

**PROVE MIKE WRONG CHALLENGE**

**EXPERT REPORT OF ROBERT ZEIDMAN**

## EXPERT REPORT OF ROBERT ZEIDMAN

### Contents

I.	Personal Experience and Background.....	1
II.	Technology Background .....	2
A.	American Standard Code for Information Interchange (ASCII).....	2
B.	Networks and Packets .....	4
C.	Packet Capture (PCAP) Files .....	4
D.	Hexadecimal (Hex) Numbers .....	5
E.	Software Source Code .....	5
III.	Scope of Report.....	6
IV.	Summary .....	6
V.	Analysis.....	7
A.	Files from Day 1 .....	7
1.	File cExtract.mp4.....	7
2.	File Election Process Taxonomy.pdf.....	7
3.	File Chinese_SourceIP_HEX.txt .....	8
4.	File FinalReult_2020_HEX.txt.....	9
5.	File Target_MachineID_HEX.txt .....	9
6.	File Targets_HEX.txt.....	10
7.	File rnx-000001.bin.....	10
B.	Files from Day 2.....	13
1.	File mx-000123.csv.....	13
2.	File Summary.rtf.....	13
3.	Files rnx-000002.bin and rnx-000003.bin .....	14
C.	Other files .....	14
D.	Conclusion.....	15
Exhibit A:	Resume of Robert Zeidman .....	A
Exhibit B:	Chinese_SourceIP_HEX.rft .....	B
Exhibit C:	FinalReult_2020_HEX.rft.....	C
Exhibit D:	Target_MachineID_HEX.rtf.....	D
Exhibit E:	Targets_HEX.rtf.....	E
Exhibit F:	Wireshark file input formats .....	F
Exhibit G:	mx-000123.csv .....	G
Exhibit H:	List of Other Files .....	H



I, Robert Zeidman, provide the following expert disclosures.

**I. PERSONAL EXPERIENCE AND BACKGROUND**

1. I am an engineer and the founder and president of Zeidman Consulting, which provides engineering consulting to high-tech companies. Among the types of services I provide are hardware and software design. My clients have included Fortune 500 computer and technology companies as well as smaller companies and startups. A copy of my resume is attached hereto as Exhibit A.

2. I hold a master's degree from Stanford University in Electrical Engineering and two bachelor's degrees from Cornell University, one in Electrical Engineering and one in Physics. I have more than 38 years of experience in electrical engineering and more than 45 years of experience in software development.

3. The software products I have developed include Internet-based training courses and web-based course administration software, an operating system synthesis tool, a source code comparison tool, a network emulation software bridge, and a remote backup system. The hardware products I have developed include semiconductor integrated circuits and printed circuit boards for bus controllers, memory interfaces, network interfaces, a video attribute controller, a data cache and register logic for a supercomputer, a processor for a telecom parallel processor system, a neural network memory, and various controllers incorporating RISC processors and DSPs.

4. I have founded several companies including Zeidman Consulting, a hardware and software development firm, eVault, a remote backup company, the Chalkboard Network, an e-learning company, Zeidman Technologies that develops software tools for enabling and improving hardware and software development, Software Analysis and Forensic Engineering Corporation that develops software forensic analysis tools, and Good Beat Games, an online poker gaming site.

5. I have written a variety of papers, books, and presentations on computer hardware

and software and other engineering subjects. I am the developer of the Universal Design Methodology, a process for efficiently developing reliable systems, about which I have written extensively. A list of my publications is included in my resume attached as Exhibit A.

6. I hold 23 patents in the areas of software analysis, software comparison, software synthesis, hardware emulation, hardware synthesis, hardware simulation, and media broadcast and advertising. I wrote a textbook on software forensics entitled *The Software IP Detective's Handbook, Measurement, Comparison, and Infringement Detection*, which is recognized as the seminal book in the field. I have created a tool called CodeSuite® for assisting in the determination of whether one computer program has been copied from another computer program.

7. I have consulted on over 240 matters involving intellectual property disputes including instances of alleged software copying, trade secret misappropriation, and patent infringement. My work in this capacity has included, among other things, reviewing and analyzing software source code, reviewing and analyzing patents, reverse engineering hardware and software, writing expert reports, and testifying in court. I have testified at deposition and at trial in a number of these cases. The specific cases can be found in my resume, attached as Exhibit A.

