Palo Alto Networks, Inc., Petitioner

v.

Finjan, Inc., Patent Owner

IPR2015-01979

U.S. Patent No. 8,141,154

IPR2016-00919 (filed by Symantec Corp.) is joined with this proceeding

Petitioner's Oral Argument Demonstratives

December 15, 2016

Overview of Argument

I. Overview of ‘154 patent

II. Summary of the state of the art

III. Sirer was readily available to the interested public

IV. Claim construction: the BRI of “content” is “code”

V. The instituted claims are obvious

VI. The estoppel statute is inapplicable to PAN’s IPRs

VII. Finjan’s motions to exclude evidence should be denied

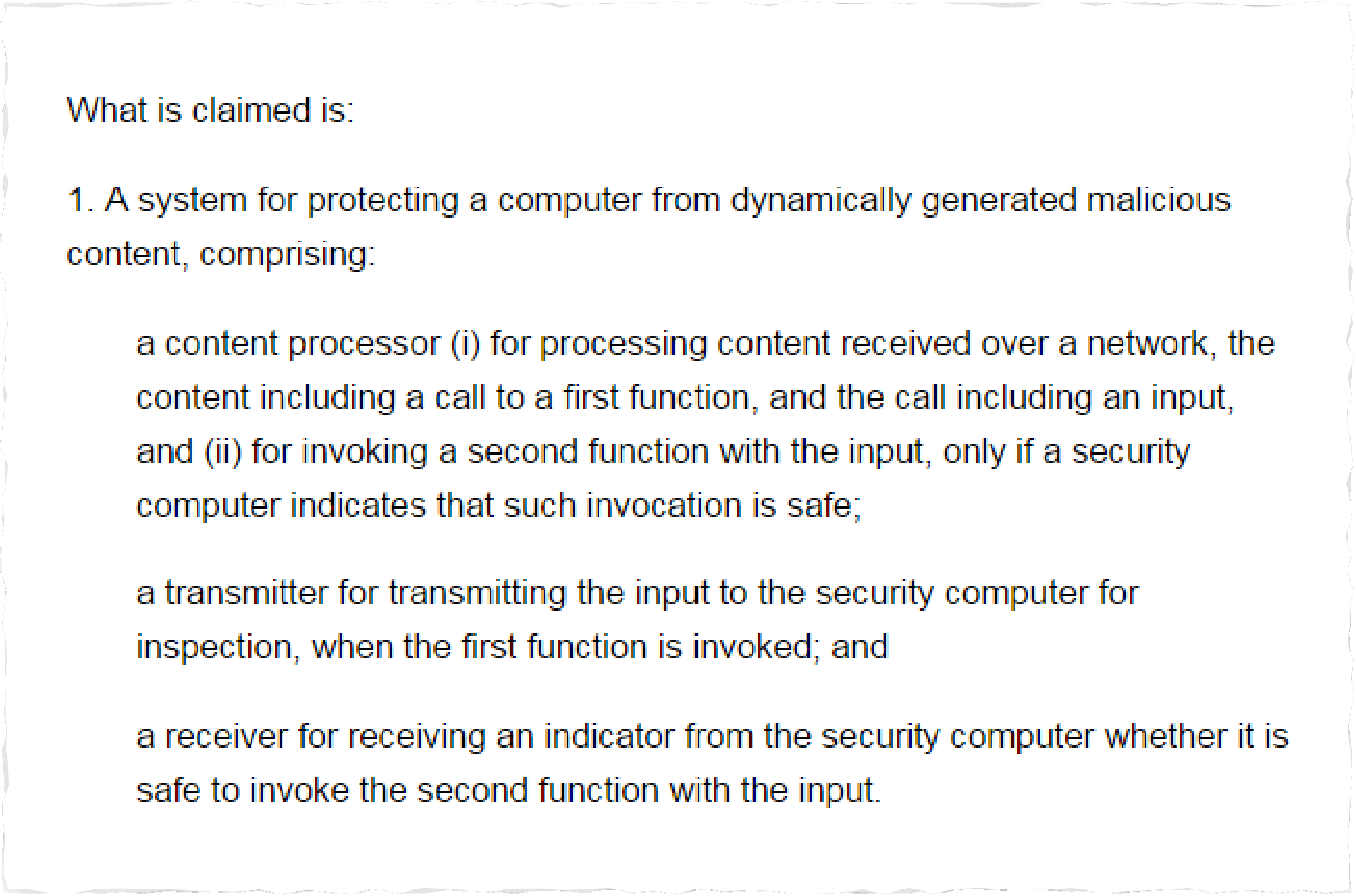
Overview of ‘154 patent

- The ’154 patent is directed at inspecting function call input variables for potentially malicious behavior to guard against malware. (See Ex. 1001 Abstract)

- Independent claims recite:
  - Static analysis (i.e., wrapping original function with substitute function)
  - Dynamic analysis (i.e., checking run-time values in the code)
  - Transmitting input variables for inspection at remote security computer
  - Invoking original function if security computer indicates it is safe

(Paper 2 at 6-8; Ex. 1002 at ¶¶ 57-62)

Exemplary Claim 1 of the ‘154 Patent



What is claimed is:

1. A system for protecting a computer from dynamically generated malicious content, comprising:

a content processor (i) for processing content received over a network, the content including a call to a first function, and the call including an input, and (ii) for invoking a second function with the input, only if a security computer indicates that such invocation is safe;

a transmitter for transmitting the input to the security computer for inspection, when the first function is invoked; and

a receiver for receiving an indicator from the security computer whether it is safe to invoke the second function with the input.

Ex. 1001 Claim 1.

Transmitters, receivers, and code inspectors were in prior art

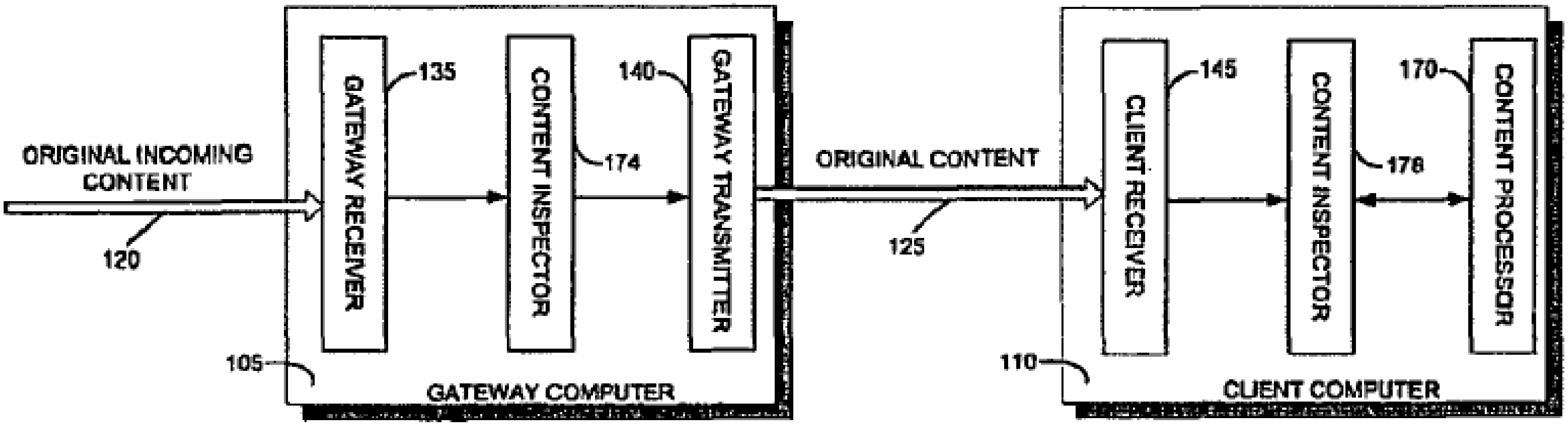
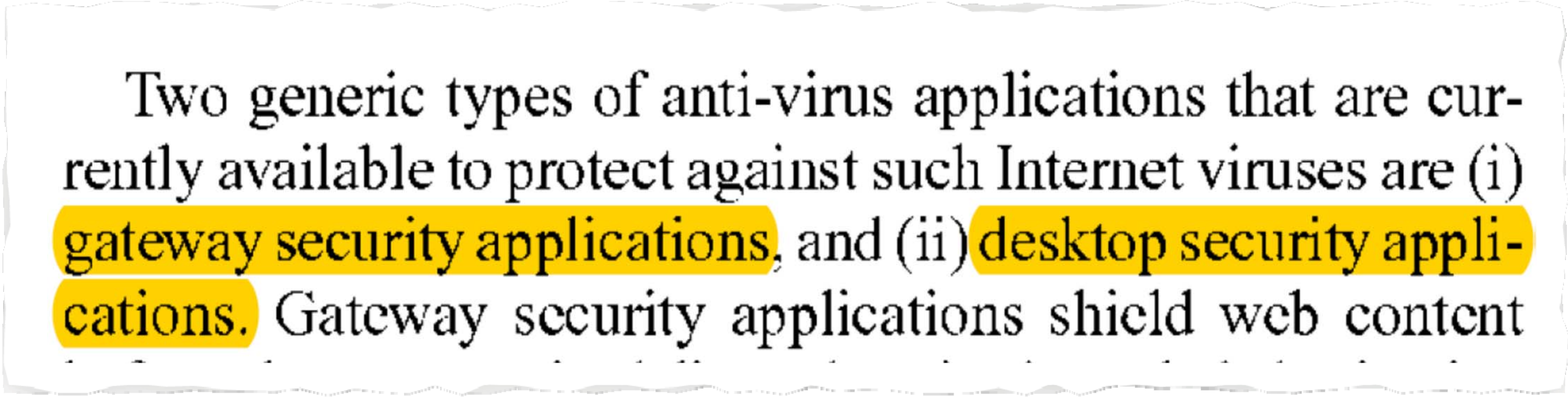


FIG. 1
(PRIOR ART)

Paper 2 at 6-9; Ex. 1001 at Fig. 1.

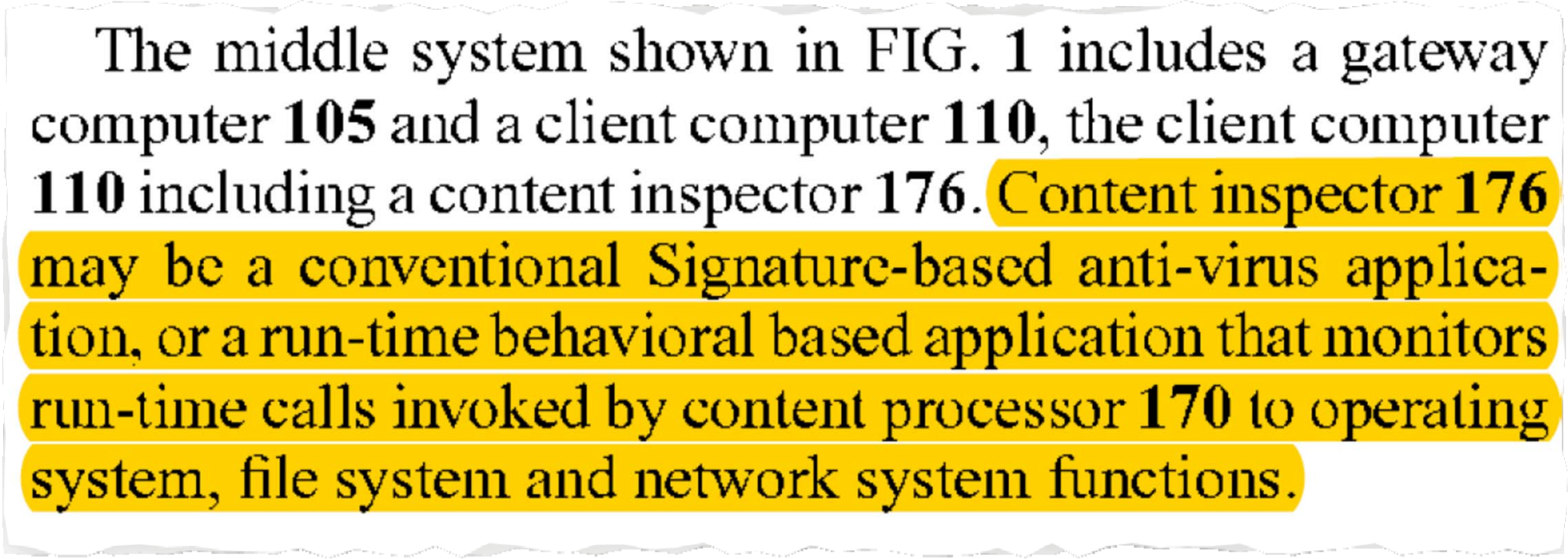
Gateway and desktop security applications were known



Two generic types of anti-virus applications that are currently available to protect against such Internet viruses are (i) gateway security applications, and (ii) desktop security applications. Gateway security applications shield web content

Paper 2 at 6-9; Ex. 1001 at 1:43-46.

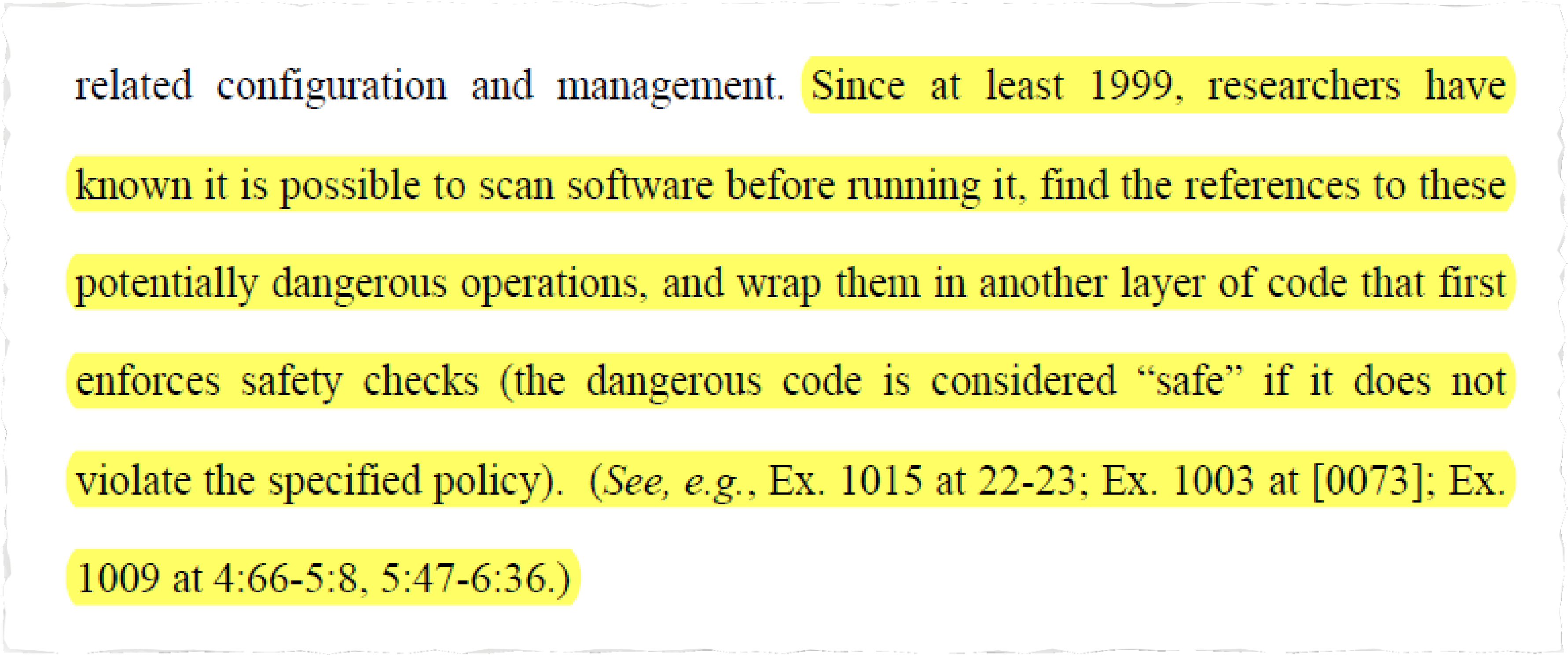
Run-time analysis of function calls was known in the art



The middle system shown in FIG. 1 includes a gateway computer **105** and a client computer **110**, the client computer **110** including a content inspector **176**. Content inspector **176** may be a conventional Signature-based anti-virus application, or a run-time behavioral based application that monitors run-time calls invoked by content processor **170** to operating system, file system and network system functions.

