority
  - This resulted in a temporary gain in performance (using vmstat CPU idle as a general guide), but saturation would creep back over the next 20 minutes
- **The system is just doing its job poorly, but the applications don't seem pathological**

NORTHROP GRUMMAN

Crashing system: CPU

Could the problem be the CPU?

- **I/O, VM, and the applications all seem to be sane**
- **We've been leaning on the urban legend for this system that RAM is the key and more processors will have no effect, but how can we test this?**
- **Fresh reboot, so the system is performing as good as it ever will; running prstat and vmstat to get two views of system performance (understanding that the measurements are having their own negative performance impact)**
- **It's a 2-way system, so let's punch out a CPU and see how it behaves:**
  - psradm –f 1 to take CPU 1 down
  - performance drops by two thirds instantly – the system as a whole is obviously sensitive to multiprocessing; we start to suspect that it's far more sensitive than we've been led to believe

NORTHROP GRUMMAN

Crashing system: Solutions

- The SunFire 240 only supports two processors, but we're lucky to have an extra 440 with four CPU and 8 gig RAM.
- The operating system as installed supports both platforms; we rack the 440, swap the disk, and boot-r
- The system boots and runs fine, so we start on the same analysis path we did previously, culminating in using psradm to knock out 1, 2, 3 processors to observe behavior
- Along the way, we get word from the developers, finally, that the application data channels are CPU hogs (we were running minimum nodes on maximum channels, vice maximum nodes on minimum channels)
- *The new hardware with increased processors, combined with applying data channel conservation, resulted in a very stable system*

NORTHROP GRUMMAN

The people behind the systems

- **A typical GCCS system administrator**
  - Prior-service or currently enlisted
  - Military training
  - Working on a degree and/or certification
  - Leadership tends to be retired senior NCO or officer if civilian, O3-O5 if active duty
- **Developers**
  - Almost exclusively contracted out to a handful of companies
  - Full-time professional programmers, not necessarily working in cleared environments
  - Northrop Grumman is the primary developer for the core of GCCS-J, but other companies also contribute

NORTHROP GRUMMAN

GCCS resources

- **Northrop Grumman**
  - http://www.northropgrumman.com/
- **Northrop Grumman Mission Systems**
  - http://www.ms.northropgrumman.com/markets/MDC4ISR.html
- **DISA PMO**
  - http://www.disa.mil/
- **USCENTCOM**
  - http://www.centcom.mil/
- **JITC**
  - http://jitc.fhu.disa.mil/gccsiop/
- **DoD**
  - http://www.defenselink.mil/
- **Joint Vision 2020**
  - http://www.dtic.mil/jointvision/jvpub2.htm

NORTHROP GRUMMAN

Questions, contacts, an event, and supper time

Andrew Seely

andrew.seely@ngc.com

+1 813 827 3226

- **The 2007 GCCS System Administration and Engineering Conference will be held on Macdill AFB in Tampa, Florida in the third week of September 2007.**

NORTHROP GRUMMAN