

UNITED STATES DISTRICT COURT
CENTRAL DISTRICT OF CALIFORNIA
HONORABLE DAVID O. CARTER, JUDGE PRESIDING

- - - - -

ECHOSTAR SATELLITE CORP., et)	
al.,)	
)	
Plaintiffs,)	
)	
vs.)	No. SACV 03-950 DOC
)	Day 12, Volume II
NDS GROUP PLC, et al.,)	
)	
Defendants.)	
_____)	

REPORTER'S TRANSCRIPT OF PROCEEDINGS

Jury Trial

Santa Ana, California

Tuesday, April 29, 2008

Debbie Gale, CSR 9472, RPR
Federal Official Court Reporter
United States District Court
411 West 4th Street, Room 1-053
Santa Ana, California 92701
(714) 558-8141

EchoStar 2008-04-29 D12V2

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25

APPEARANCES:

FOR PLAINTIFF ECHOSTAR SATELLITE CORPORATION, ET AL.:

T. WADE WELCH & ASSOCIATES

BY: CHAD M. HAGAN

CHRISTINE D. WILLETTS

DAVID NOLL

WADE WELCH

Attorneys at Law

2401 Fountainview

Suite 700

Houston, Texas 77057

(713) 952-4334

FOR DEFENDANT NDS GROUP PLC, ET AL.:

O'MELVENY & MYERS

BY: DARIN W. SNYDER

DAVID R. EBERHART

Attorneys at Law

275 Embarcadero Center West

Suite 2600

San Francisco, California 94111

(415) 984-8700

-and-

HOGAN & HARTSON

BY: RICHARD L. STONE

KENNETH D. KLEIN

Attorneys at Law

1999 Avenue of the Stars

Suite 1400

Los Angeles, California 90067

(310) 785-4600

ALSO PRESENT:

David Moskowitz

Dov Rubin

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25

I N D E X

WITNESSES	DIRECT	CROSS	REDIRECT	RECROSS
JONES, Nigel				
By Mr. Stone		5		

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25

SANTA ANA, CALIFORNIA, TUESDAY, APRIL 29, 2008

Day 12, Volume II

(10:25 a.m.)

(In the presence of the jury.)

THE COURT: We're back in session. The jury's present. All counsel are present.

Counsel, thank you for your courtesy.

Mr. Stone, on behalf of NDS, your next witness, please.

MR. STONE: Thank you, Your Honor.

Defendants call Nigel Jones.

THE COURT: Thank you, sir.

Would you step between the double doors and raise your right hand.

NIGEL JONES, DEFENSE WITNESS, SWORN

THE WITNESS: I do.

THE COURT: Thank you, sir.

Would you be kind enough to be seated in the witness box to my left.

Sir, would you state your full name for the jury, please.

THE WITNESS: Yes, sir. Nigel Andrew Jones.

THE COURT: Would you spell your first name.

THE WITNESS: N-I-G-E-L.

THE COURT: And your last name, please.

1 THE WITNESS: J-O-N-E-S.

2 THE COURT: Thank you very much.

3 This is direct examination by Mr. Stone on behalf
4 of NDS.

5 MR. STONE: Thank you, Your Honor.

6 DIRECT EXAMINATION

7 BY MR. STONE:

8 Q. Good morning, Mr. Jones.

9 Mr. Jones, how are you presently employed?

10 A. I'm president of R and B Consulting. It is a
11 consulting firm I founded about 13 years ago.

12 Q. What line of work is R and B Consulting?

13 A. R and B Consulting provides design services in the
14 field of electronics, software, embedded systems, and
15 firmware.

16 Q. Were you retained as an expert by NDS in this case?

17 A. Yes, I was.

18 Q. What were you asked to do?

19 A. My primary role was to assess the large amount of
20 technical information provided in this case and provide a
21 technical forensic analysis of it.

22 Q. I'd like to talk a little bit about your qualifications
23 and experience, sir. Can you tell us a little bit about
24 your educational background, please.

25 A. I have a first-class honors degree in engineering from

1 Brunel University in London.

2 Q. What is a first-class honors degree?

3 A. Yes. In England they have a different degree
4 classification system.

5 Honors is pretty much the same as in the United States,
6 an honors degree. First-class honors is basically
7 equivalent to a 4.0 GPA.

8 THE COURT: And would you state the university or
9 college again?

10 THE WITNESS: Yes, Brunel, B-R-U-N-E-L.

11 THE COURT: Thank you very much.

12 BY MR. STONE:

13 Q. Do you currently live in England?

14 A. No, sir. I live in Maryland.

15 Q. And how have you been employed the last 25 years or so?

16 A. Basically my role in life is designing products. I
17 design electronic circuits. I write the firmware that goes
18 in the microprocessors that go in most of those circuits and
19 provide those services to customers. So the chances are,
20 several of you have probably at one time or another used
21 something that I designed.

22 Q. Are you familiar with embedded systems?

23 A. Absolutely. My primary expertise is in the field of
24 embedded systems.

25 To give you an explanation -- what are embedded

1 systems? An embedded system is something that contains a
2 computer but isn't a computer. So, for example, I have a
3 little remote control here. Okay? It contains a
4 microprocessor. This would be an embedded system.

5 The piece of equipment the court reporter is using also
6 has a microprocessor. That would be classified as an
7 embedded system.

8 Q. Have you written programs for embedded systems?

9 A. Hundreds. That's what I do for a living.

10 Q. Is a Smart Card also considered an embedded system?

11 A. Yes, it is. In fact, a Smart Card is just about the
12 simplest form of embedded system you can have because it
13 contains just one chip. Most embedded systems contain
14 dozens or hundreds of chips. So a Smart Card is just a very
15 simple embedded system.

16 Q. Can you give us an example of some of the commercial
17 products that you've written programs for in the embedded
18 systems area?

19 A. Yes, sir. Last year, one of my clients asked me to
20 design a control system for a diesel burner that is used by
21 the United States Marine Corps on their mobile kitchens.
22 The problem that the Marines were having is that the
23 controller on the existing one would get corroded in adverse
24 environments and fail, and then they couldn't cook. And a
25 hungry Marine is an angry Marine.

1 And they came to me and said, "Can you design us a
2 better mousetrap, a better control?" And going off, I
3 designed all the electronics for it, wrote all the firmware.
4 We did the first production run right at the beginning of
5 this year. The Marines have seen that product, and they
6 can't wait to get it in the field.

7 Q. Have you written any software for scuba diving
8 equipment?

9 A. Yes, I have. One of the more interesting areas I work
10 in is, in fact, scuba, particularly highly advanced diving
11 systems. If you've ever seen anything on the Discovery
12 Channel where you've got divers doing really neat stuff down
13 deep, there's a really good chance they're wearing something
14 that I've designed.

15 My latest product that I'm working on for a company in
16 Sweden was just featured in Popular Mechanics last month.

17 Q. Do you hold any patents?

18 A. Yes. I have one patent issued and quite a few pending.

19 Q. And what are those fields in?

20 A. Yes. The patent that's being issued relates to a smart
21 battery that the U.S. military uses in all their equipment.
22 So I came up with a rather neat way to help the
23 U.S. military extend the use of those batteries. So that's
24 the patent that's issued.

25 Patents that are pending relate to this control I just

1 mentioned to you that I designed for the U.S. Marine Corps.
2 I also have some other patents pending on the diving stuff.

3 Q. Have you written any articles in the embedded systems
4 area?

5 A. Yes. In the embedded systems area there is the
6 premiere magazine called Embedded Systems Design. I've
7 written about a dozen articles for that magazine. I'm also
8 on the editorial design review board. What that means is,
9 when an article is submitted for publication, if the editor
10 thinks it's in a field that I have particular knowledge of,
11 that paper will be submitted to me for vetting or approval.

12 Q. And have you had any experience in assisting any
13 companies as an expert in satellite piracy?

14 A. Yes, I have. Four or five years ago, I was retained by
15 DirecTV, along with my colleague, Mr. Barr, who's in the
16 back here, who you'll be hearing from in a few days. The
17 two of us, plus a couple other gentlemen, spent the best
18 part of actually more than a year examining the hundreds of
19 devices used for DirecTV piracy. And so our job was to take
20 these devices, reverse-engineer them, work out what they
21 did, how they did it, and come to a conclusion whether those
22 devices were compatible with, designed for, suitable for
23 DirecTV piracy.

24 So having spent -- I think it was about a 15-month
25 period for me looking at all these devices, I learned a

1 tremendous amount about satellite piracy, how it's done, the
2 different devices that are used, and so on.

3 Q. Have you ever been retained by Bell ExpressVu or
4 EchoStar in connection with a satellite piracy case?

5 A. Yes, I have. About two or three years ago, there was a
6 joint raid in Canada between DirecTV, EchoStar, and Bell
7 ExpressVu. And what they were doing, they were going to
8 raid a printed circuit board manufacturing plant. And this
9 was a place that was suspected of making printed circuit
10 boards used in all these DirecTV and Bell ExpressVu and
11 EchoStar piracy devices, so they needed someone who, (a),
12 knew what the devices were, what they looked like; and they
13 also needed someone who designed printed circuit boards and
14 knew their way around a circuit board plant. So I went off
15 on this raid. Very dramatic -- police, lawyers turn up at
16 the door, stand back from the desks, and then they bring the
17 engineer in. A fun experience, actually.

18 Q. Do you have any experience with the microprocessors
19 used in the EchoStar access cards?

20 A. Yes, I do. Microprocessors come in families. Okay.
21 And the family of processors used in the Smart Card issue in
22 this case generically is called a 6805. The 6805 was almost
23 the first microprocessor I ever programmed, and I've written
24 hundreds of programs for the 6805 and other members of its
25 family.

1 Q. Have you also worked with encryption in your design
2 work?

3 A. Yes, I have. I use encryption in two ways. A lot of
4 the products I design include what are call encrypted
5 bootstrap loaders.

6 I also have a client that is in the car wash industry.
7 And the car wash industry, as you know, when you go up to
8 the car wash, you have a machine there where you can pay,
9 and they'll take cash, credit, or debit. Well, with debit
10 cards, Visa or MasterCard have a very stringent set of
11 encryption stuff that you have to go through in order to
12 have a debit keypad on a car wash system. If you go to
13 Wal-Mart and go through their car wash, and you pay with a
14 debit card, you'll be using software that I wrote. So if
15 you've done it, I hope it worked.

16 Q. How many programs have you written total for various
17 microprocessor families?

18 A. Oh, hundreds and hundreds. I've been doing it for 25
19 years. It's what I do every day.

20 Q. Have you ever done any reverse engineering?

21 A. Oh, yes.

22 Q. Is reverse engineering a common practice?

23 A. Oh, yes.

24 Q. Is it a secretive practice?

25 A. No, not at all.

1 Q. Can you give us any examples of that?

2 A. Oh, yes, absolutely. I'll give you two examples.

3 Whenever, say, Toyota brings out a new car, the first
4 person to buy that car is General Motors. What does General
5 Motors do? They reverse-engineer it. They rip it apart,
6 they look inside, they see what it cost, they look for new
7 technologies, they look at things they think Toyota was
8 doing badly.

9 A second example, which is much closer to home, last
10 week -- I mentioned Embedded Systems Design magazine that I
11 was involved with -- twice a year that magazine holds a
12 conference. The main conference is in Silicon Valley.
13 Unfortunately, this year it coincided with this trial, so
14 instead of being at the conference, I was here.

15 But at that trial -- excuse me -- conference, the
16 advertising literature for the conferences -- one of the
17 highlights of the conference was going to be they were going
18 to tear apart the latest Sony OLEV -- stands for Organic LED
19 TV -- this is the next big thing in TV.

20 So at the conference as a draw to bring people in,
21 "Come on in. We're going to take this thing apart." I
22 don't think that's very secretive.

23 Q. Is reverse engineering a widespread practice in your
24 industry?

25 A. Yes, it is.

1 Q. And how much time have you spent on this case?

2 A. Hundreds of hours. I think I'm up to about 800 hours.

3 Q. And what have you done with that time?

4 A. Quite frankly, I've almost gone bug-eyed. I have
5 looked at thousands of files, many of which are what are
6 called binary files. When you think of a file, most of the
7 time you think about text, okay? That's what you read.
8 Well, computers also use binary files. So I've had to look
9 at, at least, a hundred, probably more, binary files, which
10 means I have to take those binary files, put them into
11 special programs to allow me to examine them. I've gone
12 through all these different files. I've gone through what I
13 refer to as the Conus e-mails: Six-and-a-half ring binders,
14 each one that thick.

15 THE COURT: Six different --

16 THE WITNESS: Six different.

17 THE COURT: Conus?

18 THE WITNESS: C-O-N-U-S.

19 THE COURT: Conus e-mails.

20 BY MR. STONE:

21 Q. And those would be e-mails reporting on the status of
22 the DNASP-II system?

23 A. Correct.

24 Q. Have you reviewed other documents in this case as well?

25 A. A tremendous number of other documents, yes.

1 Q. And when did you prepare your original report in this
2 case?

3 A. I started work on it in March of 2007. The report was
4 submitted May 10, 2007.

5 Q. And were there any significant documents that came to
6 light after you did your original report?

7 A. Yes. Two major sets of documents come to mind. The
8 first one is what I refer to as the "black box files." And
9 the second set was the source code to the DNASP system.

10 Q. And when did you have an opportunity to review the
11 source code?

12 A. About two weeks before the trial started.

13 Q. And did any of those new documents change any of your
14 opinions?

15 A. No. Actually, they did quite the opposite. The
16 documents confirmed a lot of things that I suspected. If
17 anything, they helped to confirm my opinions.

18 Q. And you've also reviewed the deposition testimony in
19 this case?

20 A. Yes, I have. I've probably read at least eight
21 deposition transcripts.

22 Q. Now, based on your review of the evidence and the
23 documents and the files that you've testified to, were you
24 able to reach opinions in this case?

25 A. Yes, I have.

1 Q. And have you prepared a demonstrative that summarizes
2 your key opinions?

3 A. Yes, I have.

4 MR. STONE: Can we show that to him.

5 (Document displayed.)

6 THE WITNESS: This is my first opinion. Haifa was
7 not the source of the Nipper postings nor any of the other
8 information on the Internet.

9 BY MR. STONE:

10 Q. When you say "nor the -- any of the other information
11 on the Internet," what are you referring to?

12 A. Well, other than the Nipper postings that are issued in
13 this case, we also have a lot of other technical
14 information: the StuntGuy FAQ, Dover FAQ, and hundreds of
15 other bits and pieces of information that are cropping up.
16 Perhaps what's most germane are excerpts of ROM from the
17 DNASP system.

18 Q. And we'll be going through that information, and you'll
19 have an opportunity to explain why you concluded it is not
20 connected to the Headend Report, correct?

21 A. Correct.

22 Q. And have you formed any other opinions?

23 A. Yes. This is my second opinion: My second opinion is
24 that it was inevitable that the NagraVision system would be
25 hacked.

1 Q. You formed any other opinions?

2 A. Yes. My third one: NagraVision knew about the
3 problems in their system before the Nipper postings and
4 chose to do nothing about them.

5 Q. And do you have a final major opinion?

6 A. Yes. The patch that NagraVision applied to the card
7 within months after the Nipper postings was completely
8 effective in closing the buffer overflow vulnerability that
9 you've heard so much about.

10 Q. Was there also an electronic countermeasure that
11 accompanied the patch?

12 A. Yes, There was.

13 Q. Have you studied that as well?

14 A. Yes, I have.

15 Q. Have you studied both the patch code as well as the
16 electronic countermeasure information?

17 A. Yes, I have.

18 Q. Okay. Now, let's talk a little bit about the EchoStar
19 satellite system. And the folks on the jury have heard
20 some, so I'd ask that you give a very brief review.

21 A. Brief?

22 Q. But slow.

23 A. I'll try. Okay. What we have here is a basic picture
24 of how this all works. I think you've got the basic gist.

25 We have a big satellite uplink dish. It sends an

1 encrypted signal to the satellite, gets bounced off the
2 satellite down to your satellite receiver dish. That signal
3 goes into the receiver. The receiver basically says to the
4 Smart Card or access card, "Does this person have permission
5 to see the particular channel?" And if they do, you get to
6 see Shrek on your TV. Okay? So that's the basic way this
7 thing works.

8 What I'd like to do now is show you the thing that's
9 really at issue in this case. This is the receiver, so
10 there's a little bit more detail here.