Paper 2 at 6-9; Ex. 1001 at 3:17-23.

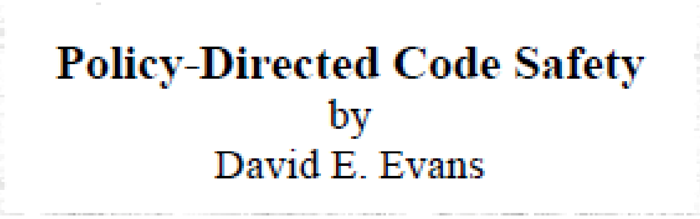
Undisputed that wrapping functions to detect dynamically generated malicious code was known in the art



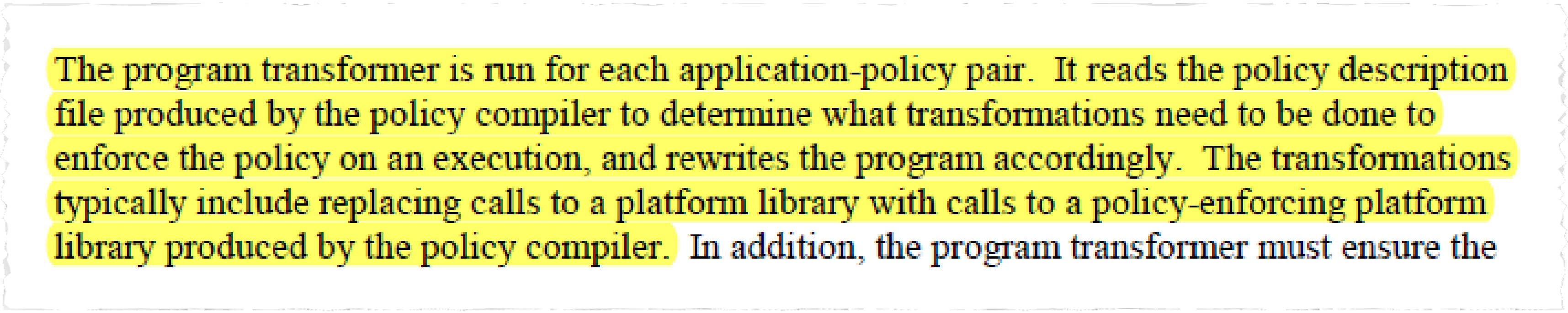
related configuration and management. Since at least 1999, researchers have known it is possible to scan software before running it, find the references to these potentially dangerous operations, and wrap them in another layer of code that first enforces safety checks (the dangerous code is considered "safe" if it does not violate the specified policy). (*See, e.g.*, Ex. 1015 at 22-23; Ex. 1003 at [0073]; Ex. 1009 at 4:66-5:8, 5:47-6:36.)

Paper 2 at 6-7; Ex. 1002, Rubin Decl. ¶ 40.

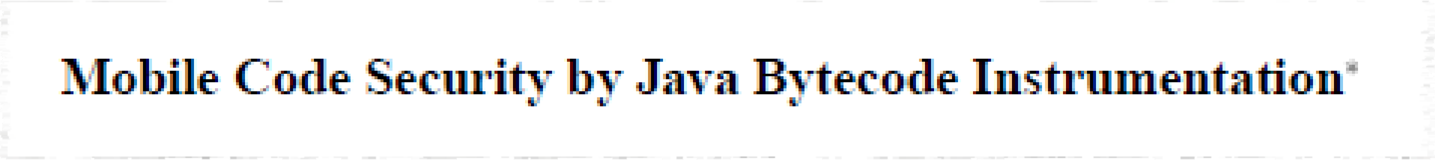
Undisputed that wrapping functions to detect dynamically generated malicious code was known in the art



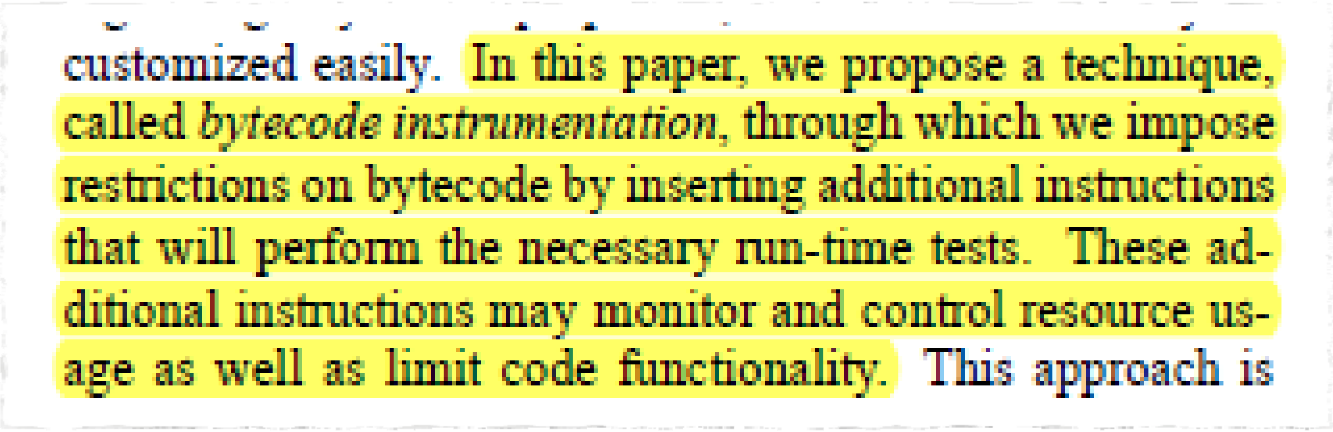
Policy-Directed Code Safety
by
David E. Evans



The program transformer is run for each application-policy pair. It reads the policy description file produced by the policy compiler to determine what transformations need to be done to enforce the policy on an execution, and rewrites the program accordingly. The transformations typically include replacing calls to a platform library with calls to a policy-enforcing platform library produced by the policy compiler. In addition, the program transformer must ensure the



Mobile Code Security by Java Bytecode Instrumentation*

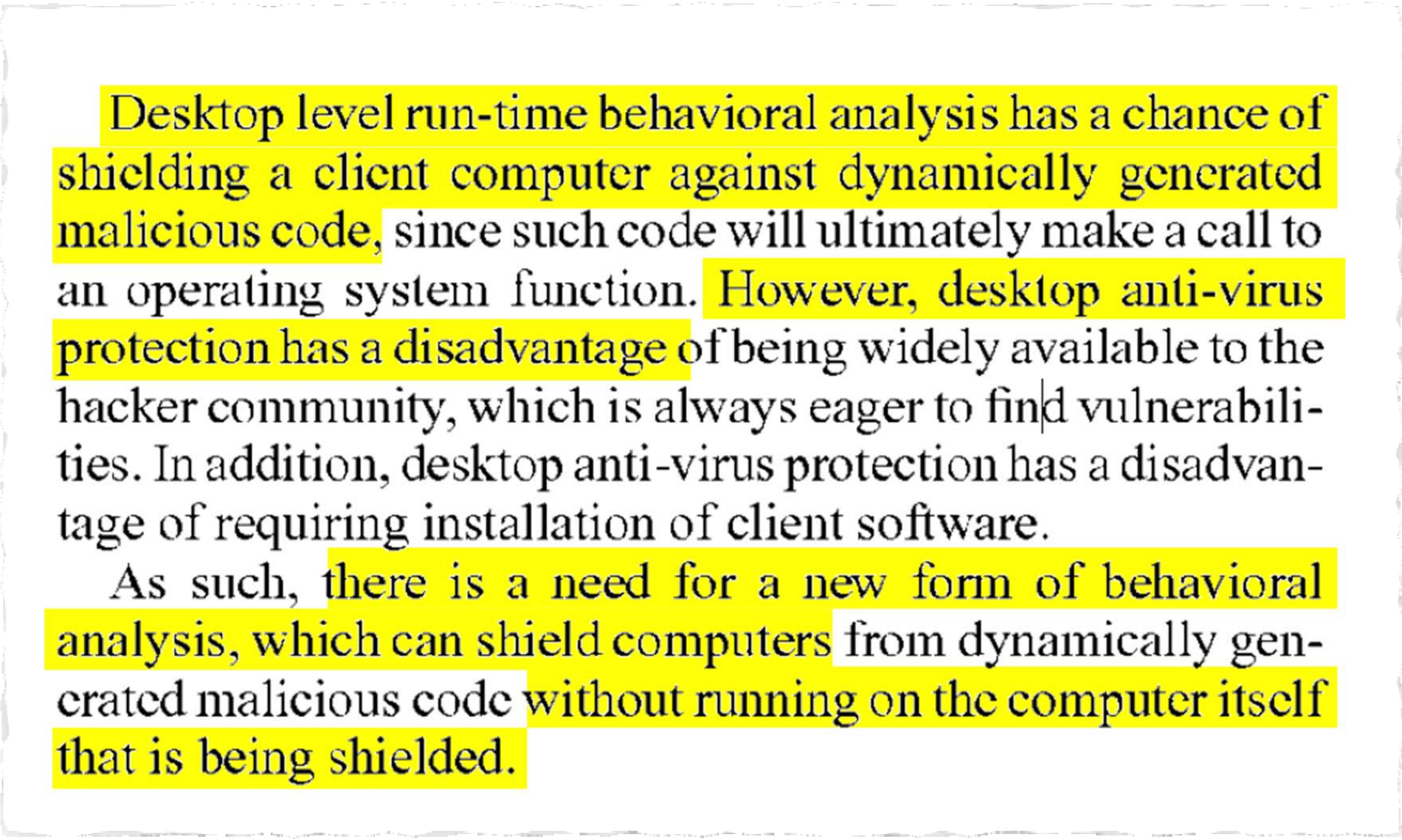


customized easily. In this paper, we propose a technique, called *bytecode instrumentation*, through which we impose restrictions on bytecode by inserting additional instructions that will perform the necessary run-time tests. These additional instructions may monitor and control resource usage as well as limit code functionality. This approach is

Paper 2 at 6-7; Ex. 1002, Rubin Decl. ¶ 40, Ex. 1015 at 22; Ex. 1011 at 4.

Sole alleged point of novelty: inspecting input at remote computer

- The sole point of novelty of the claimed invention lies in distributing the dynamic (run-time) inspection of the input variables to a remote computer. (Paper 2 at 1, citing Ex. 1001 at 4:15-26.)



Desktop level run-time behavioral analysis has a chance of shielding a client computer against dynamically generated malicious code, since such code will ultimately make a call to an operating system function. However, desktop anti-virus protection has a disadvantage of being widely available to the hacker community, which is always eager to find vulnerabilities. In addition, desktop anti-virus protection has a disadvantage of requiring installation of client software.

As such, there is a need for a new form of behavioral analysis, which can shield computers from dynamically generated malicious code without running on the computer itself that is being shielded.

Ex. 1001 at 4:15-26.

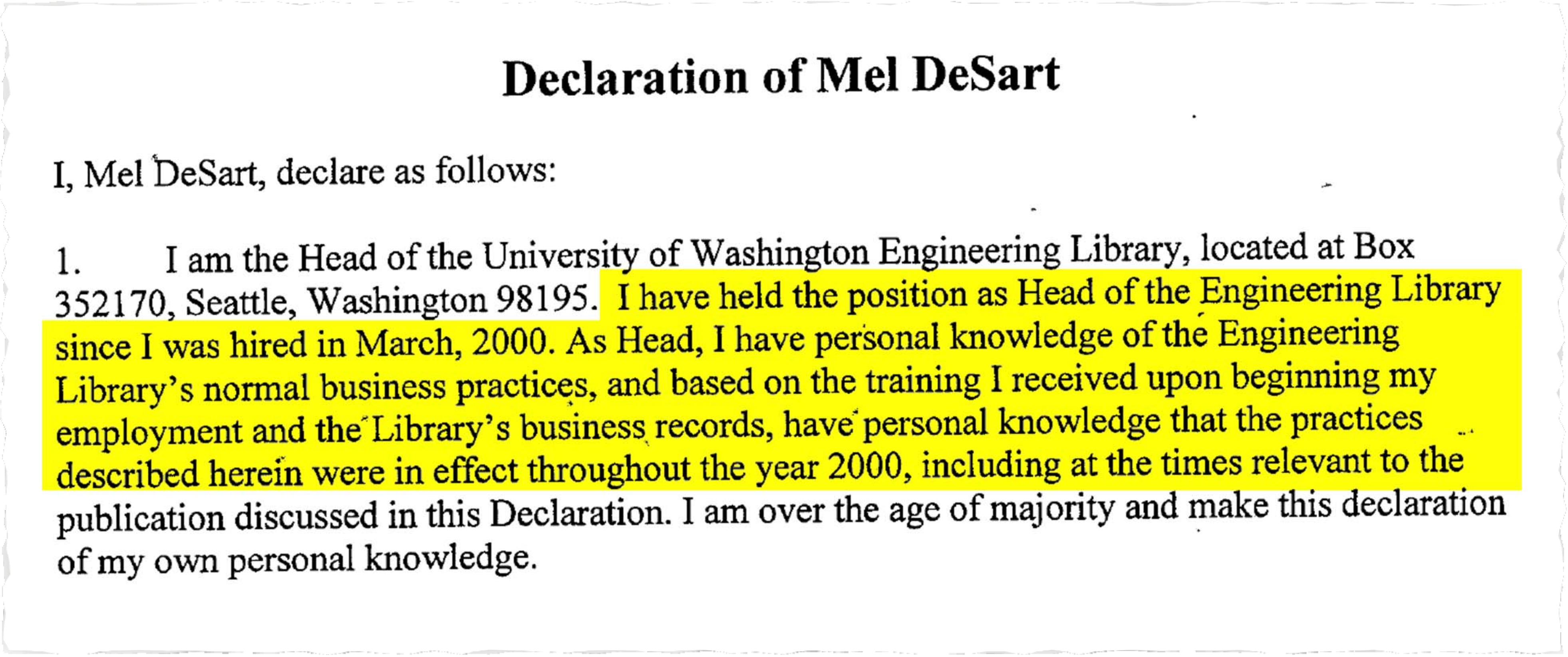
Sirer (Ex. 1004) is Prior Art

Evidence of the publication and public availability of Sirer is clear, convincing, and unrebutted

- Finjan does not dispute that
  - Sirer was published in the ACM Operating Systems Review
  - ACM Operating Systems Review is an established, respected academic journal
  - The University of Washington Engineering Library subscribed to ACM Operating Systems Review and received the journal that contained the Sirer article
  - ACM Operating Systems Review was publicly available in the University of Washington Engineering Library
  - Inspec research database indexed ACM Operating Systems Review articles
- Finjan only disputes public availability, but evidence overwhelmingly shows Sirer was available and was accessed

Ex. 1036, DeSart Declaration; Ex. 2006, DeSart Dep. Trans.; Paper 35, Petitioner's Reply, at 2-5.