## **II. TECHNOLOGY BACKGROUND**

8. This section describes the major technologies at issue in this matter.

### **A. American Standard Code for Information Interchange (ASCII)**

9. The American Standard Code for Information Interchange, or ASCII code, was created in 1963 by the “American Standards Association” Committee. It is a way to represent simple alphabetic characters, numbers, symbols, punctuation, and several unprintable characters such as a carriage return and a linefeed in binary form so that computers can store them and use them.

10. When initially developed, ASCII could only represent English-language characters but was expanded over the years to be able to represent other alphabets and a large number of symbols. Almost all computer systems today use the ASCII code to represent characters and texts.

11. A table of characters and the ASCII numbers representing them is shown in Figure 1.

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	00	Null	32	20	Space	64	40	@	96	60	`
1	01	Start of heading	33	21	!	65	41	A	97	61	a
2	02	Start of text	34	22	"	66	42	B	98	62	b
3	03	End of text	35	23	#	67	43	C	99	63	c
4	04	End of transmit	36	24	\$	68	44	D	100	64	d
5	05	Enquiry	37	25	%	69	45	E	101	65	e
6	06	Acknowledge	38	26	&	70	46	F	102	66	f
7	07	Audible bell	39	27	'	71	47	G	103	67	g
8	08	Backspace	40	28	(	72	48	H	104	68	h
9	09	Horizontal tab	41	29	)	73	49	I	105	69	i
10	0A	Line feed	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage return	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	46	2E	.	78	4E	N	110	6E	n
15	0F	Shift in	47	2F	/	79	4F	O	111	6F	o
16	10	Data link escape	48	30	0	80	50	P	112	70	p
17	11	Device control 1	49	31	1	81	51	Q	113	71	q
18	12	Device control 2	50	32	2	82	52	R	114	72	r
19	13	Device control 3	51	33	3	83	53	S	115	73	s
20	14	Device control 4	52	34	4	84	54	T	116	74	t
21	15	Neg. acknowledge	53	35	5	85	55	U	117	75	u
22	16	Synchronous idle	54	36	6	86	56	V	118	76	v
23	17	End trans. block	55	37	7	87	57	W	119	77	w
24	18	Cancel	56	38	8	88	58	X	120	78	x
25	19	End of medium	57	39	9	89	59	Y	121	79	y
26	1A	Substitution	58	3A	:	90	5A	Z	122	7A	z
27	1B	Escape	59	3B	;	91	5B	[	123	7B	{
28	1C	File separator	60	3C	<	92	5C	\	124	7C	
29	1D	Group separator	61	3D	=	93	5D	]	125	7D	}
30	1E	Record separator	62	3E	>	94	5E	^	126	7E	~
31	1F	Unit separator	63	3F	?	95	5F	_	127	7F	□

Figure 1. ASCII characters

## **B. Networks and Packets**

12. A computer network is a group of computers that are connected to each other. The Internet is a global network of millions of interconnected computers. The World Wide Web is a portion of the Internet especially suited to displaying images and sound in addition to text, with which most people are familiar. Much of the information on the World Wide Web is stored in the form of “web pages” that can be accessed through a computer that is connected to the Internet and that is equipped with a program called a “browser.” “Websites” are locations on the World Wide Web containing collections of web pages. The Internet is a single large worldwide network; there are many “Local Area Networks” (“LAN”) that connect smaller areas such as homes or offices. A LAN usually provides faster data transfer for an isolated network of computers, printers and servers.

13. Modern networks send data in the form of packets, which are small chunks of binary data. Packets include the Internet protocol (IP) address of the machine that sent the data (source address) and the IP address of the intended recipient of the data (destination address). There are many different levels of packets, each higher-level format incorporating lower-level formats. As an analogy, a book has many sentences that follow rules of grammar for a particular language. Those sentences are combined into paragraphs that have different grammatical rules. Those paragraphs are combined into chapters that have rules governed by the writer’s style. The chapters are combined into a book that has a certain flow dictated by rules of the story. Those books are printed with a cover determined by the publisher and the books are then placed in a library according to system devised by librarians. Just as the essential element of a book is a letter, the essential element of a packet is binary data of simple 1s and 0s.

## **C. Packet Capture (PCAP) Files**

14. Packet data from a network is typically captured and stored in a file that uses a standard format. The most common format is the PCAP format. The format was created by the Wireshark Foundation and its Wireshark application (formerly Ethereal), a free program used for

network analysis. This format has become a de facto standard used by most if not all network analyzers to collect and record packet data from a network. PCAP is a low-level format that also comes in a range of higher-level formats including Libpcap, WinPcap, and PCAPng.

