

CISCO *Live!*

GO BEYOND

#CiscoLiveAPJC



Big Iron and Sustainability

Measurement, Efficiency, Management, Action

LJ Wobker, Principal TME for Big Fast Routers and Switches
lwobker@cisco.com @ljwobker

BRKSPG-2318

Cisco Webex App

Questions?

Use Cisco Webex App to chat with the speaker after the session

How

- 1 Find this session in the Cisco Live Mobile App
- 2 Click “Join the Discussion”
- 3 Install the Webex App or go directly to the Webex space
- 4 Enter messages/questions in the Webex space

Webex spaces will be moderated by the speaker until November 15, 2024.



Abstract

Sustainability is a relatively recent buzzword, but it's actually one of the few places where everyone's incentives are truly aligned. Using less power is both good for the environment and saves organizations money, so what's not to love? Well, for one thing, power usage in both provisioning and consumption terms is not terribly well understood, and that's where we come in. This discussion will range across multiple topics related to how large routing systems can be optimized for efficiency. We'll cover which parts of the systems consume the most power, because if you don't know where the power is going you don't know where to look for savings. We'll discuss a Cisco project which collected millions of power data points and used those to build a highly accurate model for predicting the power consumption of systems in the wild. We'll discuss advances in silicon technology which dramatically reduce the power per gigabit on modern systems. With the "box-level" foundations in place, we'll then move to a more network-focused discussion around what mechanisms exist to reduce the power draw.

Introduction... my info



- Apologies for the accent
- TAC -> CAP -> XR TME -> AS/CX -> XR TME -> startup -> TME
- Out of scope:
 - Platform nitty-gritty details, router configs, recommendations, “do this”
- In-scope
 - Intro-engineering math, Tradeoffs, Investigative “mess around” approach
- Come find me later (MTE, etc.) to talk about:
 - Chips, packet forwarding, system design, performance

How much energy does it take to switch one more packet???

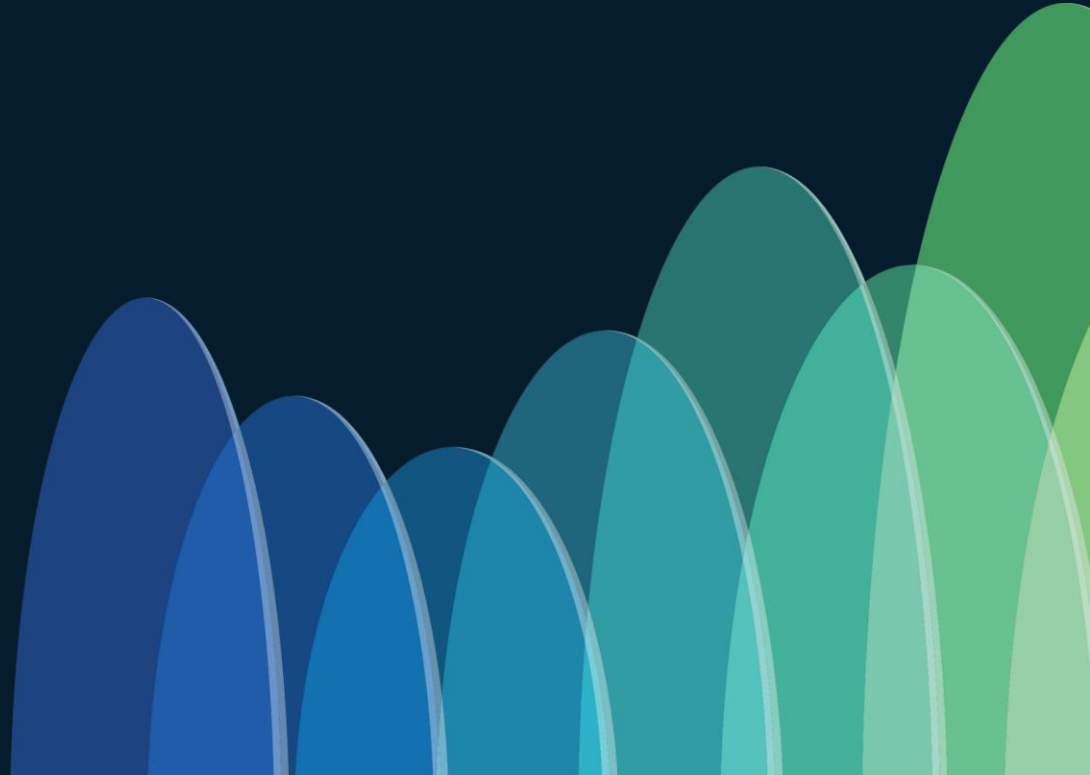
Who: -- me!

When -- early days of the Cisco 8k program

Agenda

- Physics review, power model
- Component level power
- Cisco 8000 power measurement project
- Provisioned power vs power consumption
- Reducing power requirements (systems)
- Reducing power (network)
- Conclusion

Physics review



*Never underestimate the
bandwidth of a station wagon
full of tapes hurtling down the
highway.*

Andrew S. Tanenbaum

Computer Networks, 3rd ed., p. 83

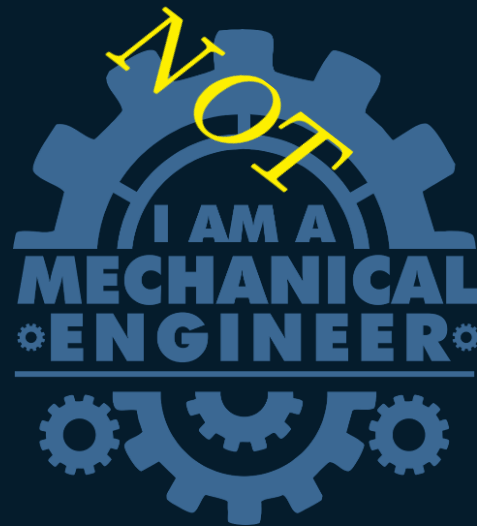
Physics review

- Power, energy, voltage, current, capacitance, resistance
- Facility planning level: power is time.
- Row/rack level: power is money.
- System level: power is heat.
- Timescales matter – a lot.



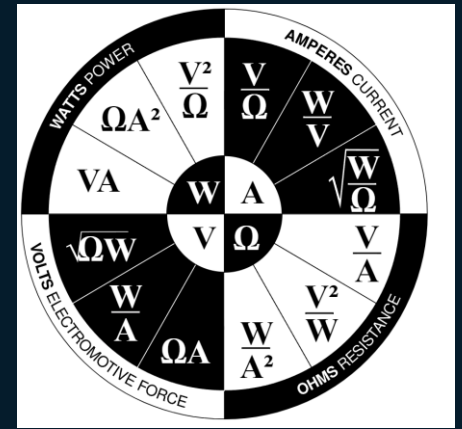
Some equations (sorry...)

- Ohm's Law
- IC power usage – static, dynamic, activity
- Fan laws
- Heat transfer delta-T



Semiconductor power “laws”

- Ohm’s Law
- static power
 - linear with voltage and current (mostly Si area)
- dynamic power
 - linear with activity, capacitance
 - losses due to switching
 - linear with frequency



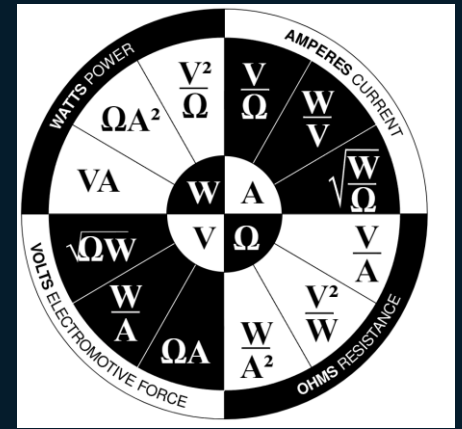
$$P_{total} = P_{static} + P_{dynamic}$$

$$P_{static} = V_{cc} \times I_{cc}$$

$$P_{dynamic} = \beta \times C_L \times V_{cc}^2 \times f$$

Semiconductor power “laws”

- static power
 - linear with voltage and current (mostly Si area)
- dynamic power
 - linear with activity, capacitance
 - losses due to switching
 - linear with frequency
 - SQUARE with voltage



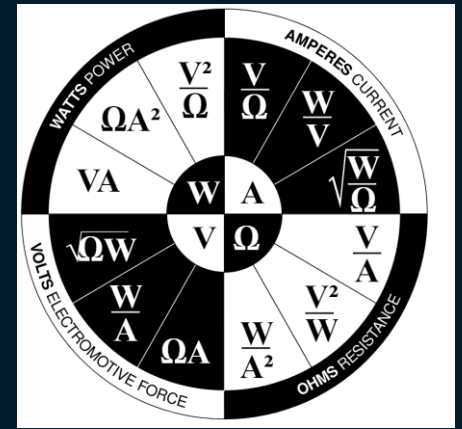
$$P_{total} = P_{static} + P_{dynamic}$$

$$P_{static} = V_{cc} \times I_{cc}$$

$$P_{dynamic} = \beta \times C_L \times V_{cc}^2 \times f$$

Semiconductor power “laws”

- static power
 - linear with voltage and current (mostly Si area)
- dynamic power
 - linear with activity, capacitance
 - losses due to switching
 - linear with frequency
 - SQUARE with voltage
- But...
 - freq / voltage are coupled!



$$P_{total} = P_{static} + P_{dynamic}$$

$$P_{static} = V_{cc} \times I_{cc}$$

$$P_{dynamic} = \beta \times C_L \times V_{cc}^2 \times f$$

Heat transfer (using air)

- Sensible heat transferred (in KW) is a function of:
- C_p – specific heat of air
- ρ – air density
- q – airflow (fans!)
- temperature difference (ambient vs component)



$$H_s(KW) = c_p \times \rho \times q \times \Delta T$$

Heat transfer (air as the fluid)

C_p - specific heat of air

ρ - air density

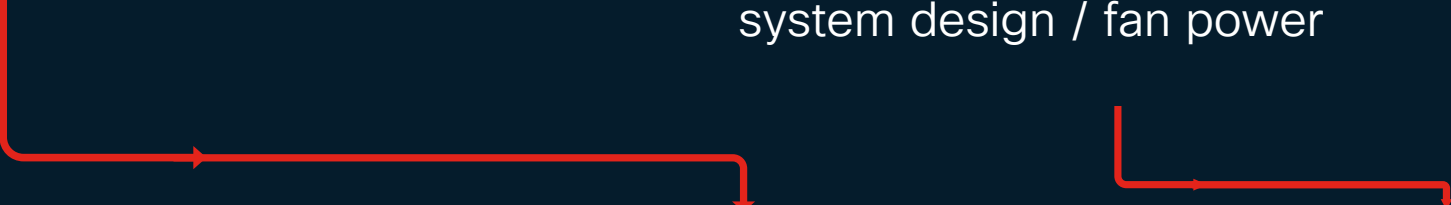
q - airflow (fans!)

dT - temperature difference
(ambient vs component)

Physical constants

hint : choose a more
conductive fluid?

ambient environment +
system design / fan power


$$H_s(KW) = c_p \times \rho \times q \times \Delta T$$

Fan affinity laws

- Airflow is linear with rotation speed (RPM)
- But power consumption is CUBED with RPM!
- Q = airflow, N = RPM, D = diameter, P = power

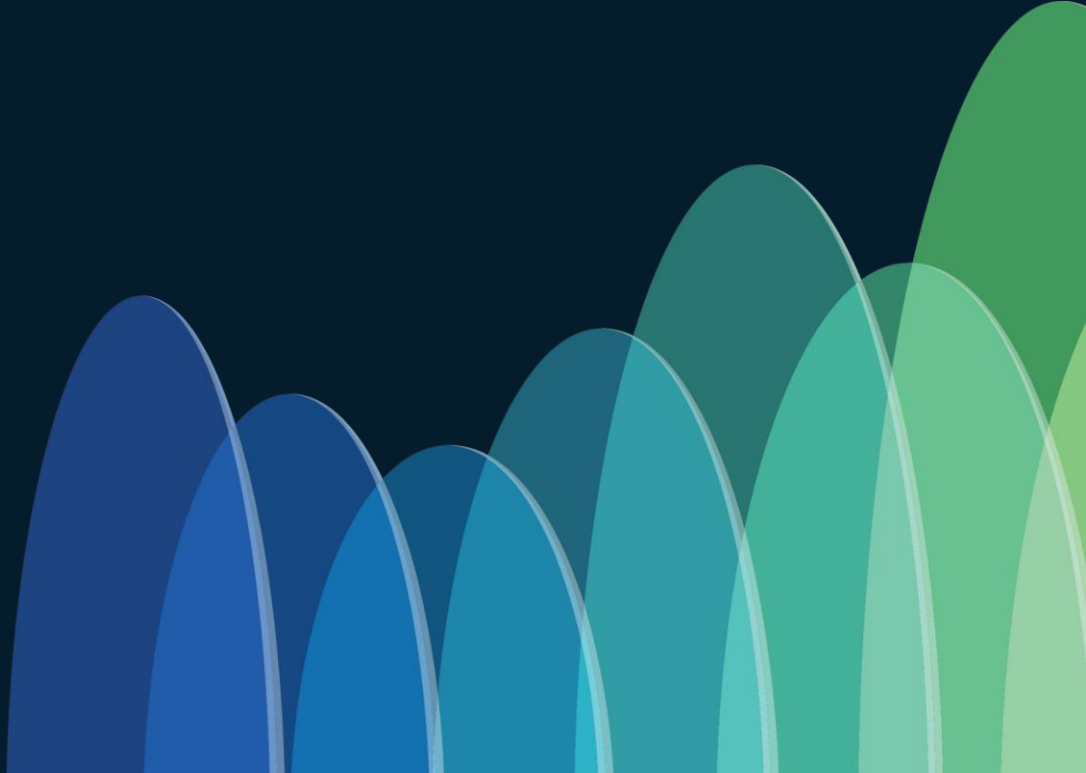


$$\frac{Q_1}{Q_2} = \frac{N_1}{N_2} * \frac{D_1}{D_2}$$

$$\frac{P_1}{P_2} = \left(\frac{N_1}{N_2} \right)^{\boxed{3}} * \left(\frac{D_1}{D_2} \right)^{\boxed{3}}$$

