

The Network Certification Description Language

University of Colorado - Colorado Springs
Master's Thesis Defense
February 16th, 2017
Cody Hanson

Executive Summary.

- Develops a new theory for formal verification of network behavior
- A New language tool is presented:
 - Network Certification Description Language (NETCDL)
- Evaluation Outcomes:
 - The language is simple to understand for human users
 - Supports the majority of common use cases
- Discusses reference design for a NETCDL certifier, and how certifiers can become important tools for network engineers
- Defined as an open standard.

Motivation:

Modern networks are complex and costly to install and verify.

Did I get my VLAN's right?

Is there connectivity to the Internet?

Are the resources my users need available?

How can you be sure?

Are my routing tables operating correctly?

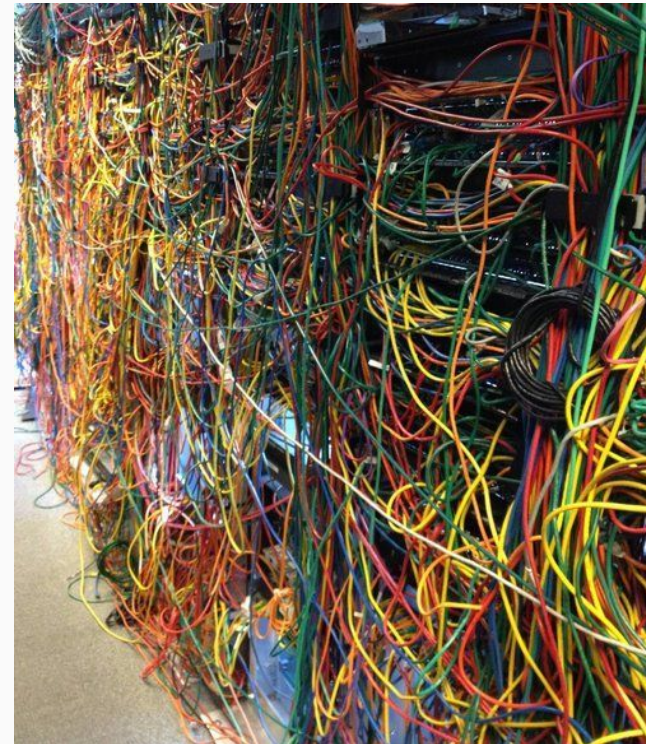
Can an unauthorized user reach a secured area?



WAIT
i'll fix it

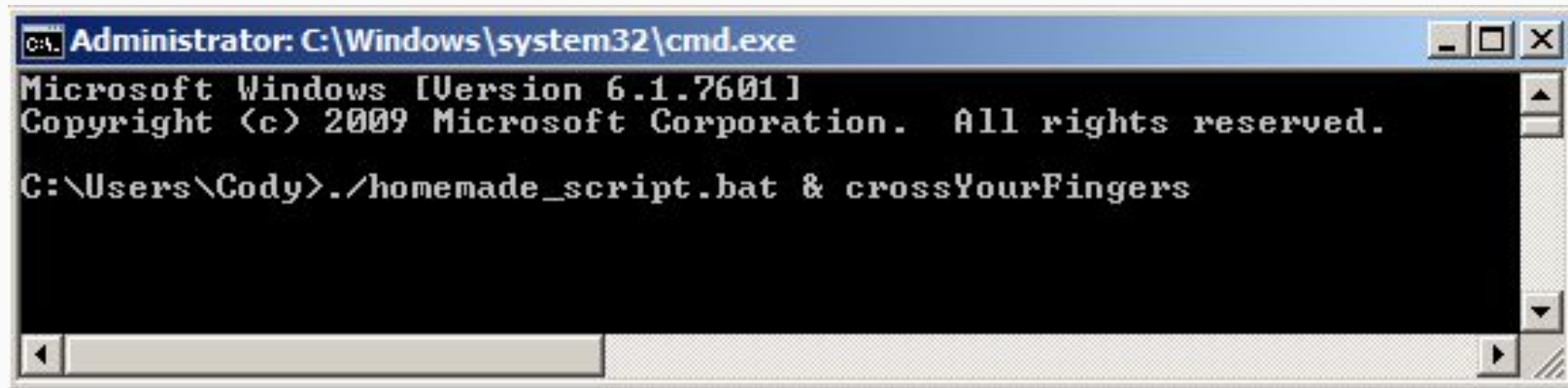
Sources of Network Complexity

- Sophisticated Routers, Switches, and Firewalls
- Evolving Wireless deployments
- Moves, Adds, and Changes
- Old, Unknown, or Unverified cabling
- Distributed campuses and remote sites
- Nobody wrote anything down