Declaration from Head of UW Eng’g Library confirms publication



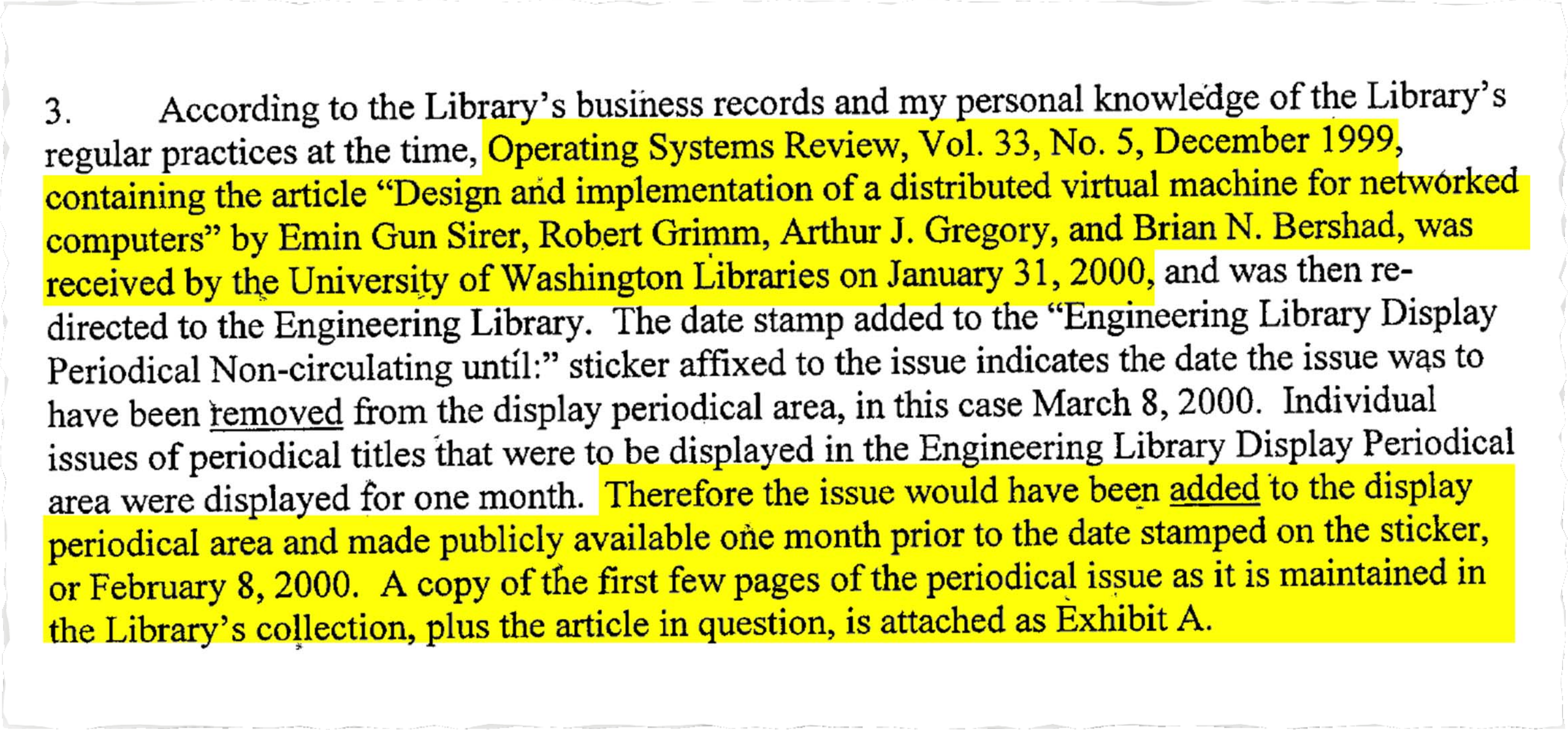
Declaration of Mel DeSart

I, Mel DeSart, declare as follows:

1. I am the Head of the University of Washington Engineering Library, located at Box 352170, Seattle, Washington 98195. I have held the position as Head of the Engineering Library since I was hired in March, 2000. As Head, I have personal knowledge of the Engineering Library’s normal business practices, and based on the training I received upon beginning my employment and the Library’s business records, have personal knowledge that the practices described herein were in effect throughout the year 2000, including at the times relevant to the publication discussed in this Declaration. I am over the age of majority and make this declaration of my own personal knowledge.

Ex. 1036

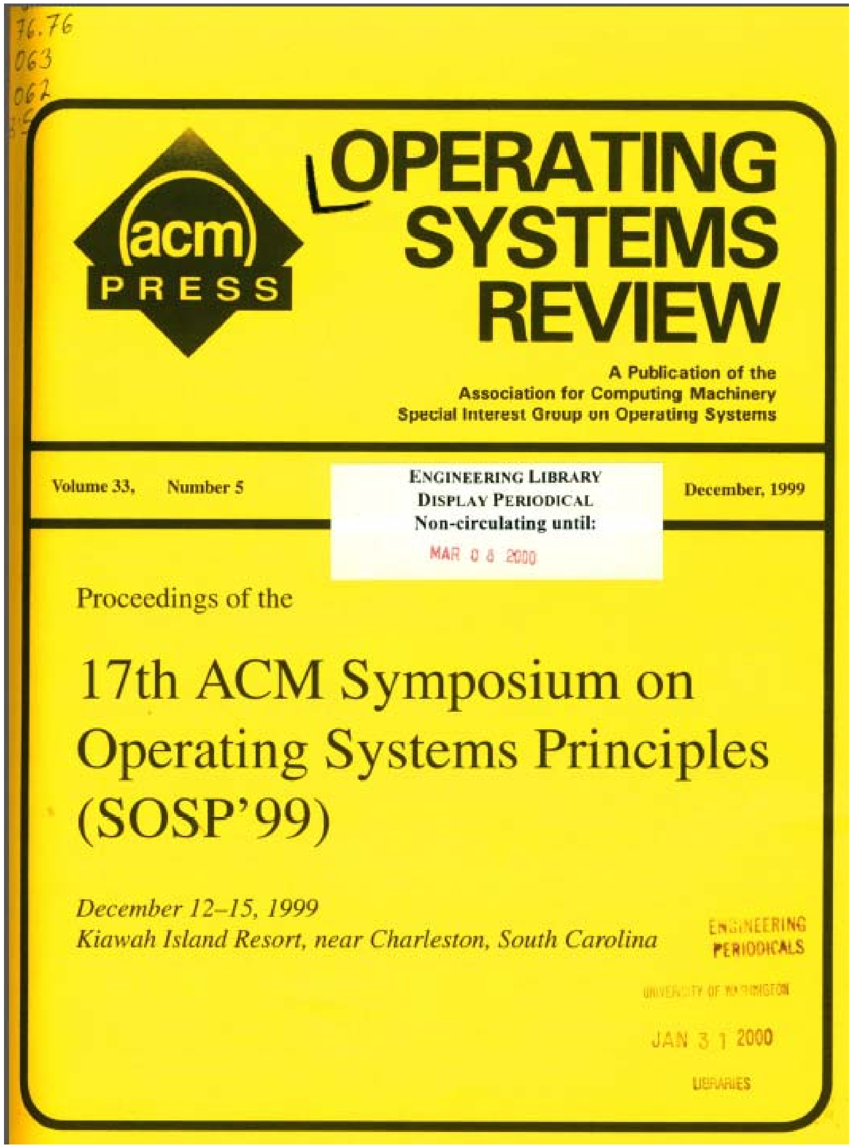
Declaration of Mel DeSart confirms publication of Sirer in 2000



3. According to the Library's business records and my personal knowledge of the Library's regular practices at the time, Operating Systems Review, Vol. 33, No. 5, December 1999, containing the article "Design and implementation of a distributed virtual machine for networked computers" by Emin Gun Sirer, Robert Grimm, Arthur J. Gregory, and Brian N. Bershad, was received by the University of Washington Libraries on January 31, 2000, and was then re-directed to the Engineering Library. The date stamp added to the "Engineering Library Display Periodical Non-circulating until:" sticker affixed to the issue indicates the date the issue was to have been removed from the display periodical area, in this case March 8, 2000. Individual issues of periodical titles that were to be displayed in the Engineering Library Display Periodical area were displayed for one month. Therefore the issue would have been added to the display periodical area and made publicly available one month prior to the date stamped on the sticker, or February 8, 2000. A copy of the first few pages of the periodical issue as it is maintained in the Library's collection, plus the article in question, is attached as Exhibit A.

Ex. 1036

Declaration of Mel DeSart confirms publication of Sirer in ACM Operating Systems Review journal received by UW Eng'g Library



OPERATING SYSTEMS REVIEW

acm PRESS

A Publication of the
Association for Computing Machinery
Special Interest Group on Operating Systems

Volume 33, Number 5 December, 1999

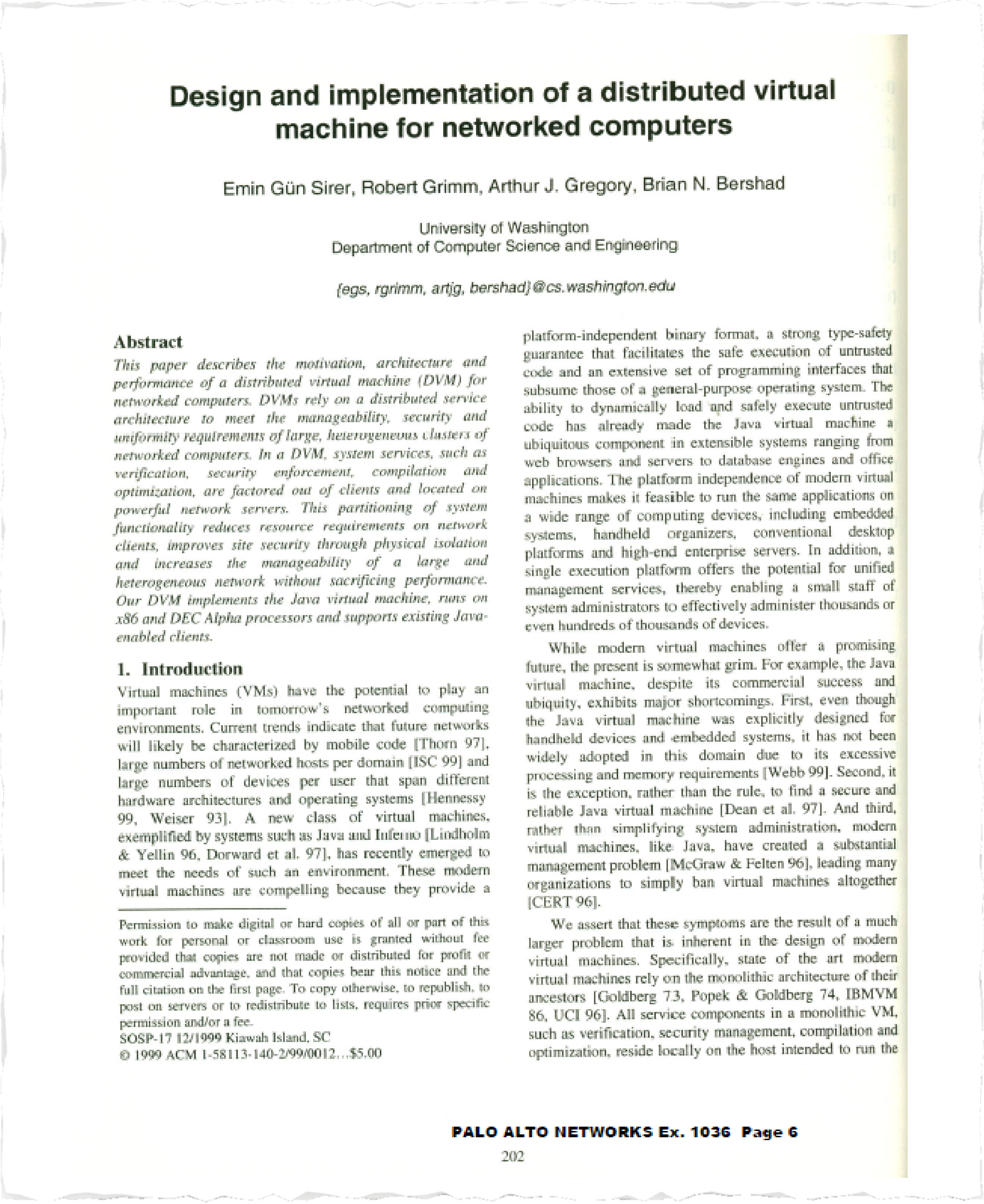
Proceedings of the

17th ACM Symposium on Operating Systems Principles (SOSP'99)

December 12–15, 1999
Kiawah Island Resort, near Charleston, South Carolina

Ex. 1036 Ex. A

Declaration of Mel DeSart confirms publication of Sirer



Design and implementation of a distributed virtual machine for networked computers

Emin Gün Sirer, Robert Grimm, Arthur J. Gregory, Brian N. Bershad

University of Washington
Department of Computer Science and Engineering

{egs, rgrimm, artjg, bershad}@cs.washington.edu

Abstract

This paper describes the motivation, architecture and performance of a distributed virtual machine (DVM) for networked computers. DVMs rely on a distributed service architecture to meet the manageability, security and uniformity requirements of large, heterogeneous clusters of networked computers. In a DVM, system services, such as verification, security enforcement, compilation and optimization, are factored out of clients and located on powerful network servers. This partitioning of system functionality reduces resource requirements on network clients, improves site security through physical isolation and increases the manageability of a large and heterogeneous network without sacrificing performance. Our DVM implements the Java virtual machine, runs on x86 and DEC Alpha processors and supports existing Java-enabled clients.

1. Introduction

Virtual machines (VMs) have the potential to play an important role in tomorrow's networked computing environments. Current trends indicate that future networks will likely be characterized by mobile code [Thorn 97], large numbers of networked hosts per domain [ISC 99] and large numbers of devices per user that span different hardware architectures and operating systems [Hennessy 99, Weiser 93]. A new class of virtual machines, exemplified by systems such as Java and Inferno [Lindholm & Yellin 96, Dorward et al. 97], has recently emerged to meet the needs of such an environment. These modern virtual machines are compelling because they provide a platform-independent binary format, a strong type-safety guarantee that facilitates the safe execution of untrusted code and an extensive set of programming interfaces that subsume those of a general-purpose operating system. The ability to dynamically load and safely execute untrusted code has already made the Java virtual machine a ubiquitous component in extensible systems ranging from web browsers and servers to database engines and office applications. The platform independence of modern virtual machines makes it feasible to run the same applications on a wide range of computing devices, including embedded systems, handheld organizers, conventional desktop platforms and high-end enterprise servers. In addition, a single execution platform offers the potential for unified management services, thereby enabling a small staff of system administrators to effectively administer thousands or even hundreds of thousands of devices.

While modern virtual machines offer a promising future, the present is somewhat grim. For example, the Java virtual machine, despite its commercial success and ubiquity, exhibits major shortcomings. First, even though the Java virtual machine was explicitly designed for handheld devices and embedded systems, it has not been widely adopted in this domain due to its excessive processing and memory requirements [Webb 99]. Second, it is the exception, rather than the rule, to find a secure and reliable Java virtual machine [Dean et al. 97]. And third, rather than simplifying system administration, modern virtual machines, like Java, have created a substantial management problem [McGraw & Felten 96], leading many organizations to simply ban virtual machines altogether [CERT 96].