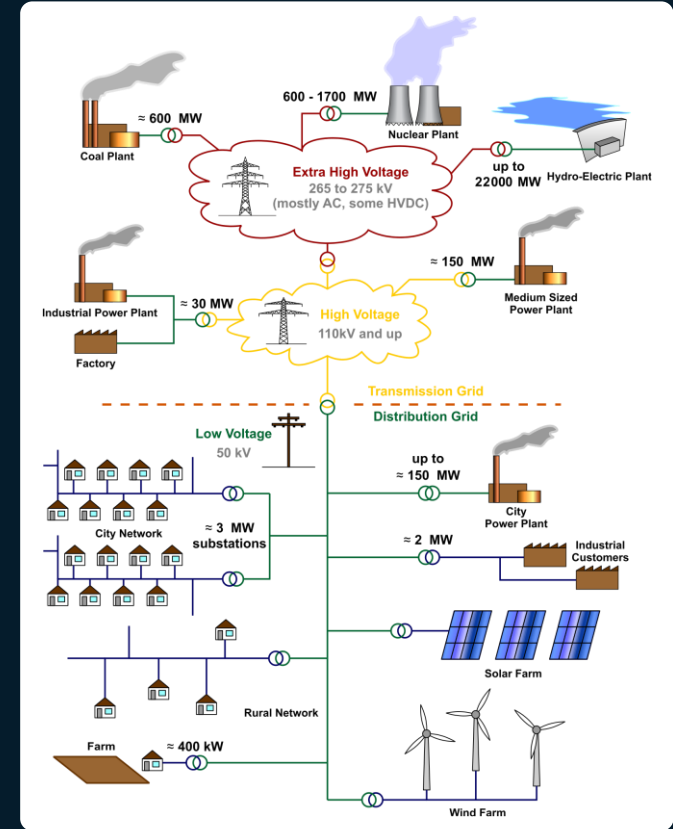
ouch

Power systems: Gigawatts to milliwatts



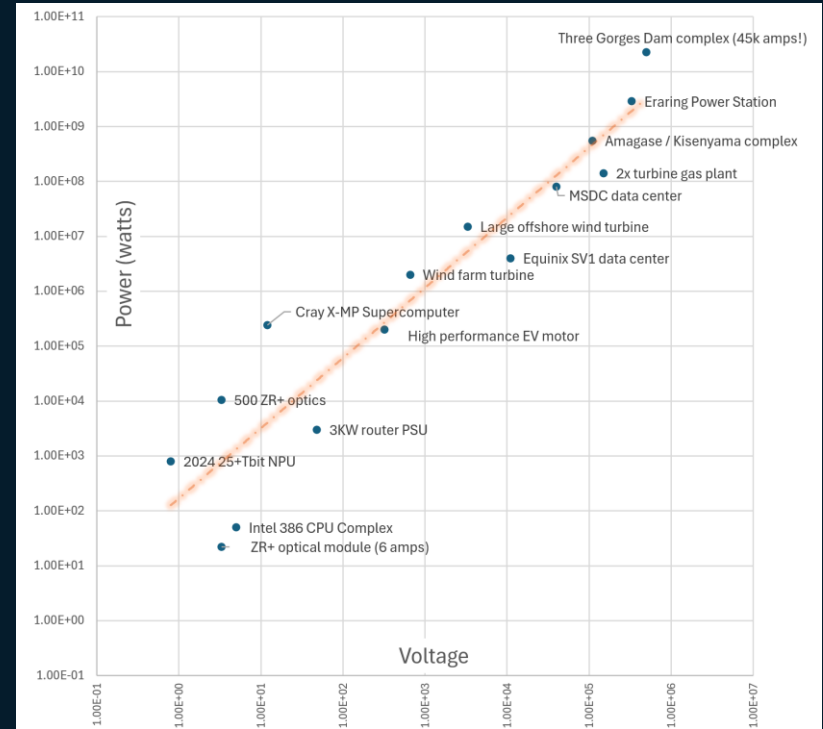
Big picture to small picture

- Commercial power generation
- Transmission / distribution networks
- Building level distribution
- Rack level distribution (PDU)
- Equipment power supply design
- Board level power trees
- Chip level



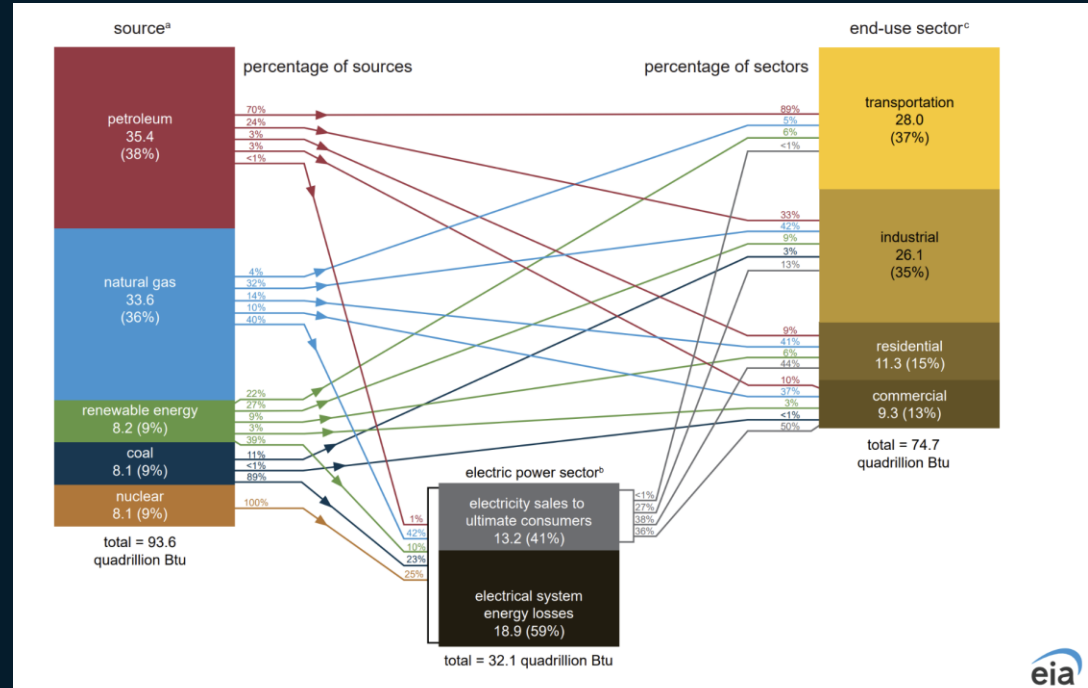
Voltage v. power for a set of engineering marvels

- Voltages : 1 to half a million
- Power : 1 to gigawatts
- An interesting observation...
- Current : MOSTLY rangebound



Commercial power generation

- Power mixes
- Government interaction
- Geopolitics



^a U.S. Energy Information Administration (EIA), *Monthly Energy Review* (April tables 1.3, 1.4c, and 2.1a-2.6).

^b The electric power sector includes electricity-only and combined-heat-and-power (CHP) plants whose primary business is to sell electricity, or electricity and heat, to the public.

Transmission / distribution networks

- Availability of power
- Facility infrastructure
- Step-down / step-up transformers
- VERY long lead times...

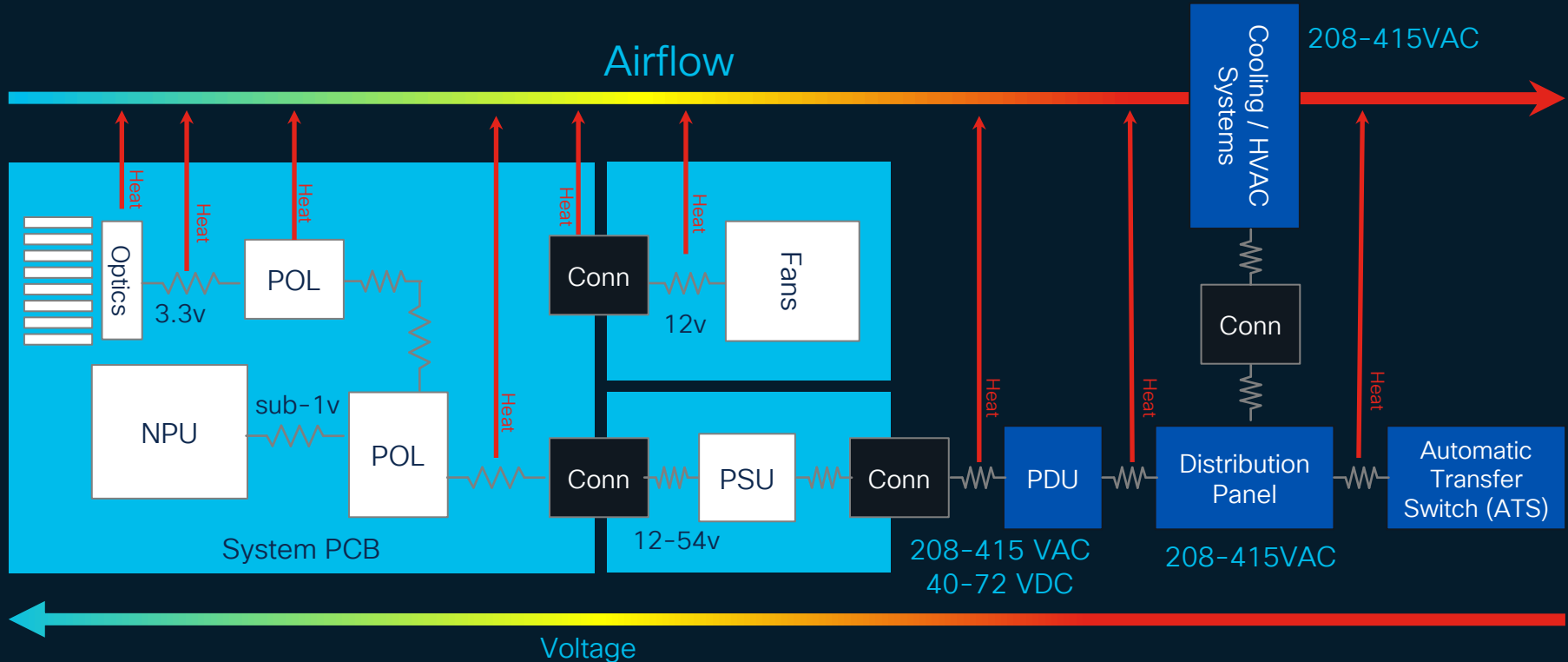


Building, rack- level distribution

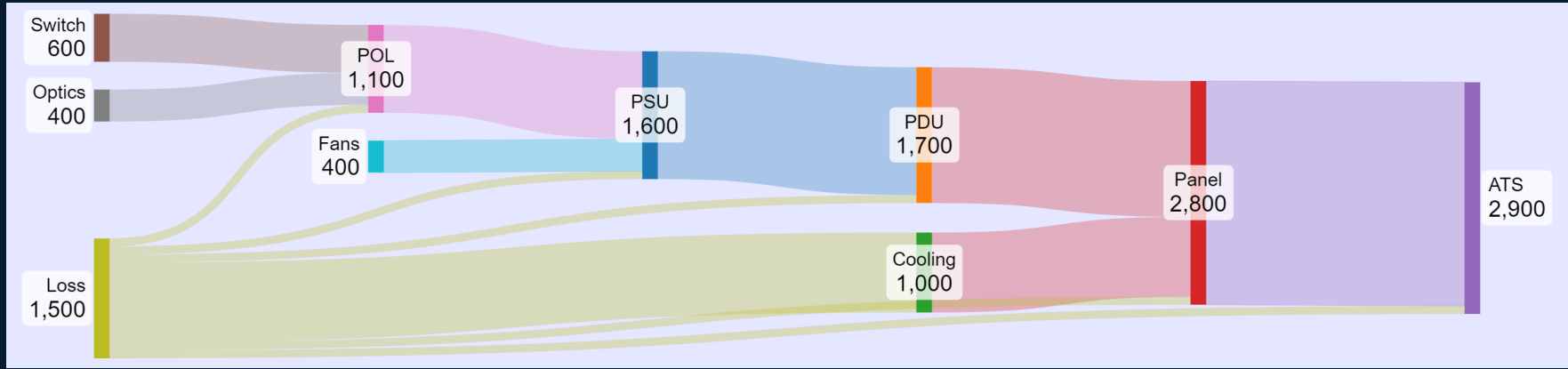
- Utility interfaces (voltages)
- Building-level switching and backup
- “Smart” PDUs?
- Integrated power racks (OCP)
- TCO for AC vs. DC power