Current Verification Strategies

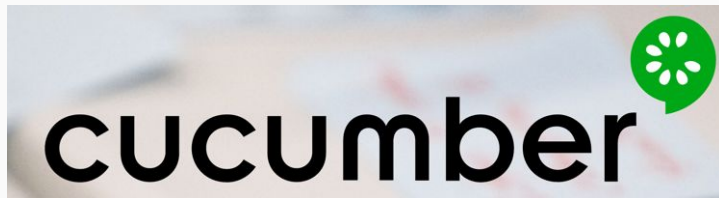
- Home-made scripts (requires programming knowledge)
- Plugging in the laptop + ad-hoc testing
- Commercial network tools (expensive and rigid)
- Waiting for a problem to be reported (impacts users)



A screenshot of a Windows command prompt window. The title bar reads "Administrator: C:\Windows\system32\cmd.exe". The window contains the following text: "Microsoft Windows [Version 6.1.7601]", "Copyright (c) 2009 Microsoft Corporation. All rights reserved.", and the command "C:\Users\Cody>./homemade_script.bat & crossYourFingers". The command prompt is currently at the end of the command line.

```
C:\Users\Cody>./homemade_script.bat & crossYourFingers
```

A wealth of verification tools are available for Hardware and Software Engineers



Other Engineers
can verify their
work, why not
Network Engineers
as well?

NETCDL

The Network Certification
Description Language

A domain specific
language for making
assertions about network
connectivity

Designed for humans

#Map IP addresses to names

```
define host 10.0.0.1 as myRouter
define host 10.0.0.105 as myWebserver
define network 192.168.1.0/24 as secured_network
```

#Describe link behavior

```
link speed should be 1000Mbps
link duplex should be full
```

#Describe VLANs

```
vlan should be 500
```

#Describe DHCP behavior

```
dhcp server should be myRouter
dhcp network should be 10.0.0.0/24
dhcp dns should be 8.8.8.8
dhcp gateway should be myRouter
```

#Describe behavior of the myRouter

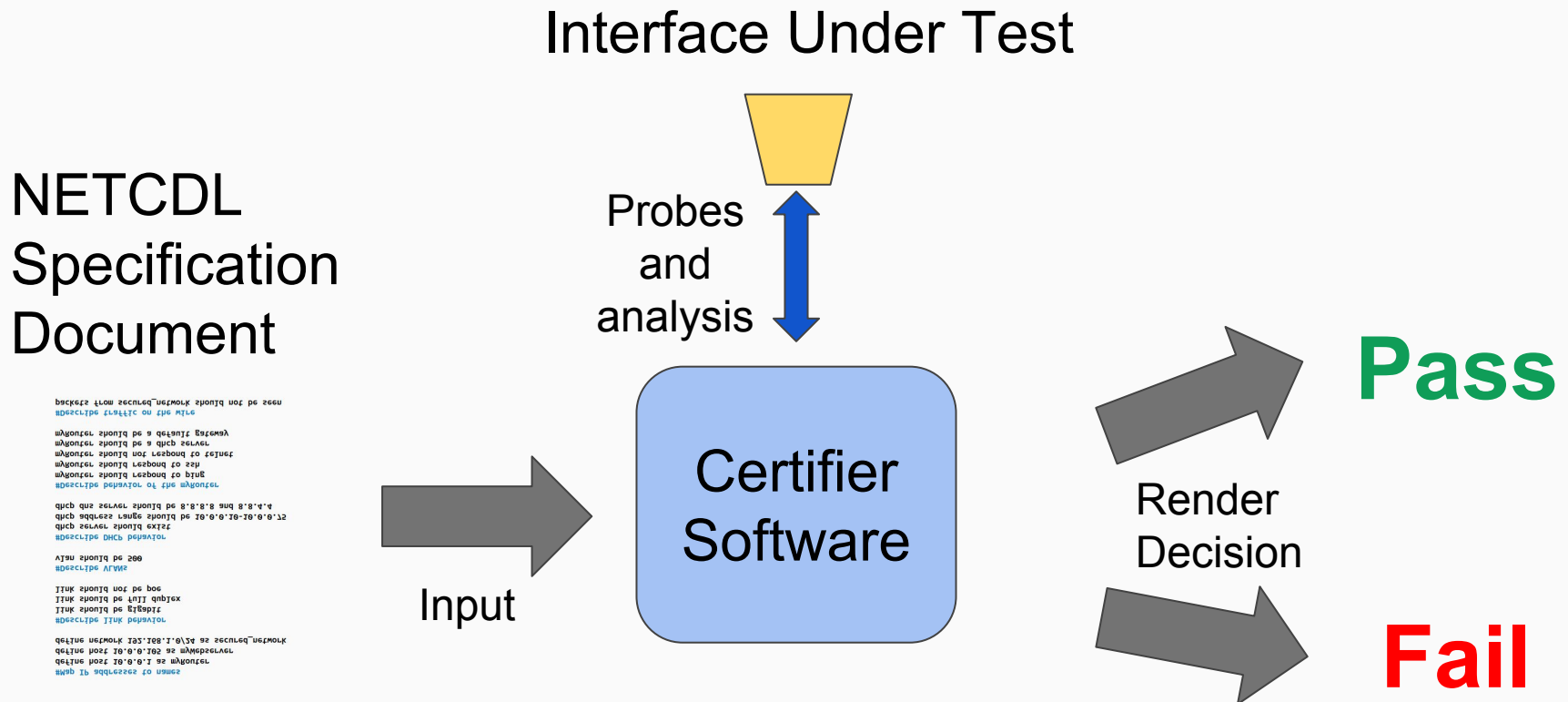
```
myRouter should be reachable by ping
myRouter should be reachable on TCP port 22
myRouter should not be reachable on TCP port 23
```