11 What we have here is messages in encrypted video coming
12 in, and the messages get routed to the access card. The
13 access card has got a couple of components that I'll talk
14 about in more detail later. The access card, if the person
15 is authorized, provides the encryption key, and that
16 encryption key allows video.

17 So to show that happening here, we've got messages in
18 encrypted video coming in, keys being provided by the access
19 card, and bingo! Decrypted video.

20 Q. Okay. And do you have an animation of the normal
21 operation?

22 A. Yes, I do.

23 Q. Okay.

24 A. So this is an animation that I'd like you to look at.
25 And before we play it, I'm gonna explain a little bit about

1 what you're going to see, and hopefully this will make a
2 little clearer to you a lot of these buzzwords you've been
3 hearing thrown around for the last few weeks.

4 You have a remote control, and the remote control is
5 gonna ask, "Please, can I watch HBO?" Okay. The message is
6 gonna go to the receiver, receiver is gonna put that message
7 into the I/O buffer. This is the buffer you've heard about
8 that's being overflowed all the time. Okay?

9 This little thing here represents the buffer filling
10 up, and you're going to see this a few times today. Okay?
11 The man in the middle is the CPU. That's the actual
12 microprocessor that's doing the work.

13 There are three what I call "sets of books" in this
14 Smart Card. Over here we have the EEPROM. Now, the EEPROM
15 contains things like decryption keys, passwords,
16 pay-per-view authorizations, and so on.

17 In the middle, we have user ROM. These are the general
18 instructions for the CPU in terms of what it must do.

19 And then we have the system ROM, which is really
20 responsible for encryption-related functions.

21 Lastly, we've got this funny little guy here called a
22 "guard." This is representing what is called the "memory
23 access control matrix." The memory access control matrix is
24 going to feature quite heavily in my testimony. The way to
25 think of it is, it's a security guard.

1 This is the guy with his hands like this. Okay?

2 THE COURT: Okay.

3 BY MR. STONE:

4 Q. And can we roll it?

5 A. Yes, sir. Can we run the animation, please?

6 Here we have the HBO request coming in, goes to the
7 receiver, into the I/O buffer. You see the I/O buffer
8 filling up. CPU goes along, picks up the message, says,
9 "Okay, what do I do with it?"

10 It goes to the user ROM to get the general instructions
11 and goes to the EEPROM and says, "Has this dude paid for
12 HBO? If he has, give me the decryption key."

13 He then takes that decryption key over to here. But
14 first off, the security guard says, "Hang on, pal. Are you
15 authorized to do this?" So he checks the credentials,
16 allows the guy in.

17 What this guy does now is, he takes the key and puts it
18 in a lock box. And it's going into a lock box because that
19 key's secret, and you've got to transmit it back to the
20 receiver in a secret way. So he's put it in a lock box. It
21 goes in over to the receiver, receiver extracts the key, and
22 guess what? We've got Shrek on TV again.

23 Okay?

24 Q. Can you give a sense to the jury how long that whole
25 process takes in real time?

1 A. Yes. In real time, we're talking about maybe half a
2 second. Okay? And I might add as well the most
3 time-consuming portion is this bit around here. Okay?

4 Q. Okay. Now, do you have an animation that illustrates
5 what we've heard a lot about, called the buffer overflow
6 attack?

7 A. Yes, I do.

8 Q. Can we run that?

9 A. Right. So in this case, things have changed a little
10 bit. We no longer have our receiver. We've got what is
11 called a Smart Card reader/writer. And attached to that is
12 the hacker's computer.

13 What I'm going to show now is what happens when a
14 hacker sends a message that is bigger than the buffer, and
15 that buffer overflows.

16 So if you could run the animation, please.

17 Here's our big red message. Okay? Comes into the I/O
18 buffer. The buffer starts filling, and then it overflows.

19 Now, the way to think of this message is like a
20 computer virus. Okay? So the CPU goes and picks it up, and
21 he is somewhat confused. I mean, this is something he
22 doesn't quite know what to do with. So the virus takes over
23 and starts commanding the CPU what to do.

24 In this case, what's happening is, he's modifying the
25 EEPROM and also getting the EEPROM contents going over to

1 the I/O buffer, sticks it in the buffer, which then goes out
2 to the hacker's computer.

3 So at this stage, okay, that is a buffer overflow in
4 practice in this card. Okay? And by doing this, the EEPROM
5 contents are now available on the hacker's computer.

6 Q. Now, we've talked about two Internet postings in this
7 case. And the first one I'd like to focus on is the
8 December 23rd posting by xbr21 of something we've been
9 calling the Nipper code.

10 Did you do anything to determine if that Nipper code
11 came from the information in the Headend Report?

12 A. Yes, I did. I ran an extensive amount of analysis on
13 it.

14 Q. Can you tell the jury in broad terms what you found
15 when you compared the Nipper code to the Headend, or Haifa,
16 Report?

17 A. Yes. So I looked at this program in many ways, and
18 what I found in broad terms was that wherever Haifa and
19 Nipper had a choice to do something, they chose differently.

20 Q. Surely they must have made some of the same choices?

21 A. Actually, no, nothing substantial.

22 Q. Now, plaintiffs have identified four things that they
23 claim prove that the Headend Report shares the same DNA, I
24 think was the reference, as the Nipper code. What is your
25 opinion on that?

1 A. Yes, this is very interesting. You heard Dr. Rubin
2 talk about what I consider the four pillars. He identified
3 four things.

4 So Dr. Rubin identified four pillars that he said were
5 characteristic of an attack that must have originated from
6 Haifa. What I will be showing you is that any buffer
7 overflow attack on this card must use those four things that
8 Dr. Rubin identified.

9 Q. Now, there is also testimony that any differences
10 between the Headend Report information and the Nipper code
11 is a result of two years' time to improve the Headend
12 information.

13 Do you have an opinion on that?

14 A. Yes, I do.

15 So what Mr. Stone is referring to here is that the
16 suggestion that any differences between Nipper and Haifa can
17 be attributed to the two-year difference in time between
18 when Mordinson wrote his code and when Nipper wrote his
19 code. Well, what I'm going to show you is that
20 David Mordinson's architecture, the way he put his program
21 together, is considerably better, superior to what the
22 Nipper architecture is.

23 Q. So there was no improvement in that intervening time
24 period?

25 A. No, the exact opposite.

1 Q. Now, can you explain to the jury the steps you went
2 through to compare the Nipper code to the Headend Report
3 information?

4 A. Yes, I think so. I'll start with the next slide.

5 What this slide is, this is the Nipper posting, okay?
6 And over here on the left are his instructions, and this was
7 literally what was published on the Internet. And even if
8 you could read it in detail, you'll see that it is just a
9 bunch of hexadecimal numbers. Okay?

10 So, because this was supposedly derived from Haifa, the
11 first thing I did was say, "Okay, let's look at David
12 Mordinson's equivalent." So on the left is Nipper code. On
13 the right, this is taken from Appendix "F" of the Headend
14 Report.

15 I believe you have this in evidence. You can go and
16 look at this. Okay? On the left, Nipper; on the right,
17 this is David Mordinson's code.

18 Q. Did you do anything to make it easier to compare the
19 two?

20 A. Right. Well, for those of you that know anything about
21 computer programming, you will recognize what's on the left
22 here is what's called a binary representation, and this is
23 what's called source code. So to do a comparison, obviously
24 I have to convert the two into the same format. I started
25 off by converting them both to binary.

1 Q. What does this show here?

2 A. The Nipper code and the Mordinson code, side by side,
3 now in the binary format.

4 Okay. Now, you don't have to know anything about
5 computer programming or chips or whatever to see immediately
6 that the Mordinson code is a different size. Okay? Well,
7 so what about the actual values that are in it? What I did
8 is, I said, "Well, I put the two programs on top of each
9 other, and the red is where codes don't match, and the gray
10 is where the codes do.

11 And so to further illustrate the point, I've now
12 removed all the places where they don't match.

13 So it -- what this illustrates here is evidently there
14 wasn't much of a match between the two at a binary level.

15 Okay?

16 Now, if you've done any computer programming, you will
17 know that you need to make a very slight change to what's
18 called the source code to make the binary image completely
19 different. So this is perhaps not a particularly fair
20 comparison, but it is an interesting one nevertheless.

21 What I did now was, I went the other way. I converted
22 Nipper's code into what is called source code. I did what
23 is called disassembly. So now Nipper and Mordinson are in
24 exactly the same format, but this time in a source code
25 format.

1 Q. And what did that show?

2 A. Well, I'm sure you can't see enough detail on the
3 screen there, ladies and gentlemen, to really be able to
4 tell, but if you went up and looked at what are called the
5 actual operation codes up there, you would see that these
6 two programs are completely different. They differ in many,
7 many ways.

8 Q. Can you tell the jury some of the other ways in which
9 the programs differ?

10 A. Yes. So the first thing I'd like to show you is this
11 line that I have highlighted here. Now --

12 Q. And what is that?

13 A. The line I have highlighted here is the call to
14 transmit a byte out of the card. So if you remember, the
15 purpose of this program is to read the contents of the
16 EEPROM book and transmit it out. And it transmitted out a
17 byte at a time. Okay? Now, if you look carefully, you will
18 see that the subroutines that are being called by Mordinson
19 and Nipper are very different. Let me show you how
20 different.

21 Nipper chose to use a routine that was built into the
22 card. Okay? He took one line of code to do it. David
23 Mordinson, by comparison, decided to write his own routine.
24 He devoted 36 lines of code to do what Nipper did in one
25 line. To me, that's a very fundamental difference in the

1 way the two people were thinking.

2 Q. Okay. Was there a difference in the way they
3 terminated the programs?

4 A. Yes. We've gone back to the slide here, and what I
5 have highlighted here is how the program ends. If you look
6 on the bottom, that is David Mordinson's code. And you see
7 this very strange thing that says "B-R-A-\$." That means in
8 assembly language branch always to yourself. In other
9 words, loop on yourself. Okay? Go into an infinite loop.

10 Now, there's only one way out of an infinite loop, and
11 that is to reset the card. Pull it out, plug it back in.
12 Not a very elegant way of finishing a program.

13 Q. Is that consistent with something called "proof of
14 concept"?

15 A. Absolutely.

16 Q. What is proof of concept?

17 A. Proof of concept is something I get to do all the time.
18 It is -- customers come to me, and they say, "We've got this
19 great idea that we think we can turn into a product. We're
20 not sure it can work. What we want you to do is just enough
21 work to show that the concept is good. Prove out the basic
22 ideas. We don't want fancy code. We don't want it well
23 documented. Just do the smallest possible amount of work to
24 prove it out." That is called proof of concept.

25 Q. How did the Nipper code terminate?

1 A. I think this was very interesting. The Nipper code
2 jumped to location 7381.

3 Q. What does that mean?

4 A. What that means is, it is jumping into part of what is
5 called the user ROM. And furthermore, this jump requires
6 you to pass what is called a parameter. Okay?

7 You'll see that strange notation, ".DBE8." That is a
8 parameter being passed to that subroutine. Now, here's the
9 rub. That subroutine 7381 does not appear in the Headend
10 Report. So the person that wrote this code must have had
11 something else other than the Headend Report. What they
12 must have had is the ROM contents. Okay? If they had the
13 ROM contents, they could do exactly what David Mordinson had
14 done.

15 Q. Are there any other differences that you saw between
16 the two programs?

17 A. Yes, many. This slide here that I have, the first four
18 we have already discussed. So program size, the actual
19 detail of the coding sequences, the write routine used, how
20 they terminate the program.

21 The fifth one is kind of easy to explain. You've heard
22 some testimony about invalid checksums before. Well, the
23 interesting thing is that Nipper and Haifa chose to use a
24 different value for the invalid checksum.

25 Q. What is the next point?

1 A. The next three points -- stack pointer, addressing use,
2 how it handles interrupts -- quite frankly, ladies and
3 gentlemen, you need a degree in electrical engineering with
4 computer science to understand those. I'll just ask that
5 you accept -- when I tell you they are significantly
6 different, that you accept that.

7 THE COURT: The jury will understand everything,
8 both you and the other expert.

9 BY MR. STONE:

10 Q. That means manipulate and interrupt, just briefly,
11 versus not using an interrupt?

12 A. Yes. So what an interrupt is, in an embedded system
13 is, when the program is running normally, and then something
14 happens that causes the program to stop doing what it's
15 doing and run off. It's a bit like when you're working at
16 your desk, or whatever, and the phone rings. The phone is
17 an interrupt. Okay? You handle the phone call, you put the
18 phone back down, and then hopefully you carry on the work
19 you were doing.

20 Okay. So interrupts feature very heavily in embedded
21 systems. They're one of the most difficult things to grasp
22 and do correctly. So the fact that the authors of these two
23 codes took different approaches to the use of interrupt
24 handling is highly significant.

25 Q. And in broad terms, what is the difference between

1 direct addressing and indexed addressing?

2 A. With direct addressing, you say, "Give me the value
3 from this specific location." With indexed addressing, you
4 say, "Give me the value at a location that is offset from a
5 base address."

6 Q. If you could move the pointer -- there you go --
7 there's a reference to "shell code" in the second column
8 from the bottom.

9 What is shell code?

10 A. You've heard the term "shell code" before. I think a
11 better term for it that's easy to understand is "virus."
12 This is the virus that we're putting into the card to take
13 it over and do its thing.

14 Now, what's important here is where Mordinson located
15 that virus and where Nipper did. Mordinson located it in
16 the communications buffer. Nipper located it in a region
17 called the stack. And I will be showing you later why that
18 is incredibly significant.

19 Q. Do we have a slide for that?

20 A. So what this shows here is, we have the Nipper code and
21 the Mordinson code side by side again. And the light blue
22 that you're looking at, that is David Mordinson's shell
23 code. And you can see it's at the top in the communications
24 buffer, whereas the Nipper code is at the bottom in the
25 stack region.

1 And I've just gone ahead and highlighted those areas.

2 Q. And just briefly, what is the significance that you'll
3 be talking about for that difference?

4 A. Basically, by David Mordinson putting the program, the
5 virus, into the communications buffer, it allowed him to
6 deliver a program faster that was bigger and was easier to
7 use. And that will be the basis of my opinion that the
8 Mordinson method is superior to the Nipper method.

9 Q. Okay. Now, if we could go back to the slide or go
10 forward to the slide with the summary. Okay, if I
11 understand it correctly, there were at least those 10
12 differences between the two programs?

13 A. That's correct. There were many more, but I felt that
14 10 was more than enough to make my point.

15 Q. And how would you summarize these differences?

16 A. To me, when you look at all these differences, it is
17 clear that these two programs were written by different
18 people independently. I see this as independent development
19 of these two codes.

20 Q. So do you think the Headend Report was the source of
21 the Nipper code?

22 A. No, I do not.

23 Q. Now, did Dr. Rubin disagree with you on the point that
24 these were different programs?

25 A. No, no, he didn't, actually. I had this excerpt from

1 his expert report, and this is what he has to say: "The
2 point of contention here is not whether or not the two
3 programs are the same, because clearly they are not."

4 Q. Now, did you find any significant error in Dr. Rubin's
5 report that might influence the assessment of his method of
6 charting the structure of the two programs?

7 A. Yes, I did. In Dr. Rubin's report, he put together
8 some graphs which showed the two programs, and he used those
9 graphs as a basis or an aid to reaching his opinions.

10 Unfortunately, there were some errors in those graphs
11 which I think are quite significant.

12 MR. STONE: Okay. And, Your Honor, may I approach
13 the easel?

14 THE COURT: You may.

15 BY MR. STONE:

16 Q. Mr. Jones, what I'd ask that you do is step down and
17 demonstrate how you found the error and what the consequence
18 of that error is in the analysis.

19 For the record, we have two blowups of Page 35 and 36
20 from the Appendix "F" of the Headend Report.

21 THE COURT: Just a moment. I want to see if
22 Dr. Rubin can see also.

23 DR. RUBIN: Yes, I can.

24 THE COURT: If you need to get closer, either
25 expert, that's fine.

1 THE WITNESS: Okay. Ladies and gentlemen, what
2 I'd like to show you is this excerpt from the
3 Headend Report, which is David Mordinson's code. You can go
4 into the jury room and look at this and do what I'm about to
5 do just for yourself. Okay?