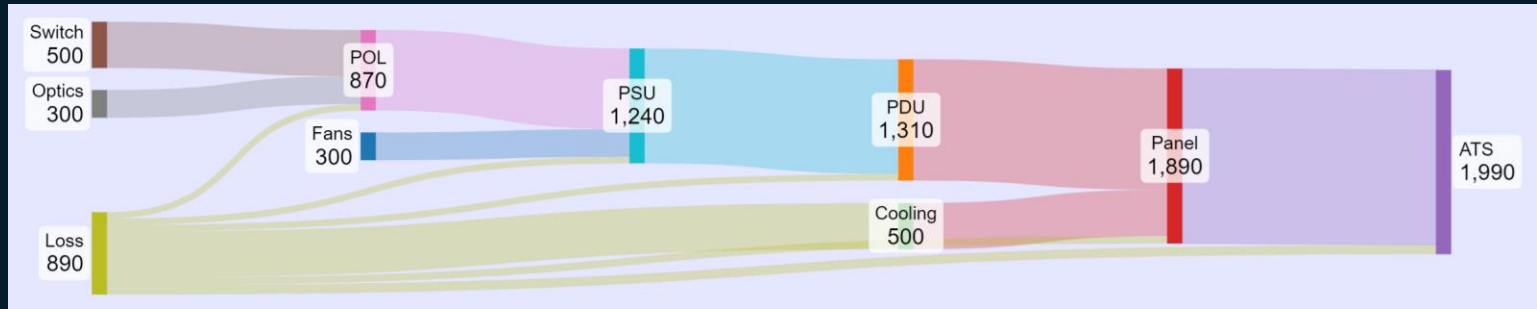
Building/system components and voltage ranges



PUE : Power Usage Effectiveness

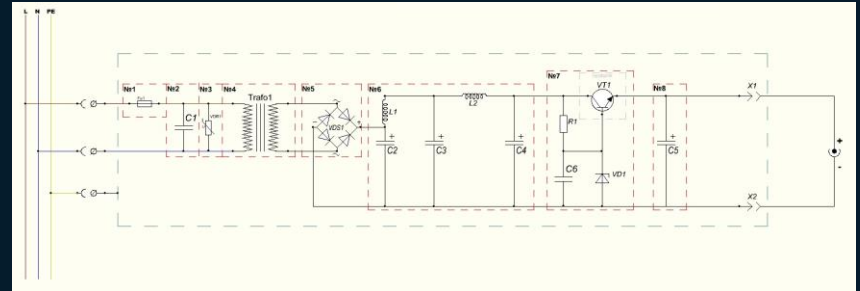


VS...



Equipment power supply design

- Design discussion
 - Capacitors, diodes, & inverters -- oh my!
- Efficiency curves
- Various certifications
- Redundancy vs. efficiency
 - Optimized for ~50% vs ~90% ?



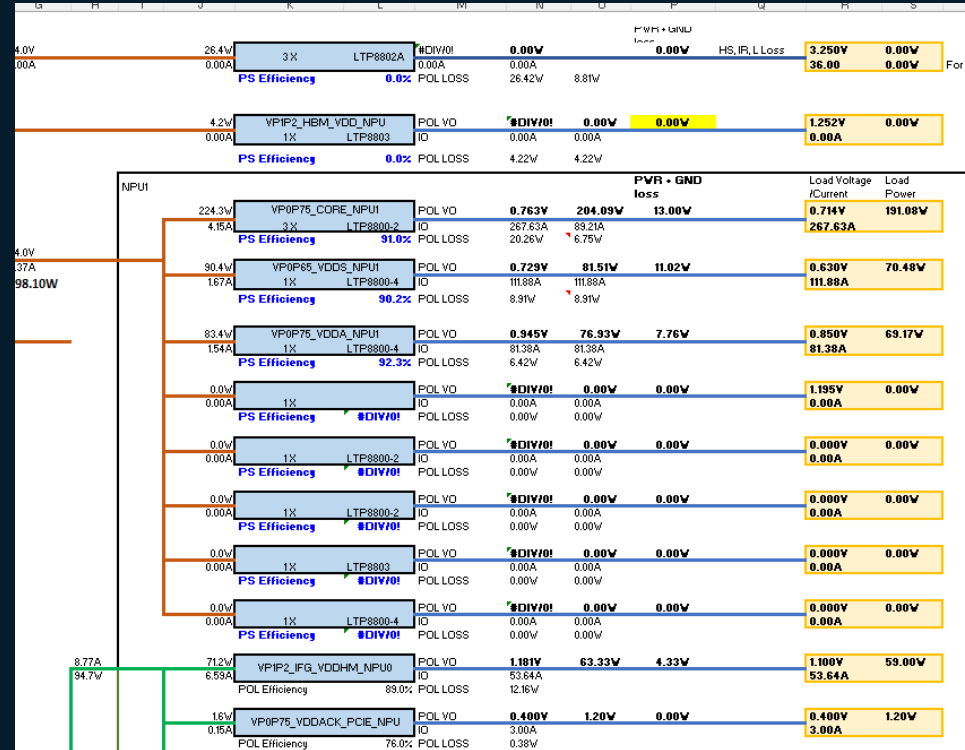
Board level power trees

- Multiple components (100s!)
- Multiple voltage rails
- Dependencies in power-up order
- Expertise is efficiency – art vs. science

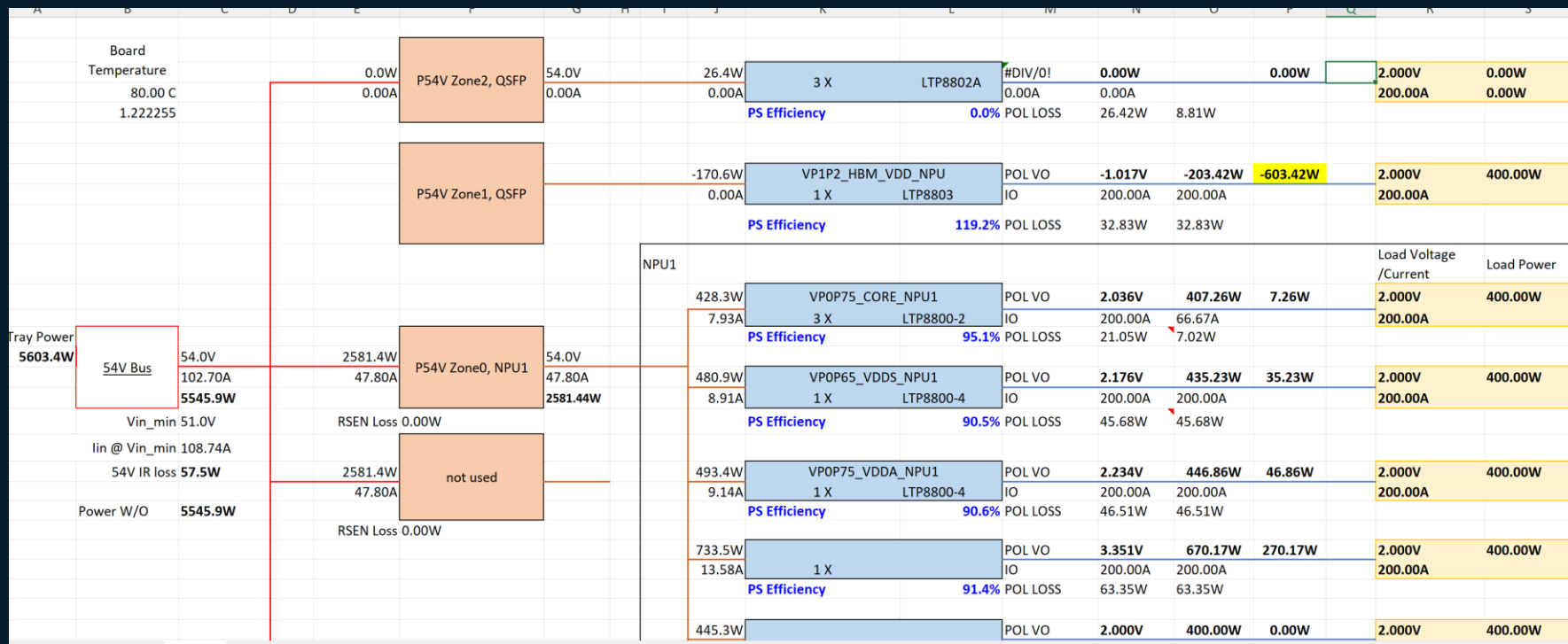


Example XLS model of board power tree

Location	VOLTAGE Sensor	Value (mV)	Crit (Lo)	Minor (Lo)	Minor (Hi)	Crit (Hi)
0/RP0/CPU0						
IBV		10316	8928	NA	NA	11984
VP1P8_OCX0		1806	1638	NA	NA	1962
P1_8V		1814	1638	NA	NA	1962
P1_0V_ALDRIN_SD		1009	910	NA	NA	1090
P1_0V_ALDRIN_CRE		984	930	NA	NA	1070
P1V		1005	910	NA	NA	1090
P0_9V		897	819	NA	NA	981
ALDRIN_P1P8_LDO		1796	1638	NA	NA	1962
OMG_P1P8_LDO		1798	1638	NA	NA	1962
V3P3_RTC_C		3297	3003	NA	NA	3597
VP1P0_Z		1002	910	NA	NA	1090
VP1P2_CPU_C		1221	1134	NA	NA	1305
P1_2V_A_PMB_CRE		1206	1092	NA	NA	1308
P1_2V_A_PMB_SER		1203	1092	NA	NA	1308
P2_5V_A_PMB_PLL		2495	2275	NA	NA	2725
P1_8V_A		1798	1638	NA	NA	1962
P2_5V_A		2520	2275	NA	NA	2725
P3_3V_A		3310	3069	NA	NA	3531
P5V		5039	4550	NA	NA	5450
P7V		7046	6370	NA	NA	7630
V3P3_CPU_C		3302	3003	NA	NA	3597
VP1P8_VCCIN_C		1795	1488	NA	NA	2086
VP1P05_CPU		1054	976	NA	NA	1123
VP1P7_C		1705	1547	NA	NA	1853
VP1P5_C		1504	1365	NA	NA	1635
VP1P3_C		1303	1183	NA	NA	1417
MB_VPS4P0V		54525	48298	NA	NA	59913
VP0P6_VTT_MEM_C		611	558	NA	NA	642
VP1P2_MGTAVTT		1206	1092	NA	NA	1308
VP1P0_MGTAVCC		1002	910	NA	NA	1090
VP3P3_OCX0		3321	3003	NA	NA	3597
P3_3V		3308	3003	NA	NA	3597
P2_5V		2511	2275	NA	NA	2725
0/0/CPU0						
VP0P94_VDDA_NPU0		961	818	NA	NA	1081
VP1P8_PLLVDD N1		1799	1566	NA	NA	2070

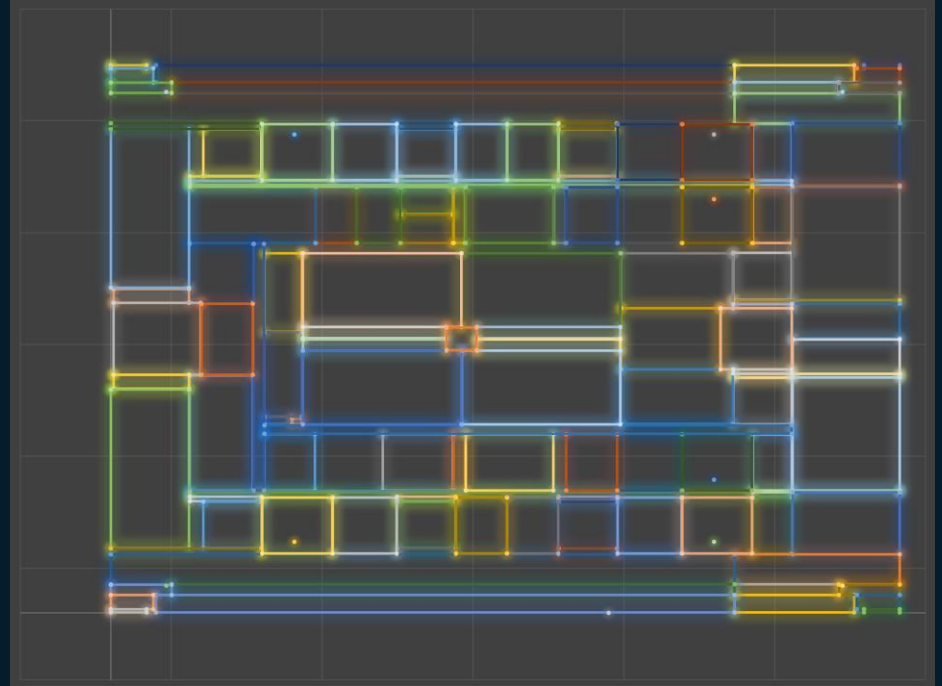


Board power tree (zoomed, sanitized)



Chip level power

- Power maps
- Low voltages -> high current
- Power is heat!
- Range: 1W – 1000W



Cisco 8000 Power Measurement Project

What are we measuring?