#Describe traffic on the wire

```
packets from secured_network should not be seen
Packets with type 0x18 should not be seen
```

#And More...

Network Certification Use Model



Impact of NETCDL

Installing and maintaining a network becomes...

- Documented
- Version Controlled
- Reproducible
- More Accessible

Language Grammar Design

Language Goals

1. Enable users to describe how networks should behave
2. Be easy for humans to write. Easier than programming.
3. Be easy for humans to read. Even if not familiar with the language.

English Phrases;
No complicated syntax;
Context free grammar

`link speed should be 1000Mb/s`

`dhcp network should be 192.168.1.0/24`

`SecuredServer should not be reachable by ping`

`http server at 10.0.0.10 should serve "/index.html"`

`packets from host 10.250.0.1 should not be seen`

```
<should-expr> ::= "should" | "should not"
<link-statement> ::= <link-speed> | <link-duplex>

<link-speed> ::= "link speed" <should-expr> "be"
    <speed>
<link-speed> ::= "link duplex" <should-expr> "be"
    <duplex>

<duplex> ::= "full" | "half"
<speed> ::= "10Mb" | "100Mb" | "1000Mb" | "1Gb"

<port-open-statement> ::= <reachable> "on"
    <transport> "port" <port-number>
<ping-statement> ::= <reachable> "by ping"
```