#### **D. Hexadecimal (Hex) Numbers**

15. Hexadecimal numbers, also called hex numbers, are numbers that are base 16 rather than the usual base 10 that people use in everyday counting and arithmetic. When we count, we count up to 9 and then put a 1 in the tens place like this: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10. With hexadecimal, we count up to 15 and then put a 1 in the 16s place. For the digits after 9, we use the letters, A, B, C, D, E, and F. Because of the way that modern digital computers are designed, they use hexadecimal numbers rather than decimal numbers to be faster and more efficient. Counting in decimal, and the equivalent hex numbers, is shown in Figure 2.

Decimal	Hex	Decimal	Hex
0	0	16	10
1	1	17	11
2	2	18	12
3	3	19	13
4	4	20	14
5	5	21	15
6	6	22	16
7	7	23	17
8	8	24	18
9	9	25	19
10	A	26	20
11	B	27	21
12	C	28	22
13	D	29	23
14	E	30	24
15	F	31	25

**Figure 2. Decimal and hexadecimal counting**

#### **E. Software Source Code**

16. Computer programs can be written using complex instructions that look like English. For example, the instruction `a = b*c+2` tells the computer to take the number stored in memory



and represented by variable `b`, multiply that by the number stored in memory and represented by the variable `c`, add 2 and store the result in memory represented by the variable `a`. Similarly, the statement `printf("Hello world!")` tells the computer to print the words "Hello world!" to the computer screen. These high-level, English-like instructions are called "source code." Computer programs are made up of many lines of source code and the process of writing these lines of code is called programming. Eventually these lines of source code are turned into instructions that a computer understands, consisting of sequences of electronic ones and zeroes. The process of turning human-readable source code into a file containing computer instructions is called "compiling" and is performed by a special computer program called a "compiler." In some cases, source code is run directly by a computer, without creating any file of computer instructions, in which case the program is "interpreted" by a special computer program called an "interpreter" that converts each line of source code to computer instructions one at a time to be executed by the computer.

### **III. SCOPE OF REPORT**

17. I was invited to Mike Lindell's Cybersecurity Forum to listen to speakers and get information about alleged fraud in the 2020 U.S. presidential election. I signed the form for the official rules for the "Prove Mike Wrong Challenge" to prove that the data he provided allegedly showing fraud in the election. The wording states that to win the prize, one must "prove that the data Lindell provides, and represents reflects information from the November 2020 election, unequivocally does NOT reflect information related to the November 2020 election."

### **IV. SUMMARY**

18. I have proven that the data Lindell provides, and represents reflects information from the November 2020 election, unequivocally does not reflect information related to the November 2020 election. The proof is explained in the following sections of this report.

## V. ANALYSIS

19. In this section I describe the analysis that leads to my conclusions.

### A. Files from Day 1

20. At the conference, I was taken to a breakout room where I joined a number of other software cybersecurity and forensics experts. We were instructed to examine files in the folder Pod\_Dist on the network at the network location [\\USAServer01\Cyber](#). At that location, I found the following files:

1. cExtract.mp4
2. Chinese\_SourceIP\_HEX.txt
3. Election Process Taxonomy.pdf
4. FinalReult\_2020\_HEX.txt
5. rnx-000001.bin
6. Targets\_HEX.txt
7. Target\_MachineID\_HEX.txt

#### 1. *File cExtract.mp4*

21. The file cExtract.mp4 is a video without sound that shows a user performing some kind of manipulation of software source code on a capture of a computer screen. There is no sound and no explanation of what this represents. It is not purported to be election data and it does not contain information about the November 2020 election.

#### 2. *File Election Process Taxonomy.pdf*

22. The file Election Process Taxonomy.pdf is a document showing configurations of voting machines and network connections without explanation. It is not purported to be election data and it does not contain information about the November 2020 election.