- Well, duh – how much power the thing uses.
- And, along the way...
- About a million other things...



What are we measuring... and why?

- Save money. Yours and ours.
- Climate, out of both nobility and regulation.
- Competitive differentiation
- Design improvements / system optimizations.
- What knobs exist?
- And are worth turning?



Why do we measure?

- When we can probably compute from:
 - First principles?
 - Data sheets?
 - Educated guesses?



Why we measure...

- Primary sources are often unreliable (including ours!)
- Interactions are frequent and often complex
- There's an important distinction between...
- “Nondeterministic” and “really complicated”
- . . . and . . .
- TRUST



(Almost) everything can be squished to:

- A “few term” linear equation
- Example:
- Each pair is a “material component” of power draw...

$$P_{total} = W * \alpha + X * \beta + Y * \rho + Z * \lambda$$

(Almost) everything can be squished to:

- A “few term” linear equation
- Example:

$$P_{total} = W * \alpha + X * \beta + Y * \rho + Z * \lambda$$

- The **coefficients** are what we really care about

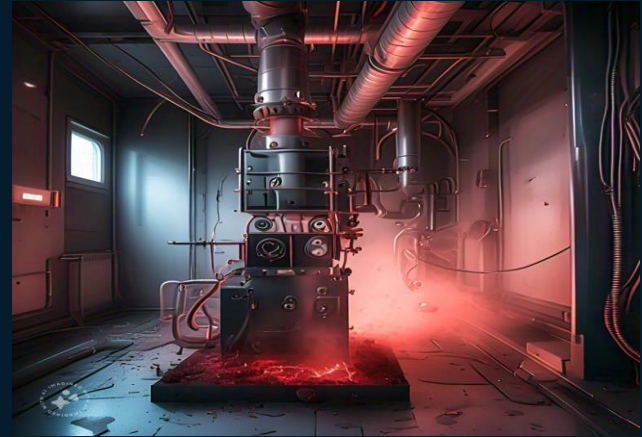
Power model for the test router

- Number of serdes powered up
 - Power for each one
- Number of optics powered up
 - Watts / module
- Total traffic load (bits / second)
 - Coefficient for data rate
- Total traffic load (pkts / second)
 - Coefficient for packet rate

$$P_{total} = W * \alpha + X * \beta + Y * \rho + Z * \lambda$$

The project

- Systems to test
 - Thermal chambers, optics, test ports, etc.
- Data collection script
- Data analysis script
- Philosophies (constraints?) :
 - Minimal assumptions about test systems (scrape, not modeled)
 - Get the raw data, everything else is secondary (practical)



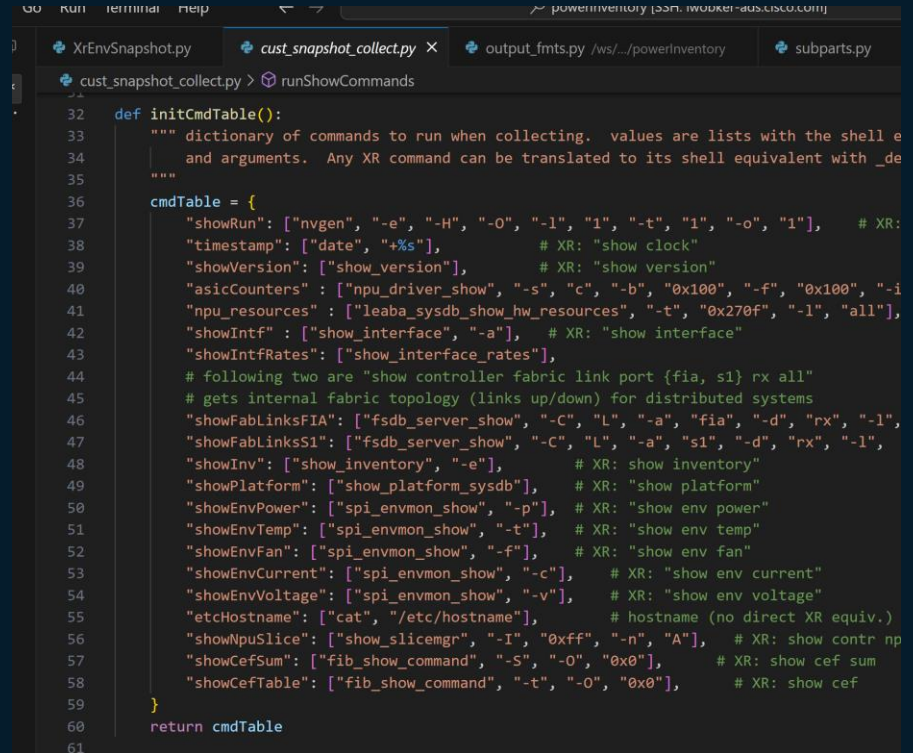
Key variables for tests

- Ambient temperature -> fan speeds (thermal chamber)
- Physical system configuration (which components)
- Traffic load (bandwidth)
- Packet rates (feature sets)



Let's get some data...

- On the Cisco 8000
- Linux shell for XR commands
- Native python interpreter
- Thread for speed
- Save raw text output to /tmp
- Compress to JSON



```
def initCmdTable():
    """
    dictionary of commands to run when collecting. values are lists with the shell e
    and arguments. Any XR command can be translated to its shell equivalent with de
    """
    cmdTable = {
        "showRun": ["nvgen", "-e", "-H", "-O", "-l", "1", "-t", "1", "-o", "1"], # XR:
        "timestamp": ["date", "%s"], # XR: "show clock"
        "showVersion": ["show_version"], # XR: "show version"
        "asicCounters": ["npd_driver_show", "-s", "c", "-b", "0x100", "-f", "0x100", "-i
        "npd_resources": ["leaba_sysdb_show_hw_resources", "-t", "0x270f", "-l", "all"],
        "showIntf": ["show_interface", "-a"], # XR: "show interface"
        "showIntfRates": ["show_interface_rates"],
        # following two are "show controller fabric link port {fia, s1} rx all"
        # gets internal fabric topology (links up/down) for distributed systems
        "showFabLinksFIA": ["fsdb_server_show", "-C", "L", "-a", "fia", "-d", "rx", "-l",
        "showFabLinksS1": ["fsdb_server_show", "-C", "L", "-a", "s1", "-d", "rx", "-l",
        "showInv": ["show_inventory", "-e"], # XR: show inventory"
        "showPlatform": ["show_platform_sysdb"], # XR: "show platform"
        "showEnvPower": ["spi_envmon_show", "-p"], # XR: "show env power"
        "showEnvTemp": ["spi_envmon_show", "-t"], # XR: "show env temp"
        "showEnvFan": ["spi_envmon_show", "-f"], # XR: "show env fan"
        "showEnvCurrent": ["spi_envmon_show", "-c"], # XR: "show env current"
        "showEnvVoltage": ["spi_envmon_show", "-v"], # XR: "show env voltage"
        "etcHostname": ["cat", "/etc/hostname"], # hostname (no direct XR equiv.)
        "showNpuSlice": ["show_slidemgr", "-I", "0xff", "-n", "A"], # XR: show contr np
        "showCefSum": ["fib_show_command", "-S", "-O", "0x0"], # XR: show cef sum
        "showCefTable": ["fib_show_command", "-t", "-O", "0x0"], # XR: show cef
    }
    return cmdTable
```

How much data do we have?

```
./k2/K2_BB8_768B_50LRD8K2_cmds_240611-134359.json.xz  
./k2/K2_BB8_768B_50LRD8K2_cmds_240611-134452.json.xz  
./k2/K2_BB8_768B_50LRD8K2_cmds_240611-134545.json.xz  
./k2/K2_BB8_768B_50LRD8K2_cmds_240611-134639.json.xz  
./k2/K2_BB8_768B_50LRD8K2_cmds_240611-134733.json.xz  
./k2/K2_BB8_768B_75LRD8K2_cmds_240611-142227.json.xz  
./k2/K2_BB8_768B_75LRD8K2_cmds_240611-142321.json.xz  
./k2/K2_BB8_768B_75LRD8K2_cmds_240611-142415.json.xz  
./k2/K2_BB8_768B_75LRD8K2_cmds_240611-142509.json.xz  
./k2/K2_BB8_768B_75LRD8K2_cmds_240611-142603.json.xz
```

```
rtp-ads-1024:~/ws/gitz/powerInventory > find ./data -name \*.json.xz | wc -l  
8671
```

(By now, well over 9000 snapshots...)

What do we DO with all of it?

- “Build” system (show inventory)
- Add interfaces (show interfaces)
- Parse environmentals (show environment ...)
- Grab gooky NPU / chip level details (show controller ...)
 - Chip frequencies, revisions, software versions...
- Compute component level power (show env voltage / current)

What do we DO with all of it?

- .py classes for each part
- Find power components
- Parse PID, S/N, etc.
- Store as key/value pairs

```
409 def addFromShowInv(self, invList: list):
410     """From a string containing the output of XR's show inventory, parse the various parts and
411     add them to the snapshot's interface dict. the RE match strings are stored in the invDefines
412     file. The ordering is relevant as some components are substrings of others."""
413
414     patterns = invDefines.showInvTypesRegex
415     for basepart in invList: # each line in show inventory is a basePart in the list
416         basepart.phantom = False # if a part is in show inventory it's "real"
417         if re.search(patterns.rack, basepart.name):
418             self["rack"][basepart.name] = Rack(basepart)
419             self.rack = self["rack"][basepart.name] # convenience!
420             self.buildChassisFramework(self.rack.pid) # hardcoded -- everything is Rack 0
421         elif re.search(patterns.optic, basepart.name):
422             self["optic"][basepart.name] = Optic(basepart)
423         elif re.search(patterns.lc, basepart.name):
424             self["lc"][basepart.name] = LineCard(basepart)
425         elif re.search(patterns.rp, basepart.name):
426             self["rp"][basepart.name] = RouteProcessor(basepart)
427         elif re.search(patterns.fantray, basepart.name):
428             self["fantray"][basepart.name] = FanTray(basepart)
429         elif re.search(patterns.fc, basepart.name):
430             self["fc"][basepart.name] = FabricCard(basepart)
431         elif re.search(patterns.pwrmod_f, basepart.name):
432             self["pwrmod"][basepart.name] = PowerModule(basepart)
433         elif re.search(patterns.pwrmod_dist, basepart.name):
434             self["pwrmod"][basepart.name] = PowerModule(basepart)
435         elif re.search(patterns.pwrtray, basepart.name): # must be after! - powertray is a subst
436             self["pwrtray"][basepart.name] = PowerTray(basepart)
```

What do we DO with all of it?



```
▼ VARIABLES
  ▼ Locals
    > special variables
    > class variables
    > argparse = <module 'argparse' from '/auto/mcp-tools-global/python/rhel8/...
    > args = Namespace(all_cmds=False, all_keys=False, commands=None, filename=
    > cmdOutput = {'annotation': 'n/a', 'args_annot': '768B_25LR', 'args_bw':
    > json = <module 'json' from '/auto/mcp-tools-global/python/rhel8/3.8.12-b
    k = 'totalSize'
    > lzma = <module 'lzma' from '/auto/mcp-tools-global/python/rhel8/3.8.12-b
    > parser = ArgumentParser(prog='dumpsnap.py', usage=None, description=None,
  ▼ snap = {'optic': {'FourHundredGigE0/0/0/0': <subparts.Optic object at 0x
    > special variables
    > function variables
      XrVersion = '24.3.1.36I'
    > cmdOutput = {'annotation': 'n/a', 'args_annot': '768B_25LR', 'args_bw':
    > counters = {}
      hostname = 'slugger_4'
      originFile = '/users/lwobker/ws/git/z/powerInventory/test.xz'
    > partTypes = ['optic', 'rack', 'rp', 'lc', 'intf', 'fantray', 'fc', 'pwr
    > rack = <subparts.Rack object at 0x7fe2e9fad700>
    > realLCs = []
    > snapTime = datetime.datetime(2024, 9, 3, 19, 39, 19)
    > 'optic' = {'FourHundredGigE0/0/0/0': <subparts.Optic object at 0x7fe2e9
    > 'rack' = defaultdict(None, {'Rack 0': <subparts.Rack object at 0x7fe2e9
    > 'rp' = {'0/RP0/CPU0': <subparts.RouteProcessor object at 0x7fe2e9fad9a0
    > 'lc' = {'0/RP0/CPU0': <subparts.LineCard object at 0x7fe2e9fad5b0>}
    > 'intf' = {'FourHundredGigE0/0/0/0': <subparts.Interface object at 0x7fe
    > 'fantray' = {'0/FT0': <subparts.FanTray object at 0x7fe2e9fad6d0>, '0/F
    > 'fc' = {}
    > 'pwrtray' = {}
    > 'pwrmod' = {'0/PM0': <subparts.PowerModule object at 0x7fe2e9fada90>, '
    len() = 9
```

cmdOutput

```
▼ cmdOutput = {'annotation': 'n/a', 'args_annot': '768B_25LR', 'args_bw': '25', 'args_leader...
  > special variables
  > function variables
    'annotation' = 'n/a'
    'args_annot' = '768B_25LR'
    'args_bw' = '25'
    'args_leader' = 'Slugger_Traffic'
    'args_pps' = '768'
    'cpaFanAlgo' = 'Optics info: ZRP Upper 0, ZRP Lower 0, ZR Upper 0, ZR Lower 0, 400G 2, 800...
    'envmonTest' = 'connected to RACK scope service\nDump debug set successfully\n'
    'etcHostname' = 'slugger_4\n'
    'etcNodeName' = 'node0_RP0_CPU0\n'
    'piThermalDataRP0' = 'pi thermal_data 0\n\nASIC = 0/RP0/CPU0/0\nThermal Algo State = runni...
    'piThermalDataRP1' = 'pi thermal_data 0\n\nUnable to extract fields from nodeid\n'
```