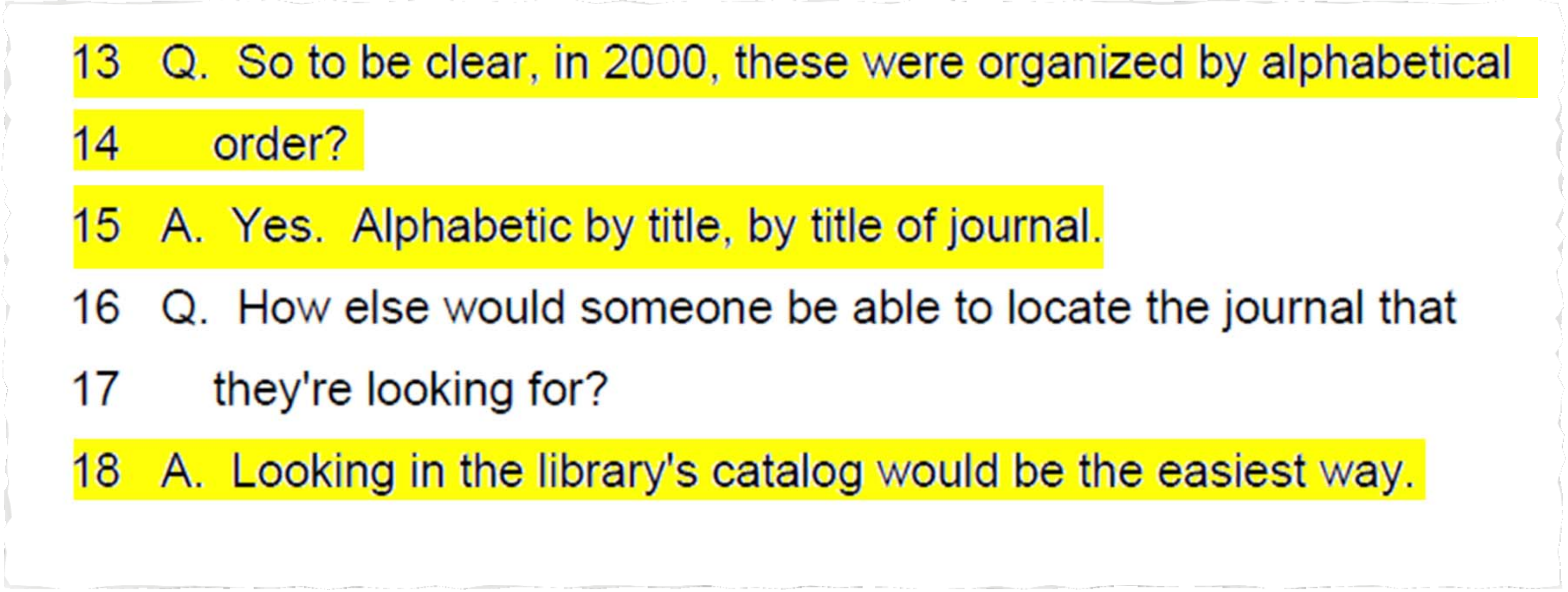
We assert that these symptoms are the result of a much larger problem that is inherent in the design of modern virtual machines. Specifically, state of the art modern virtual machines rely on the monolithic architecture of their ancestors [Goldberg 73, Popek & Goldberg 74, IBMVM 86, UCI 96]. All service components in a monolithic VM, such as verification, security management, compilation and optimization, reside locally on the host intended to run the

SOSP-17 12/1999 Kiawah Island, SC
© 1999 ACM 1-58113-140-2/99/0012…$5.00

Ex. 1036 Ex. A

DeSart testimony confirms Sirer was available to the public on the shelf in alphabetical order or by searching catalog for journal

- At UW Engineering Library, ACM Operating Systems Review journal was on the shelf in alphabetical order by title of journal and in library’s catalog



Q. So to be clear, in 2000, these were organized by alphabetical order?

A. Yes. Alphabetic by title, by title of journal.

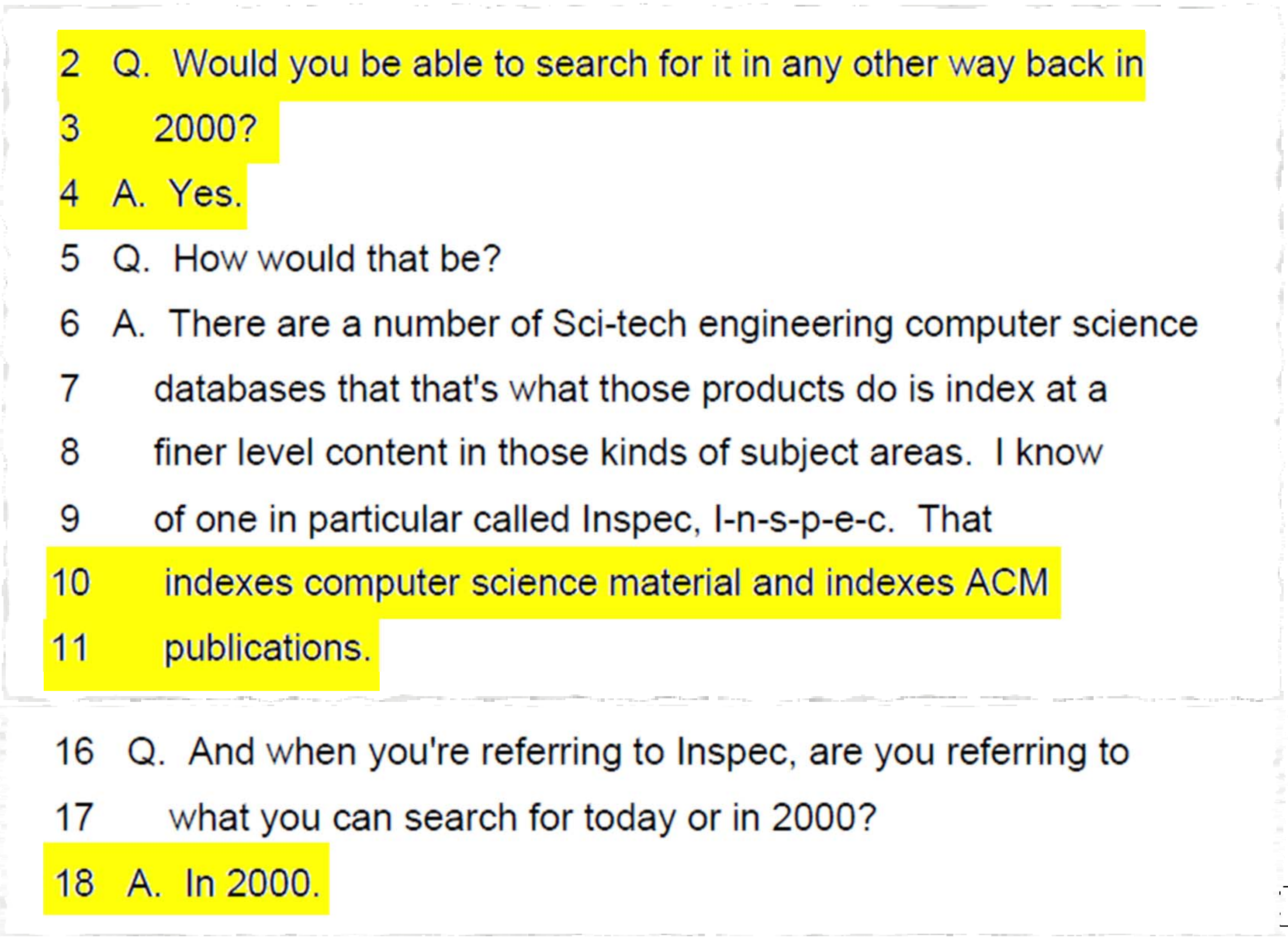
Q. How else would someone be able to locate the journal that they're looking for?

A. Looking in the library's catalog would be the easiest way.

Ex. 2006, DeSart Tr. at 10:13-18.

DeSart testimony confirms Sirer was indexed and searchable in the Inspec database

- Sirer article was indexed and searchable in Inspec database



Q. Would you be able to search for it in any other way back in 2000?

A. Yes.

Q. How would that be?

A. There are a number of Sci-tech engineering computer science databases that that's what those products do is index at a finer level content in those kinds of subject areas. I know of one in particular called Inspec, I-n-s-p-e-c. That indexes computer science material and indexes ACM publications.

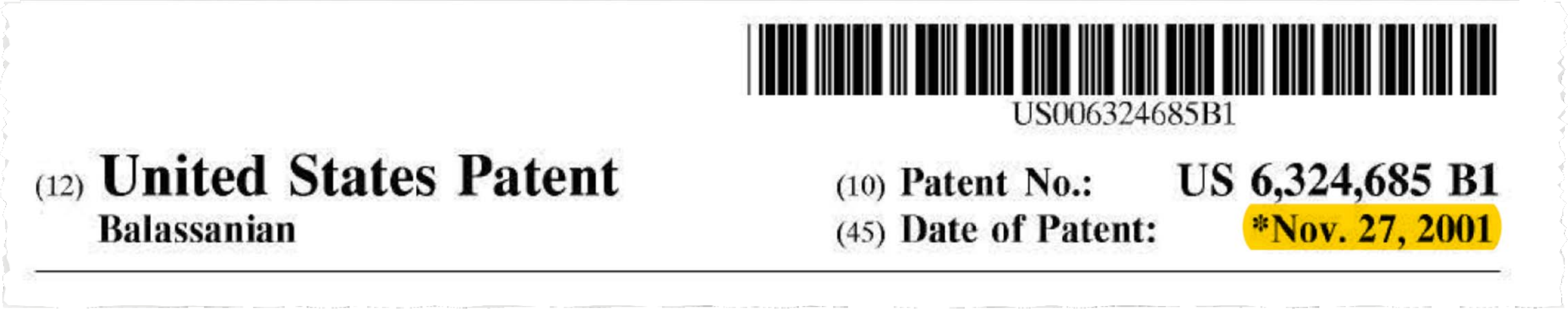
Q. And when you're referring to Inspec, are you referring to what you can search for today or in 2000?

A. In 2000.

Ex. 2006, DeSart Tr. at 12:2-18.

U.S. Patent 6,324,685 issued in 2001 and cited Sirer—confirming public availability and accessibility

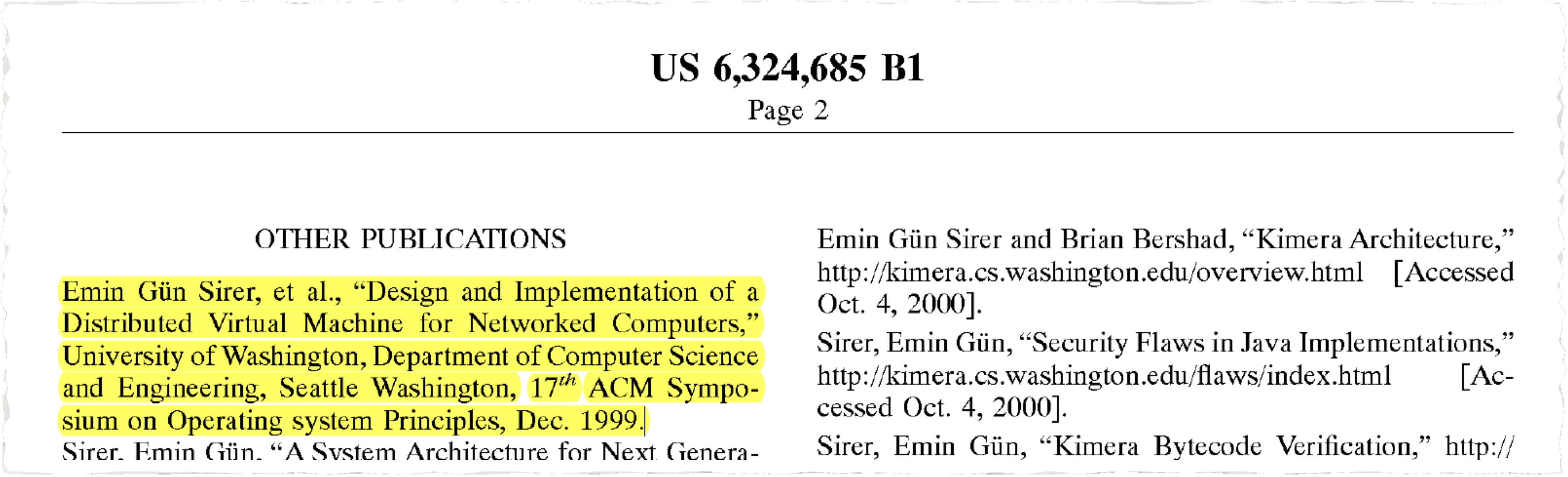
- '685 patent confirms that Sirer was available to the interested public in 2001



US006324685B1

(12) **United States Patent**
Balassanian

(10) **Patent No.:** **US 6,324,685 B1**
(45) **Date of Patent:** ***Nov. 27, 2001**



US 6,324,685 B1
Page 2

OTHER PUBLICATIONS

Emin Gün Sirer, et al., "Design and Implementation of a Distributed Virtual Machine for Networked Computers," University of Washington, Department of Computer Science and Engineering, Seattle Washington, 17th ACM Symposium on Operating system Principles, Dec. 1999.

Sirer, Emin Gün, "A System Architecture for Next Genera-

Emin Gün Sirer and Brian Bershad, "Kimera Architecture," http://kimera.cs.washington.edu/overview.html [Accessed Oct. 4, 2000].

Sirer, Emin Gün, "Security Flaws in Java Implementations," http://kimera.cs.washington.edu/flaws/index.html [Accessed Oct. 4, 2000].