Reference Certifier Design

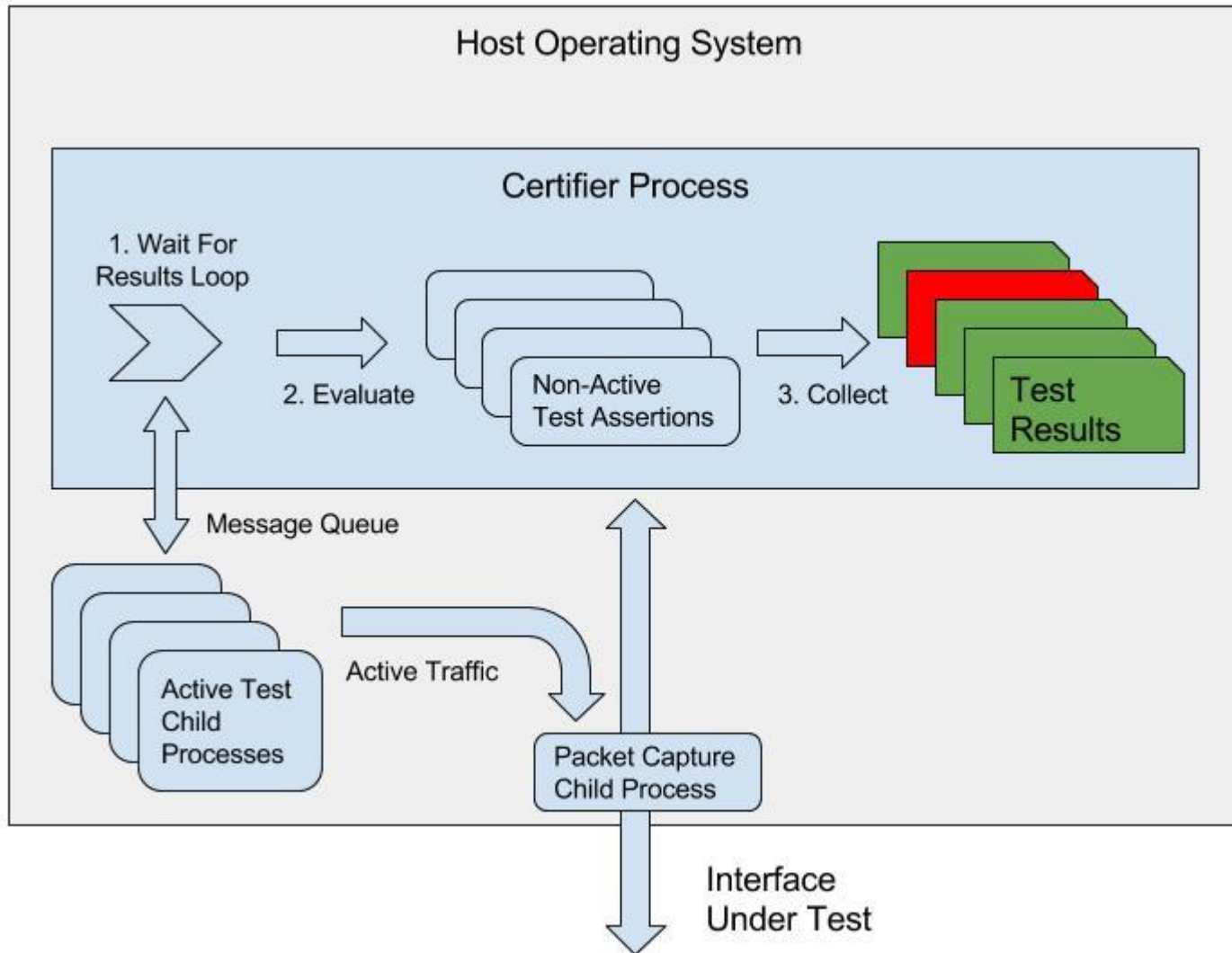
A Certifier's Job:

1. Parse the input NETCDL statements
2. Evaluate the statements for truth
3. Report results
4. “Leave no trace” -- Relinquish key resources

Reference Design Philosophy

- Promote Extensible design
- Promote Performant design
- Illustrate use of the language,
demonstrate the certification concept.

Certifier Software Blocks



Python

textX[16]

Scapy[6]

Example: Simple Certifier Output

```
link speed should be 1000mb/s
link speed should not be 10mb/s
link speed should not be 100mb/s
link duplex should be full
link duplex should not be half
access vlan should be 500
domain name google.com should resolve using server 8.8.8.8
domain name ent.local should resolve using server 8.8.8.8
iperf download from ent.local should be at most 30Kbps
domain name public.company.com should not resolve using server 8.8.8.8
domain name ent.local should resolve using server 192.168.1.1
http server at 192.168.1.1 should not serve /path/to/file on port 8080
http server at google.com should serve /index.html on port 80
tftp server at ent.local should serve afile on port 69
tftp server at ent.local should not serve missingfile on port 69
tftp server at 192.168.1.144 should serve afile on port 69
iperf upload to ent.local should be at least 30Kbps
ftp server at speedtest.tele2.net should not serve missingfile on port 21
ftp server at speedtest.tele2.net should serve 1KB.zip on port 21
iperf upload to fake.local should be at least 30Kbps
ping to 192.168.1.1 should succeed:True
traceroute to 10.0.0.1 should traverse 1.2.3.4
google.com should not be reachable on TCP port 25
google.com should not be reachable on UDP port 100
1.2.3.4 should be reachable on UDP port 100
192.168.1.1 should not be reachable on TCP port 23
```