Data model from a snapshot

- Mostly basic python classes
- Some methods to do math
 - Add up power totals
 - Find average of temp sensors

```
> json = module 'json' from '/auto/mcp-tools-global/python/rhel8/3.8.12-b
> k = 'totalSize'
> lzma = <module 'lzma' from '/auto/mcp-tools-global/python/rhel8/3.8.12-b
> parser = ArgumentParser(prog='dumpsnap.py', usage=None, description=None,
✓ > snap = {'optic': {'FourHundredGigE0/0/0/0': <subparts.Optic object at 0x7fe2e9f
> > special variables
> > function variables
XrVersion = '24.3.1.36I'
> cmdOutput = {'annotation': 'n/a', 'args_annot': '768B_25LR', 'args_bw':
> counters = {}
hostname = 'slugger_4'
originFile = '/users/lwobker/ws/git/pwr/powerInventory/test.xz'
> partTypes = ['optic', 'rack', 'rp', 'lc', 'intf', 'fantray', 'fc', 'pwr'
> rack = <subparts.Rack object at 0x7fe2e9fad700>
> realLCs = []
> snapTime = datetime.datetime(2024, 9, 3, 19, 39, 19)
> 'optic' = {'FourHundredGigE0/0/0/0': <subparts.Optic object at 0x7fe2e9f
> 'rack' = defaultdict(None, {'Rack 0': <subparts.Rack object at 0x7fe2e9f
> 'rp' = {'0/RP0/CPU0': <subparts.RouteProcessor object at 0x7fe2e9fad9a0
> 'lc' = {'0/RP0/CPU0': <subparts.LineCard object at 0x7fe2e9fad5b0>}
> 'intf' = {'FourHundredGigE0/0/0/0': <subparts.Interface object at 0x7fe
> 'fantray' = {'0/FT0': <subparts.FanTray object at 0x7fe2e9fad6d0>, '0/F
> 'fc' = {}
> 'pwrtray' = {}
> 'pwrmod' = {'0/PM0': <subparts.PowerModule object at 0x7fe2e9fada90>, '
len() = 9
```

Parsed data we can use later

```
✓ VARIABLES
  ✓ Locals
    > special variables
    > class variables
    > argparse = <module 'argparse' from '/auto/mcp-tools-global/python/rhel8/...
    > args = Namespace(all_cmds=False, all_keys=False, commands=None, filename:
    > cmdOutput = {'annotation': 'n/a', 'args_annot': '768B_25LR', 'args_bw':
    > json = <module 'json' from '/auto/mcp-tools-global/python/rhel8/3.8.12-bi
    > k = 'totalSize'
    > lzma = <module 'lzma' from '/auto/mcp-tools-global/python/rhel8/3.8.12-bi
    > parser = ArgumentParser(prog='dumpsnap.py', usage=None, description=None,
    > snap = {'optic': {'FourHundredGigE0/0/0/0': <subparts.Optic object at 0x
    > special variables
    > function variables
    > XnVersion = '24.3.1.36I'
    > cmdOutput = {'annotation': 'n/a', 'args_annot': '768B_25LR', 'args_bw':
    > counters = {}
    > hostname = 'slugger_4'
    > originFile = '/users/lwobker/ws/gitz/powerInventory/test.xz'
    > partTypes = ['optic', 'rack', 'rp', 'lc', 'intf', 'fantray', 'fc', 'pwr
    > rack = <subparts.Rack object at 0x7fe2e9fad700>
    > reallCs = []
    > snapTime = datetime.datetime(2024, 9, 3, 19, 39, 19)
    > 'optic' = {'FourHundredGigE0/0/0/0': <subparts.Optic object at 0x7fe2e9
    > 'rack' = defaultdict(None, {'Rack 0': <subparts.Rack object at 0x7fe2e9
    > 'rp' = {'0/RP0/CPU0': <subparts.RouteProcessor object at 0x7fe2e9fad9a0
    > 'lc' = {'0/RP0/CPU0': <subparts.LineCard object at 0x7fe2e9fad5b0>}
    > 'intf' = {'FourHundredGigE0/0/0/0': <subparts.Interface object at 0x7fe
    > 'fantray' = {'0/FT0': <subparts.FanTray object at 0x7fe2e9fad6d0>, '0/F
    > 'fc' = {}
    > 'pwrtray' = {}
    > 'pwrmod' = {'0/PM0': <subparts.PowerModule object at 0x7fe2e9fada90>, '
    len() = 9
```

```
✓ 'pwrmod' = {'0/PM0': <subparts.PowerModule object at 0x7fe2e9fada90>, '0/PM1':
  > special variables
  > function variables
  ✓ '0/PM0' = <subparts.PowerModule object at 0x7fe2e9fada90>
    > special variables
      IinA = 1.5
      IinB = 0.0
      Iout = 24.9
      VinA = 238.5
      VinB = 0.0
      Vout = 12.0
      descr = '300W AC/HVAC/HVDC Power Module with Port-side Air Intake'
      fanSpeed = 'Traceback (most recent call last):\n File "/ws/lwobker-rtp/.vsco
    > fanSpeeds = []
    > inTemps = []
      loadFraction = -298799.99999999994
      modEff = 0.8352201257861634
      name = '0/PM0'
      phantom = False
      pid = 'PSU3KW-HVPI'
      pwrIn = 357.75
      pwrOut = 298.79999999999995
      ratedPwr = -0.001
      serial = 'DTM271801D2'
      status = 'OK'
      temp = -1
    > temps = {'HSNK_Temp': 51, 'Inlet_Temp': 30, 'Outlet_Temp': 39}
      vid = 'V01'
```

Parser .XLS output (one of many formats)

- Pick what fields we want
- Build XLS from those
- Do math!

	B	C	D	E	F	I	J	K	L	M
1	timestamp	rack_pid	annotation	cli_bw	cli_paks	pwrln	pwrOut	total pwr	Inlet Ter	avg_fans
2	08/15 21:07	8711-32FH-M	SteadyState	0	0	1237	1138	92.0%	22	16425
3	08/15 21:08	8711-32FH-M	SteadyState	0	0	1237	1135	91.8%	22	16425
4	08/15 21:08	8711-32FH-M	SteadyState	0	0	1239	1138	91.8%	22	16440
5	08/15 21:09	8711-32FH-M	SteadyState	0	0	1239	1135	91.6%	22	16425
6	08/15 21:10	8711-32FH-M	SteadyState	0	0	1237	1138	92.0%	22	16440
7	08/15 21:22	8711-32FH-M	768B_15LR	15	768	1260	1157	91.9%	21	16455
8	08/15 21:23	8711-32FH-M	768B_15LR	15	768	1260	1157	91.9%	22	16455
9	08/15 21:24	8711-32FH-M	768B_15LR	15	768	1260	1157	91.9%	22	16455
10	08/15 21:25	8711-32FH-M	768B_15LR	15	768	1260	1157	91.9%	22	16440
11	08/15 21:25	8711-32FH-M	768B_15LR	15	768	1262	1157	91.7%	22	16440
12	08/15 21:30	8711-32FH-M	768B_25LR	25	768	1260	1171	93.0%	22	16440
13	08/15 21:30	8711-32FH-M	768B_25LR	25	768	1260	1171	93.0%	22	16425
14	08/15 21:31	8711-32FH-M	768B_25LR	25	768	1260	1171	93.0%	22	16440
15	08/15 21:32	8711-32FH-M	768B_25LR	25	768	1262	1171	92.8%	22	16440
16	08/15 21:33	8711-32FH-M	768B_25LR	25	768	1262	1171	92.8%	22	16440
17	08/15 21:37	8711-32FH-M	768B_50LR	50	768	1305	1205	92.3%	22	16455
18	08/15 21:38	8711-32FH-M	768B_50LR	50	768	1305	1205	92.3%	22	16440
19	08/15 21:39	8711-32FH-M	768B_50LR	50	768	1305	1205	92.3%	22	16440
20	08/15 21:40	8711-32FH-M	768B_50LR	50	768	1305	1206	92.4%	22	16425
21	08/15 21:41	8711-32FH-M	768B_50LR	50	768	1305	1205	92.3%	22	16425
22	08/15 21:43	8711-32FH-M	768B_75LR	75	768	1328	1234	92.9%	22	16425
23	08/15 21:44	8711-32FH-M	768B_75LR	75	768	1328	1235	93.0%	22	16305
24	08/15 21:45	8711-32FH-M	768B_75LR	75	768	1328	1235	93.0%	22	16425
25	08/15 21:46	8711-32FH-M	768B_75LR	75	768	1328	1235	93.0%	22	16425
26	08/15 21:46	8711-32FH-M	768B_75LR	75	768	1328	1236	93.1%	23	16425
27	08/15 22:05	8711-32FH-M	768B_100LR	100	768	1374	1260	91.7%	23	16425
28	08/15 22:06	8711-32FH-M	768B_100LR	100	768	1351	1260	93.3%	23	16425
29	08/15 22:07	8711-32FH-M	768B_100LR	100	768	1351	1261	93.4%	23	16425

Use case / example:

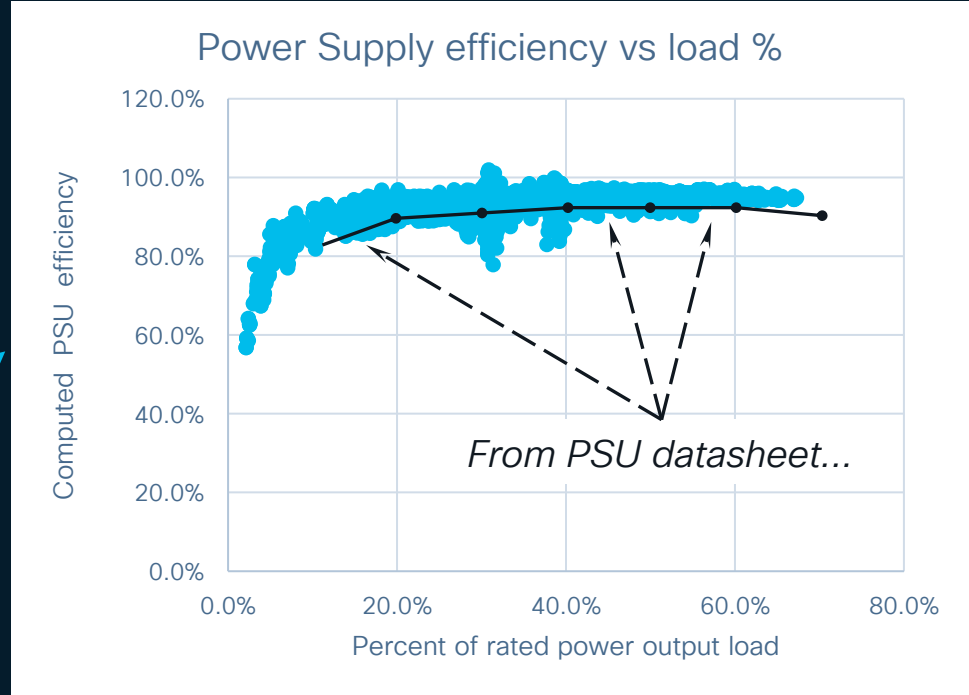
Verify PSU efficiency
data

Data snapshots (~54,000 of them...)