### 3. *File Chinese\_SourceIP\_HEX.txt*

23. The remaining text files, `Chinese_SourceIP_HEX.txt`, `FinalReult_2020_HEX.txt`, `Targets_HEX.txt`, and `Target_MachineID_HEX.txt` were represented to us by Joshua Merritt and Colonel Phil Waldron that these were excerpts of packet data in hex format, which means it is binary data (typically PCAP file format) that has been converted to ASCII representations of hexadecimal numbers so that it can be read by a human. Opening the file `Chinese_SourceIP_HEX.txt`, in a text editor, the first line is the following hex numbers:

```
7B 5C 72 74 66 31 5C 61 6E 73 69 5C 61 6E 73 69
```

24. Referring back to Figure 1, converting these hexadecimal ASCII numbers to text results in the following string:

```
{\rtf1\ansi\ansi
```

25. This is odd because ASCII is a small subset of all hex numbers. Most hex numbers should be unprintable, but these hex numbers are all valid ASCII characters. Not only that, but the term `ansi` typically refers to the American National Standards Institute, a private, non-profit organization that administers and coordinates U.S. voluntary standards including software standards (see <https://www.ansi.org/about/introduction>). And `rtf1` typically refers to rich text format, a format developed by Microsoft for simple documents (see [https://en.wikipedia.org/wiki/Rich\\_Text\\_Format](https://en.wikipedia.org/wiki/Rich_Text_Format)). Further, `{\rtf1\ansi\`, is the text that usually appears at the beginning of an RTF formatted document (see <http://www.pindari.com/rtf1.html>).

26. Using a program called `hex2bin` that I created years ago, to convert ASCII-represented hex to binary, I converted the entire file to binary, although I could have done that manually but it would have taken a long time. That binary, as shown in the small example above, conforms to ASCII that represents an RTF document that can be opened in Microsoft Word and

that I have attached as Exhibit B. It is simply a long table with two columns of numbers. The left side appears to be IP addresses and the right side is a single digit number ranked from highest (7) to lowest (3). This is not packet data because there are no packets and no data, only IP addresses and single digits. It contains no information about the November 2020 election.

#### **4. *File FinalReult\_2020\_HEX.txt***

27. Using the same conversion technique I described above, I also converted this file. The beginning of the file (the “header”) describes it as another RTF file:

```
{\rtf1\ansi\ansicpg1252\cocoartf2513
\cocoatextscaling0\cocoaplatform0{\fonttbl\f0\fswiss\fcharset
0 Helvetica;}
{\colortbl;\red255\green255\blue255;}
{\*\expandedcolortbl;;}
\margl1440\margr1440\vieww10800\viewh8400\viewkind0
\pard\tx720\tx1440\tx2160\tx2880\tx3600\tx4320\tx5040\tx5760\
tx6480\tx7200\tx7920\tx8640\pardirnatural\partightenfactor0
```

28. I have attached this RTF document as Exhibit C. While the display of the file appears to be gibberish, all of the contents of the file are RTF codes. It is an RTF file and does not contain packet data or information about the November 2020 election.

#### **5. *File Target\_MachineID\_HEX.txt***

29. Using the same conversion technique I described above, I also converted this file. The beginning of the file describes it as another RTF file, with the same exact header in the file:

```
{\rtf1\ansi\ansicpg1252\cocoartf2513
\cocoatextscaling0\cocoaplatform0{\fonttbl\f0\fswiss\fcharset
0 Helvetica;}
{\colortbl;\red255\green255\blue255;}
{\*\expandedcolortbl;;}
\margl1440\margr1440\vieww10800\viewh8400\viewkind0
\pard\tx720\tx1440\tx2160\tx2880\tx3600\tx4320\tx5040\tx5760\
tx6480\tx7200\tx7920\tx8640\pardirnatural\partightenfactor0
```

30. I have attached this RTF document as Exhibit D. While the display of the file appears to be gibberish, all of the contents of the file are RTF codes. It is an RTF file and does not

contain packet data or information about the November 2020 election.

## **6. *File Targets\_HEX.txt***

31. Using the same conversion technique I described above, I also converted this file. The beginning of the file describes it as another RTF file, with the same exact header in the previous two files:

```
{\rtf1\ansi\ansicpg1252\cocoartf2513
\cocoatextscaling0\cocoaplatform0{\fonttbl\f0\fswiss\fcharset
0 Helvetica;}
{\colortbl;\red255\green255\blue255;}
{\*\expandedcolortbl;;}
\margl1440\margr1440\vieww10800\viewh8400\viewkind0
\pard\tx720\tx1440\tx2160\tx2880\tx3600\tx4320\tx5040\tx5760\
tx6480\tx7200\tx7920\tx8640\pardirnatural\partightenfactor0
```

32. I have attached this RTF document as Exhibit E. While the display of the file appears to be gibberish, all of the contents of the file are RTF codes. It is an RTF file and does not contain packet data or information about the November 2020 election.