6 The way to look at this is, in this column here,
7 these numbers here are the numbers that appear over here.
8 Okay? These are what we call pneumonics or the actual op
9 codes that the computer executes. These are variables
10 associated with those op codes.

11 And over here we have comments. A typical
12 comment, load to high byte, check the EEPROM boundary, and
13 so on.

14 What was done with Dr. Rubin's report is, he
15 looked at this and started at the top, and he saw "2100A8."
16 And that's what you see here.

17 BY MR. STONE:

18 Q. So these three bytes match. And then we come down to
19 this byte, and you see a whole series of 9Ds. And here you
20 have a whole series of 9Ds. And we come all the way down to
21 here, CC01A0. And that's these three bytes here.

22 And you notice that Dr. Rubin then says that is the end
23 of what he calls the shell code, or the virus.

24 Well, he didn't look below the line. This is a
25 subroutine here that's very important, and you can see the

1 subroutine now. We get all these 9Ds, and then we come to
2 1100, and, in fact, we go all the way down here, all the way
3 up here, and all the way down to the last 81, which is here.

4 So in reality, the shell code isn't here; the shell
5 code includes all of this.

6 Now, I thought that was quite significant. We're not
7 talking about a few bytes here. We've missed well over half
8 the program. And the importance -- that I can now show you
9 on an animation.

10 Could we step to the first part of the animation,
11 please?

12 So what you see here, this is exactly the same thing we
13 just had on the board. Okay? This is an excerpt from
14 Dr. Rubin's report. All the labels and things are his.
15 Okay?

16 We'll now step the animation, please.

17 All I've done now is add color. Okay? And what I want
18 you to understand is that the different colored regions do
19 different things. Okay? I haven't changed anything, just
20 colorized it. So as this shows right now, the light blue is
21 where Dr. Rubin says the shell code is.

22 We step the animation, please.

23 So what I've done now is go ahead and correct the shell
24 code representation. As you can see, it's quite dramatic.

25 Can you step the animation, please.

1 As you see here, Dr. Rubin has identified this yellow
2 area as what he calls overflow. It isn't overflow. There's
3 some padding in there, but there's setup of some very
4 important variables in low memory which will become
5 important later.

6 Step the animation, please.

7 So what we're now showing, the stuff in red is what
8 David Mordinson considered important setup of that memory
9 location. The stuff in yellow are padding bytes. Okay? So
10 this represents a much more accurate and detailed
11 representation of David Mordinson's code.

12 When I realized there were mistakes in the Mordinson
13 representation, I asked myself, well, is there a similar
14 problem with the Nipper representation?

15 And I have an animation that shows that. So this one's
16 a bit shorter. So same thing, this is from Dr. Rubin's
17 report, and the first thing I do is colorize it.

18 Now, the thing I want you to know is the colors I've
19 added are consistent. So the shell code is still light
20 blue, overflow is in the same color, and so on. Okay.

21 So can we step the animation, please.

22 Again, Dr. Rubin had a bit of an error in his
23 description of these overflow bytes. And what you see here
24 is the yellow is indeed padding, but the red is what Nipper
25 considers to be important memory setup.

1 Okay. So what?

2 So can we go to the next animation, please. Oh, I'm
3 sorry. I hadn't quite finished that. We had a terminology
4 problem as well that got corrected.

5 So what I've done now is I've put these two side by
6 side for you to see the bigger picture. So I've dispensed
7 with the monochrome version, and I've gone straight to the
8 colorized version. So what I'm going to do is correct the
9 errors one by one.

10 Can we step it, please.

11 So there's the shell code.

12 Next, please.

13 That's the stack setup.

14 Next, please.

15 That's the correct representation of the Nipper code.
16 Now, to show the significance of that, could we go back to
17 the first step of that, please.

18 That's where we started.

19 Now -- I'm sorry. Can we stay on the first one,
20 please.

21 I don't know about you, but if you don't know much
22 about computers, you just look at those two pictures and you
23 say, well, yeah, they're basically the same. You just move
24 the light blue up to the top and you've got the same thing,
25 right?

1 Now, one thing I must stress here, ladies and
2 gentlemen, is that the data within each of the colored
3 regions are different. I'm not saying that these two blue
4 regions are identical. They're not. It's simply their
5 basic function we're talking about. So that's where it
6 started.

7 Can we go to the end now, please.

8 I think that's a considerably different representation
9 of the two programs.

10 BY MR. STONE:

11 Q. Mr. Jones, there are some colors that are the same in
12 the same place. Like up at the top there's a white box.

13 A. Yes, sir.

14 Q. What does that represent?

15 A. Right. So in some cases -- that white box, for
16 instance, is the ISO7816 mandated header. In other words,
17 the international standards say you've got to have that
18 there. You have no choice.

19 Q. Are there any other no-choice areas between those?

20 A. Yes. You can surely have noticed that towards the
21 bottom third there's this dark blue region. You've all
22 heard about the buffer overflow and memory aliasing. This
23 is that buffer overflow region where you have no choice in
24 the matter. So the hardware in the card is dictating what
25 you see there. You can do nothing about it.

1 Q. So once again, where the authors had a choice, did they
2 make completely different choices?

3 A. Correct, they did.

4 Q. What's the significance of that?

5 A. Well, to me, if you have got two people making
6 completely different choices wherever they have a choice,
7 the logical conclusion is they developed these things
8 differently, independently. There was no cross-coupling
9 between them.

10 Q. Now, the next area I'd like to shift to is the
11 plaintiff's claim that were four characteristics between
12 Headend Report and Nipper that show that they share the same
13 DNA, the four pillars, as you've described them.

14 A. Yes. I think we have a slide here that shows my
15 understanding of what Dr. Rubin said.

16 Q. And those four things are the use of a buffer overflow
17 attack, the use of memory aliasing, knowledge and use of the
18 index variable, and knowledge and use of the exception
19 handler, correct?

20 A. Correct.

21 Q. And if you analyzed each of those four pillars, as it
22 were --

23 A. Yes, I have.

24 So let's take the first pillar. The claim is, because
25 the buffer overflow attack was used, this is indicative it

1 came from Haifa. Well, you've heard testimony from
2 Mr. Nicolas and Dr. Rubin that a buffer overflow attack is
3 the most common form of attack on any computer system. That
4 was true back in 1980; it was true in 1990; it's true in
5 2000; it's still true today. Okay?

6 If you get this little update from Microsoft that says
7 "Windows has been updated," there's a good chance that
8 they've just patched a buffer overflow vulnerability in
9 their code.

10 Q. The second item is memory aliasing. Can you explain a
11 little bit about memory aliasing?

12 A. Memory aliasing is a strange topic. And so I have some
13 slides here which I hope will help you better understand
14 what memory aliasing is.

15 So consider this: We've got ourselves a street, Memory
16 Lane, and on there we've got four houses. And our person,
17 our mailman here, is going to deliver a letter addressed to
18 120 Memory Lane. And you can see the mailman has absolutely
19 no difficulty in doing it because 120 Memory Lane is there,
20 and the letter will be delivered.

21 Well, what happens if you send a letter to 180 and
22 relay? The mailman doesn't know what to do with it, so he
23 will mark it as "Return to Sender, Not Deliverable."

24 Okay. But what happens if you get a letter addressed
25 to 220 Memory Lane? The mailman could do one of two things

1 here. He could say, "You know what? 220 doesn't exist.
2 I'll mark it 'return to Sender.'" Or he could say, "You
3 know what? I bet they meant 120 Memory Lane, so I'll
4 deliver it to 120 Memory Lane."

5 That, ladies and gentlemen, is memory aliasing, where
6 something designed for one address gets sent to another
7 address.

8 Q. And why do chip manufacturers allow memory aliasing to
9 occur?

10 A. Fundamentally, it's a cost-savings measure.

11 I'll have to explain a little bit about how chips are
12 designed. When you design a chip, in there you build in
13 something called a memory management unit. And the memory
14 management unit, as its name suggests, is a piece of a chip
15 whose job it is to manage memory.

16 Now, when you design a family of microprocessors, you
17 typically design the memory management unit such that it can
18 address, say, this much memory.

19 Now, if you don't need that much memory in your chip,
20 say this much or this much, they don't actually change the
21 memory management unit. They just say just don't install so
22 much memory. And that is exactly what happened on this
23 chip. Okay?

24 Q. Is there an easy way to tell if a particular chip is
25 memory aliasing?

1 A. Potentially. Sometimes you can just read it in the
2 data sheet. But regardless of that, the easiest way is to
3 run a test, do the equivalent of send a letter to 220 and
4 then go to 120's mailbox and see if it got it. It takes you
5 two, three hours tops to run that test.

6 Q. Is memory aliasing used a lot in the industry?

7 A. Certainly when I first graduated, first 10, 15 years,
8 yes, it was used a lot. It was almost standard. Today,
9 with the way they design chips differently and so on, it's
10 becoming less and less common.

11 Q. Now, is memory aliasing something that would be a
12 unique characteristic in the buffer overflow attack on this
13 particular chip?

14 A. Ah. Well, here's where it gets interesting. Okay? As
15 soon as you write beyond the end of the communications
16 buffer on this chip, aliasing occurs. Okay? You have no
17 choice in the matter. You can't say to the chip, "This byte
18 I'm writing I don't want you to alias. This byte I would
19 like you to alias."

20 It happens regardless. You have no control. So what's
21 the implication of this? The implication of this is, if you
22 perform a buffer overflow attack on this card, you have to
23 exploit aliasing because you have no choice.

24 Q. Well, just to be clear, is there any way to perform a
25 buffer overflow on this card without having the data memory

1 alias beyond the buffer?

2 A. No.

3 Q. There was a third point, the use of the index variable.

4 A. Yes. We've heard a lot about the index variable.

5 THE WITNESS: What I'd like to do now, Your Honor,
6 with your permission, is to go in front of the jury and do
7 some drawings.

8 THE COURT: Certainly.

9 THE WITNESS: Okay. What we have here is what
10 engineers like to call a memory map. So I'm going to give
11 you an analogy first so you can better understand it.
12 Imagine you've got a big apartment building with lots of
13 residents, and all their mail is delivered in mailboxes at
14 the bottom of the building. Okay? Each resident has one
15 mailbox.

16 Think of this as all the mailboxes for that
17 building. Okay? And just as you can say -- refer to, say,
18 the tenant who's in apartment 3C, or you can say Mrs. Smith,
19 so you can with memory.

20 So our famous index variable -- we can either
21 refer to the index variable or we can refer to its address
22 or, if you like, the apartment that it's in. Okay. In this
23 case, apartment 3F.

24 Now, Dr. Rubin was kind enough to explain
25 hexadecimal numbers to you. So even though these numbers

1 look a little strange for what we are used to counting in,
2 they are real numbers.

3 And so what I want to show you here on this
4 picture -- this is the memory of the microprocessor in this
5 card. Now, at the bottom we have what are called the
6 registers. We then have some memory. We get our index
7 variable, a location called top of stack. And then up at
8 location 19C it's the start of the I/O buffer. This is the
9 famous buffer that is being overflowed. Okay?

10 What I'm going to do now is show you how the index
11 variable fits into all this.

12 So when a message is received, okay, start of a
13 message is received, the code in the user ROM -- remember
14 the instruction book on the animation -- is going to store
15 the first bytes it receives in the I/O buffer at the offset
16 given by the index available.

17 Wow! That was a bit of a mouthful.

18 So let's put it in practice. When the message
19 first comes in, the index variable has the value zero. And
20 so what will happen is the first byte will come in. It will
21 be stored in the I/O buffer at offset zero, here, the first
22 location.

23 Okay. We then increment the index variable to 1.
24 The next byte that comes in will be stored in the I/O buffer
25 at the offset given by the index variable, offset 1.

1 So far so good.

2 Now, the I/O buffer is a hundred bytes long, and
3 so I think you can see that by the time you get to the end
4 of the I/O buffer, the index variable is going to have a
5 value of 99. Okay?

6 But what happens now? Well, the next byte we send
7 gets aliased; that is, it comes off the end here, all the
8 way down into here.

9 Well, it so happens this region called registers
10 is special. Okay? It can't be touched by the aliasing.

11 And so you keep on sending bytes, and nothing much
12 happens until the index variable has got a value of 132. At
13 that point you are now into memory. So what that means is
14 when your index variable is 132, the next byte that is
15 received gets stored here. Okay.

16 Well, I think you can see what's coming here,
17 folks. As we overflow the buffer more and more, when the
18 index value -- variable has a value of 162, we're here, just
19 below the index variable.

20 Now, Dr. Rubin says if we use the index variable,
21 the attack must come from Haifa. So to not use it, at this
22 point I stop.

23 Anyone have any idea what will happen at this
24 point? Well, I understand you're not allowed to answer my
25 questions, and I don't want to upset the judge here, so I'll

1 answer the question. Okay? The answer is nothing happens,
2 or nothing substantial.

3 Okay. We've gone to all this trouble of
4 overflowing the buffer. We've got all the way to here, and
5 nothing happens. Well, it seems to me, then, we don't have
6 any choice. We need to overwrite the index variable. Okay.

7 So the next byte we send overwrites the index
8 variable. Well, what happens if we overwrote the index
9 variable with zero? Where would the next byte go? Well,
10 the next byte is stored in the communications buffer at the
11 offset given by the index variable. It goes right back
12 there.

13 Well, that's not very useful. In fact, if you
14 chose any value of the index variable between zero and 162,
15 you just end up somewhere around here, and you just go in an
16 infinite loop all day long. Okay? Patently not very
17 useful.

18 So you have to do something with this index
19 variable. Well, so what did David Mordinson do? What David
20 Mordinson did is, he said, you know, "I'm gonna modify the
21 index variable such that the next byte gets written to the
22 top of stack." Okay? That's what David Mordinson did.

23 Well, what did Nipper do? What Nipper did is he
24 modified the index variable to point to here. Well, why did
25 he point it there? Because that's where his program wanted

1 to go. And his program was just big enough such that the
2 last byte of the program overwrote the top of the stack.

3 Okay.

4 Well, you've also heard about black box. What did
5 black box do? Well, black box modified the index variable
6 such that the next byte got written immediately thereafter
7 it. Okay.

8 So all three of them modified the index variable,
9 but they modified it in different ways.

10 Now, the question is: How did David Mordinson
11 know how to modify the index variable? I presume he didn't
12 call anybody up. Okay? So the only way he worked out how
13 to use the index variable was to take the ROM contents,
14 study them, work out how the program works, and go for it.

15 I might add, this whole use of index variables and
16 things is a very standard procedure. Okay? There's nothing
17 complicated or difficult about it. If I was implementing
18 code like this, this is exactly how I would do it.

19 So David Mordinson worked it out by reading the
20 ROM.

21 Well, does that mean that Nipper could have worked
22 it out by reading the ROM? Of course. And also the same
23 for black box.

24 So the point I want you to take away from this,
25 ladies and gentlemen, is I see no way of constructing a

1 buffer overflow attack against this card which doesn't use
2 the index variable. It is impossible.

3 Q. Now, the fourth point was the exception handling. Can
4 you explain that?

5 A. Yes. So far the three steps we've talked about, all
6 they do is get the virus into the card. Okay? But it's
7 dormant at this stage. It can do nothing.

8 And so we have to somehow persuade the CPU, the
9 microprocessor, to activate that virus. Well, it turns out
10 that there's a fairly standard way of doing this, and that
11 is what's called forcing an exception.

12 As its name suggests, what an exception is, is when you
13 cause something to happen to a computer system that is
14 unusual. Okay? Now, when you write these programs, you
15 like to take care of all contingencies, and you build in
16 what is called an exception handler whose job is to handle
17 exceptions.

18 And so the interesting thing about this card is, is the
19 exception handler was designed in such a way that if you
20 were to put a pointer to your shell code at the top of the
21 stack -- remember my picture we had of the top of stack --
22 if you put the pointer to your shell code there and then
23 force an exception, then the exception handler will end up
24 running your code or, to put it in the vernacular,
25 activating the virus.

1 So the question is: Is this unique?

2 Well, I've studied the code. I've gone through all the
3 documentation and so on. And as far as I can tell, there
4 are only two ways of generating an exception that would
5 cause that behavior to happen.

6 The first is to send an invalid checksum. Now, I think
7 you've had this explained to you before. Checksum is
8 nothing more than where you sort of add up all the bytes
9 that have gone beforehand, and if they equal what you've
10 previously got, there is no error. So you deliberately send
11 an invalid checksum and you say there is an error in the
12 message. So that's one way to do it.