Sirer, Emin Gün, "Kimera Bytecode Verification," http://

Ex. 1024

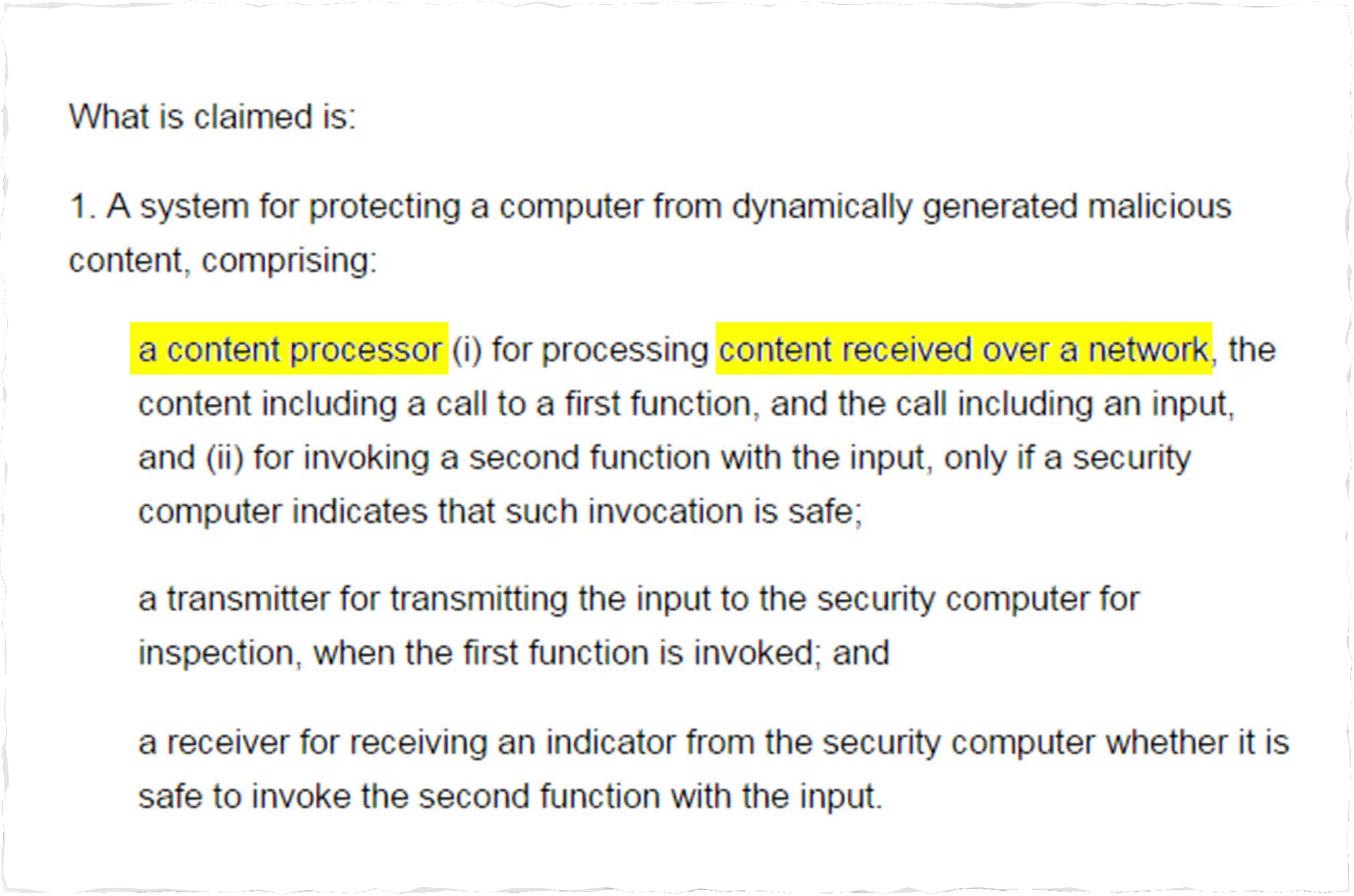
The Broadest Reasonable Interpretation of “Content” is “Code”

Finjan’s proposed construction of “content” is not the broadest reasonable interpretation

- **Finjan**: “a data container that can be rendered by a client web browser”

- **PAN**: “code”

Claims of the ‘154 patent do not limit meaning of “content”



What is claimed is:

1. A system for protecting a computer from dynamically generated malicious content, comprising:

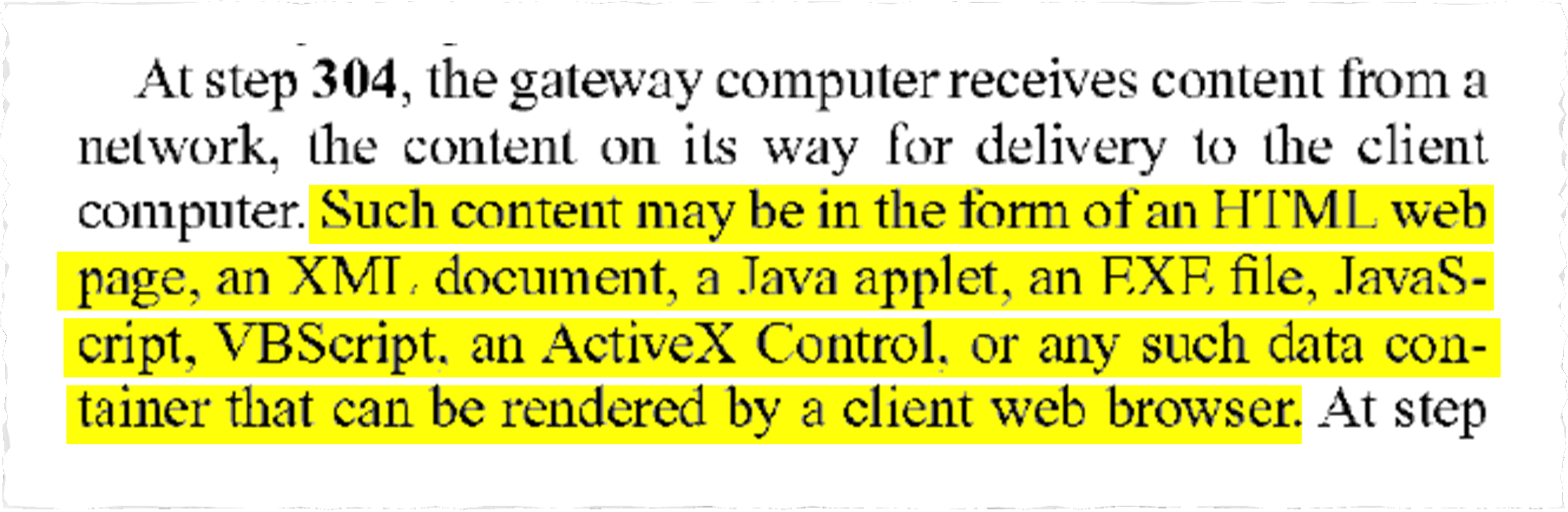
a content processor (i) for processing content received over a network, the content including a call to a first function, and the call including an input, and (ii) for invoking a second function with the input, only if a security computer indicates that such invocation is safe;

a transmitter for transmitting the input to the security computer for inspection, when the first function is invoked; and

a receiver for receiving an indicator from the security computer whether it is safe to invoke the second function with the input.

Ex. 1001 Claim 1.

‘154 patent specification describes “content” broadly and without limits



At step **304**, the gateway computer receives content from a network, the content on its way for delivery to the client computer. Such content may be in the form of an HTML web page, an XML document, a Java applet, an EXE file, JavaScript, VBScript, an ActiveX Control, or any such data container that can be rendered by a client web browser. At step

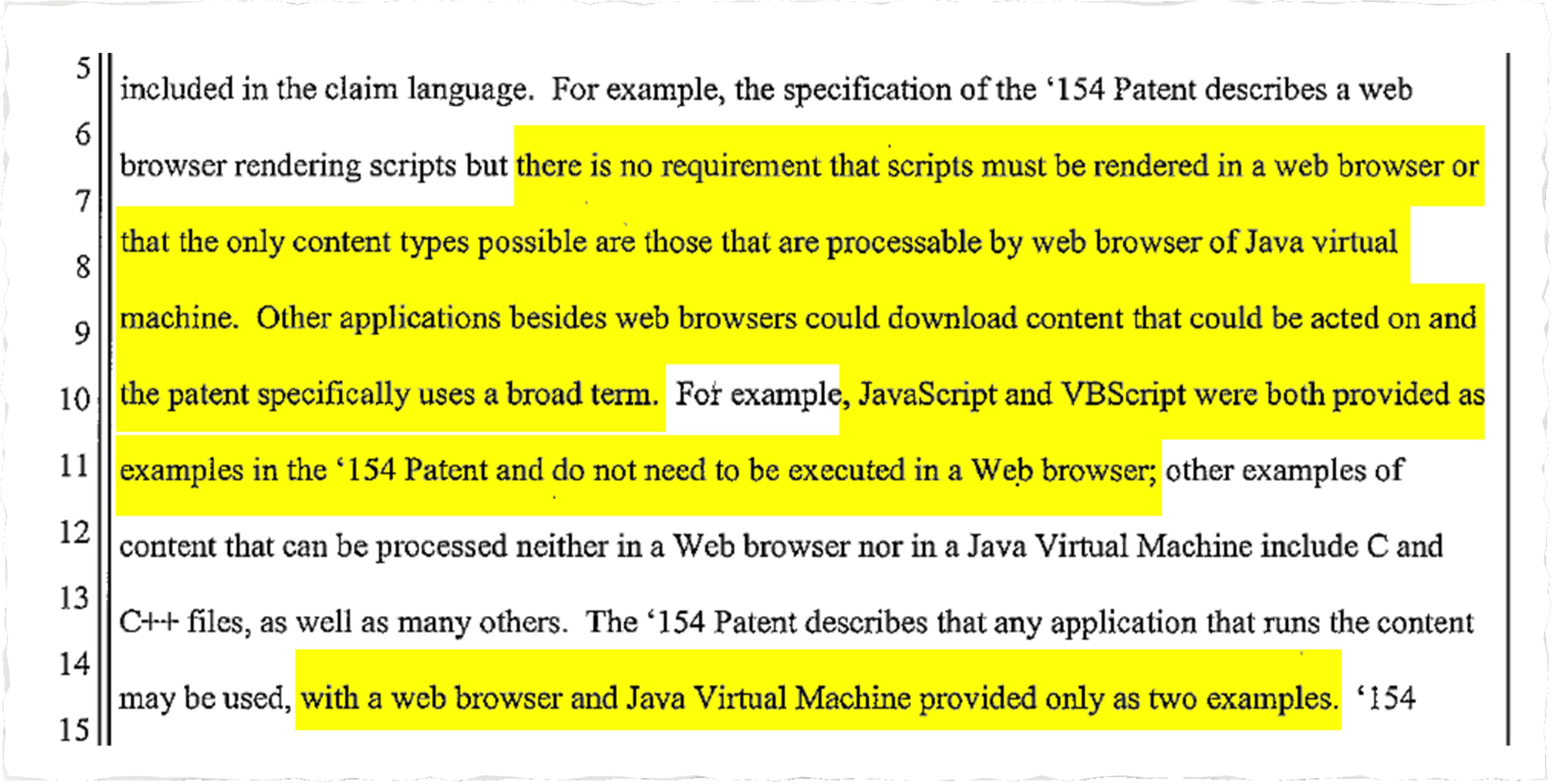
Ex. 1001 at 13:47-52.

There is no disavowal of claim scope

- *Hill-Rom Services, Inc. v. Stryker Corp.*, 755 F.3d 1367 (Fed. Cir. 2014)
  - "Disavowal requires that the specification or prosecution history make clear that the invention does not include a particular feature."
- *i4i Partnership v. Microsoft Corp.*, 598 F.3d 831, 844 (Fed. Cir. 2010)
  - "The specification's permissive language, 'could be edited,' 'can be created,' and 'ability to work,' does not clearly disclaim systems lacking these benefits."

Finjan’s construction improperly excludes preferred embodiments

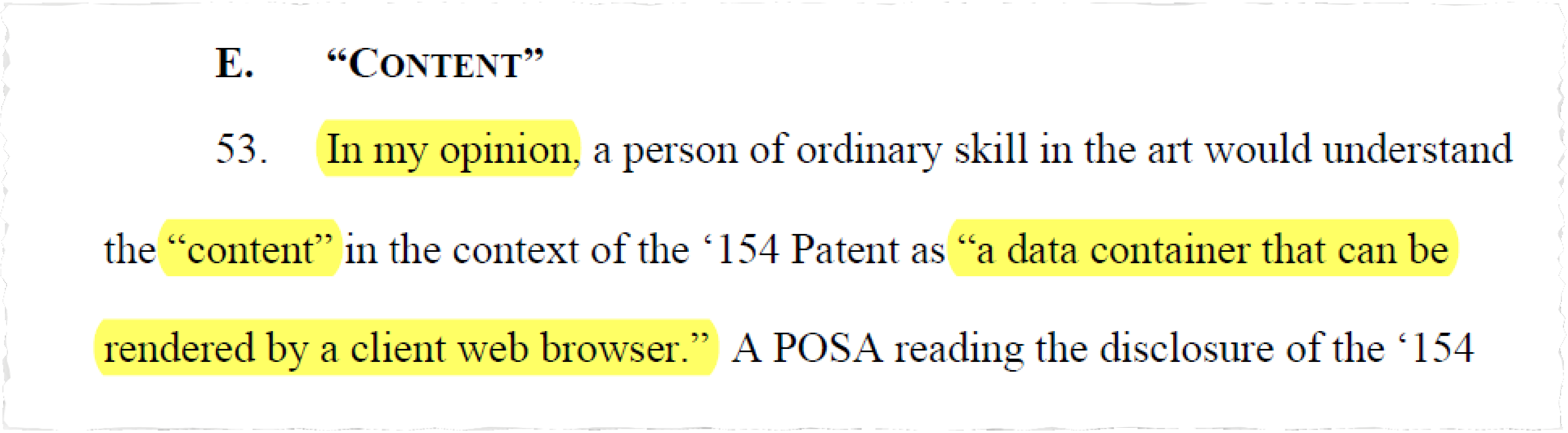
- Finjan’s expert Dr. Medvidovic: EXE files, JavaScript, and VBScript do not need to be rendered in a Web browser



included in the claim language. For example, the specification of the ‘154 Patent describes a web browser rendering scripts but there is no requirement that scripts must be rendered in a web browser or that the only content types possible are those that are processable by web browser of Java virtual machine. Other applications besides web browsers could download content that could be acted on and the patent specifically uses a broad term. For example, JavaScript and VBScript were both provided as examples in the ‘154 Patent and do not need to be executed in a Web browser; other examples of content that can be processed neither in a Web browser nor in a Java Virtual Machine include C and C++ files, as well as many others. The ‘154 Patent describes that any application that runs the content may be used, with a web browser and Java Virtual Machine provided only as two examples. ‘154

Ex. 1039, Medvidovic Decl. at 16.

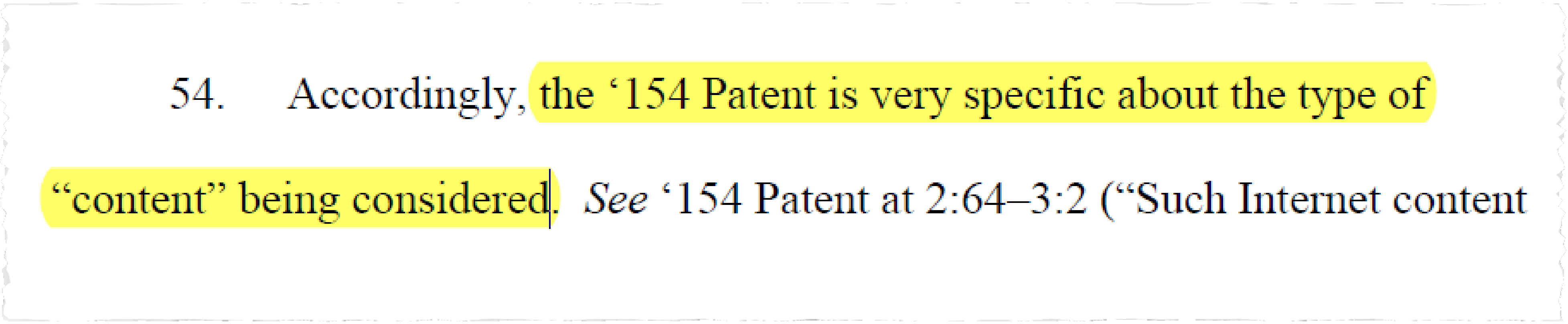
Dr. Medvidovic's sworn testimony to the Board



E. "CONTENT"

53. In my opinion, a person of ordinary skill in the art would understand the "content" in the context of the '154 Patent as "a data container that can be rendered by a client web browser." A POSA reading the disclosure of the '154