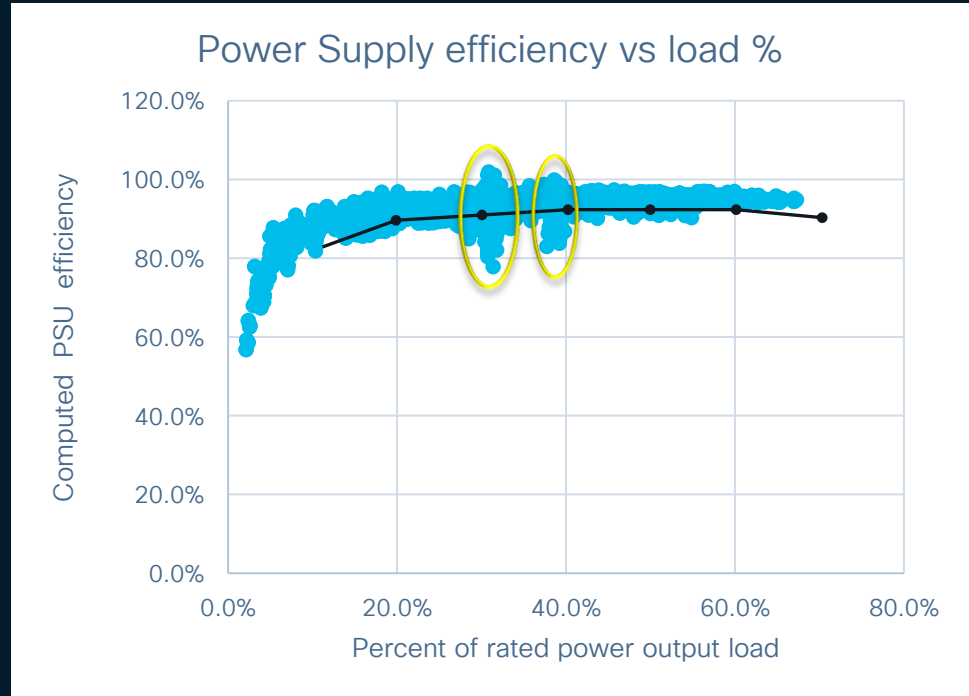
	B	C	G	H	J	K	L	N	O	P	Q	R	S	T	U	V
1	timestamp	rack_pid	Eff	PSU slot	rating	load	PSU eff	linA	linB	VinA	VinB	Iout	Vout	PsuPwrIn	PsuPwrOut	PSU SN
53867	05/23 10:08	8808	88.7%	0/PT1-PM2	6300	16.8%	90.4%	2.6	3	208.8	209.7	19.3	54.9	1172.0	1059.6	DTM222304PS
53868	05/23 11:03	8808	89.7%	0/PT0-PM0	6300	19.4%	89.9%	3.2	3.3	209.7	209.1	22.2	55.1	1361.1	1223.2	DTM2351006N
53869	05/23 11:03	8808	89.7%	0/PT0-PM1	6300	18.6%	89.0%	3.2	3.1	209.7	209.1	21.3	55.1	1319.3	1173.6	DTM271907YW
53870	05/23 11:03	8808	89.7%	0/PT0-PM2	6300	18.6%	89.2%	3.1	3.2	208.5	209.1	21.3	55.1	1315.5	1173.6	DTM271907P0
53871	05/23 11:03	8808	89.7%	0/PT1-PM0	6300	18.2%	89.9%	3	3.1	209.1	209.4	20.9	54.9	1276.4	1147.4	DTM235200AY
53872	05/23 11:03	8808	89.7%	0/PT1-PM1	6300	20.1%	89.7%	3.3	3.4	210.3	210.8	23	55	1410.7	1265.0	DTM2346014T
53873	05/23 11:03	8808	89.7%	0/PT1-PM2	6300	18.6%	90.4%	3	3.2	209.1	210.3	21.4	54.9	1300.3	1174.9	DTM222304PS
53874	05/23 11:29	8808	90.3%	0/PT0-PM0	6300	20.4%	91.6%	3.3	3.4	209.7	208.8	23.3	55.1	1401.9	1283.8	DTM2351006N
53875	05/23 11:29	8808	90.3%	0/PT0-PM1	6300	19.7%	88.5%	3.4	3.3	209.4	208.8	22.5	55.1	1401.0	1239.8	DTM271907YW
53876	05/23 11:29	8808	90.3%	0/PT0-PM2	6300	19.6%	89.6%	3.3	3.3	208.5	208.8	22.4	55.1	1377.1	1234.2	DTM271907P0
53877	05/23 11:29	8808	90.3%	0/PT1-PM0	6300	19.0%	89.5%	3.1	3.3	208.8	209.1	21.8	54.9	1337.3	1196.8	DTM235200AY
53878	05/23 11:29	8808	90.3%	0/PT1-PM1	6300	21.0%	91.3%	3.4	3.5	210	210.6	24.1	55	1451.1	1325.5	DTM2346014T
53879	05/23 11:29	8808	90.3%	0/PT1-PM2	6300	19.7%	91.2%	3.1	3.4	209.1	209.7	22.6	54.9	1361.2	1240.7	DTM222304PS
53880	05/23 11:56	8808	90.6%	0/PT0-PM0	6300	21.3%	91.7%	3.4	3.6	209.7	209.1	24.4	55.1	1465.7	1344.4	DTM2351006N
53881	05/23 11:56	8808	90.6%	0/PT0-PM1	6300	20.6%	90.0%	3.5	3.4	209.7	209.1	23.6	55.1	1444.9	1300.4	DTM271907YW
53882	05/23 11:56	8808	90.6%	0/PT0-PM2	6300	20.6%	89.8%	3.4	3.5	208.8	209.1	23.5	55.1	1441.8	1294.9	DTM271907P0
53883	05/23 11:56	8808	90.6%	0/PT1-PM0	6300	19.9%	89.4%	3.3	3.4	209.4	209.4	22.9	54.8	1403.0	1254.9	DTM235200AY
53884	05/23 11:56	8808	90.6%	0/PT1-PM1	6300	21.9%	90.8%	3.6	3.6	210.3	211.1	25.1	54.9	1517.0	1378.0	DTM2346014T
53885	05/23 11:56	8808	90.6%	0/PT1-PM2	6300	20.5%	92.0%	3.2	3.5	209.4	210.3	23.6	54.8	1406.1	1293.3	DTM222304PS
53886	05/23 10:41	8808	89.3%	0/PT0-PM0	6300	18.5%	90.4%	3	3.1	211.1	210.6	21.1	55.1	1286.2	1162.6	DTM2351006N
53887	05/23 10:41	8808	89.3%	0/PT0-PM1	6300	17.8%	87.0%	3.1	3	211.1	210.6	20.3	55.1	1286.2	1118.5	DTM271907YW
53888	05/23 10:41	8808	89.3%	0/PT0-PM2	6300	17.8%	88.6%	3	3	210.3	210.3	20.3	55.1	1261.8	1118.5	DTM271907P0
53889	05/23 10:41	8808	89.3%	0/PT1-PM0	6300	17.3%	89.1%	2.8	3	210.6	210.8	19.8	55	1222.1	1089.0	DTM235200AY
53890	05/23 10:41	8808	89.3%	0/PT1-PM1	6300	19.2%	90.5%	3.1	3.2	211.4	212	21.9	55.1	1333.7	1206.7	DTM2346014T

Scatter plots are human-analysis-friendly

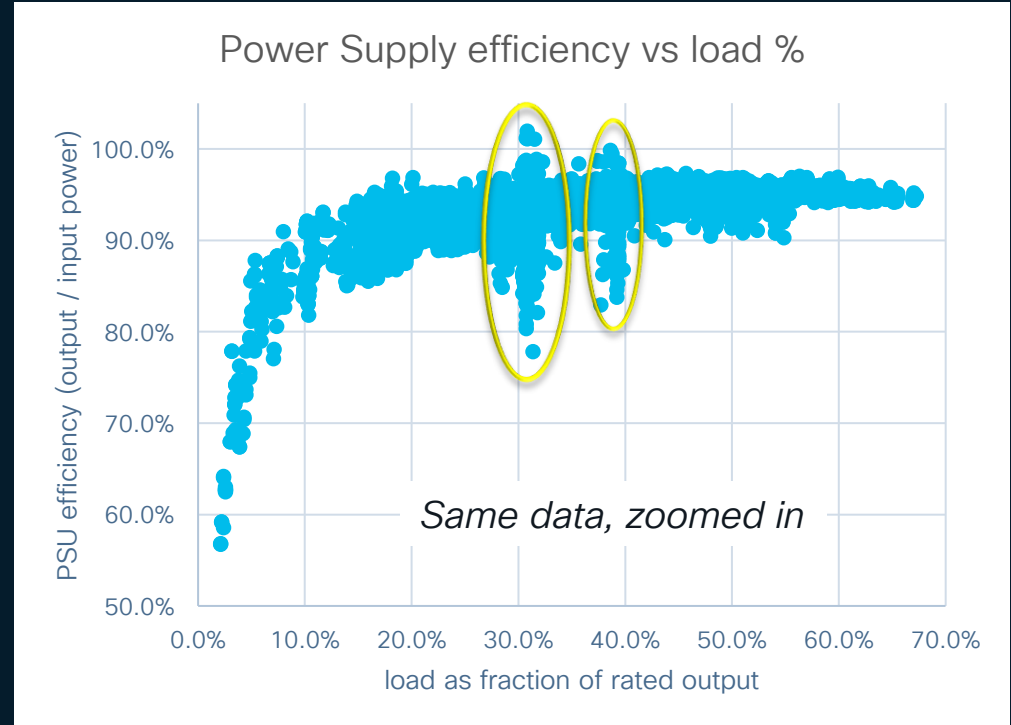
	B	C	G	H	J	K	L	N
1	timestamp	rack_pid	Eff	PSU slot	rating	load	PSU eff	linA
53867	05/23 10:08	8808	88.7%	0/PT1-PM2	6300	16.8%	90.4%	2.6
53868	05/23 11:03	8808	89.7%	0/PT0-PM0	6300	19.4%	89.9%	3.2
53869	05/23 11:03	8808	89.7%	0/PT0-PM1	6300	18.6%	89.0%	3.2
53870	05/23 11:03	8808	89.7%	0/PT0-PM2	6300	18.6%	89.2%	3.1
53871	05/23 11:03	8808	89.7%	0/PT1-PM0	6300	18.2%	89.9%	3
53872	05/23 11:03	8808	89.7%	0/PT1-PM1	6300	20.1%	89.7%	3.3
53873	05/23 11:03	8808	89.7%	0/PT1-PM2	6300	18.6%	90.4%	3
53874	05/23 11:29	8808	90.3%	0/PT0-PM0	6300	20.4%	91.6%	3.3
53875	05/23 11:29	8808	90.3%	0/PT0-PM1	6300	19.7%	88.5%	3.4
53876	05/23 11:29	8808	90.3%	0/PT0-PM2	6300	19.6%	89.6%	3.3
53877	05/23 11:29	8808	90.3%	0/PT1-PM0	6300	19.0%	89.5%	3.1
53878	05/23 11:29	8808	90.3%	0/PT1-PM1	6300	21.0%	91.3%	3.4
53879	05/23 11:29	8808	90.3%	0/PT1-PM2	6300	19.7%	91.2%	3.1
53880	05/23 11:56	8808	90.6%	0/PT0-PM0	6300	21.3%	91.7%	3.4
53881	05/23 11:56	8808	90.6%	0/PT0-PM1	6300	20.6%	90.0%	3.5
53882	05/23 11:56	8808	90.6%	0/PT0-PM2	6300	20.6%	89.8%	3.4
53883	05/23 11:56	8808	90.6%	0/PT1-PM0	6300	19.9%	89.4%	3.3
53884	05/23 11:56	8808	90.6%	0/PT1-PM1	6300	21.9%	90.8%	3.6
53885	05/23 11:56	8808	90.6%	0/PT1-PM2	6300	20.5%	92.0%	3.6
53886	05/23 10:41	8808	89.3%	0/PT0-PM0	6300	18.5%	90.4%	3
53887	05/23 10:41	8808	89.3%	0/PT0-PM1	6300	17.8%	87.0%	3.1
53888	05/23 10:41	8808	89.3%	0/PT0-PM2	6300	17.8%	88.6%	3
53889	05/23 10:41	8808	89.3%	0/PT1-PM0	6300	17.3%	89.1%	2.8
53890	05/23 10:41	8808	89.3%	0/PT1-PM1	6300	19.2%	90.5%	3.1
53891	05/23 10:41	8808	89.3%	0/PT1-PM2	6300	17.8%	90.0%	2.8
53892	05/23 09:13	8808	89.7%	0/PT0-PM0	6300	13.8%	89.6%	2.2
53893	05/23 09:13	8808	89.7%	0/PT0-PM1	6300	13.3%	88.5%	2.3
53894	05/23 09:13	8808	89.7%	0/PT0-PM2	6300	13.1%	89.2%	2.2
53895	05/23 09:13	8808	89.7%	0/PT1-PM0	6300	12.8%	89.0%	2.1



Plot. What's wrong with this picture?



OK, so now we KNOW something is off...