Evaluation

Evaluation - Key Questions

- How complex is the language?
 - We want it to feel like a natural language
 - Improves adoption and widens audience
- Can users express their ideas?
 - Must adapt to real life situations and demands
- Is the certifier software high quality?
 - To serve as reference implementation

Measuring Grammar Complexity

- An objective evaluation of language properties
- Grammar metrics suite SdfMetz [3]
- Compare results to other well known computer languages

EBNF: `<ping-statement> ::= <reachable> "by ping"`

SDF: `ShouldBeReachable ByPing -> PingStatement`

Grammar Complexity Metrics

Metric	Description	Practical Impact
TERM	Number of unique terminals	Impacts the size of the vocabulary a user must comprehend
VAR	Number of defined non-terminals	Large VAR can increase program maintenance cost due to cascading effects to rest of grammar
PROD	Number of defined production rules	More production rules imply more rules governing the structure of the grammar
MCC	McCabe's Cyclomatic Complexity. Number of linearly independent paths (or decisions) through a graph. Related to PROD.	Measures cognitive impact on user, due to branch complexity
TIMP	Tree Impurity	Measures to what degree that the parse tree of the grammar is actually a tree. 0% means the graph is a perfect tree, 100% means the graph is fully connected
DEP	Size of largest level	Maximum Number of non-terminals in a level of the parse tree. Higher numbers denote wider trees, which increase grammar complexity
HEI	Maximum Height of Parse Tree	Taller parse trees denote more complex grammar structure
E	Program Effort - Sometimes referred to as Halstead Effort [39]	Computation that combines frequency of occurrence for operators and operands into a single number. Useful for comparing complexity between grammars of different sizes.

Grammar Metrics Comparison

[Reference Dataset](#)

Metric Name	NETCDL Value	Ranking out of 31 (lower is better)	Comparable Languages
TERM	49.0	5th	MatLab, XPath
VAR	24.0	3rd (Tie)	C (Grammar Base/SDF)
PROD	42.0	3rd	FORTRAN 77
MCC	29.0	2nd	FORTRAN 77
TIMP	3.162%	1st	-
DEP	1	1st (Tie)	BibTex
HEI	6	3rd (Tie)	BibTex, MatLab
E (thousands)	12.913	3rd	MatLab

Table 2: NETCDL Grammar Performance vs SdfMetz Grammar Dataset

Evaluating Language Expressiveness

- Identify important use cases
- Reference respected texts
 - Cisco Interconnecting Network Devices Vol 1 and 2 [35][36]
 - Network Maintenance and Troubleshooting Guide Fluke Networks [2]



Use Case	Description	Sources	NETCDL Support
Ping	ICMP Echo, verifies layer 3 connectivity	[2][35]	Yes
TCP Port Open	Verifies that a TCP port on a remote host is open and reachable	[2]	Yes
UDP Port Open	Verifies that a UDP port on a remote host is open and reachable	[2]	Yes
HTTP Get	Verifies that a web server is up and running and can serve a file.	[35]	Yes
FTP Get	Verifies that an FTP server is up and running and can serve a file.	[36]	Yes
TFTP Get	Verifies that an TFTP server is up and running and can serve a file. TFTP Is commonly used to bootstrap and update networking and VoIP equipment.	[35]	Yes
Traceroute	Verifies correct order of layer 3 hops, using packets with increasing TTL	[36][2]	Yes
VLAN Trunking	Verifies that port on a router or switch is tagging vlans and acting as a “Trunk”.	[36]	No
Access Vlan ID	Verifies that a port belongs to the correct access vlan	[36]	No
DHCP Server testing	Verifies that DHCP services are operating properly	[35] [2]	Yes
DNS Server testing	Verifies that DNS services are operating properly	[2]	Yes
Link Speed	Verifies that the network interface negotiates to the correct bitrate	[36][2]	Yes
Link Duplex	Verifies that the network interface negotiates to the correct duplex (full or half)	[36][2]	Yes
Link Power	Verifies that the correct Power Over Ethernet voltage is present	[2]	No
Nearest Switch	Verifies that the next Layer 2 hop is the correct network device.	[2]	No
Network Bandwidth	Verifies that the correct thresholds for upload and download bandwidth for a client can be achieved.	[2]	Yes
Packet Presence Forensics	Examine contents of a packet capture to look for presence of desired packets.	[36]	Yes
Packet Sequencing Forensics	Similar to prescence forensics but ensuring packets arrive in proper order.	[2]	No
Physical Cabling Fitness	Includes verifying wiring order, signal quality, and other physical characteristics	[2]	No