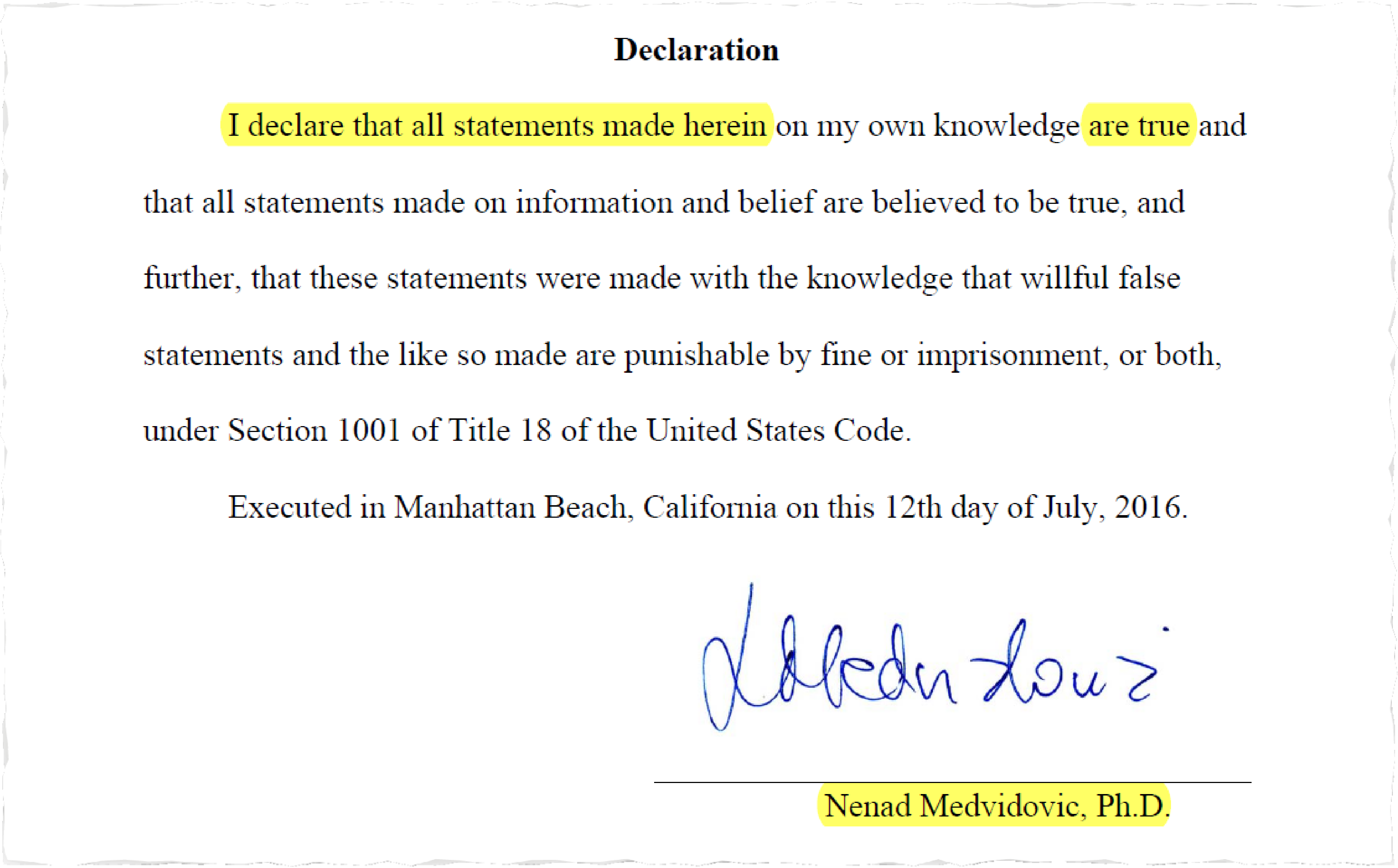
Ex. 2002, Medvidovic Decl. ¶ 53.



54. Accordingly, the '154 Patent is very specific about the type of "content" being considered. *See* '154 Patent at 2:64–3:2 ("Such Internet content

Ex. 2002, Medvidovic Decl. ¶ 54.

Dr. Medvidovic's sworn testimony to the Board



Declaration

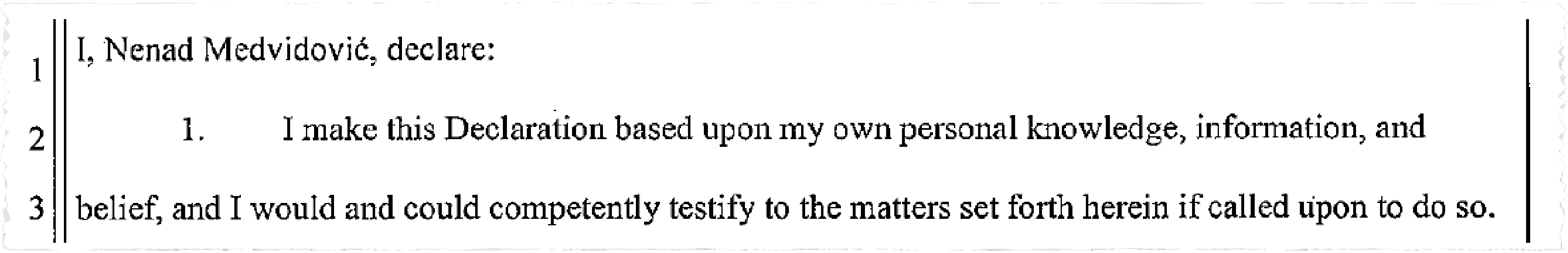
I declare that all statements made herein on my own knowledge are true and that all statements made on information and belief are believed to be true, and further, that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code.

Executed in Manhattan Beach, California on this 12th day of July, 2016.

Nenad Medvidovic, Ph.D.

Ex. 2002, Medvidovic Decl. at 81.

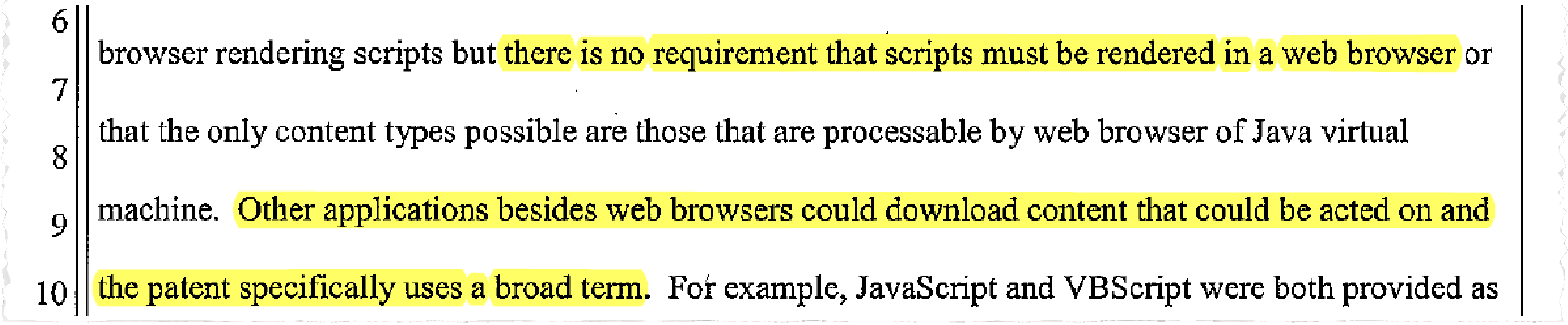
Dr. Medvidovic's contradictory sworn testimony in U.S. District Court, under the narrower *Phillips* claim construction standard



I, Nenad Medvidović, declare:

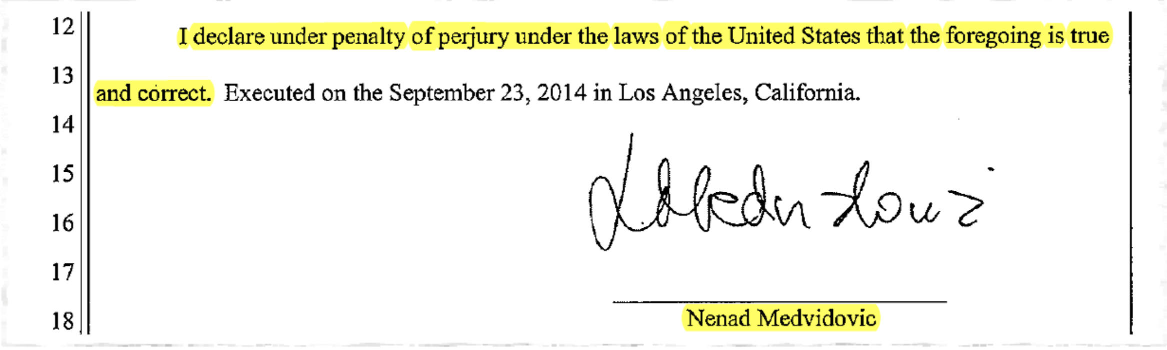
1. I make this Declaration based upon my own personal knowledge, information, and belief, and I would and could competently testify to the matters set forth herein if called upon to do so.

Ex. 1039, Medvidovic Decl. at 1.



browser rendering scripts but there is no requirement that scripts must be rendered in a web browser or that the only content types possible are those that are processable by web browser of Java virtual machine. Other applications besides web browsers could download content that could be acted on and the patent specifically uses a broad term. For example, JavaScript and VBScript were both provided as

Ex. 1039, Medvidovic Decl. at 16.

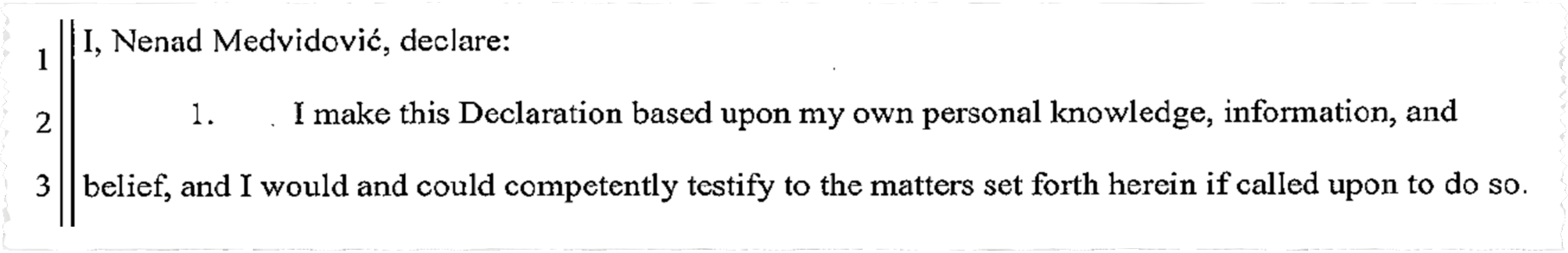


I declare under penalty of perjury under the laws of the United States that the foregoing is true and correct. Executed on the September 23, 2014 in Los Angeles, California.

Nenad Medvidovic

Ex. 1039, Medvidovic Decl. at 23.

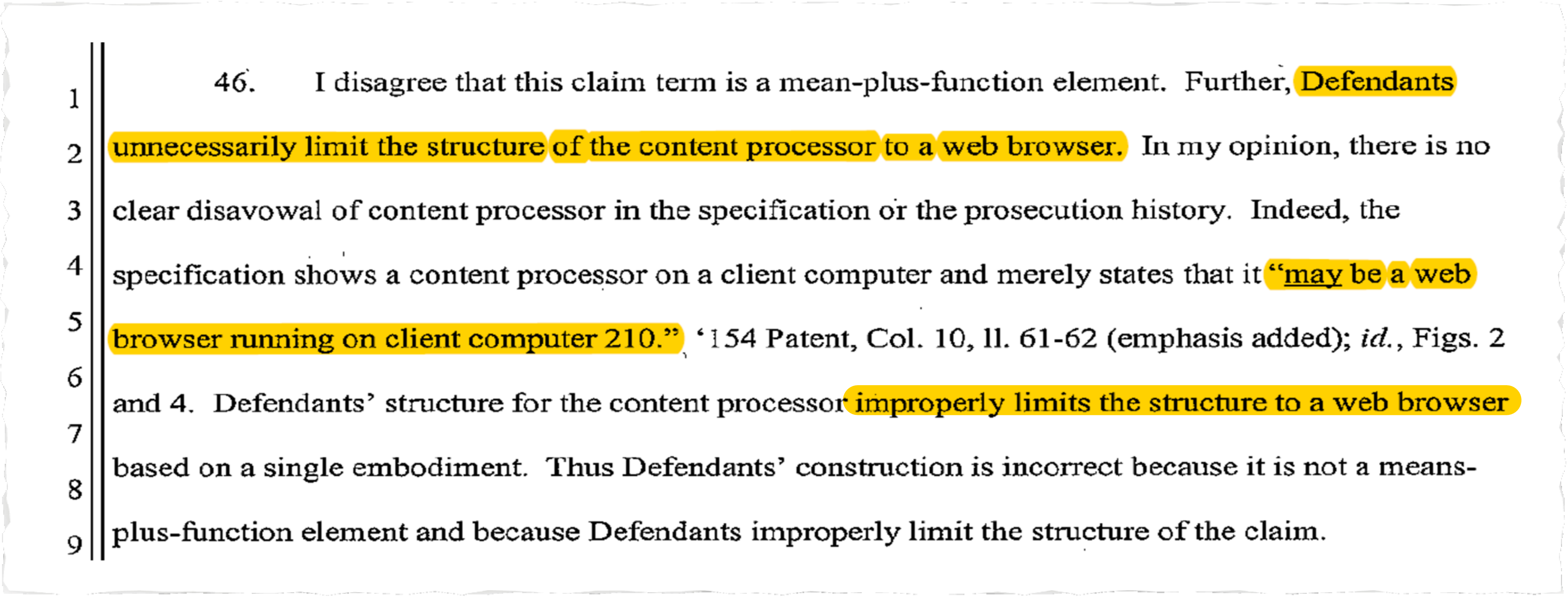
Dr. Medvidovic's contradictory sworn testimony in U.S. District Court, under the narrower *Phillips* claim construction standard



I, Nenad Medvidović, declare:

1. I make this Declaration based upon my own personal knowledge, information, and belief, and I would and could competently testify to the matters set forth herein if called upon to do so.

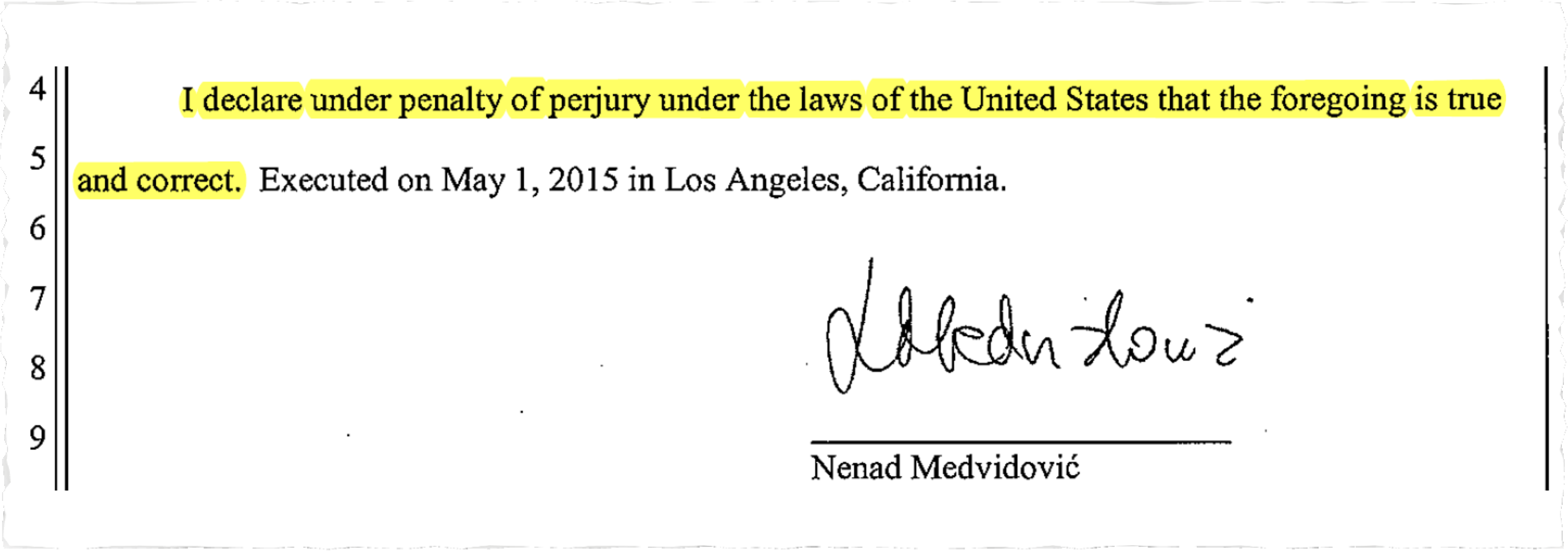
Ex. 1041, Medvidovic Decl. at 1.