## **7. *File rnx-000001.bin***

33. The file `rnx-000001.bin` was represented by Mike Lindell's cybersecurity expert Joshua Merritt to be some kind of binary network packet data that comprised a decrypted version of the original encrypted PCAP file data. This file is over 22 Gbytes in size. I loaded the file into Wireshark, the standard tool for analyzing network packet data that is available from the Wireshark Foundation at <https://www.wireshark.org>. The Wireshark program can be used to analyze packet data in all common packet data formats, listed below (*see* [https://www.wireshark.org/docs/wsug\\_html\\_chunked/ChIOOpenSection.html](https://www.wireshark.org/docs/wsug_html_chunked/ChIOOpenSection.html), attached as Exhibit F):

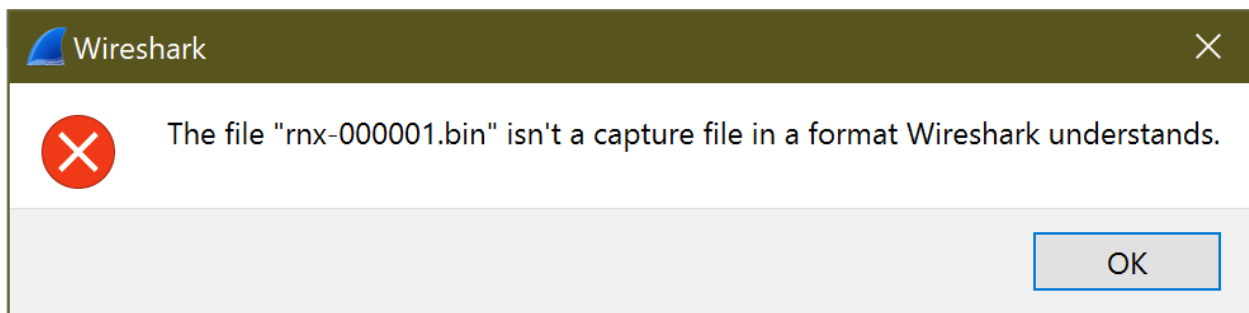
- 1) pcapng. A flexible, extensible successor to the libpcap format. Wireshark 1.8 and later save files as pcapng by default. Versions prior to 1.8 used libpcap.
- 2) libpcap. The default format used by the *libpcap* packet capture library. Used by

*tcpdump*, *\_Snort*, *Nmap*, *Ntop*, and many other tools.

- 3) Oracle (previously Sun) *snoop* and *atmsnoop*
- 4) Finisar (previously Shomiti) *Surveyor* captures
- 5) Microsoft *Network Monitor* captures
- 6) Novell *LANalyzer* captures
- 7) AIX *iptrace* captures
- 8) Cinco Networks *NetXray* captures
- 9) Network Associates Windows-based *Sniffer* and *Sniffer Pro* captures
- 10) Network General/Network Associates DOS-based *Sniffer* (compressed or uncompressed) captures
- 11) AG Group/*WildPackets*/*Savvius*  
    *EtherPeek*/*TokenPeek*/*AiroPeek*/*EtherHelp*/*PacketGrabber* captures
- 12) RADCOM's WAN/LAN Analyzer captures
- 13) Network Instruments *Observer* version 9 captures
- 14) Lucent/Ascend router debug output
- 15) HP-UX's *nettl*
- 16) Toshiba's ISDN routers dump output
- 17) ISDN4BSD *i4btrace* utility
- 18) traces from the EyeSDN USB S0
- 19) IPLog format from the Cisco Secure Intrusion Detection System
- 20) *pppd* logs (*pppdump* format)
- 21) the output from VMS's *TCPIPtrace*/*TCPtrace*/*UCX\$TRACE* utilities
- 22) the text output from the DBS *Etherwatch* VMS utility
- 23) Visual Networks' *Visual UpTime* traffic capture
- 24) the output from CoSine L2 debug
- 25) the output from Accellent's *5Views* LAN agents
- 26) Endace Measurement Systems' *ERF* format captures

- 27) Linux Bluez Bluetooth stack hcidump -w traces
- 28) Catapult DCT2000 .out files
- 29) Gammu generated text output from Nokia DCT3 phones in Netmonitor mode
- 30) IBM Series (OS/400) Comm traces (ASCII & UNICODE)
- 31) Juniper Netscreen snoop captures
- 32) Symbian OS btsnoop captures
- 33) Tamosoft CommView captures
- 34) Textronix K12xx 32bit .rf5 format captures
- 35) Textronix K12 text file format captures
- 36) Apple PacketLogger captures
- 37) Captures from Aethra Telecommunications' PC108 software for their test instruments

34. However, I received the following error message in Figure 3, confirming that the file did not contain packet data in any known format for packet data.