Common Use Case Coverage

- 68% coverage for identified use cases.
- Coverage == Language and Certifier support
- Hardware Difficulties:
 - Power Over Ethernet
 - Copper/Fiber Time-domain Reflectometry
 - Line-rate packet capture

Evaluating Reference Certifier Quality

- Comprehensive Unit Test Suite
- Adherence to Certifier Standards Document

APPENDIX D. NETCDL Certifier Implementation Standards

This document, while not a formal RFC, should serve as a guide for implementors of future NETCDL Certifier Software. The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC2119[7].

A NETCDL certifier **MUST** recognize and parse a body of text that conforms to the official NETCDL grammar specification. If a parse error occurs, the certifier **SHOULD** inform the user the cause of the error, so that they may fix it.

During certification the certifier **SHALL** evaluate every assertion in the input document. The certifier **MAY** carry out the assertions in any order. The certifier **SHOULD** attempt to minimize the total assertion time, or the performance impact on the network, according to user preferences.



Software Quality Indicators

- Less code is better
- Write less, reuse more
- Modular designs promote extensibility

Code Coverage Report Generated by py.test			
Name	Stmts	Miss	Cover

netcdl/ActiveTest.py	14	4	71%
netcdl/Certifier.py	97	22	77%
netcdl/DHCPTTest.py	69	0	100%
netcdl/DNSTest.py	30	4	87%
netcdl/DefineMap.py	10	0	100%
netcdl/EthtoolParser.py	16	0	100%
netcdl/FileFetchTest.py	52	0	100%
netcdl/FrameTypeTest.py	17	0	100%
netcdl/IperfTest.py	37	0	100%
netcdl/LinkControl.py	7	0	100%
netcdl/LinkDuplexTest.py	18	0	100%
netcdl/LinkSpeedTest.py	19	0	100%
netcdl/PacketCapture.py	25	0	100%
netcdl/PacketFromTest.py	28	0	100%
netcdl/PacketPortTest.py	29	0	100%
netcdl/PacketTypeTest.py	20	0	100%
netcdl/PingTest.py	19	0	100%
netcdl/PortOpenTest.py	45	11	76%
netcdl/Report.py	18	0	100%
netcdl/Test.py	16	0	100%
netcdl/TestResult.py	11	0	100%
netcdl/TraceRouteTest.py	25	0	100%
netcdl/__init__.py	0	0	100%
netcdl/netcdl.py	32	16	50%

TOTAL	654	57	91%

Discussion

Threats to Validity

Key

Assumptions:

- Low grammar complexity metrics correlate with language ease of use in real life
- Benefits of formal verification outweigh costs
- Network maintenance and installation will continue to be performed manually (i.e. mistakes are not automated out of the process)

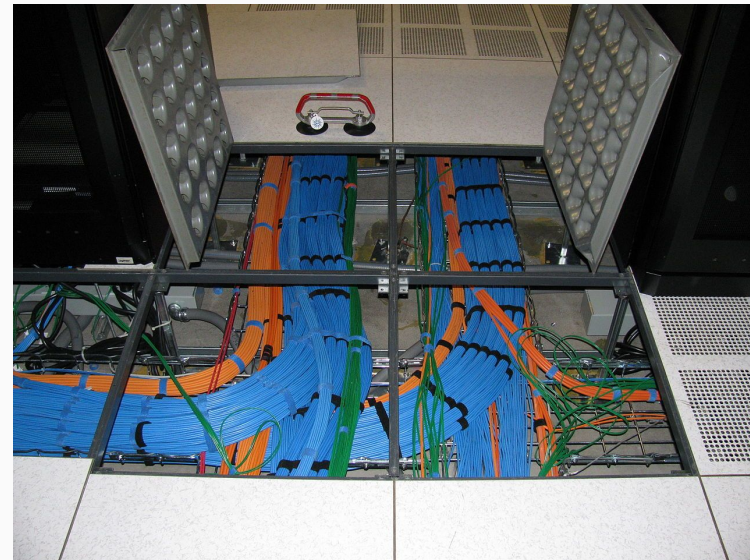
Concluding Thoughts

Thesis Outcomes

- Top 5 ranking for low grammar complexity ✓
- Majority of common use cases supported ✓
- Quality reference implementation available ✓
- Open Standard and Open Source ✓