13 There is another way. You don't send a checksum. And
14 what happens then is the card says, "Hang on a sec. Where's
15 my checksum?" And it will wait for a large fraction of a
16 second, eventually decide it's not getting the checksum, and
17 run the same exception handler.

18 So two ways of doing it. Sending the invalid checksum
19 is obviously superior because you don't waste the time
20 waiting for the timeout. And so what we have here is, in my
21 opinion, the only way of activating that virus in this card
22 is to exploit the exception handler by sending an invalid
23 checksum. There is no other way.

24 So if there is no other way, it is not surprising that
25 a buffer overflow attack that was independently developed

1 exploited that characteristic.

2 Q. So taking these four pillars, the buffer overflow
3 attack itself, that's the most common form of attack on
4 computers?

5 A. Correct.

6 Q. Would anyone engaging in such an attack have any choice
7 as to memory aliasing for this chip?

8 A. No, they would have no choice at all.

9 Q. Would there be any choice about using the index
10 variable to engage in such an attack?

11 A. Not that I can see.

12 Q. And for the exception handling, would there be any
13 choice as well?

14 A. None.

15 Q. So memory aliasing, modifying the index variable, and
16 exception handling are all necessary structural features for
17 a buffer overflow attack?

18 A. On this particular chip, yes.

19 Q. Now, when you discussed the index variable and the
20 exception handling, you mentioned that you could deduce
21 those from having the ROM contents, correct?

22 A. Correct.

23 Q. Did you see any evidence that the ROM contents from
24 satellite cards were obtained by pirates?

25 A. Yes. There's a tremendous amount of information

1 supporting that position.

2 Q. Do we have a demonstrative to start off with?

3 A. I think so.

4 What we have here, ladies and gentlemen, is a
5 historical perspective. What this is, is a list of all the
6 Smart Cards for satellite TV that I know about that had
7 their ROMs extracted by pirates.

8 As you can see, it's an extensive list. It starts 1993
9 and goes all the way through 2002. I cut it off there
10 because I ran out of slide.

11 Q. In what ways can pirates obtain ROM contents?

12 A. In general there are three ways of getting ROM
13 contents: physical extraction, theft, and glitching.

14 Q. Okay. Let's talk about physical extraction, or
15 invasive attacks. How difficult are those?

16 A. Invasive attacks, this is the testimony you've heard
17 about where you use a FIB or a scanning electron microscope,
18 or so on, to get in and extract the chip's contents. So
19 that is a moderately -- well, a reasonably difficult, fairly
20 expensive means of extracting the ROM contents.

21 Q. Have you seen evidence that that has been in the public
22 literature well before the Haifa report?

23 A. Yes, I have.

24 What I'd like to show you, ladies and gentlemen, is a
25 paper written by Ross Anderson in 1996.

1 Q. Who was Ross Anderson?

2 A. Ross Anderson is professor of computer security
3 engineering, or something like that, at Cambridge University
4 in England. He is arguably the world's foremost authority
5 on computer security.

6 Q. Are you in agreement with Dr. Rubin on that point?

7 A. Yes. Actually, I'll show you this slide. This is what
8 I found on amazon.com where Dr. Rubin was talking about Ross
9 Anderson's book, Security Engineering. You can see
10 Dr. Rubin has some very nice things to say about the book.
11 I must agree. It's a terrific book. If you have any
12 interest at all in this topic, it's quite readable, and I
13 quite recommend it.

14 Q. What did the paper that -- Mr. Anderson have to say?

15 A. This paper was fascinating. Okay? What we have here
16 is the front page of the paper, and in the abstract -- you
17 can read it all, but the key thing is Smart Cards are broken
18 routinely. Okay? So this is Professor Anderson talking in
19 1996.

20 Q. And where was this paper presented?

21 A. This paper was presented at USENIX. Okay? USENIX is
22 not some obscure body. This is the premier advanced
23 computing society in the world. In fact, Dr. Rubin sat on
24 the board of USENIX for a few years.

25 Q. Is there anything else interesting about Mr. Anderson's

1 1996 article?

2 A. Oh, many things. Let me show you this one. He goes on
3 to say "Smart Cards are broken routinely, and even a device
4 that was described by a government signals agency as the
5 'most secure processor generally available' turns out to be
6 vulnerable."

7 So that is the best the government can do. The NSA and
8 the rest of it, in '96, isn't good enough.

9 Q. Is there anything else about this article that you
10 relied upon?

11 A. Yes. In this paper Professor Anderson described some
12 of the ways in which you can attack Smart Cards. And here's
13 an interesting quote. "We will now briefly describe some of
14 the techniques available in professionally equipped
15 semiconductor laboratories, of which there are several
16 hundred worldwide."

17 So this is in 1996. Professor Anderson is saying
18 there's hundreds of labs worldwide with sophisticated
19 equipment capable of attacking Smart Cards.

20 Q. And does he discuss the ability to rent time on such
21 equipment?

22 A. Yes. How's about this for an interesting quote? "We
23 understand, for example, that production attacks carried out
24 by some pay-TV pirates involve the use of a focused ion
25 beam, or FIB, workstation. Low budget attackers can rent

1 time on them from various semiconductor companies."

2 So here's the thing, ladies and gentlemen. In 1996
3 Professor Anderson was writing that satellite TV pirates had
4 already used a FIB to attack a satellite TV card.

5 Q. Does Mr. Anderson's paper spell out exactly how to use
6 a FIB to extract ROM contents?

7 A. Yes, it does. The method described in the paper is
8 very similar to what was ultimately used by Haifa.

9 Q. And if a chip memory aliases, would using the Anderson
10 method to extract show to one the memory aliasing?

11 A. Yes. That was a rather complicated question. But what
12 happens, if you use the technique that's described, as you
13 are extracting the memory contents, you will inherently see
14 this memory aliasing occurring. And you'll see it because
15 you'll see the same values appearing time and again when you
16 would only expect them to appear once.

17 Q. And did you also look at documents in this case that
18 referenced an analysis by TNO?

19 A. Yes, I did. In late 1997 and early 1998, DirecTV
20 contracted with a firm called TNO. And what they basically
21 said to TNO was, "We want you to hack our P3 card if you
22 can." Okay? And TNO wrote a report on their attempts.

23 Q. What was the upshot of that?

24 A. Well, here's an interesting quote from TNO. They
25 were -- they found it possible to rent time on a FIB for

1 \$2500 a day. Okay. Now, bear in mind, what we're talking
2 about here, this is full commercial rates. If you happen to
3 be a grad student at the university that has this FIB, you
4 don't pay much to use it.

5 Q. Okay. Do we have a slide showing a summary of the
6 invasive attack information?

7 A. Yes. So these are some of the things that I'd like you
8 to take away regarding extracting ROM contents using
9 invasive techniques.

10 Number one, reverse engineering is a routine procedure.
11 It is done every day in industry.

12 Number two, the technique used by Haifa was described
13 in Anderson's paper in '96.

14 Anderson has reported that pay-TV pirates had used a
15 FIB prior to 1996 to hack satellite TV. He says there are
16 hundreds of labs around the world capable of deploying
17 advanced attacks against these cards.

18 And then in 1998 TNO could rent time on a FIB for \$2500
19 a day. Now, this equipment is available at universities.
20 And that's not just my opinion. The plaintiff's consultant,
21 Ron Ereiser, also made this point. And as I mentioned,
22 access is typically free to this equipment for grad
23 students.

24 Q. Now, what is the optical technique that's referenced at
25 No. 8?

1 A. Yes, sir. As well as using a FIB to extract the ROM
2 contents, you can also use what are called optical
3 techniques. What you do here is you literally take a very,
4 very high resolution picture of the chip and then, by using
5 staining techniques and taking more pictures, you can work
6 out whether the ROM is a zero or a 1. You get all those
7 zeros and 1's, you've got the program.

8 Q. Did you see any evidence that a real live hacker in
9 this case had used optical techniques for examining the ROM
10 contents of a NagraVision card?

11 A. Yes. I believe in a day or two you'll be hearing from
12 someone called StuntGuy. StuntGuy was the biggest hacker on
13 the scene. He wrote a document called "The StuntGuy FAQ,"
14 frequently asked questions, which literally told you
15 everything you needed to know about how to hack an EchoStar
16 card.

17 In there he describes receiving photomicrographs of
18 ST16 chip that's at issue in this case. And indeed, he even
19 produced pictures of them at his deposition.

20 Q. Now, the TNO report dealt with invasive attack on the
21 P3 card. Would that be more difficult or less difficult
22 than such an attack on the NagraVision card?

23 A. Yeah. The P3 card would have been dramatically harder
24 for two reasons. First of all, it's a newer generation
25 design. Obviously this is brand-new, whereas the

1 NagraVision system by '98 was three, four years old.

2 The second thing is, in the DirecTV system, the card
3 included what is called an ASIC. ASIC stands for
4 application-specific integrated circuit or, in the
5 vernacular, a custom chip. Okay.

6 And so TNO not only had to hack the CPU, but they also
7 had to hack the ASIC. That is a dramatically harder thing
8 to do.

9 Q. Are there noninvasive ways to obtain the ROM contents?

10 A. Yes, there are. The most well-known method is
11 something called glitching.

12 Q. Just briefly, what is glitching?

13 A. Glitching is when you -- I'll back up. A
14 microprocessor is designed to operate at a certain voltage
15 with a certain clock. It's called a clock. If you force
16 the microprocessor to work outside the design envelope, so
17 you set the voltage too high or too low, you set the clock
18 too fast or too slow, then you can literally confuse the
19 electronics or "hiccup" it. That is called glitching.

20 Q. Do we have an animation that shows that?

21 A. Yes, we do.

22 What I'm going to show you here is arguably the world's
23 stupidest bank teller, but we'll go with it anyway.

24 If we'd start the animation, please.

25 The customer is asking the bank teller, "Could I have

1 \$10, please." And the bank teller counts out the money, and
2 they're done.

3 Now let's add a glitcher to the mix.

4 So if we would continue the animation, please.

5 Customer is asking for \$10. Bank teller starts to
6 count. Along comes our glitcher. He yells, "Look." The
7 bank teller turns around and doesn't remember that they've
8 already handed over the money. And the glitcher continues
9 to do that and does it and does it and does it and
10 eventually gets all the money in the bank or, in this case,
11 all the ROM contents. Okay.

12 Q. How much does it cost to build a glitcher?

13 A. Well, you can buy commercial glitchers for about a
14 hundred dollars. You can build your own for about a
15 thousand.

16 Q. Is glitching widely used in the pirate community?

17 A. Yes.

18 Q. And did StuntGuy discuss it in his frequently asked
19 questions?

20 A. Yes, he did. Not only did he discuss it, but I've seen
21 circuit diagrams of glitches that he designed and built.

22 Q. Is it also possible to steal the ROM contents?

23 A. Yes, of course. You can steal just about anything.

24 Q. And do you have a slide showing the three ways of doing
25 that?

1 A. Yes, I think so.

2 So the first method, breaking and entering, I think we
3 all understand this is where you break into the building,
4 burglarize it, and off you go.

5 The second one is much more subtle. Because a ROM is a
6 ROM is a ROM, if you take a copy of the ROM, it's as good as
7 the original. And furthermore, the person you've copied it
8 from is none the wiser that it's been taken. Okay.

9 So anybody that's got access to the ROM, if they can
10 copy it and walk out the door with it, then that's as good
11 as renting a FIB and, you know, a lab and doing the rest of
12 it. Okay.

13 So the question is: Who had potential access to copy?
14 And this was a list of people that I came up with. Okay?
15 So -- and you can read it, but essentially actually the guys
16 designing the system, computer backup people, cleaning
17 staff, security personnel -- security people have to be able
18 to get into secure areas. Senior management, a lot of leaks
19 in companies come via senior management.

20 Now, here's the thing. When NagraVision designed this
21 code, they had to send it to STMicro to have it put into
22 chips. So, then, we've got all the employees of STMicro.

23 And then finally a couple others. There's some good
24 evidence to show that the code at various times was at
25 EchoStar and also at a company called DiviCom in Sunnyvale.

1 So it's a pretty wide circle of people that could
2 potentially steal this.

3 Q. What is a tempest attack?

4 A. Right.

5 So hopefully, if you learn nothing else today, ladies
6 and gentlemen, you'll enjoy this one. This is a tempest
7 attack. The way this works is that when the electrons hit
8 your TV screen to draw the letters on it, they emit
9 electromagnetic radiation. If you sit outside the building
10 with a receiver like this, you can pick up those
11 transmissions and see them on your TV screen.

12 Now, this was first demonstrated in 1995. At the time,
13 I was living in the Washington, D.C. area; and I can tell
14 you, the federal government went nuts because they suddenly
15 realized that all these secure computers they had were
16 vulnerable to this type of attack. Okay? So very
17 interesting form of attack. It's there, nonetheless.

18 Q. Now, have you seen any evidence to suggest the ROM
19 contents were out in the pirate community prior to
20 December 2000?

21 A. Yes. There's a huge amount of evidence suggesting that
22 the ROM contents was out in the community.

23 Q. What did that evidence consist of?

24 A. In 1999 -- I think it was September of 1999 -- there
25 was published on an IRC channel a list of the ROM fragments

1 from NagraVision cards. These ROM fragments were published
2 by six different people, and they covered both ROM 2 and
3 ROM 3. And in several cases the people publishing the
4 fragments said, "We have it all, and to prove it, we're just
5 giving you an excerpt."

6 Okay. So this is in September 1999.

7 Q. Did those fragments include what's called system ROM
8 fragments?

9 A. Yes, they did. If you remember the animation of normal
10 operation, you had EEPROM, user ROM, and system ROM. Well,
11 a large number of the fragments that were published were
12 from the system ROM.

13 Q. Can a buffer overflow on this card be used to obtain
14 system ROM?

15 A. No, it can't. Remember our little security card, the
16 MACM? It turns out that if you do a buffer overflow attack
17 on this card, one of the few things you can't do is extract
18 the system ROM. Okay? You can't use buffer overflow to
19 extract system ROM.

20 So the fact that the pirates had it meant they didn't
21 use buffer overflow. They used either an invasive technique
22 or glitching or they stole it from someone.

23 Q. Now, let's go back to the famous, or infamous,
24 StuntGuy. Did you see any evidence that StuntGuy had the
25 full ROM images?

1 A. Yes, I did.

2 Q. And do we have a slide on that?

3 A. You're going to be hearing a bit about StuntGuy today.

4 So this is an excerpt from the StuntGuy FAQ. And one
5 of the things StuntGuy was very kind to do was he had what's
6 called a change log in this document. Every time he updated
7 it, he said why he was updating it and the information that
8 was added.

9 And so you can see here: "July 15th, 2000. Completed
10 analysis of all commands based on EROM288-02 ROM dump."

11 Well, what you need to know is a 288-02 is the official
12 designation for a ROM 3 card. So StuntGuy's saying, "Hey, I
13 finished analyzing all the commands."

14 And then below that we've got this excerpt where he
15 says, "...the EROM guys, for providing a good environment in
16 which to work, good information and good sounding boards.
17 In addition, as of 25th of August, 2000, the EROM group has
18 managed to gain full access, including back-door commands to
19 the EchoStar 288-02 cards."

20 Now, let me explain to you the significance of the back
21 door. I'll be talking about that later. Once you have
22 access through the back door, you've got complete control of
23 this card. Okay? Complete control.

24 Q. Now, was there other evidence of ROM contents that you
25 saw as well?

1 A. Yes. I have a slide that shows this. Okay?

2 You may recognize this. This was something that was
3 shown to Mr. Nicolas when he was testifying. And this is an
4 e-mail sent, I believe, by Suzanne Guggenheim. And the
5 title is essentially Publicly Available EchoStar ROM Dump
6 and Commented Disassembly.

7 Q. Okay. Have you had an opportunity to look at the
8 attachments to that e-mail?

9 A. Yes, I have.

10 Q. And do you have a slide on that?

11 A. Yes. There was a readme file in here, and there was a
12 couple of things I thought you should see.

13 So this first one, it says, "This file contains all of
14 the ROM dumps of the EchoStar 288-01 cards that have been
15 available on the Net, as well as some ROM information we got
16 from other sources." So a 288-01, that's the official
17 designation for a ROM 2 card.