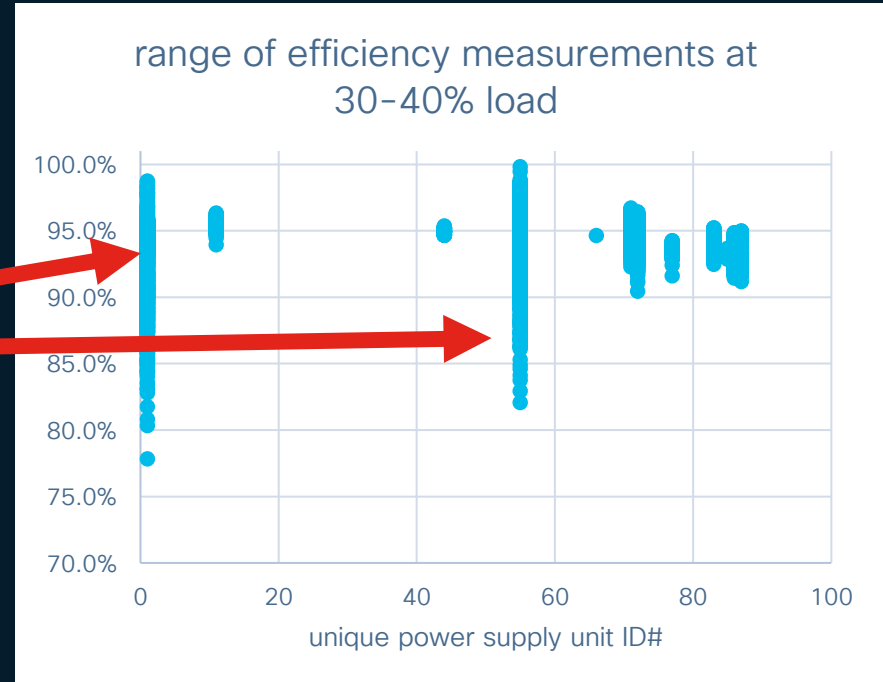


Filter for a known load range and look for outliers

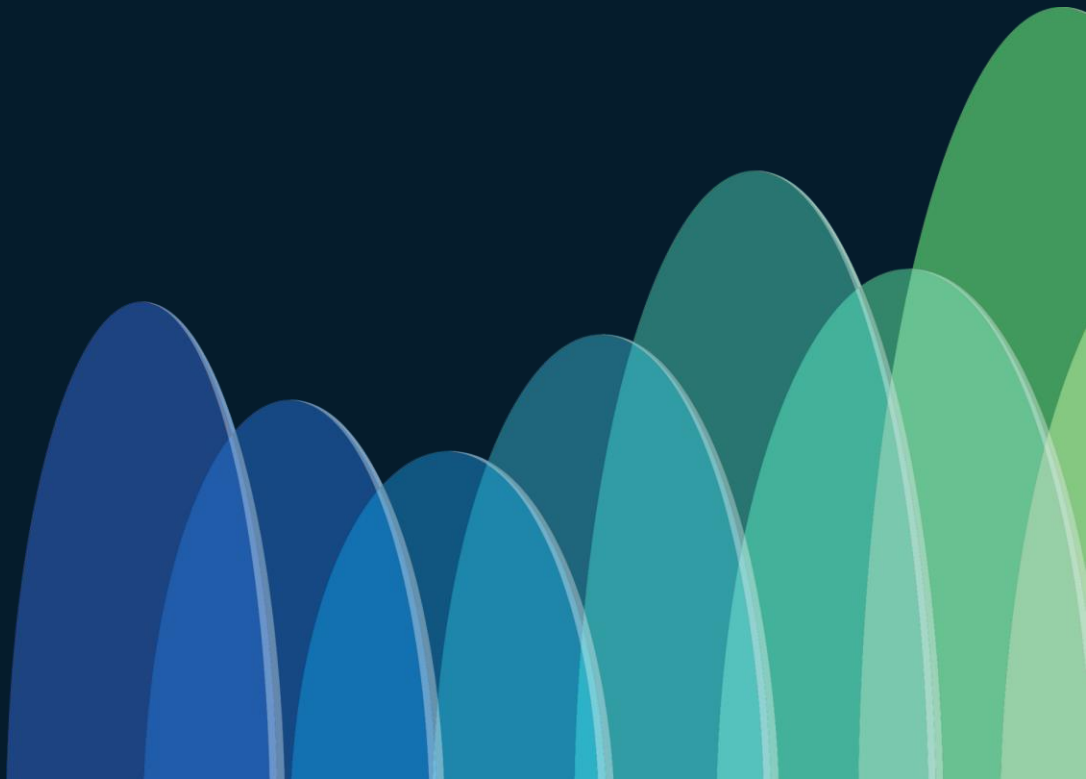
> 100% is not possible!

2 outliers with WAY too wide a range

These two are the troublemakers!



Power usage by component



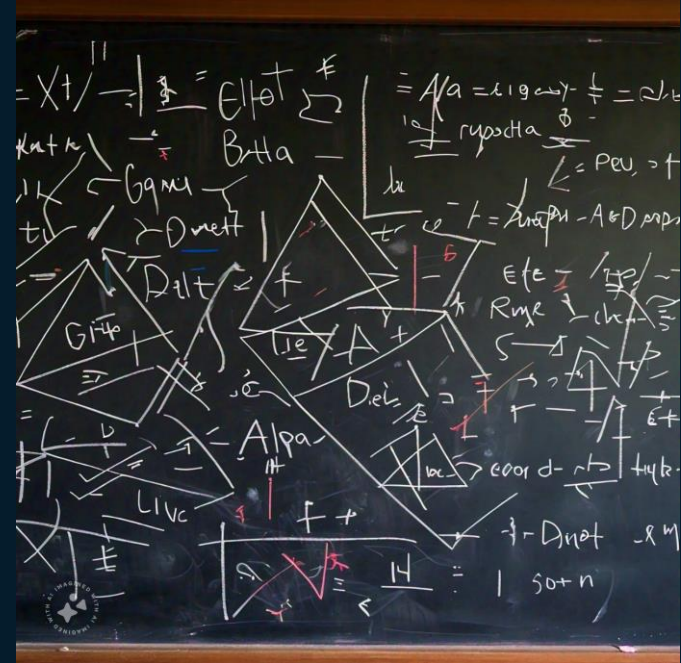
What are the major components?

- Linecards – move packets
 - NPU's, optics, other “big” chips
- Fabric cards – data path interconnect
 - Fabric elements
- Route Processors – control plane
- Fans and Fan trays – move air
- Power supplies – supply power (ok, sorry...)
- Lots of copper – leaks power as heat



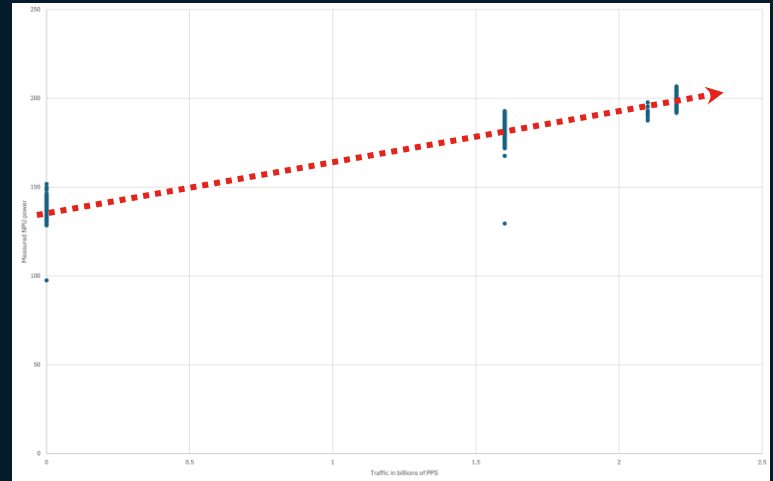
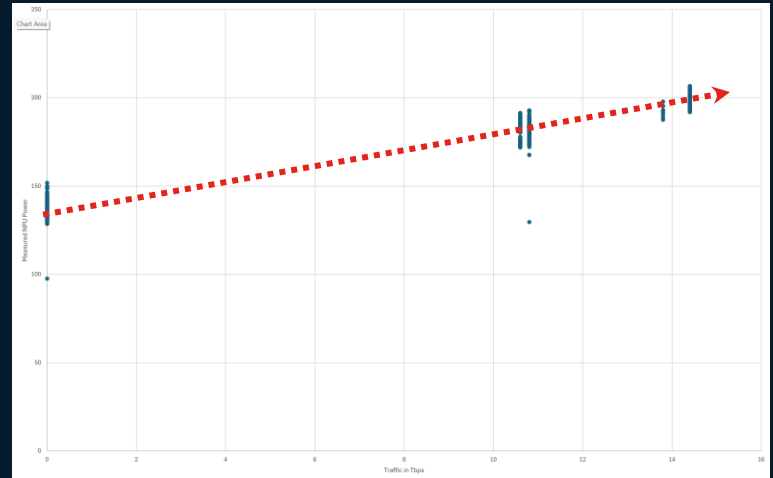
What are the major variables?

- System configuration
 - cards, fans, fabric, etc.
- Ambient temperature
- Optics modules, interface states
- “Touch every bit” features
 - particularly crypto
- Bandwidth load
- Packet rates



NPU power

- NPU power is a function of:
 - Static power (always on)
 - # of serdes on
 - Traffic rate
 - Bandwidth rate
- Power is Y-axis
- X axis is:
 - Billions of PPS (top)
 - Terabits / sec (bottom)



Multivariate analysis

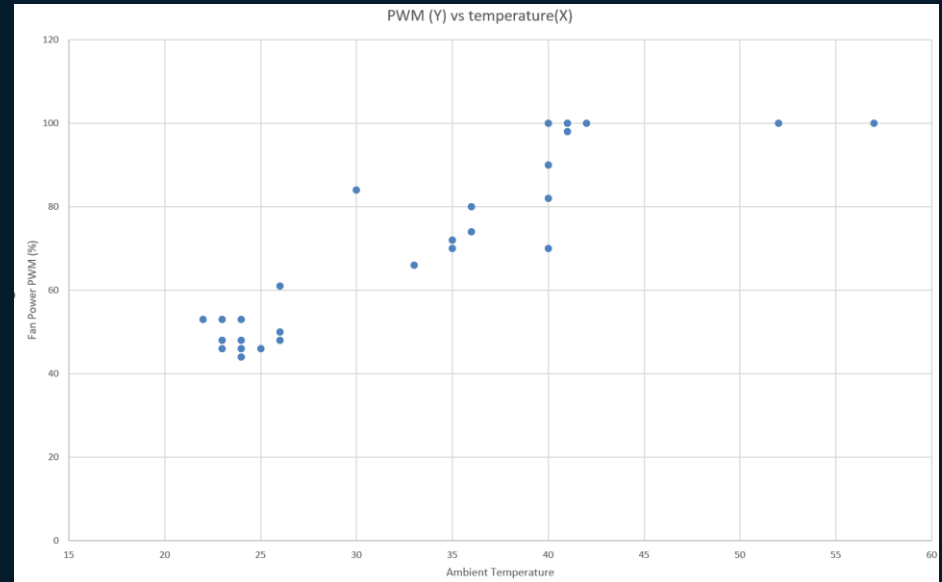
- Lots of data
- Isolate the variables you care about
- Run regression (excel, python, etc.)
- Sanity check coefficients
- R-square values



SUMMARY OUTPUT						
Regression Statistics						
Multiple R	0.992938927					
R Square	0.985927713					
Adjusted R Square	0.985702556					
Standard Error	3.377064167					
Observations	128					
ANOVA						
	df	SS	MS	F	Significance F	
Regression	2	99877.81169	49938.90584	4378.853317	1.8741E-116	
Residual	125	1425.570298	11.40456239			
Total	127	101303.382				
	Coefficients	Standard Error	t Stat	P-value	Lower 95%	Upper 95%
Intercept	1.539084469	0.636014358	2.419889502	0.016964128	0.280333119	2.79783582
cli_tbps	3.289766369	0.088285996	37.26260698	1.50997E-69	3.115037426	3.464495312
cli_Bpps	12.78342241	0.235298537	54.32852472	8.2038E-89	12.3177374	13.24910742

Fan power

- Fan speed controlled by PWM setting
- PWM is a function of:
 - Ambient temp sensor
 - Hundreds of sensor readings
 - System HW config
 - System SW config



(Slightly) easier ones...

- Optics
 - How many optics?
 - Which types?
 - Measure marginal power and divide...
- Internal serdes links
 - How many are there?
 - How many are up/up?
 - Measure marginal power and divide...



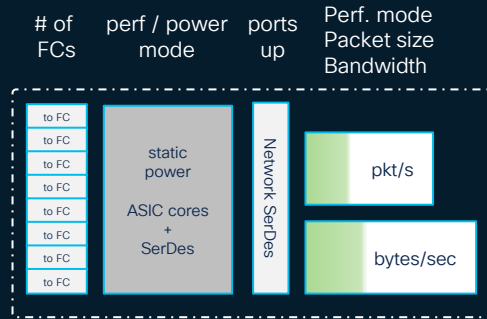
Results?

- Coefficients we can use
- Build power model for planning
- Track over time

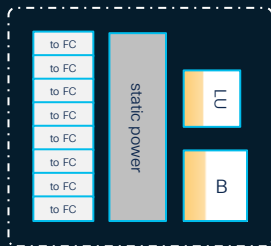


$$W * \alpha + X * \beta + Y * \rho + Z * \lambda$$

Power usage by component – to scale



Line Card
88-LC0-36FH

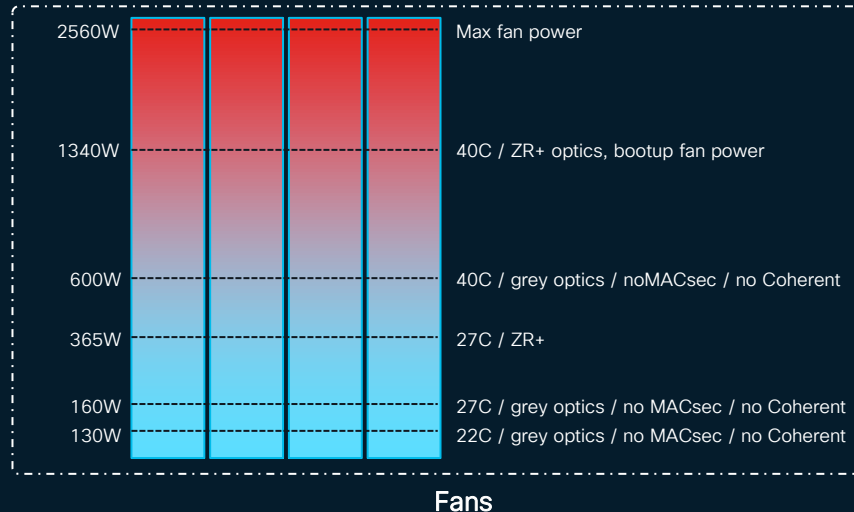
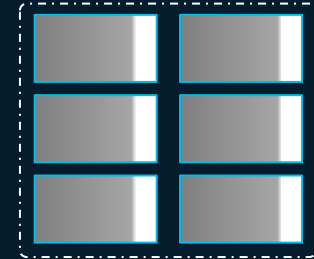


Fabric Card
8808-FC0

Optics

100G	3.5W
400G	11W
400G ZR	19W
400G ZR+	23W
Modules + POL	

Ambient Temp
Line cards installed
Optics installed



PSU : 450-550W

- ASIC core idle power
- SerDes power (~static when on)
- Dynamic LC power (lookups + bytes)
- FC / LC power (lookups + bytes)
- Fan power (dynamic @ runtime)
- PSU efficiency losses

Provisioned power vs. consumed power

Provisioned vs consumed power

- Why do we care?
- Who is paying for what?
- How wide can the range be?