Future Work

- Real-world trials of the NETCDL workflow
- Wireless networking Applications
- More applications of 'natural language' computing



NETCDL

The Open Standard for Network Verification

- Documentation, Standards, and Source Code
<https://github.com/netcdl>
- Project Homepage
<http://netcdl.org>



Backup slides

Document Links

- [Example NETCDL Document](#)
- [Certifier Implementation Standards](#)

Tools

Program:

'begin'

commands*=Command

'end'

;

Command:

InitialCommand | MoveCommand

;

InitialCommand:

'initial' x=INT ',' y=INT

;

MoveCommand:

direction=Direction (steps=INT)?

;

Direction:

"up"|"down"|"left"|"right"

;

Comment:

/\\V.*\$/

;

```
begin
  initial 3, 1
  up 4
  left 9
  down
  right 1
end
```



Generated
Python
Classes

Selected Related Works

Cucumber [23]

Feature: Accumulator addition

In order to increase the value of the Accumulator
As a user of the class

I want to be able to add an integer to an Accumulator object

Scenario: Add positive integer

Given an Accumulator initialized with 1

When I add 5 to the Accumulator

Then the value of the Accumulator should be 6

Cucumber Step Definition [23]

```
#Test Step definitions for the Accumulator class
```

```
Given /^I have an initial value of (\d+)$/ do |arg1|
```

```
  $starting_value = arg1.to_i
```

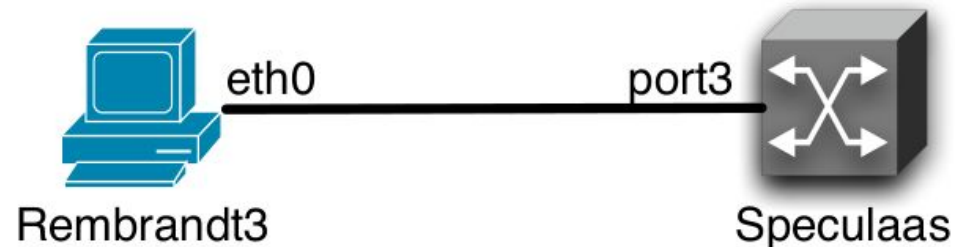
```
end
```

```
Given /^I have constructed a new Accumulator with the parameter (\d+)$/ do |arg1|
```

```
  $accumulator = Accumulator.new(arg1.to_i)
```

```
end
```

The Network Description Language [43]



```
1  <?xml version="1.0" encoding="UTF-8"?>
   <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
           xmlns:ndl="http://www.science.uva.nl/research/air/ndl#"
5   <ndl:Location rdf:about="#Lighthouse">
   <ndl:name>Lighthouse</ndl:name>
   </ndl:Location>
   <ndl:Device rdf:about="#Rembrandt3">
       <ndl:name>Rembrandt3</ndl:name>
       <ndl:locatedAt rdf:resource="#Lighthouse"/>
10      <ndl:hasInterface rdf:resource="#Rembrandt3:eth0"/>
   </ndl:Device>
   <ndl:Interface rdf:about="#Rembrandt3:eth0">
       <ndl:name>Rembrandt3:eth0</ndl:name>
       <ndl:connectedTo rdf:resource="#Speculaas:port3"/>
15  </ndl:Interface>

   <ndl:Device rdf:about="#Speculaas">
       <ndl:name>Speculaas</ndl:name>
       <ndl:locatedAt rdf:resource="#Lighthouse"/>
20      <ndl:hasInterface rdf:resource="#Speculaas:port3"/>
   </ndl:Device>
   <ndl:Interface rdf:about="#Speculaas:port3">
       <ndl:name>Speculaas:port3</ndl:name>
       <ndl:connectedTo rdf:resource="#Rembrandt3:eth0"/>
25  </ndl:Interface>
   </rdf:RDF>
```

Grammar Metrics Collected by SdfMetz[3]