18 Q. And how similar is the ROM 2 code to the ROM 3 code?

19 A. Oh, it's very similar. Essentially the ROM 3, they
20 just took the ROM 2 and fixed all the bugs in it and issued
21 it as ROM 3. So essentially the same card.

22 Q. And do we have another slide?

23 A. Yes. They go on to say why they're doing this. And
24 here's the quote at the end: "Until eventually a working
25 hack emerges at the far end of all of this."

1 Q. Is there another slide?

2 A. Yes. And this is where they're discussing who they
3 are. And a couple of things I'd like you to see. "This
4 information wasn't all discovered by just one person."

5 And then at the bottom there, "We and others have put a
6 lot of time into this." So you go through this readme file,
7 and it is very clear that there are lots of people with this
8 ROM working on the problem of hacking the system.

9 Q. Now, was there something else in the zip file other
10 than the readme file?

11 A. Well, absolutely. What was in there was a commented
12 disassembly of a NagraVision ROM 2 card, including system
13 ROM, user ROM, and EEPROM.

14 Q. Can system ROM be obtained using any buffer overflow
15 attack?

16 A. No, absolutely not.

17 Q. And what is a commented disassembly of ROM?

18 A. Yes. So remember when I put up David Mordinson's code
19 side by side with the Nipper code? The Nipper code you just
20 saw was a binary image. And on the right we had David
21 Mordinson's code, which was very nicely formatted and had
22 all sorts of comments and explanations.

23 So the process of disassembly is taking the binary
24 image, just those numbers, and back-converting it into a
25 meaningful program that a human can read. Significant

1 undertaking.

2 MR. STONE: Your Honor, I'm going to shift to
3 another topic. I don't know if this would be a good time.

4 THE COURT: This is a good time.

5 Ladies and gentlemen, why don't we resume at
6 1:00 o'clock. You're admonished not to discuss this matter
7 amongst yourselves nor to form or express any opinion
8 concerning this case.

9 Sir, why don't you step down.

10 THE WITNESS: Thank you.

11 THE COURT: All right. Counsel, have a nice
12 lunch.

13 (Lunch recess held at 11:56 a.m.)

14 (Further proceedings reported by Jane Rule in
15 Volume III.)

16 -oOo-

17

18

19

20

21

22

23

24

25

1 -oOo-

2
3 CERTIFICATE

4
5 I hereby certify that pursuant to Section 753,
6 Title 28, United States Code, the foregoing is a true and
7 correct transcript of the stenographically reported
8 proceedings held in the above-entitled matter and that the
9 transcript page format is in conformance with the
10 regulations of the Judicial Conference of the United States.

11
12 Date: Month Date , Year

13
14
15 _____
16 DEBBIE GALE, U.S. COURT REPORTER

17 CSR NO. 9472, RPR
18
19
20
21
22
23
24
25

A	<p>41:1 aliased 43:7 aliases 52:9 aliasing 36:22 37:17 38:10,11 38:12,14 39:5,8 39:25 40:6,11 40:16,23 43:10 48:7,15 52:10 52:14 allow 13:11 39:8 allowed 30:5 43:24 allows 17:16 19:16 amazon.com 50:8 amount 5:19 10:1 21:12 26:23 48:25 58:21 Ana 1:16,23 4:1 analogy 41:11 analysis 5:21 21:12 31:18 52:18 60:10 analyzed 37:21 analyzing 60:13 Anderson 49:25 50:1,2,14,18 51:11,17 52:3,9 53:14 Anderson's 50:9 50:25 52:5 53:13 Andrew 4:22 Angeles 2:21 angry 7:25 animation 17:20 17:24 19:5 20:4 20:16 33:9,10 33:16,22,25 34:6,15,21 35:2 42:14 55:20,24 56:4 59:9 answer 43:24 44:1,1 anybody 45:12 57:9 anyway 55:23 apart 12:5,18,21 apartment 41:12</p>	<p>41:18,22,23 appear 27:9 32:7 52:16 APPEARANCES 2:1 appearing 52:15 Appendix 23:13 31:20 application-spe... 55:4 applied 16:6 approach 31:12 approaches 28:23 approval 9:11 April 1:17 4:1 architecture 22:20,22 area 7:18 9:4,5 34:2 37:10 58:13 areas 8:9 30:1 36:19 57:18 arguably 50:4 55:22 article 9:9 51:1,9 articles 9:3,7 ASIC 55:3,3,7 asked 5:18 7:19 34:13 54:14 56:18 asking 55:25 56:5 assembly 26:8 assess 5:19 assessment 31:5 assisting 9:12 associated 32:10 ASSOCIATES 2:4 attached 20:11 attachments 61:8 attack 20:6 22:5,7 37:17,25 38:2,3 40:12,22 43:21 46:1 47:25 48:3 48:3,6,10,17 51:12 52:4 53:6 54:20,22 58:3,7 58:16,17 59:16 62:15 attackers 51:25</p>	<p>attacking 51:19 attacks 49:15,16 51:23 53:17 attempts 52:22 Attorneys 2:7,14 2:19 attributed 22:17 August 60:17 authority 50:4 authorizations 18:16 authorized 17:15 19:15 authors 28:22 37:1 available 21:5 42:16 51:5,14 53:19 61:5,15 Avenue 2:20 a.m 4:3 63:13</p>	<p>becoming 40:10 beginning 8:4 behalf 4:8 5:3 behavior 47:5 believe 23:15 54:11 61:4 Bell 10:3,6,10 best 9:17 51:7 bet 39:3 better 8:2,2 22:21 29:11 38:13 41:11 beyond 40:15 41:1 big 12:19 16:25 20:17 41:12 45:1 bigger 20:14 30:6 35:6 biggest 54:12 binary 13:6,8,9 13:10 23:22,25 24:3,14,18 62:20,23 binders 13:13 bingo 17:19 bit 5:22,23 16:18 17:10,25 20:3 20:10 28:15 34:16,22 38:11 39:11 42:17 60:3 bits 15:15 black 14:8 45:4,5 45:5,23 blowups 31:19 blue 29:21 33:20 34:20 35:24 36:3,21 board 9:8 10:8,14 33:13 50:24 boards 10:10,13 60:16 body 50:22 book 25:16 42:14 50:9,10,11 books 18:13 bootstrap 11:5 bottom 26:6 29:8 29:24 36:21</p>
			B	
			<p>B 5:10,12,13 back 4:5 9:16 10:16 19:19 26:4,11 28:18 30:9 35:16 38:4 44:11 55:13 59:23 60:20,22 background 5:24 backup 57:16 back-converting 62:24 back-door 60:18 badly 12:8 bank 55:23,25 56:1,5,7,10 Barr 9:15 base 29:5 based 14:22 60:10 basic 16:23,24 17:6 26:21 36:5 basically 6:6,16 17:3 30:4 35:23 52:20 basis 30:7 31:9 batteries 8:23 battery 8:21 beam 51:25 bear 53:1</p>	

<p>41:14 42:5 62:5 bounced 17:1 boundary 32:12 box 4:19 14:8 19:18,18,20 36:12,15 45:4,5 45:5,23 branch 26:8 brand-new 54:25 break 57:3 breaking 57:2 brief 16:20,21 briefly 28:10 30:2 51:13 55:12 bring 10:16 12:20 brings 12:3 broad 21:14,18 28:25 broken 50:17 51:3 Brunel 6:1,10 budget 51:25 buffer 16:8 18:7,7 18:9 19:7,7 20:5 20:14,15,18,18 21:1,1,3 22:6 29:16,24 30:5 36:22,23 37:16 37:25 38:2,8 40:12,16,22,25 41:1 42:8,9,15 42:21,24 43:2,4 43:17 44:4,10 46:1 47:25 48:2 48:17 59:13,16 59:18,21 62:14 bugs 61:20 bug-eyed 13:4 build 39:12 46:15 56:12,14 building 41:12,14 41:17 57:3 58:9 built 25:21 56:21 bunch 23:9 burglarize 57:4 burner 7:20 buy 12:4 56:13 buzzwords 18:2 byte 25:14,17 32:12,19 40:17</p>	<p>40:18 42:20,24 43:6,14 44:7,9 44:10,21 45:2,6 bytes 32:18,21 33:7 34:9,23 42:15 43:2,11 47:8 B-R-A 26:7 B-R-U-N-E-L 6:10</p> <hr/> <p style="text-align: center;">C</p> <p>California 1:2,16 1:23 2:15,21 4:1 call 4:11 11:4 18:13 25:13 28:17 32:8 41:10 45:12 called 9:6 10:22 13:6 18:21,22 20:5,11 23:22 23:23 24:18,22 24:23 25:4,18 26:13,24 27:5,6 29:17 39:13 42:5,7 43:9 46:11,16 52:20 54:2,12,13 55:3 55:11,15,19 57:25 59:7 60:6 calling 21:9 calls 32:23 34:2 Cambridge 50:3 Canada 10:6 capable 51:19 53:16 car 11:6,7,8,12,13 12:3,4 card 7:10,11,14 10:21 11:14 16:6 17:4,4,12 17:13,14,19 18:14 20:11 21:4 22:7 25:14 25:22 26:11 29:12 36:24 40:22,25 42:5 46:1,6,18 47:14 47:21 52:4,21 54:10,16,21,22</p>	<p>54:23 55:2 59:13,15,17 60:12,23 61:17 61:21 62:12 cards 10:19 11:10 48:24 49:6 50:17 51:3,12 51:19 53:17 59:1 60:19 61:14 care 46:15 carefully 25:17 carried 51:23 carry 28:18 CARTER 1:3 case 5:16,20 10:4 10:22 13:1,24 14:2,19,24 15:13 17:9 20:9 20:24 21:7 41:23 52:17 54:9,18 56:10 63:8 cases 36:15 59:3 cash 11:9 cause 46:13 47:5 causes 28:14 CC01A0 32:21 Center 2:14 CENTRAL 1:2 certain 55:14,15 Certainly 40:7 41:8 CERTIFICATE 64:3 certify 64:5 CHAD 2:5 chance 8:13 38:7 chances 6:19 change 14:13 24:17 39:20 60:6 changed 20:9 33:19 channel 8:12 17:5 58:25 characteristic 22:5 40:12 48:1 characteristics 37:11</p>	<p>charting 31:6 check 32:12 checks 19:15 checksum 27:24 47:6,7,11,13,15 47:16,18,23 checksums 27:22 chip 7:13 39:8,12 39:14,19,23,24 40:13,16,17 48:7,18 52:9 54:4,18 55:5 chips 7:14 24:5 39:11 40:9 57:22 chip's 49:18 choice 21:19 36:18,23 37:1,6 40:17,23 44:6 48:6,8,9,13 choices 21:20 37:2,6 chose 16:4 21:19 25:21 27:23 44:14 CHRISTINE 2:5 circle 58:1 circuit 10:8,9,13 10:14 55:4 56:21 circuits 6:17,18 claim 21:23 37:11 37:24 classification 6:4 classified 7:6 cleaning 57:16 clear 30:17 40:24 62:7 clearer 18:2 clearly 31:3 client 11:6 clients 7:19 clock 55:15,15,17 closer 12:9 31:24 closing 16:8 code 14:9,11 16:15 21:9,10 21:15,24 22:10 22:18,19 23:2 23:12,17,23</p>	<p>24:2,2,6,18,22 24:22,24 25:22 25:24 26:6,22 26:25 27:1,10 29:7,9,10,20,21 29:23,24 30:21 32:3,23 33:4,5 33:21,24 34:11 34:19 35:11,15 38:9 42:13 45:18 46:20,22 46:24 47:2 57:21,24 61:18 61:18 62:18,19 62:19,21 64:6 codes 24:9,10 25:5 28:23 30:19 32:9,10 coding 27:19 coincided 12:13 colleague 9:15 college 6:9 color 33:17 34:20 colored 33:18 36:2 colorize 34:17 colored 33:20 35:8 colors 34:18 36:11 column 29:7 32:6 come 9:21 10:20 12:21 14:7 26:18 32:18,20 33:1 42:20 43:21 57:19 comes 20:17 42:19,24 43:7 56:6 coming 17:11,18 19:6 43:16 commanding 20:23 commands 60:10 60:13,18 comment 32:12 commented 61:6 62:11,17 comments 32:11 62:22</p>
--	--	--	---	--

<p>commercial 7:16 53:2 56:13 common 11:22 38:3 40:10 48:3 communications 29:16,23 30:5 40:15 44:10 community 56:16 58:19,22 companies 9:13 52:1 57:19 company 8:15 57:25 compare 23:2,18 compared 21:15 comparison 23:23 24:20 25:23 compatible 9:22 complete 60:22 60:23 Completed 60:9 completely 16:7 24:18 25:6 37:2 37:6 complicated 45:17 52:11 components 17:13 computer 7:2,2 20:12,20 21:2,5 23:21 24:5,16 28:4 32:9 38:3 46:13 50:2,5 57:16 computers 13:8 35:22 48:4 58:15 computing 50:23 concept 26:14,16 26:17,21,24 concerning 63:8 concluded 15:19 conclusion 9:21 37:7 conference 12:12 12:12,14,15,17 12:20 64:10 conferences 12:16 confirm 14:17 confirmed 14:16</p>	<p>conformance 64:9 confuse 55:18 confused 20:21 connected 15:20 connection 10:4 consequence 31:17 consider 22:2 38:15 considerably 22:21 36:8 considered 7:10 34:8 considers 34:25 consist 58:23 consistent 26:13 34:19 constructing 45:25 consultant 53:20 consulting 5:10 5:11,12,13 contain 7:13 contains 7:1,3,13 18:15 61:13 contention 31:2 contents 20:25 21:5 25:15 27:12,13 45:13 48:21,23 49:11 49:13,18,20 52:6,13 53:8 54:2,10 55:9 56:11,22 58:19 58:22 60:24 contingencies 46:15 continue 56:4 continues 56:8 contracted 52:20 control 7:3,20 8:2 8:25 18:4,4,23 18:23 40:20 60:22,23 controller 7:23 Conus 13:13,17 13:19 convert 23:24 converted 24:21</p>	<p>converting 23:25 cook 7:24 copied 57:7 copy 57:6,10,13 CORP 1:5 CORPORATI... 2:3 Corps 7:21 9:1 correct 13:23 15:20,21 30:13 33:23 35:8,15 37:3,19,20 48:5 48:21,22 64:7 corrected 35:4 correctly 28:22 30:11 corroded 7:23 cost 12:6 56:12 cost-savings 39:10 counsel 4:6,7 63:11 count 56:6 countermeasure 16:10,16 counting 42:1 counts 56:1 couple 9:17 17:13 57:23 61:12 62:3 course 45:22 56:23 court 1:1,21,22 4:5,12,17,23,25 5:2 6:8,11 7:5 13:15,17,19 19:2 28:7 31:14 31:21,24 41:8 63:4,11 64:15 courtesy 4:7 covered 59:2 CPU 18:11,18 19:8 20:20,23 46:8 55:6 credentials 19:15 credit 11:9 cropping 15:15 CROSS 3:4 cross-coupling 37:8</p>	<p>CSR 1:21 64:16 currently 6:13 custom 55:5 customer 55:25 56:5 customers 6:19 26:18 cut 49:9 C-O-N-U-S 13:18</p> <hr/> <p style="text-align: center;">D</p> <hr/> <p>D 2:5,19 3:1 DARIN 2:13 dark 36:21 data 36:2 40:2,25 Date 64:12,12 David 1:3 2:6,13 2:24 22:20 23:11,17 25:22 26:6 27:13 29:22 30:4 32:3 34:8,11 44:19 44:19,22 45:10 45:19 62:18,20 day 1:8 4:2 11:19 44:16 53:1,11 53:19 54:11 days 9:16 DBE8 27:7 dealt 54:20 Debbie 1:21 64:15 debit 11:9,9,12,14 December 21:8 58:20 decide 47:16 decided 25:23 Decrypted 17:19 decryption 18:15 19:12,13 deduce 48:20 deep 8:13 DEFENDANT 2:11 Defendants 1:9 4:11 DEFENSE 4:15 degree 5:25 6:2,3 6:6 28:3 deliberately 47:10</p>	<p>deliver 30:6 38:17 39:4 Deliverable 38:23 delivered 38:20 41:13 demonstrate 31:17 demonstrated 58:12 demonstrative 15:1 49:2 deploying 53:16 deposition 14:18 14:21 54:19 derived 23:10 describe 51:13 described 37:13 51:4,11 52:7,12 53:12 describes 54:17 description 34:23 design 5:13 6:17 7:20 8:1 9:6,8 11:1,4 12:10 39:12,16,17 40:9 54:25 55:16 designation 60:12 61:17 designed 6:21 8:3 8:14 9:1,22 10:13 39:6,12 46:19 55:14 56:21 57:20 designing 6:16 57:16 desk 28:16 desks 10:16 detail 17:10,14 23:8 25:2 27:19 detailed 34:10 determine 21:10 developed 37:7 47:25 development 30:18 device 51:3 devices 9:19,20 9:22,25 10:2,11 10:12</p>
--	---	--	--	---