**Figure 3. Wireshark error message**

35. I also confirmed that Wireshark does not care what the file extension is. I renamed a normal PCAP file that Wireshark does understand, having the normal extension `.pcap`, to a file name with the extension `.bin`, but Wireshark successfully opened the file despite the changed file extension. Thus this file does not contain packet data; otherwise Wireshark would have been able to open it. It does not contain information about the November 2020 election.

## **B. Files from Day 2**

36. On Day 2, four more files were supplied. They were:

- 1) `mx-000123.csv`
- 2) `Summary.rtf`
- 3) `rnx-000002.bin`
- 4) `rnx-000003.bin`

### **1. *File mx-000123.csv***

37. The first file, `mx-000123.csv`, was simply a spreadsheet in the “comma-separated values” format. I have included a printout of this spreadsheet as Exhibit G. This spreadsheet contains names of cities in the United States and numbers that are labeled “latn” and “lngn,” where n is a number, with no explanations. Checking online, the lat and lng values are the latitude and longitude of each city.

38. The files include columns labeled “Source IP,” “TargetIP,” and “SourceLocation.” There is no explanation about what these columns are, though presumably the SourceIP is the origination IP of a packet and TargetIP is the IP address of some computer being attacked (this is the meaning of a target IP address). There is no explanation of information about what kind of attack was occurring or how this was determined. SourceLocation would, I presume, be the location where the attack originated, though the list contains the names of most, if not all, Fortune 500 companies and many other companies. There is no information about what this information represents or how it was obtained, there is no way to determine how it was obtained, and the information contains no packet data or any information about the November 2020 election.

### **2. *File Summary.rtf***

39. The second file, `Summary.rtf`, is an RTF format document that when opened in Microsoft Word, simply states:



mx-000001.bin

mx-000002.bin

mx-000003.bin

Create: 11/2/20

Modified: 11/2/20

Last Opened: 11/2/20

Output from passive acquisition of election data. cExtract is used to export to csv for analysis. A lot of Networks have been altered since Nov, 3, 2020 for please look at the historical record if looking at ip's. Not every single line will be perfect but a lot are.

Mx-000123.csv

40. This information does not clarify any of the data in the other files and does not contain packet data or any information about the November 2020 election.

**3. Files *rnx-000002.bin* and *rnx-000003.bin***

41. The remaining two files, *rnx-000002.bin* and *rnx-000003.bin* are each 21.6 Gbyte binary files that appear to be in the same unreadable format as file *rnx-000001.bin*. Attempting to open these files in Wireshark again produces an error message stating that the files do not contain packet data in any known format for packet data. They do not contain packet data or any information about the November 2020 election.

**C. Other files**

42. After the previous set of files were given to us, other files were presented to us on a hard drive. Copies were made and distributed, and I obtained a copy. The list of files is attached in Exhibit H. These files cannot be related to the November 2020 election for at least these reasons:

1. Most of the files are unrelated to network data or anything that would have to do with

the November 2020 election. For example, there are source code files, spreadsheets, and document files.

2. The modification dates for the files are all August 2021. This means though they may have been created any time before that, the data within them was modified within the last two weeks. Thus they cannot represent data from the November 2020 election.
3. Members of Mike Lindell's team, including Dr. Douglas Frank and Joshua Merritt informed us when giving us the files that these were not the original files from the November 2020 election. In particular, the original PCAP files were still not available to us and were never made available to us throughout the conference.

#### **D. Conclusion**

43. I have proven that the data Lindell provides, and represents reflects information from the November 2020 election, unequivocally does not contain packet data of any kind and do not contain any information related to the November 2020 election.

44. I await the acknowledgement of this report and its contents and my share of the \$5 million financial reward that has been offered.

Dated: August 11, 2021

---

Robert Zeidman

## **Exhibit A: Resume of Robert Zeidman**

## **Exhibit B: Chinese\_SourceIP\_HEX.rft**

## **Exhibit C: FinalReult\_2020\_HEX.rft**

**Exhibit D: Target\_MachineID\_HEX.rtf**

## **Exhibit E: Targets\_HEX.rtf**

## **Exhibit F: Wireshark file input formats**



**Exhibit G: mx-000123.csv**

## **Exhibit H: List of Other Files**