Language	Origin	Type	TERM	VAR	PROD	MCC	NPATH	E	TIMPi	TIMP	LEV	CLEV	NSLEV	DEP	HEI
XPath	GCC	Bison	47	28	86	58	86	7.8	2.3	93.7	9	32.1	1	20	4
BibTeX	GB	SDF	20	6	16	21	32	8.9	20.0	100.0	6	100.0	0	1	6
MatLab	DK	Bison	44	34	92	58	92	13.0	4.0	62.3	15	44.1	2	16	6
Fortran 77	GB	SDF	41	16	41	32	58	18.9	4.8	42.9	15	93.8	1	2	7
C	P&M	BNF	86	65	-	149	-	51	-	64.1	22	33.8	3	38	13
C	ALR	ANTLR	122	67	67	197	285	59.4	1.4	94.9	27	40.9	3	37	13
Java 1.3	ALR	ANTLR	104	68	68	171	263	74.0	2.1	86.1	24	35.3	1	45	9
EcmaScript	SD	DMS	144	90	344	254	344	80.8	1.6	83.7	29	32.2	3	57	10
Ada	HF	Bison	93	238	458	220	458	87.4	0.8	63.3	156	65.5	4	42	24
Java	P&M	BNF	100	149	-	213	-	95	-	32.7	89	59.7	4	33	23
PHP 5	SD	DMS	159	112	418	306	418	96.5	1.6	76.1	55	49.1	2	37	15
Java 1.5	ALR	ANTLR	108	102	102	245	450	132.2	1.9	93.1	24	23.5	2	73	9
SDL	GB	SDF	89	91	174	170	273	132.5	1.1	39.8	76	83.5	2	13	14
Java	SD	DMS	99	144	460	316	460	137.9	1.3	93.9	41	28.5	2	87	17
C	GB	SDF	102	24	148	162	196	146.4	5.9	75.9	15	62.5	1	10	10
C++	P&M	BNF	116	141	-	368	-	173.0	-	85.8	21	14.9	1	121	4
EcmaScript	ALR	ANTLR	185	198	198	406	612	175.6	0.6	43.5	104	52.5	4	89	10
DB2	SIG	SDF	214	98	292	311	478	185.0	1.0	42.6	69	71.1	1	29	16
C#	ALR	ANTLR	133	219	219	408	660	202.3	0.7	55.6	159	72.9	4	30	27
PL/SQL	ALR	ANTLR	196	157	157	449	4683	215.5	1.5	45.6	119	76.3	6	15	21
C#	P&M	BNF	138	245	-	466	-	228	-	29.7	159	64.9	5	44	28
VDM-SL	TA	SDF	143	71	227	232	316	247.6	2.8	78.7	35	49.3	3	27	13
Java 1.5	GB	SDF	105	122	327	318	616	265.4	2.1	79.8	53	43.4	2	63	10
Ada	ALR	ANTLR	99	209	209	326	537	295.4	1.1	56.9	143	68.4	5	36	19
VB.net	JV	SDF	170	206	446	466	1554	390.0	1.0	42.9	154	74.8	6	25	26
Verilog 2001	SD	DMS	232	488	1248	760	1248	455.7	0.5	46.1	266	54.5	10	117	19
SDL	ALR	ANTLR	174	461	461	822	2043	708.9	0.6	35.4	357	77.4	6	43	28
PL/SQL	SIG	SDF	456	499	1094	888	1564	710.9	0.3	24.5	434	87.0	2	38	29
Cobol	GB	SDF	479	774	1330	1122	2194	909.4	0.2	22.2	627	81.2	7	70	26
C++	SD	DMS	173	436	2122	1686	2122	1300.6	0.7	96.3	42	9.6	2	393	10

Fluke LinkSprinter and OneTouch AT [13,14]



Power Over Ethernet (PoE)

Check to make sure you can power a phone, security camera or Access Point through a specific port. The LinkSprinter Network Tester can even run without batteries on PoE.



Link to the Switch

Perform a switch test, which indicates switch name, model, slot, port and VLAN you are connected to using CDP/LLDP/EDP. Know your available speed and duplex settings.



DHCP Connection

Confirm that the DHCP server is running and responsive. Request an IP address, get your subnet information, and identify the default gateway and DNS server.



Gateway Connection

Verify the gateway/router address and reachability by pinging the device.



Internet Connection

Confirm cloud connectivity or internal service reachability. Verify DNS server lookup and application port connectivity.



References

Reference numbers as cited in this presentation correspond to those in the bibliography in the Master's Thesis.