<p>devoted 25:24 diagrams 56:21 dictating 36:24 diesel 7:20 differ 25:6,9 difference 22:17 25:25 26:2 28:25 30:3 differences 22:9 22:16 27:15 30:12,15,16 different 6:3 10:2 13:12,15,16 24:6,19 25:6,19 25:20 27:24 28:6,23 30:17 30:24 33:18,19 36:3,8 37:2,6 45:9 59:2 differently 21:19 37:8 40:9 difficult 28:21 45:17 49:15,19 54:21,21 difficulty 38:19 direct 3:4 5:3,6 29:1,2 DirecTV 9:15,19 9:23 10:6,10 52:19 55:2 disagree 30:23 disassembly 24:23 61:6 62:12,17,23 discovered 62:4 Discovery 8:11 discuss 51:20 56:18,20 63:6 discussed 27:18 48:19 discussing 62:2 dish 16:25 17:2 dispensed 35:6 displayed 15:5 District 1:1,2,22 divers 8:12 DiviCom 57:25 diving 8:7,10 9:2 DNA 21:23 37:13 DNASP 14:9</p>	<p>15:17 DNASP-II 13:22 DOC 1:7 document 15:5 54:13 60:6 documentation 47:3 documented 26:23 documents 13:24 13:25 14:5,7,13 14:16,23 52:17 doing 8:12 10:7 11:18 12:8 18:12 21:4 28:14,15,19 38:19 46:10 47:18 56:24 57:11 61:23 dollars 56:14 door 10:16 57:10 60:21,22 doors 4:13 dormant 46:7 double 4:13 Dov 2:24 Dover 15:14 dozen 9:7 dozens 7:14 Dr 22:1,4,8 30:23 31:4,7,22,23 32:14,22 33:14 33:21 34:1,16 34:22 37:15 38:2 41:24 43:20 50:6,8,10 50:23 dramatic 10:15 33:24 dramatically 54:23 55:7 draw 12:20 58:8 drawings 41:7 dude 19:11 dump 60:10 61:5 dumps 61:14 D.C 58:13 D12V2 1:25</p> <hr/> <p style="text-align: center;">E</p> <hr/>	<p>E 3:1 early 52:19 easel 31:13 easier 23:18 30:6 easiest 40:2 easy 27:21 29:11 39:24 EBERHART 2:13 EchoStar 1:5,25 2:3 10:4,6,11,19 16:18 54:15 57:25 60:19 61:5,14 editor 9:9 editorial 9:8 educational 5:24 EEPROM 18:14 18:14 19:11 20:25,25 21:4 25:16 32:12 59:10 62:13 effective 16:8 eight 14:20 either 31:24 41:20 59:21 electrical 28:3 electromagnetic 58:9 electron 49:17 electronic 6:17 16:10,16 electronics 5:14 8:3 55:19 electrons 58:7 elegant 26:12 Embarcadero 2:14 embedded 5:14 6:22,24,25 7:1,4 7:7,8,10,12,13 7:15,17 9:3,5,6 12:10 28:12,20 emerges 61:25 emit 58:8 employed 5:9 6:15 employees 57:22 encrypted 11:4 17:1,11,18</p>	<p>encryption 11:1,3 11:11 17:15,16 encryption-rela... 18:20 ends 26:5 engage 48:10 engaging 48:6 engineer 10:17 engineering 5:25 11:20,22 12:23 28:3 50:3,9 53:10 engineers 41:10 England 6:3,13 50:4 enjoy 58:6 entering 57:2 envelope 55:16 environment 60:15 environments 7:24 equal 47:9 equipment 7:5 8:8,21 51:19,21 53:19,22 equipped 51:14 equivalent 6:7 23:12 40:3 Ereiser 53:21 EROM 60:15,17 EROM288-02 60:10 error 31:4,17,18 34:22 47:10,11 errors 31:10 35:9 essentially 57:15 61:5,19,21 et 1:5,8 2:3,11 eventually 47:16 56:10 61:24 evidence 14:22 23:15 48:23 49:21 54:8 57:24 58:18,21 58:23 59:24 60:24 evidently 24:13 exact 22:25 exactly 24:24</p>	<p>27:13 33:12 39:22 45:18 52:5 examination 5:3,6 examine 13:11 examining 9:18 54:9 example 7:2,16 12:9 51:23 examples 12:1,2 exception 37:18 46:3,11,12,16 46:19,23,23 47:4,17,22 48:12,16,20 exceptions 46:17 excerpt 30:25 32:2 33:13 59:5 60:4,14 excerpts 15:16 excuse 12:15 executes 32:9 exist 39:1 existing 7:23 expect 52:16 expensive 49:20 experience 5:23 9:12 10:17,18 expert 5:16 9:13 28:8 31:1,25 expertise 6:23 explain 15:19 17:25 23:1 27:21 38:10 39:11 41:24 46:4 60:20 explained 47:7 explanation 6:25 62:22 exploit 40:23 47:22 exploited 48:1 express 63:7 ExpressVu 10:3,7 10:10 extend 8:23 extensive 21:12 49:8 extract 49:18 52:6</p>
--	---	---	--	---

<p>52:10 54:1 59:17,19 extracted 49:7 extracting 49:20 52:13 53:8 extraction 49:13 49:14 extracts 19:21 e-mail 61:4,8 e-mails 13:13,19 13:21</p> <hr/> <p style="text-align: center;">F</p> <hr/> <p>F 23:13 31:20 fact 7:11 8:10 28:22 33:2 44:13 50:23 59:20 fail 7:24 fair 24:19 fairly 46:10 49:19 familiar 6:22 families 10:20 11:17 family 10:21,25 39:16 famous 41:20 42:9 59:23 fancy 26:22 FAQ 15:14,14 54:13 60:4 far 43:1 46:5 47:3 61:25 fascinating 50:15 fast 55:18 faster 30:6 feature 18:24 28:20 featured 8:16 features 48:16 federal 1:21 58:14 felt 30:13 FIB 49:17 51:25 52:4,6,25 53:3 53:15,18 54:1 57:11 field 5:14 6:23 8:6 9:10 fields 8:19 fifth 27:21</p>	<p>file 13:6 61:11,13 62:6,9,10 files 13:5,6,8,9,10 13:12 14:8,23 filling 18:9 19:8 20:18 final 16:5 finally 57:23 find 31:4 fine 31:25 finished 35:3 60:13 finishing 26:12 firm 5:11 52:20 firmware 5:15 6:17 8:3 first 4:23 8:4 10:23 12:3 14:8 15:6 19:14 21:7 23:11 25:10 27:17 33:10 34:17 35:17,19 37:24 40:7,7 41:11 42:15,19 42:20,21 47:6 54:24 57:2 58:12 61:13 first-class 5:25 6:2,6 fits 42:11 five 9:14 fixed 61:20 focus 21:7 focused 51:24 folks 16:19 43:17 force 46:23 55:15 forcing 46:11 foregoing 64:6 foremost 50:4 forensic 5:21 form 7:12 38:3 48:3 58:17 63:7 format 23:24 24:3 24:24,25 64:9 formatted 62:21 formed 15:22 16:1 forward 30:10 found 21:14,18 31:17 50:8</p>	<p>52:25 founded 5:11 Fountainview 2:7 four 9:14 21:22 22:2,3,4,7 27:17 37:11,13,16,21 38:16 48:2 55:1 fourth 46:3 fraction 47:15 fragments 58:25 59:1,4,7,8,11 Francisco 2:15 frankly 13:4 28:2 free 53:22 frequently 54:14 56:18 front 41:6 50:16 full 4:20 53:2 59:25 60:18 fun 10:17 function 36:5 functions 18:20 fundamental 25:25 Fundamentally 39:10 funny 18:21 further 24:11 63:14 furthermore 27:5 57:7</p> <hr/> <p style="text-align: center;">G</p> <hr/> <p>gain 60:18 Gale 1:21 64:15 general 12:4,4 18:17 19:10 49:12 generally 51:5 generating 47:4 generation 54:24 generically 10:22 gentlemen 9:17 25:3 28:3 32:1 36:2 39:5 45:25 49:4,24 52:2 58:6 63:5 germane 15:16 getting 20:25 47:16 49:12</p>	<p>gist 16:24 give 6:25 7:16 12:1,2 16:20 19:12,24 29:2,4 41:10 given 42:16,25 44:11 giving 59:5 glitcher 56:3,6,8 56:12 glitchers 56:13 glitches 56:21 glitching 49:13 55:11,12,13,19 56:16 59:22 go 6:18 11:7,11,12 11:13 18:6 23:15 26:9 29:6 30:9,9 32:3 33:2 33:23 35:2,16 36:7 40:4 41:6 44:9,15 45:1,14 55:23 57:4 59:23 61:23 62:6 goes 6:17 17:3 19:6,8,10,11,21 20:20 21:1 44:11 49:9 51:2 going 8:2 10:7 12:17,17,21 15:18 18:1,10 18:24 19:18 20:13,25 22:19 35:8 38:17 41:10 42:10,14 43:4 55:22 60:3 63:2 gonna 17:25 18:5 18:6,6 44:20 good 5:8 8:13 26:21 38:7 43:1 51:8 57:6,10,23 60:15,16,16 63:3,4 government 51:4 51:7 58:14 GPA 6:7 grad 53:3,22 graduated 40:7</p>	<p>graphs 31:8,9,10 grasp 28:21 gray 24:9 great 26:19 group 1:8 2:11 60:17 guard 18:22,25 19:14 guess 19:22 Guggenheim 61:4 guy 18:21 19:1,16 19:17 guys 57:15 60:15</p> <hr/> <p style="text-align: center;">H</p> <hr/> <p>hack 52:21 53:15 54:15 55:6,7 61:25 hacked 15:25 hacker 20:14 54:8 54:12 hacker's 20:12 21:2,5 hacking 62:8 HAGAN 2:5 Haifa 15:6 21:15 21:18 22:6,16 23:10 27:23 38:1 43:21 49:22 52:8 53:12 half 20:1 33:7 hand 4:14 handed 56:8 handle 28:17 46:16 handler 37:19 46:16,19,23 47:17,22 handles 28:2 handling 28:24 46:3 48:12,16 48:20 hands 19:1 Hang 19:14 47:14 happen 42:20 43:23 46:13 47:5 53:2 happened 39:22 happening 17:17</p>
---	--	--	--	--

<p>20:24 happens 20:13 28:14 38:21,24 40:20 43:6,9,12 44:1,5,8 47:14 52:12 harder 54:23 55:7 hardware 36:24 HARTSON 2:18 HBO 18:5 19:6,12 Headend 15:20 21:11,15,23 22:10,11 23:2 23:13 27:9,11 30:20 31:20 32:3 37:12 header 36:16 heard 16:9,19 18:7 20:5 22:1 27:21 29:10 36:22 38:1 41:4 45:4 49:16 hearing 9:16 18:3 54:11 60:3 heavily 18:24 28:20 held 63:13 64:8 help 8:22 38:13 helped 14:17 hexadecimal 23:9 41:25 Hey 60:12 hiccup 55:19 high 32:12 54:4 55:17 highlighted 25:11 25:13 26:5 30:1 highlights 12:17 highly 8:10 28:24 historical 49:5 hit 58:7 HOGAN 2:18 hold 8:17 holds 12:11 home 12:9 Honor 4:10 5:5 31:12 41:5 63:2 HONORABLE 1:3 honors 5:25 6:2,5</p>	<p>6:6,6 hope 11:15 38:13 hopefully 18:1 28:18 58:5 hours 13:2,2 40:5 houses 38:16 Houston 2:8 How's 51:22 huge 58:21 human 62:25 hundred 13:9 43:2 51:16 56:14 hundreds 7:9,14 9:18 10:24 11:18,18 13:2 15:14 51:18 53:16 hungry 7:25</p> <hr/> <p style="text-align: center;">I</p> <p>idea 26:19 43:23 ideas 26:22 identical 36:4 identified 21:22 22:2,4,8 34:1 II 1:8 4:2 III 63:15 illustrate 24:11 illustrates 20:4 24:13 image 24:18 62:20,24 images 59:25 Imagine 41:12 immediately 24:5 45:6 implementing 45:17 implication 40:21 40:21 importance 33:8 important 29:14 32:25 34:4,5,8 34:25 impossible 46:2 improve 22:11 improvement 22:23 include 11:4 59:7</p>	<p>included 55:3 includes 33:5 including 60:18 62:12 incredibly 29:18 increment 42:23 independent 30:18 independently 30:18 37:8 47:25 index 37:18 41:3 41:4,20,21 42:6 42:10,16,19,23 42:25 43:4,12 43:14,18,19,20 44:6,7,8,11,14 44:18,21,24 45:5,8,11,13,15 46:2 48:9,15,19 indexed 29:1,3 indicative 37:25 industry 11:6,7 12:24 40:6 53:11 inevitable 15:24 infamous 59:23 infinite 26:9,10 44:16 influence 31:5 information 5:20 15:8,10,14,15 15:18 16:16 21:11 22:10,12 23:3 48:25 53:6 60:7,16 61:15 62:4 inherently 52:13 inside 12:6 install 39:21 instance 36:16 instruction 42:14 instructions 18:18 19:10 23:6 integrated 55:4 interest 50:12 interesting 8:9 22:1 24:20 27:1 27:23 40:14</p>	<p>46:18 50:25 51:13,22 52:24 58:17 international 36:17 Internet 15:8,11 21:6 23:7 interrupt 28:10 28:11,12,17,23 interrupts 28:2 28:20 intervening 22:23 invalid 27:22,24 47:6,11,18,22 invasive 49:15,16 53:6,9 54:20 59:21 involve 51:24 involved 12:11 ion 51:24 IRC 58:25 ISO7816 36:16 issue 10:21 17:9 54:18 issued 8:18,20,24 15:12 61:20 item 38:10 I/O 18:7 19:7,7 20:17 21:1 42:8 42:15,21,24 43:2,4</p> <hr/> <p style="text-align: center;">J</p> <p>Jane 63:14 job 9:19 39:15 46:16 joint 10:6 Jones 3:5 4:11,15 4:22 5:8,9 31:16 36:11 judge 1:3 43:25 Judicial 64:10 July 60:9 jump 27:5 jumped 27:2 jumping 27:4 jury 1:15 4:4,20 16:19 19:24 21:14 23:1 25:8 28:7 32:4 41:6</p>	<p>jury's 4:5 J-O-N-E-S 5:1</p> <hr/> <p style="text-align: center;">K</p> <p>keep 43:11 KENNETH 2:19 key 15:2 17:15,16 19:12,13,17,21 50:17 keypad 11:12 keys 17:18 18:15 key's 19:19 kind 4:18 27:21 41:24 60:5 kitchens 7:21 KLEIN 2:19 knew 10:12,14 16:2 know 11:7 20:22 23:20 24:4,17 34:18 35:21,21 38:22 39:1,3 44:20 45:11 49:6 54:15 57:11 60:11 63:3 knowledge 9:10 37:17,18</p> <hr/> <p style="text-align: center;">L</p> <p>L 2:18 lab 57:11 labels 33:14 laboratories 51:15 labs 51:18 53:16 ladies 25:3 28:2 32:1 36:1 39:5 45:25 49:4,24 52:2 58:5 63:5 Lane 38:16,18,19 38:25 39:3,4 language 26:8 large 5:19 47:15 59:11 Lastly 18:21 late 52:19 latest 8:15 12:18 Law 2:7,14,19 lawyers 10:15</p>
---	--	---	---	--