46. I disagree that this claim term is a mean-plus-function element. Further, Defendants unnecessarily limit the structure of the content processor to a web browser. In my opinion, there is no clear disavowal of content processor in the specification or the prosecution history. Indeed, the specification shows a content processor on a client computer and merely states that it "may be a web browser running on client computer 210." '154 Patent, Col. 10, ll. 61-62 (emphasis added); *id.*, Figs. 2 and 4. Defendants' structure for the content processor improperly limits the structure to a web browser based on a single embodiment. Thus Defendants' construction is incorrect because it is not a means-plus-function element and because Defendants improperly limit the structure of the claim.

Ex. 1041, Medvidovic Decl. at 19.

Dr. Medvidovic’s contradictory sworn testimony in U.S. District Court, under the narrower *Phillips* claim construction standard

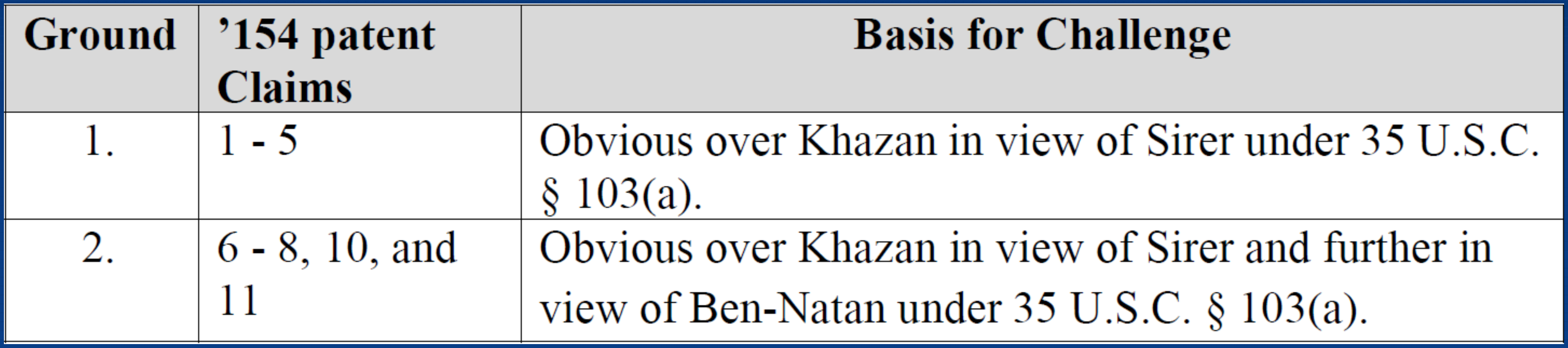


I declare under penalty of perjury under the laws of the United States that the foregoing is true and correct. Executed on May 1, 2015 in Los Angeles, California.

Nenad Medvidović

Ex. 1041, Medvidovic Decl. at 23.

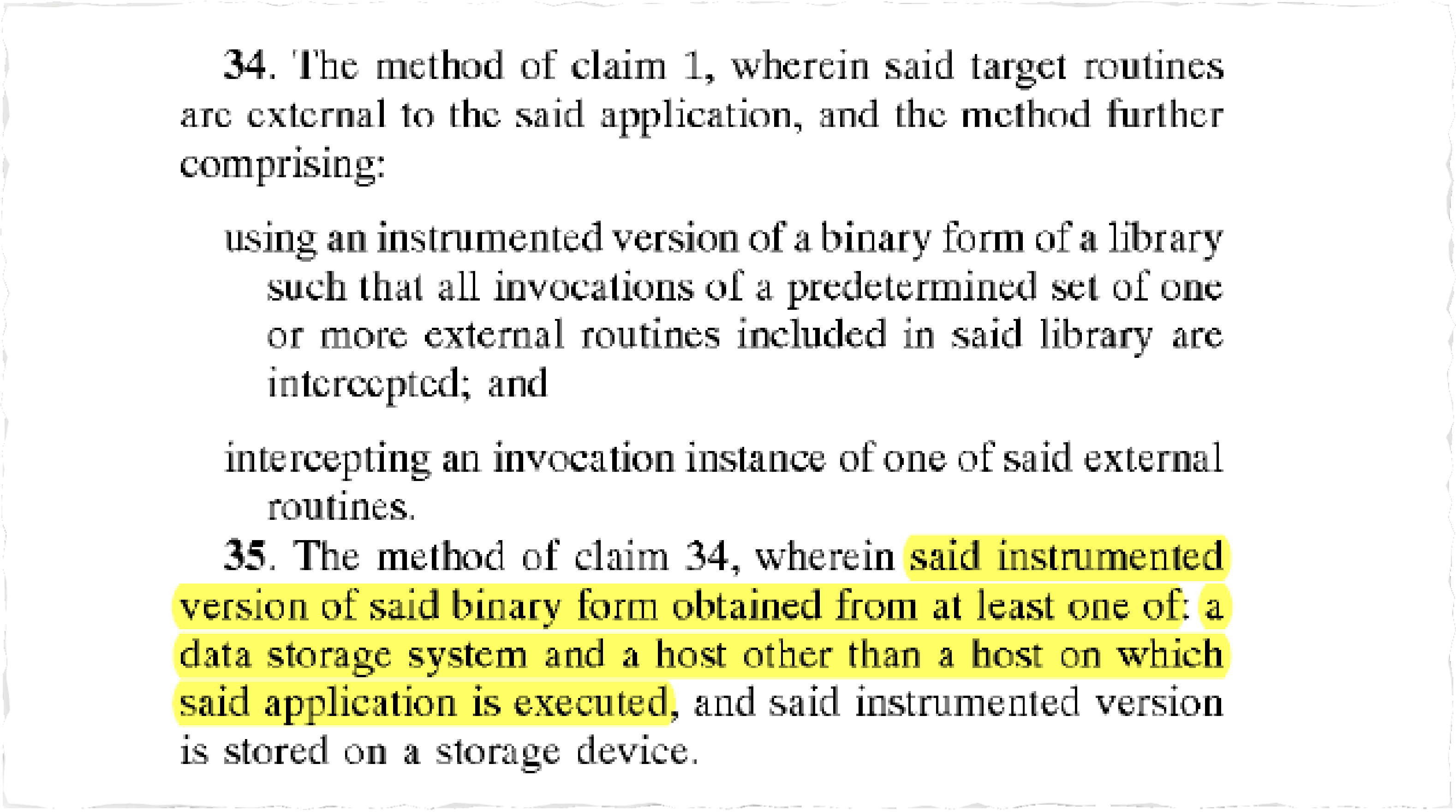
The Instituted Obviousness Grounds



Ground	’154 patent Claims	Basis for Challenge
1.	1 - 5	Obvious over Khazan in view of Sirer under 35 U.S.C. § 103(a).
2.	6 - 8, 10, and 11	Obvious over Khazan in view of Sirer and further in view of Ben-Natan under 35 U.S.C. § 103(a).

Khazan's libraries are "content received over a network"

- If the Board construes "content" as "code," Khazan's libraries are "content received over a network"



34. The method of claim 1, wherein said target routines are external to the said application, and the method further comprising:

using an instrumented version of a binary form of a library such that all invocations of a predetermined set of one or more external routines included in said library are intercepted; and

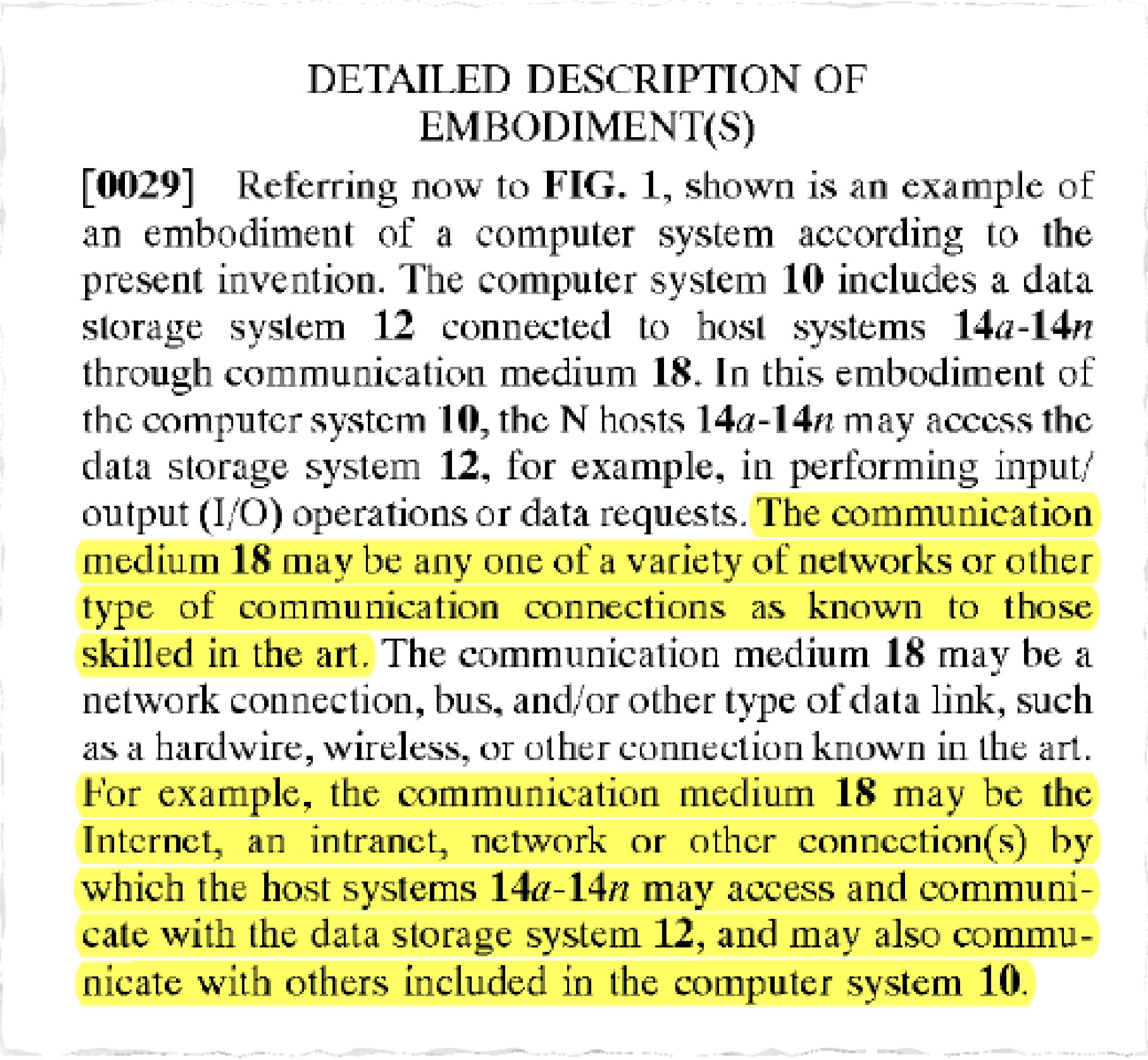
intercepting an invocation instance of one of said external routines.

35. The method of claim 34, wherein said instrumented version of said binary form obtained from at least one of: a data storage system and a host other than a host on which said application is executed, and said instrumented version is stored on a storage device.

Ex. 1003.

Khazan's libraries are "content received over a network"

- Khazan broadly discloses using its system over all types of networks



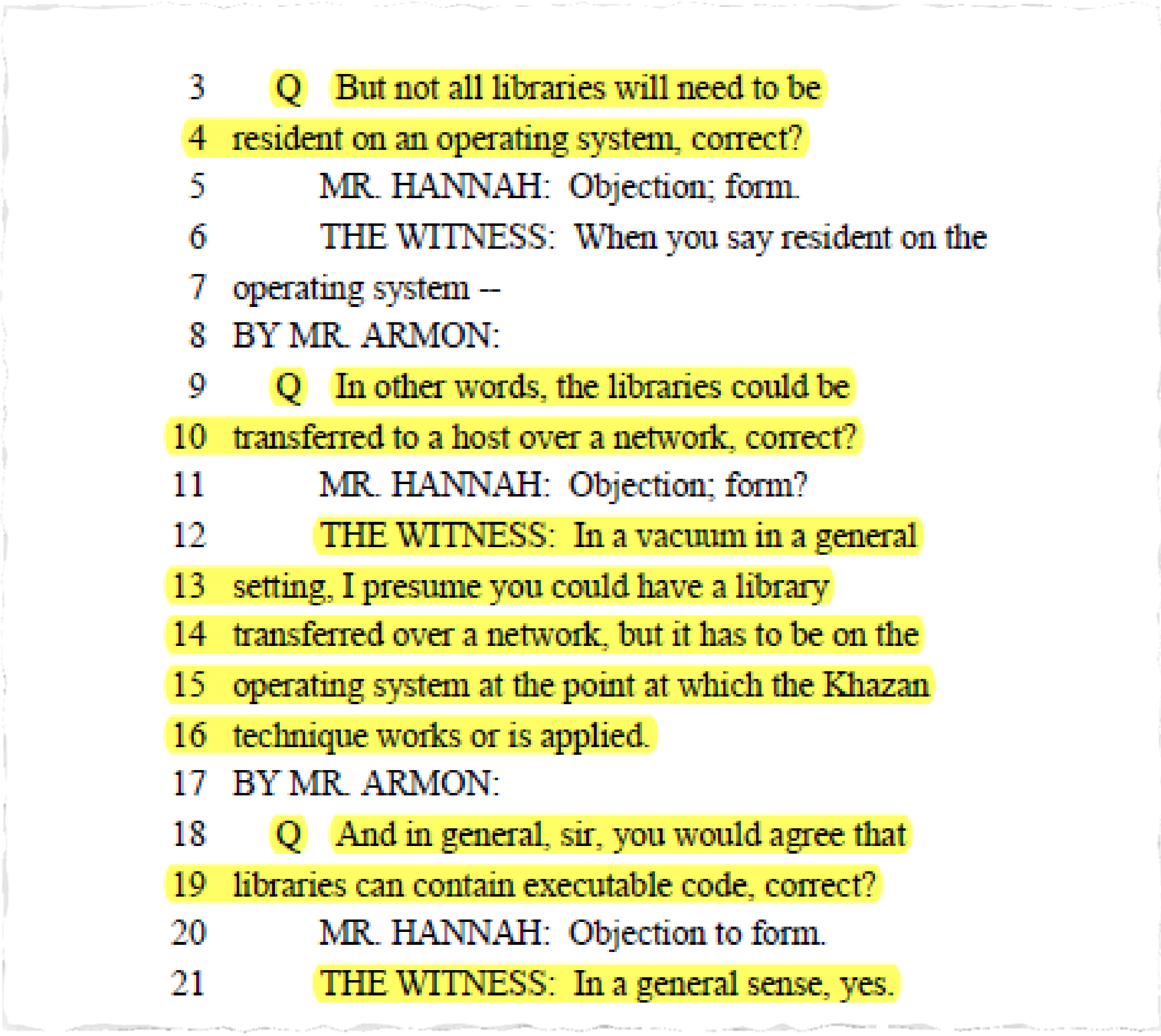
DETAILED DESCRIPTION OF EMBODIMENT(S)

[0029] Referring now to **FIG. 1**, shown is an example of an embodiment of a computer system according to the present invention. The computer system **10** includes a data storage system **12** connected to host systems **14***a*-**14***n* through communication medium **18**. In this embodiment of the computer system **10**, the N hosts **14***a*-**14***n* may access the data storage system **12**, for example, in performing input/output (I/O) operations or data requests. The communication medium **18** may be any one of a variety of networks or other type of communication connections as known to those skilled in the art. The communication medium **18** may be a network connection, bus, and/or other type of data link, such as a hardwire, wireless, or other connection known in the art. For example, the communication medium **18** may be the Internet, an intranet, network or other connection(s) by which the host systems **14***a*-**14***n* may access and communicate with the data storage system **12**, and may also communicate with others included in the computer system **10**.