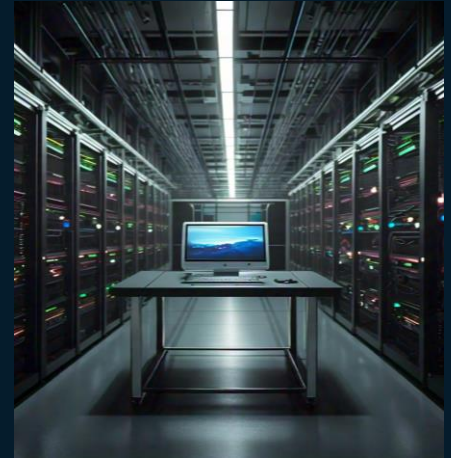
Redundancy models (and why they aren't)

- What can fail?
- That you can control...
- Overlap analysis
- “Big” vs “small” failures



What are you paying for?

- Space?
- Power?
- Future power?
- UNUSED power?!?!?
- We used to “size” systems by ... size.
- But now we should probably...
- “Size” systems by “power”.



Reducing power...

Saving power : context setting



Saving power : context setting

- Balance technical answers and business realities
- Reminder: this presentation is mostly technical
 - (But your business objectives may not be)
- Understand “who is paying for what”
- How are second-order effects measured?

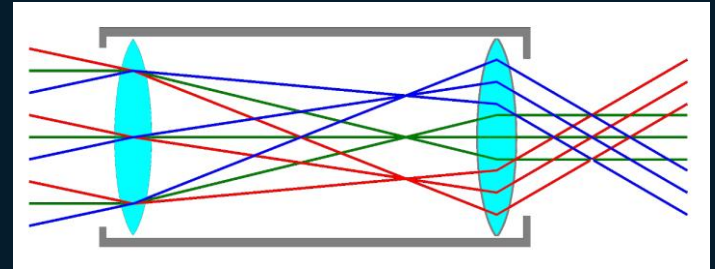
Saving power : facilities

- HVAC/HVDC is slightly more efficient
- Provision proper numbers of PSUs
 - Don't load up the system “just because”
- Ambient temperature ranges matter
 - And they sometimes matter A LOT!



Saving power : optics

- Saving entire systems matters -- A LOT
- Yes, that was a sneaky advertisement for RON...
- High power optics are thermally sensitive
- Planning where they go can (sometimes) save big



Saving power : vendor / OS

- Can the system turn off things that aren't being used?
- DOES the system turn them off?
- Do you have to do anything?
- How do you track / report this?



Saving power : exchange rates

- What is cheap today may not be cheap tomorrow...
- Logic v. analog v. optics
- Step functions like whole systems



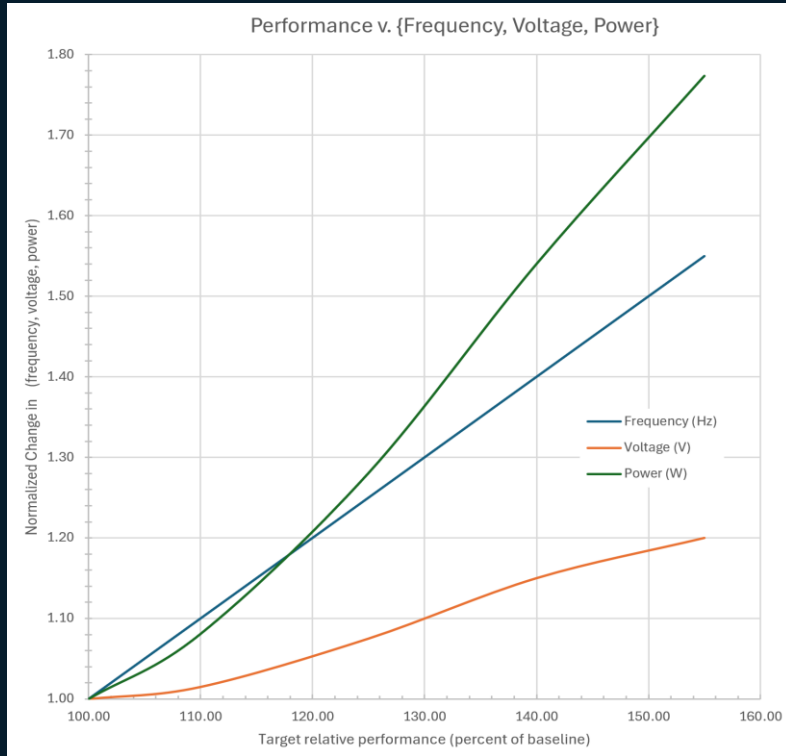
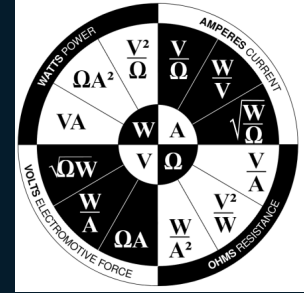
Frequency & voltage savings

$$P_{dynamic} = \beta \times C_L \times V_{cc}^2 \times f$$

- I like fast! let's increase performance by 10% ...
 - 10% higher frequency, 2% increase in voltage
 - ~ 8% increase in total power
- I like faster! let's increase by 10% again! ...
 - 10% higher frequency, but now 4% in voltage
- and so on...



How many PPS do you actually need?



$$P_{total} = P_{static} + P_{dynamic}$$

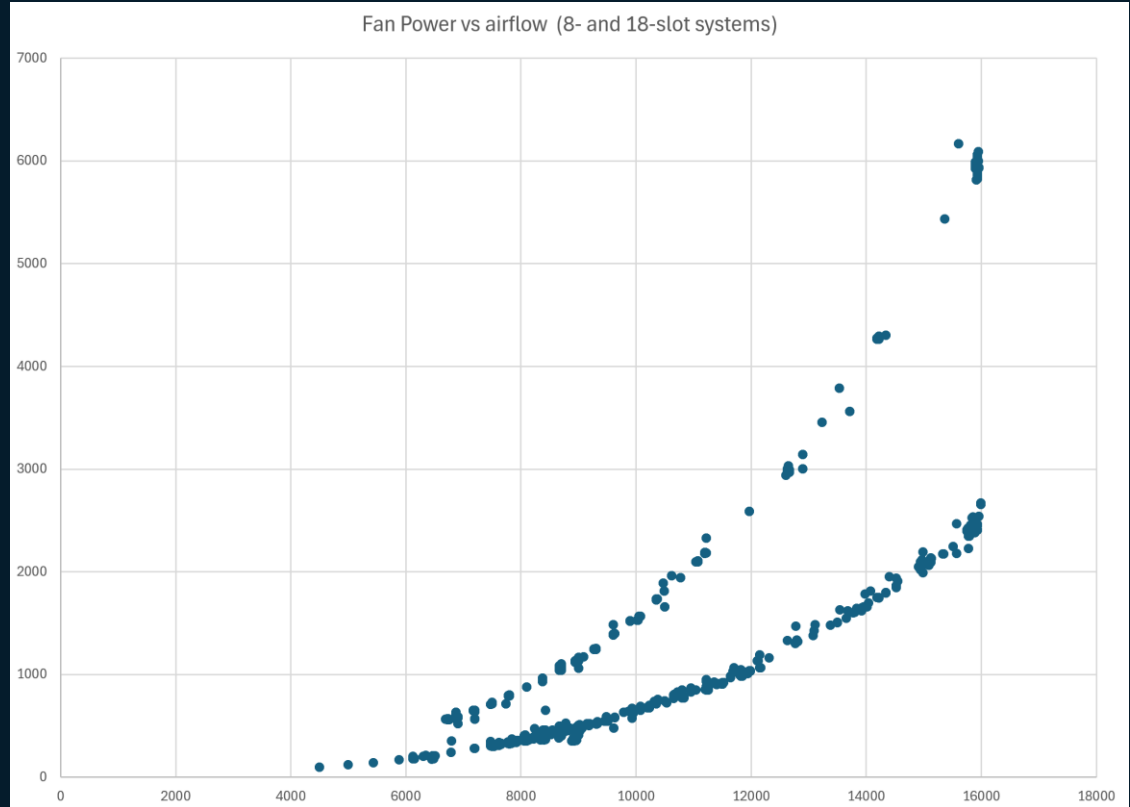
$$P_{static} = V_{cc} \times I_{cc}$$

$$P_{dynamic} = \beta \times C_L \times V_{cc}^2 \times f$$

High ambient temperatures are expensive!

$$\frac{P_1}{P_2} = \left(\frac{N_1}{N_2} \right)^3 * \left(\frac{D_1}{D_2} \right)^3$$

$$\frac{Q_1}{Q_2} = \frac{N_1}{N_2} * \frac{D_1}{D_2}$$



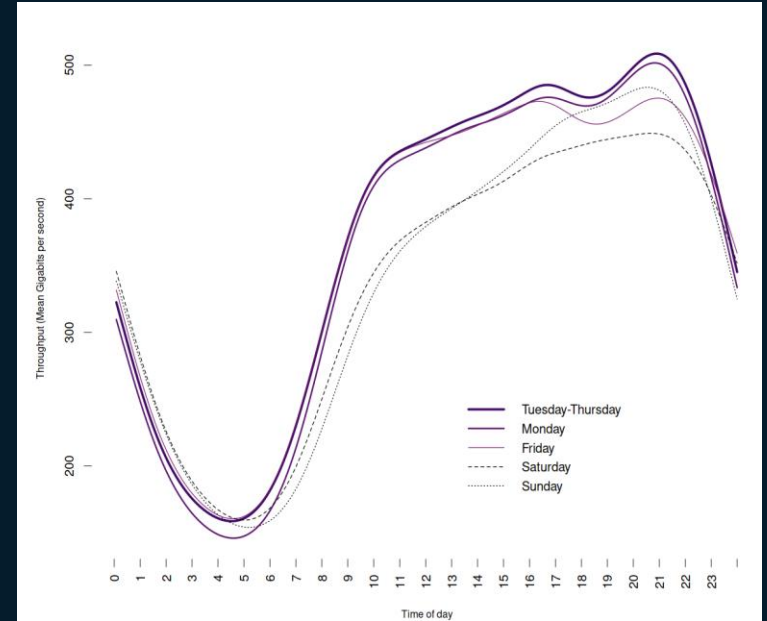
Saving power : network

- System selection : “horses for courses”
- Small / fixed systems vs large / distributed
- “Network” vs “facilities”



Saving power : diurnal / annual patterns

- Most provider networks have strong daily activity patterns
- ~5x load in 8-hour span (9pm-5am)
- Often also have seasonal peaks and troughs



Toolboxes

- Where is the power-to-value ratio wrong?
- Choose systems that make sense
- What can you get rid of?
- What can you collapse?
- Start pricing services accordingly (!!!)



*We can switch about 231 million
packets per joule of energy:
or ~5 nanojoules / packet*

Who: -- me!

When - after running a bunch of these tests!



Thank you



#CiscoLiveAPJC

Keynote Deep Dives

Wednesday
10:30am – 11:30am



Experiences Amplified:
How AI Can Fuel
Better Employee
and Customer
Experiences

Level 1
Room 106



Smart, Secure,
Seamless:
Transforming
Experiences with
Next-Generation
Networking

Level 2
Room 204



Harness a Bold
New Era:
Transform Data
Centre and
Service Provider
Connectivity

Level 2
Room 203



Securing User to
Application and
Everything in
Between

Level 2
Melbourne Room 2



Unlocking
Digital Resilience
through Unified
Observability

The HUB
Centre Stage

Complete Your Session Evaluations



Complete a minimum of 4 session surveys and the Overall Event Survey to claim a **Cisco Live T-Shirt**.



Complete your surveys in the **Cisco Live mobile app**.



Continue your education

- Visit the Cisco Stand for related demos
- Book your one-on-one Meet the Expert meeting
- Attend the interactive education with DevNet, Capture the Flag, and Walk-in Labs
- Visit the On-Demand Library for more sessions at www.CiscoLive.com/on-demand



Thank you

CISCO *Live!*

#CiscoLiveAPJC

CISCO *Live!*

GO BEYOND

#CiscoLiveAPJC