<p>leaks 57:18 learn 58:5 learned 9:25 LED 12:18 left 4:19 23:6,12 23:16,21 letter 38:17,20,21 38:24 40:3 letters 58:8 let's 16:18 23:11 37:24 42:18 49:14 56:3 59:23 level 24:14 life 6:16 light 14:6 29:21 33:20 34:19 35:24 line 5:12 25:11,13 25:22,25 32:24 lines 25:24 list 49:5,8 57:14 58:25 literally 23:7 54:3 54:14 55:18 literature 12:16 49:22 little 5:22,23 7:3 16:18 17:10,25 18:2,9,21 20:9 38:6,11 39:11 42:1 59:15 live 6:13,14 54:8 living 7:9 58:13 load 32:12 loaders 11:5 located 29:14,15 29:16 location 27:2 29:3 29:4 34:9 42:7,8 42:22 lock 19:18,18,20 log 60:6 logical 37:7 London 6:1 long 19:24 43:2 44:16 longer 20:10 look 12:6,6,7 13:8 17:24 23:11,16</p>	<p>25:17 26:5 30:16 32:4,6,24 35:22 42:1 52:17 56:6 61:7 looked 10:12 13:5 21:17 25:4 32:15 looking 9:25 29:22 loop 26:9,9,10 44:16 Los 2:21 lot 11:3 14:16 15:13 18:2 20:5 40:6,8 41:4 57:18 62:6 lots 41:12 62:7 low 34:4 51:25 55:17 lunch 63:12,13</p> <hr/> <p style="text-align: center;">M</p> <hr/> <p>M 2:5 machine 11:8 MACM 59:16 magazine 9:6,7 12:10,11 mail 41:13 mailbox 40:4 41:15 mailboxes 41:13 41:16 mailman 38:17,18 38:22,25 main 12:12 major 14:7 16:5 making 10:9 37:5 man 18:11 manage 39:15 managed 60:18 management 39:13,14,17,21 57:18,19 mandated 36:16 manipulate 28:10 manufacturers 39:8 manufacturing 10:8 map 41:10</p>	<p>March 14:3 Marine 7:21,25 7:25 9:1 Marines 7:22 8:5 mark 38:23 39:2 Maryland 6:14 MasterCard 11:10 match 24:9,12,14 32:18 matrix 18:23,23 matter 36:24 40:17 63:6 64:8 mean 20:21 27:3 45:21 meaningful 62:25 means 9:8 13:10 26:7 27:4 28:10 43:13 49:20 meant 39:3 59:20 measure 39:10 Mechanics 8:16 members 10:24 memory 18:22,23 34:4,8,25 36:22 37:17 38:10,11 38:12,14,15,18 38:19,25 39:3,4 39:5,8,13,13,15 39:17,18,19,21 39:22,25 40:6 40:11,25 41:10 41:19 42:4,6 43:13 48:7,15 52:9,10,13,14 mentioned 9:1 12:10 48:20 53:21 message 18:5,6 19:8 20:14,17 20:19 42:12,13 42:18 47:12 messages 17:11 17:12,17 method 30:8,8 31:5 52:7,10 55:10 57:2 microprocessor 7:4,6 10:23 11:17 18:12</p>	<p>42:4 46:9 55:14 55:16 microprocessors 6:18 10:18,20 39:16 microscope 49:17 Microsoft 38:6 middle 18:11,17 military 8:21,23 mind 14:7 53:1 missed 33:7 mistakes 34:12 mix 56:3 mobile 7:21 moderately 49:19 modified 44:24 45:5,8,9 modify 44:20 45:11 modifying 20:24 48:15 moment 31:21 money 56:1,8,10 monochrome 35:7 month 8:16 64:12 months 16:7 Mordinson 22:18 24:2,6,23 25:18 25:23 27:13 29:14,15,21 30:4,8 34:8,12 44:19,20,22 45:10,19 Mordinson's 22:20 23:12,17 26:6 29:22 32:3 34:11 62:18,21 morning 5:8 Moskowitz 2:24 Motors 12:4,5 mousetrap 8:2 mouthful 42:17 move 29:6 35:23 MYERS 2:12</p> <hr/> <p style="text-align: center;">N</p> <hr/> <p>N 3:1 NagraVision 15:24 16:2,6</p>	<p>54:10,22 55:1 57:20 59:1 62:12 name 4:20,23,25 39:14 46:12 NDS 1:8 2:11 4:8 5:4,16 neat 8:12,22 necessary 48:16 need 24:17 28:3 31:24 39:19 44:6 60:11 needed 10:11,13 54:15 Net 61:15 nevertheless 24:20 new 12:3,6 14:13 newer 54:24 nice 50:10 63:11 nicely 62:21 Nicolas 38:2 61:3 Nigel 3:5 4:11,15 4:22 Nipper 15:7,12 16:3,7 21:9,10 21:15,19,24 22:10,16,18,22 23:2,5,12,16 24:2,23 25:19 25:21,24 26:25 27:1,23 29:15 29:16,20,24 30:8,21 34:14 34:24 35:15 37:12 44:23,23 45:21 62:19,19 Nipper's 24:22 NOLL 2:6 noninvasive 55:9 normal 17:20 59:9 normally 28:13 notation 27:7 notice 32:22 noticed 36:20 no-choice 36:19 NSA 51:7 number 13:25 53:10,12 59:11</p>
---	---	---	---	--

<p>numbers 23:9 32:7,7 41:25,25 42:2 62:24 nuts 58:14 N-I-G-E-L 4:24</p> <hr/> <p style="text-align: center;">O</p> <hr/> <p>O 1:3 obscure 50:22 obtain 49:11 55:9 59:13 obtained 48:24 62:14 obviously 23:23 47:19 54:25 occur 39:9 occurring 52:14 occurs 40:16 official 1:21 60:11 61:16 offset 29:4 42:15 42:21,25,25 44:11 Oh 11:18,21,23 12:2 35:2 51:2 61:19 okay 7:3 10:20 13:7 16:18,23 17:6,20,23 18:5 18:8,10 19:1,2,9 19:23 20:2,3,4 20:17,20 21:3,4 23:5,9,11,16 24:4,6,15 25:17 25:22 26:2,9 27:6,12 28:17 28:20 30:9,10 31:12 32:1,5,8 33:13,15,17,19 34:9,20 35:1 38:5,24 39:23 40:14,16 41:9 41:14,17,22 42:9,12,23 43:5 43:10,15 44:1,3 44:6,16,22 45:3 45:7,12,16 46:6 46:14 49:14 50:15,18,21 52:22 53:1,5</p>	<p>55:5 56:11 57:8 57:12,14 58:16 59:6,18 60:23 61:1,7 old 55:1 OLEV 12:18 once 37:1 52:16 60:21 one's 34:15 oOo 63:16 64:1 op 32:8,10 operate 55:14 operation 17:21 25:5 59:10 opinion 15:6,23 15:23 16:5 21:25 22:13 30:7 47:21 53:20 63:7 opinions 14:14,17 14:24 15:2,22 16:1 31:9 opportunity 14:10 15:19 61:7 opposite 14:15 22:25 optical 53:24 54:2 54:9 order 11:11 Organic 12:18 original 14:1,6 57:7 originated 22:5 outside 55:16 58:9 overflow 16:8 20:5 21:3 22:7 34:2,2,20,23 36:22,23 37:16 37:25 38:2,8 40:12,22,25 43:17 46:1 47:25 48:2,17 59:13,16,18,21 62:14 overflowing 44:4 overflown 18:8 42:9 overflows 20:15</p>	<p>20:18 overwrite 44:6 overwrites 44:7 overwrote 44:8 45:2 o'clock 63:6 O'MELVENY 2:12</p> <hr/> <p style="text-align: center;">P</p> <hr/> <p>padding 34:3,9,24 page 31:19 50:16 64:9 paid 19:11 pal 19:14 paper 9:11 49:25 50:14,15,16,20 50:21 51:11 52:5,7 53:13 parameter 27:6,8 part 9:18 27:4 33:10 particular 9:10 17:5 39:24 40:13 48:18 particularly 8:10 24:19 pass 27:6 passed 27:8 passwords 18:15 patch 16:6,11,15 patched 38:8 patent 8:18,20,24 Patently 44:16 patents 8:17,25 9:2 pay 11:8,13 53:4 pay-per-view 18:16 pay-TV 51:24 53:14 pending 8:18,25 9:2 people 12:20 26:1 30:18 37:5 57:14,16,17 58:1 59:2,3 62:7 perform 40:22,24 period 9:25 22:24 permission 17:4</p>	<p>41:6 person 12:4 17:4 17:14 27:10 38:16 57:7 62:4 personnel 57:17 perspective 49:5 persuade 46:8 phone 28:16,16 28:17,18 photomicrogra... 54:17 physical 49:13,14 pick 58:10 picks 19:8 20:20 picture 16:23 35:6 42:4 46:21 54:4 pictures 35:22 54:5,19 piece 7:5 39:14 pieces 15:15 pillar 37:24 pillars 22:2,4 37:13,21 48:2 piracy 9:13,19,23 10:1,4,11 pirate 56:16 58:19 pirates 48:24 49:7 49:11 51:24 52:3 53:14 59:20 place 10:9 36:12 places 24:12 PLAINTIFF 2:3 plaintiffs 1:6 21:22 plaintiff's 37:11 53:20 plant 10:8,14 play 17:25 PLC 1:8 2:11 please 4:9,21,25 5:24 18:5 19:5 20:16 33:11,16 33:22,25 34:6 34:21 35:2,10 35:12,14,17,20 36:7 55:24 56:1 56:4</p>	<p>plug 26:11 plus 9:17 pneumonics 32:8 point 24:11 27:25 30:14,23 31:2 41:3 43:13,22 43:24 44:24,25 45:24 46:3 50:6 53:21 pointer 28:1 29:6 46:20,22 points 28:1 police 10:15 Popular 8:16 portion 20:3 position 49:1 possible 26:23 52:25 56:22 posting 21:8 23:5 postings 15:7,12 16:3,7 21:6 potential 57:13 potentially 40:1 58:2 practice 11:22,24 12:23 21:4 42:18 premier 50:22 premiere 9:6 prepare 14:1 prepared 15:1 presence 4:4 present 2:23 4:6,6 presented 50:20 50:21 presently 5:9 president 5:10 PRESIDING 1:3 presume 45:11 pretty 6:5 58:1 previously 47:10 primary 5:19 6:23 printed 10:8,9,13 prior 53:15 58:19 probably 6:20 13:9 14:20 problem 7:22 34:14 35:4 62:8 problems 16:3</p>
--	---	--	---	--

procedure 45:16 53:10	published 23:7 58:25 59:1,11	readme 61:11 62:6,10	relied 51:10	12:23 53:10
proceedings 1:14 63:14 64:8	publishing 59:3 Pull 26:11	real 19:25 20:1 42:2 54:8	remember 25:14 42:13 46:21 56:7 59:9,15 62:18	reverse-engineer 9:20 12:5
process 19:25 62:23	purpose 25:15	reality 33:4	remote 7:3 18:4,4	review 9:8 14:10 14:22 16:20
processor 51:5	pursuant 64:5	realized 34:12 58:15	removed 24:12	reviewed 13:24 14:18
processors 10:21	put 13:10 18:6 19:20 22:20 24:8 28:17 31:7 35:5 42:18 46:20,22,24 57:21 62:5,18	really 8:12,13 17:9 18:19 25:3	rent 51:20,25 52:25 53:18	RICHARD 2:18
produced 54:19	puts 19:17	reasonably 49:19	renting 57:11	right 4:14 8:4 20:9 23:13,16 23:20 33:20 35:25 36:15 44:11 58:4 62:20 63:11
product 8:5,15 26:19	putting 29:12 30:4	reasons 54:24	report 14:1,3,6 15:20 21:11,16 21:23 22:10 23:2,14 27:10 27:11 30:20 31:1,5,7,20 32:3 32:14 33:14 34:17 37:12 49:22 52:22 54:20	ring 13:13
production 8:4 51:23	P3 52:21 54:21,23	received 42:12,13 43:15	reported 53:14 63:14 64:7	rings 28:16
products 6:16 7:17 11:4	<hr/> Q <hr/>	receiver 17:2,3,3 17:9 18:6,6 19:7 19:20,21,21 20:10 58:10	reporter 1:21 7:5 64:15	rip 12:5
professionally 51:14	qualifications 5:22	receives 42:15	REPORTER'S 1:14	role 5:19 6:16
professor 50:2,18 51:11,17 52:3	question 44:1 45:10 47:1 52:11 57:13	receiving 54:17	reporting 13:21	roll 19:4
program 21:17 22:20 25:15 26:5,12 27:18 27:20 28:13,14 30:4,6 33:8 44:25 45:1,2,14 54:7 62:25	questions 43:25 54:14 56:19	recess 63:13	represent 36:14	ROM 15:16 18:17 18:19 19:10 27:5,12,13 42:13 45:13,20 45:22 48:21,23 49:11,12,20 52:6 53:8 54:1,6 54:9 55:9 56:11 56:22 57:5,6,6,6 57:9 58:18,22 58:25 59:1,2,3,7 59:10,10,12,14 59:18,19,25 60:10,12,24 61:5,14,15,17 61:18,18,19,20 61:21 62:8,12 62:13,13,14,17
programmed 10:23	quite 8:18 13:4 14:15 18:24 20:22 28:2 31:11 33:6,24 35:3 50:12,13	recognize 23:21 61:2	representation 23:22 33:24 34:11,13,14 35:15 36:8	ROMs 49:7
programming 23:21 24:5,16	quote 51:13,22 52:24 61:24	recommend 50:13	representing 18:22	Ron 53:21
programs 7:8,17 10:24 11:16 13:11 24:8 25:6 25:9 26:3 27:16 30:12,17,24 31:3,6,8 36:9 46:14	<hr/> R <hr/>	record 31:19	represents 18:9 34:10	room 1:22 32:4
proof 26:13,16,17 26:24	R 2:13 5:10,12,13	RECROSS 3:4	request 19:6	Ross 49:25 50:1,2 50:8
prove 21:23 26:21 26:24 59:4	radiation 58:9	red 20:17 24:9 34:7,24	requires 27:5	routed 17:12
provide 5:20 6:19	raid 10:6,8,15	REDIRECT 3:4	reset 26:11	routine 25:21,23 27:19 53:10
provided 5:20 17:18	raise 4:13	refer 13:13 14:8 41:17,21,21	resident 41:14	routinely 50:18 51:3
provides 5:13 17:15	ran 21:12 49:10	reference 21:24 29:7	residents 41:13	RPR 1:21 64:16
providing 60:15	rates 53:2	referenced 52:18 53:24	resolution 54:4	rub 27:9
public 49:21	reach 14:24	referring 15:11 22:15	responsible 18:20	Rubin 2:24 22:1,4
publication 9:9	reaching 31:9	regarding 53:8	result 22:11	
Publicly 61:5	read 13:7 14:20 23:8 25:15 40:1 50:17 57:15 62:25	regardless 40:2 40:20	resume 63:5	
	reading 45:19,22	region 29:16,25 36:21,23 43:9	retained 5:16 9:14 10:3	
		regions 33:18 36:3,4	return 38:23 39:2	
		registers 42:6 43:9	reverse 11:20,22	
		regulations 64:10		
		relate 8:25		
		relates 8:20		
		relay 38:22		

<p>22:8 30:23 31:22,23 32:22 33:21 34:1,22 37:15 38:2 41:24 43:20 50:6,8,10,23 Rubin's 31:4,7 32:14 33:14 34:16 Rule 63:14 run 8:4 19:5 20:8 20:16 28:15 40:3,5 47:17 running 28:13 46:24</p> <hr/> <p style="text-align: center;">S</p> <p>SACV 1:7 San 2:15 Santa 1:16,23 4:1 sat 50:23 satellite 1:5 2:3 9:13 10:1,4 16:19,25 17:1,2 17:2 48:24 49:6 52:3,4 53:15 saw 27:15 32:15 60:25 62:20 saying 36:3 51:17 60:12 says 17:3 19:8,11 19:14 26:7 32:22 33:21 38:6 43:20 47:14 53:15 60:15 61:13 scanning 49:17 scene 54:13 science 28:4 screen 25:3 58:8 58:11 scuba 8:7,10 seated 4:18 sec 47:14 second 12:9 14:9 15:23,23 20:2 29:7 38:10 47:16 55:2 57:5 secret 19:19,20 secretive 11:24</p>	<p>12:22 Section 64:5 secure 51:5 57:18 58:15 security 18:25 19:14 50:2,5,9 57:17,17 59:15 see 12:6 17:5,6 18:1,10 19:7 23:8 24:5 25:2,5 25:18 26:6 27:7 29:23 30:18 31:21,22 32:16 32:19,25 33:12 33:24 34:1,23 35:6 36:25 38:18 40:4 43:3 43:16 45:25 48:11,23 49:8 50:9 52:13,14 52:15 54:8 58:11 59:24 60:9 61:12 62:3 seen 8:5,11 49:21 56:20 58:18 semiconductor 51:15 52:1 send 38:21 40:3 43:6 44:7 47:6 47:10,13 57:21 Sender 38:23 39:2 sending 43:11 47:18,22 sends 16:25 20:14 senior 57:18,19 sense 19:24 sent 39:6 61:4 September 58:24 59:6 sequences 27:19 series 32:19,20 services 5:13 6:19 session 4:5 set 11:10 14:9 55:17,17 sets 14:7 18:13 setup 34:3,8,25 35:13 share 37:12 shares 21:23</p>	<p>sheet 40:2 shell 29:7,9,10,22 32:23 33:4,4,21 33:23 34:19 35:11 46:20,22 shift 37:10 63:2 shorter 34:16 show 15:4 17:8,17 20:13 22:19 24:1 25:1,10,19 26:21 32:2 33:8 35:16 37:12 42:3,10 49:24 50:7 51:2 52:10 55:22 57:24 showed 31:8 showing 22:6 29:17 34:7 53:5 56:24 shown 61:3 shows 29:20 33:20 34:15 37:14 55:20 61:1 Shrek 17:6 19:22 side 24:2,2 29:21 29:21 35:5,6 62:19,19 signal 17:1,2 signals 51:4 significance 30:2 35:16 37:4 60:20 significant 14:5 28:24 29:18 31:4,11 33:6 62:25 significantly 28:5 Silicon 12:12 similar 34:13 52:8 61:18,19 simple 7:15 simplest 7:12 simply 36:4 sir 4:12,17,20,22 5:23 6:14 7:19 19:5 36:13 54:1 63:9 sit 58:9 six 13:15,16 59:2</p>	<p>Six-and-a-half 13:13 size 24:6 27:18 slide 23:4,5 26:4 27:17 29:19 30:9,10 37:14 49:10 50:7 53:5 56:24 60:2 61:1 61:10,22 62:1 slides 38:13 slight 24:17 slow 16:22 55:18 smallest 26:23 smart 7:10,11,14 8:20 10:21 17:4 18:14 20:11 49:6 50:17 51:3 51:12,19 Smith 41:18 SNYDER 2:13 society 50:23 software 5:14 8:7 11:14 somewhat 20:21 Sony 12:18 soon 40:15 sophisticated 51:18 sorry 35:3,19 sort 47:8 sorts 62:22 sounding 60:16 source 14:9,11 15:7 23:23 24:18,22,24 30:20 sources 61:16 special 13:11 43:10 specific 29:3 spell 4:23 52:5 spent 9:17,24 13:1 stack 28:1 29:17 29:25 35:13 42:7 44:22 45:2 46:21,21 staff 57:17 stage 21:3 46:7 staining 54:5</p>	<p>stand 10:16 standard 40:8 45:16 46:10 standards 36:17 stands 12:18 55:3 Stars 2:20 start 23:4 42:8,12 49:2 55:24 started 14:3,12 23:24 32:15 35:18 36:6 starts 20:18,23 49:8 56:5 state 4:20 6:8 States 1:1,22 6:5 7:21 64:6,10 status 13:21 stay 35:19 steal 56:22,23 58:2 stenographically 64:7 step 4:13 31:16 33:10,16,22,25 34:6,21 35:10 35:17 63:9 steps 23:1 46:5 sticks 21:1 STMICRO 57:21 57:22 stole 59:22 Stone 2:18 3:5 4:8 4:10 5:3,5,7 6:12 13:20 15:4 15:9 19:3 22:15 28:9 31:12,15 32:17 36:10 63:2 stop 28:14 43:22 store 42:14 stored 42:21,24 43:15 44:10 straight 35:7 strange 26:7 27:7 38:12 42:1 street 1:22 38:15 stress 36:1 stringent 11:10 structural 48:16 structure 31:6</p>
---	--	---	---	--

<p>student 53:3 students 53:23 studied 16:13,15 47:2 study 45:14 stuff 8:12 9:2 11:11 34:7,9 StuntGuy 15:14 54:12,12,13 56:18 59:24,24 60:3,4,5 StuntGuy's 60:12 stupidest 55:23 ST16 54:18 submitted 9:9,11 14:4 subroutine 27:8,9 32:25 33:1 subroutines 25:18 substantial 21:21 44:2 subtle 57:5 suddenly 58:14 suggest 58:18 suggesting 58:21 suggestion 22:16 suggests 39:14 46:12 suitable 9:22 Suite 2:8,15,20 summarize 30:15 summarizes 15:1 summary 30:10 53:5 Sunnyvale 57:25 superior 22:21 30:8 47:19 supporting 49:1 supposedly 23:10 sure 25:2 26:20 surely 21:20 36:20 surprising 47:24 suspected 10:9 14:16 Suzanne 61:4 Sweden 8:16 SWORN 4:15 system 6:4 7:1,4,7 7:10,12,15,20</p>	<p>11:12 13:22 14:9 15:17,24 16:3,19 18:19 28:12 38:3 46:13 55:1,2 57:16 59:7,10 59:12,14,18,19 62:8,12,14 systems 5:14 6:22 6:24 7:1,8,13,18 8:11 9:3,5,6 12:10 28:21</p> <hr/> <p style="text-align: center;">T</p> <hr/> <p>T 2:4 take 9:19 11:9 12:21 13:10 29:12 37:24 45:13,24 46:15 53:8 54:3 57:6 taken 23:13 57:8 takes 19:13,17,25 20:22 40:4 talk 5:22 16:18 17:13 22:2 49:14 talked 21:6 46:5 talking 20:1 30:3 33:7 36:5 50:8 50:18 53:1 60:21 tear 12:18 technical 5:20,21 15:13 technique 52:12 53:12,24 59:21 techniques 51:14 53:9 54:3,5,9 technologies 12:7 tell 5:23 21:14 25:4,8 28:5 39:24 47:3 58:13 teller 55:23,25 56:1,5,7 tempest 58:3,6 tenant 41:18 term 29:10,11 terminate 26:25 27:20</p>	<p>terminated 26:3 terminology 35:3 terms 18:18 21:14 21:18 28:25 terrific 50:11 test 40:3,5 testified 14:23 testifying 61:3 testimony 14:18 18:24 22:9 27:22 38:1 49:16 Texas 2:8 text 13:7 thank 4:7,10,12 4:17 5:2,5 6:11 63:10 theft 49:13 thick 13:14 thing 12:19,21 17:7,8 18:9 23:11 25:10 26:7 27:23 29:13 33:12 34:16,17,18 35:24 36:1 46:18 50:17 52:2 55:2,7 57:20 things 12:7 14:16 18:15 20:9 21:22 22:3,7 28:21 33:14,19 37:7,16 38:25 45:16 50:10 51:2 53:7 59:17 60:5 61:12 62:3 think 9:24 12:7 12:22 13:2,6,7 16:24 18:25 20:19 21:24 23:4 26:19 27:1 29:10 30:20 31:11 36:8 37:14 41:16 43:3,16 47:6 49:3 57:1,2 58:24 thinking 26:1 thinks 9:10</p>	<p>third 16:2 36:21 41:3 thought 33:6 61:12 thousand 56:15 thousands 13:5 three 10:5 18:13 28:1 32:18,21 40:5 45:8 46:5 49:12 55:1 56:24 thrown 18:3 time 6:20 13:1,3,7 18:8 19:25 20:1 22:11,17,23 24:24 25:17 26:17 43:3 47:19 51:20 52:1,15,25 53:18 58:12 60:6 62:6 63:3,4 timeout 47:20 times 18:10 57:24 time-consuming 20:3 title 61:5 64:6 TNO 52:18,20,21 52:22,24 53:18 54:20 55:6 today 18:10 38:5 40:8 58:5 60:3 told 54:14 top 24:8 29:23 32:15 35:24 36:12 42:7 44:22 45:2 46:20,21 topic 38:12 50:12 63:3 tops 40:5 total 11:16 touched 43:10 Toyota 12:3,7 transcript 1:14 64:7,9 transcripts 14:21 transmissions 58:11 transmit 19:19 25:14,16</p>	<p>transmitted 25:16 tremendous 10:1 13:25 48:25 trial 1:15 12:13 12:15 14:12 trouble 44:3 true 38:4,4,4,5 64:6 try 16:23 Tuesday 1:17 4:1 turn 10:15 26:19 turns 46:9 51:5 56:7 59:16 TV 12:19,19 17:6 19:22 49:6 52:3 52:4 53:15 58:8 58:11 twice 12:11 two 9:17 10:5 11:3 12:2 14:7 14:12 21:6 22:11 23:19,24 24:8,14 25:6 26:1 27:16 28:22 30:12,17 30:19 31:2,6,8 31:19 35:5,22 36:3,9 37:5 38:25 40:5 47:4 47:18 53:12 54:11,24 two-year 22:17 type 58:16 typical 32:11 typically 39:17 53:22</p> <hr/> <p style="text-align: center;">U</p> <hr/> <p>ultimately 52:8 understand 28:4 28:7 29:11 30:11 33:18 38:13 41:11 43:24 51:23 57:3 understanding 37:15 undertaking 63:1 Unfortunately 12:13 31:10</p>
--	--	--	--	--

<p>unique 40:12 47:1 unit 39:13,14,17 39:21 United 1:1,22 6:5 7:21 64:6,10 universities 53:19 university 6:1,8 50:3 53:3 unusual 46:14 update 38:6 updated 38:7 60:6 updating 60:7 uplink 16:25 upset 43:25 upshot 52:23 use 8:23 11:3 13:8 22:7 25:21 27:23 28:1,23 30:7 37:16,17 37:17,18 41:3 43:20,21 45:13 45:15 46:1 49:17 51:24 52:5,12 53:4 54:2 59:18,21 useful 44:13,17 USENIX 50:21,21 50:24 user 18:17 19:10 27:5 42:13 59:10 62:13 uses 8:21 U.S 8:21,23 9:1 64:15</p> <hr/> <p style="text-align: center;">V</p> <p>Valley 12:12 value 27:24 29:2 29:4 42:19 43:5 43:12,18,18 44:14 values 24:7 52:15 variable 37:18 41:3,4,20,21 42:7,11,19,23 42:25 43:4,12 43:14,18,19,20 44:6,8,9,11,14 44:19,21,24 45:5,8,11,13</p>	<p>46:2 48:10,15 48:19 variables 32:9 34:4 45:15 various 11:16 52:1 57:24 vernacular 46:24 55:5 version 35:7,8 versus 28:11 vetting 9:11 video 17:11,16,18 17:19 virus 20:20,22 29:11,12,15 30:5 32:23 46:6 46:9,25 47:21 Visa 11:10 voltage 55:14,17 Volume 1:8 4:2 63:15 vs 1:7 vulnerability 16:8 38:8 vulnerable 51:6 58:16</p> <hr/> <p style="text-align: center;">W</p> <p>W 2:13 WADE 2:4,6 wait 8:6 47:15 waiting 47:20 walk 57:10 Wal-Mart 11:13 want 26:20,22,22 31:21 33:17 34:18 40:18 42:3 43:25 45:24 52:21 wanted 44:25 wash 11:6,7,8,12 11:13 Washington 58:13 wasn't 24:14 62:4 waste 47:19 watch 18:5 way 8:22 10:14 17:6 18:24 19:20 20:19</p>	<p>22:20 24:21 26:1,2,10,12 32:6,20 33:2,2,3 39:24 40:2,9,24 43:8 44:4 45:12 45:25 46:10,19 47:12,13,21,23 47:24 49:9 58:7 ways 11:3 21:17 25:7,8 45:9 47:4 47:18 49:11,12 51:12 55:9 56:24 wearing 8:13 week 12:10 weeks 14:12 18:3 WELCH 2:4,6 well-known 55:10 went 10:14 23:1 24:21 25:4 58:14 West 1:22 2:14 we'll 15:18 33:16 55:23 we're 4:5 12:21 20:1 26:19 29:12 33:6 34:7 36:5 43:18 53:1 59:4 we've 17:17 18:21 19:22 20:5,10 21:6,8 26:4,18 33:7 38:15,16 41:4 44:3,4 46:5 57:22 60:14 white 36:12,15 wide 58:1 widely 56:16 widespread 12:23 WILLETTS 2:5 Windows 38:7 wiser 57:8 witness 4:8,15,16 4:19,22,24 5:1 6:10 13:16,18 15:6 32:1 41:5,9 63:10 WITNESSES 3:4 words 26:9 36:16 work 5:12 8:9</p>	<p>9:20 11:2 14:3 18:12 26:20,21 26:23 28:18 45:14 54:5 55:16 60:16 worked 11:1,15 45:12,19,21 working 8:15 28:15 61:24 62:8 works 16:24 17:7 45:14 58:7 workstation 51:25 world 50:23 53:16 worldwide 51:16 51:18 world's 50:4 55:22 Wow 42:17 write 6:17 25:23 27:19 40:15 46:14 writing 40:18 52:3 written 7:8,17 8:7 9:3,7 10:23 11:16 30:17 44:21 45:6 49:25 wrote 8:3 11:14 22:18,18 27:10 52:22 54:13</p> <hr/> <p style="text-align: center;">X</p> <p>X 3:1 xbr21 21:8</p> <hr/> <p style="text-align: center;">Y</p> <p>yeah 35:23 54:23 year 7:19 8:5 9:18 12:11,13 64:12 years 5:11 6:15 9:14 10:5 11:19 22:11 40:7 50:24 55:1 yellow 34:1,9,24 yells 56:6</p> <hr/> <p style="text-align: center;">Z</p>	<p>zero 42:19,21 44:9,14 54:6 zeros 54:7 zip 62:9</p> <hr/> <p style="text-align: center;">\$</p> <p>\$10 56:1,5 \$2500 53:1,18</p> <hr/> <p style="text-align: center;">0</p> <p>03-950 1:7</p> <hr/> <p style="text-align: center;">1</p> <p>1 42:23,25 54:6 1's 54:7 1-053 1:22 1:00 63:6 10 14:4 30:11,14 40:7 10:25 4:3 11:56 63:13 1100 33:2 12 1:8 4:2 120 38:18,19 39:3 39:4 120's 40:4 13 5:11 132 43:12,14 1400 2:20 15 40:7 15th 60:9 15-month 9:24 162 43:18 44:14 180 38:21 19C 42:8 1980 38:4 1990 38:4 1993 49:8 1995 58:12 1996 49:25 50:19 51:1,17 52:2 53:15 1997 52:19 1998 52:19 53:18 1999 2:20 58:24 58:24 59:6</p> <hr/> <p style="text-align: center;">2</p> <p>2 59:2 61:17,18,20 62:12</p>
---	---	---	--	--

2000 38:5 58:20	<hr/> 8 <hr/>			
60:9,17	8 53:25			
2002 49:9	800 13:2			
2007 14:3,4	81 33:3			
2008 1:17 4:1				
2008-04-29 1:25	<hr/> 9 <hr/>			
2100A8 32:15	9Ds 32:19,20 33:1			
220 38:25 39:1	90067 2:21			
40:3	92701 1:23			
23rd 21:8	94111 2:15			
2401 2:7	9472 1:21 64:16			
25 6:15 11:18	952-4334 2:9			
25th 60:17	96 51:8 53:13			
2600 2:15	98 55:1			
275 2:14	984-8700 2:16			
28 64:6	99 43:5			
288-01 61:14,16				
288-02 60:11,19				
29 1:17 4:1				
<hr/> 3 <hr/>				
3 59:3 60:12				
61:18,19,21				
3C 41:18				
3F 41:23				
310 2:21				
35 31:19				
36 25:24 31:19				
<hr/> 4 <hr/>				
4th 1:22				
4.0 6:7				
411 1:22				
415 2:16				
<hr/> 5 <hr/>				
5 3:5				
558-8141 1:23				
<hr/> 6 <hr/>				
6805 10:22,22,24				
<hr/> 7 <hr/>				
700 2:8				
713 2:9				
714 1:23				
7381 27:2,9				
753 64:5				
77057 2:8				
785-4600 2:21				