Ex. 1003.

Khazan's libraries are "content received over a network"

- Finjan's expert Dr. Medvidovic admitted that libraries contain executable code and can be transferred over a network



Q But not all libraries will need to be resident on an operating system, correct?

MR. HANNAH: Objection; form.

THE WITNESS: When you say resident on the operating system --

BY MR. ARMON:

Q In other words, the libraries could be transferred to a host over a network, correct?

MR. HANNAH: Objection; form?

THE WITNESS: In a vacuum in a general setting, I presume you could have a library transferred over a network, but it has to be on the operating system at the point at which the Khazan technique works or is applied.

BY MR. ARMON:

Q And in general, sir, you would agree that libraries can contain executable code, correct?

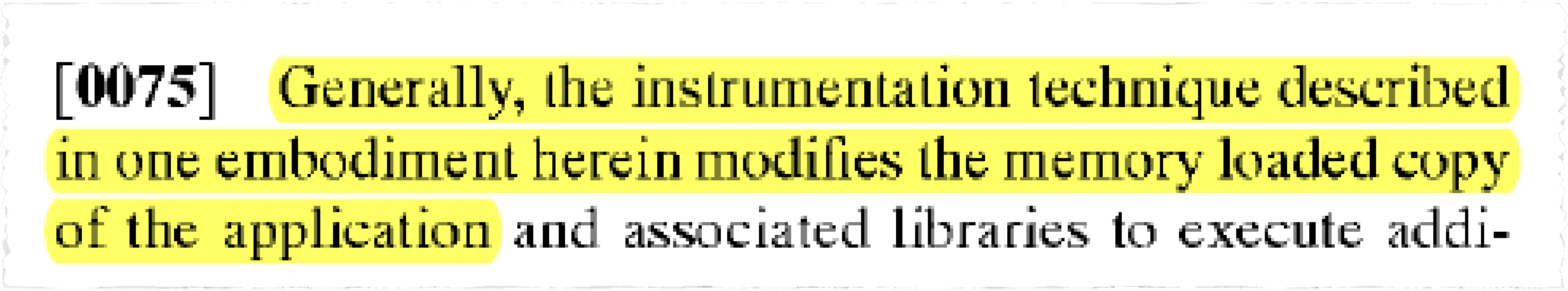
MR. HANNAH: Objection to form.

THE WITNESS: In a general sense, yes.

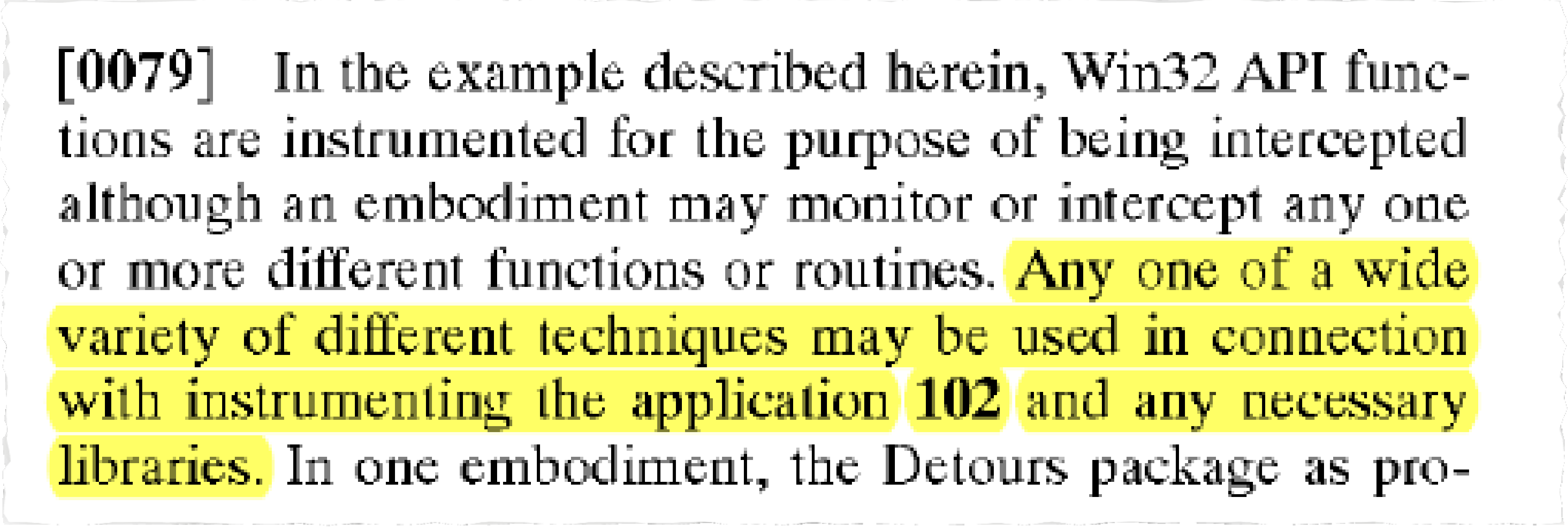
Ex. 1038, Medvidovic Tr. at 45:3-21.

The "content received over a network" limitations are obvious in view of Khazan's instrumented application

- Khazan's disclosure explicitly teaches instrumenting applications



[0075] Generally, the instrumentation technique described in one embodiment herein modifies the memory loaded copy of the application and associated libraries to execute addi-

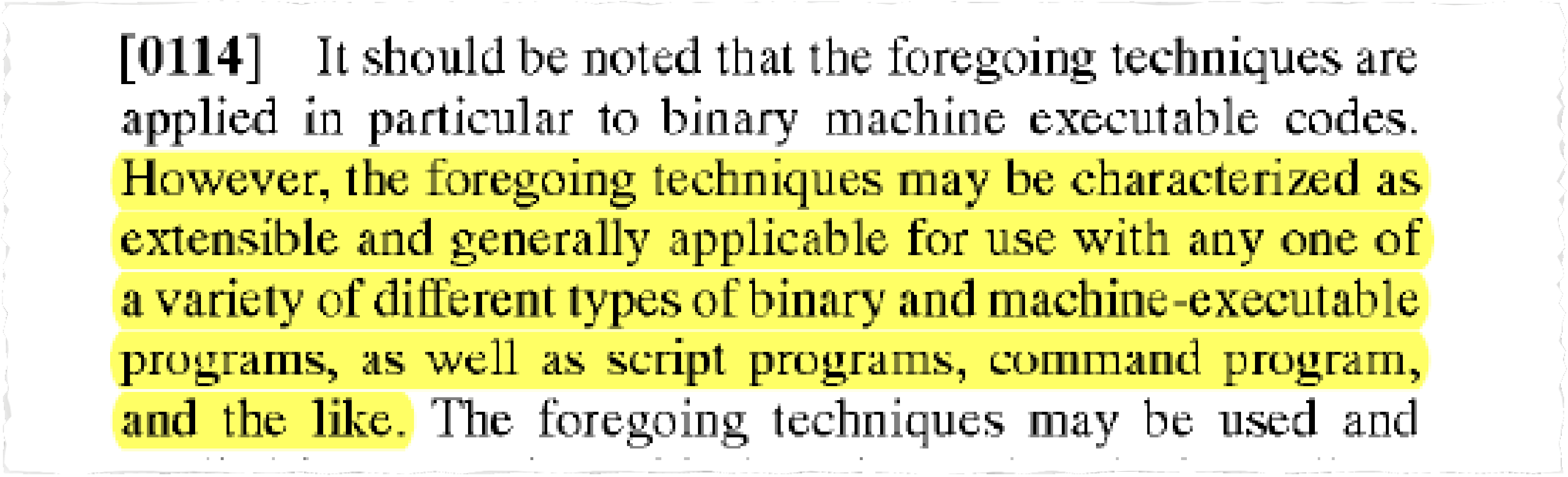


[0079] In the example described herein, Win32 API functions are instrumented for the purpose of being intercepted although an embodiment may monitor or intercept any one or more different functions or routines. Any one of a wide variety of different techniques may be used in connection with instrumenting the application 102 and any necessary libraries. In one embodiment, the Detours package as pro-

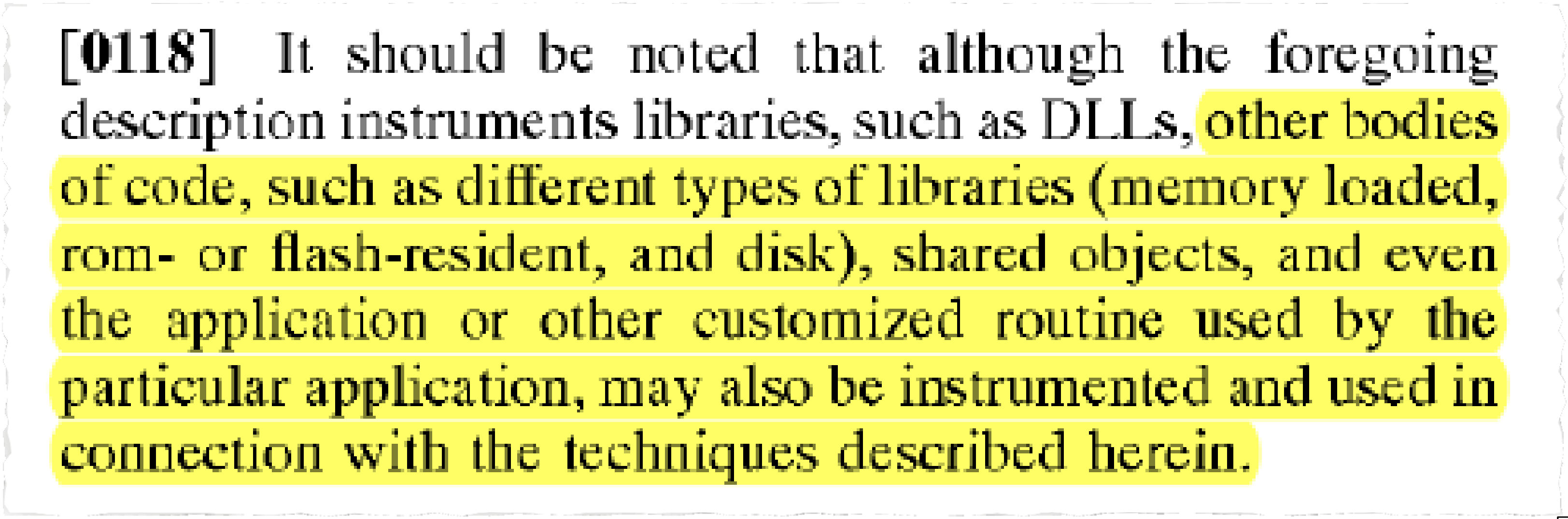
Ex. 1003.

The "content received over a network" limitations are obvious in view of Khazan's instrumented application

- Khazan's disclosure explicitly teaches instrumenting applications



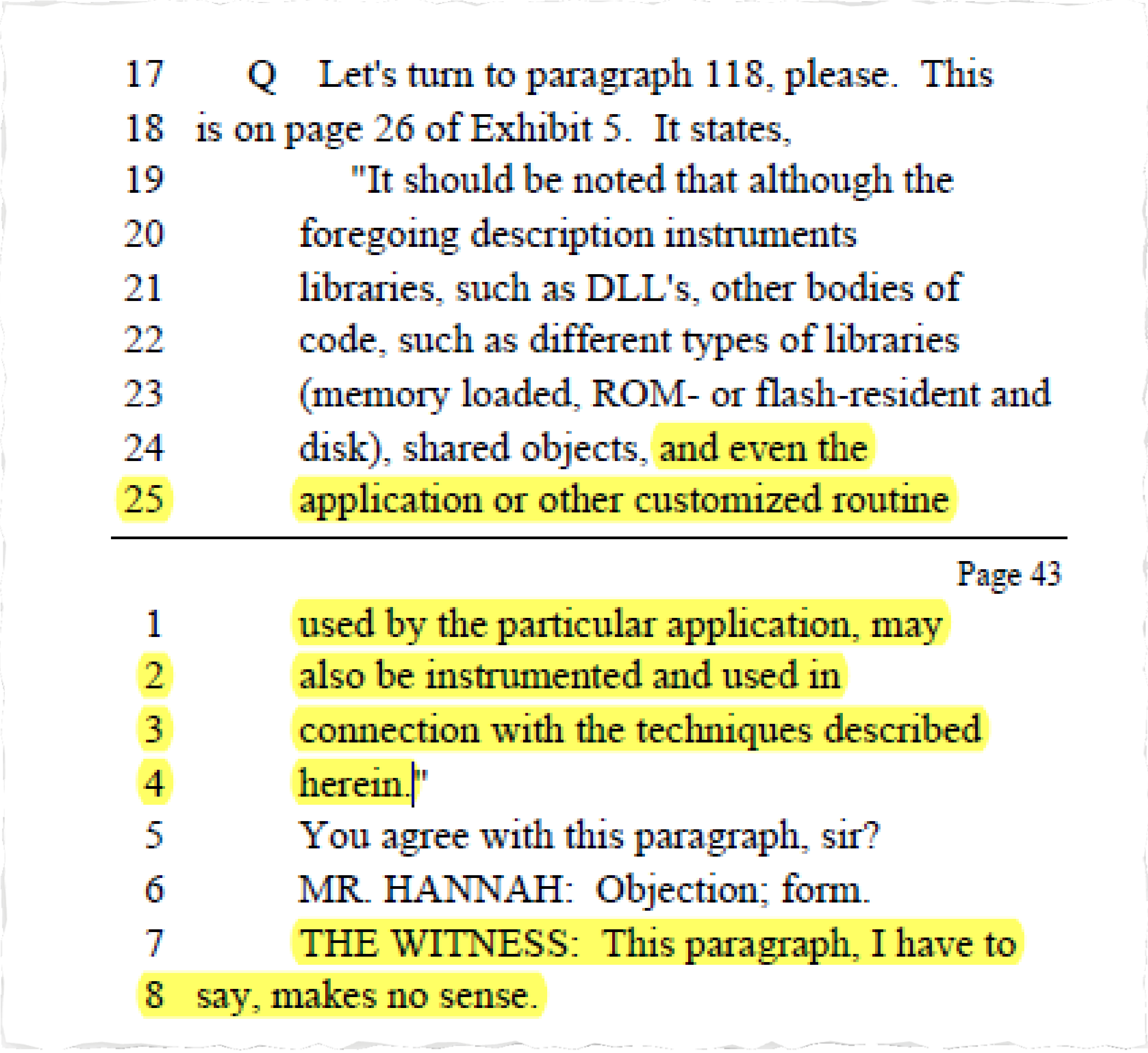
[0114] It should be noted that the foregoing techniques are applied in particular to binary machine executable codes. However, the foregoing techniques may be characterized as extensible and generally applicable for use with any one of a variety of different types of binary and machine-executable programs, as well as script programs, command program, and the like. The foregoing techniques may be used and



[0118] It should be noted that although the foregoing description instruments libraries, such as DLLs, other bodies of code, such as different types of libraries (memory loaded, rom- or flash-resident, and disk), shared objects, and even the application or other customized routine used by the particular application, may also be instrumented and used in connection with the techniques described herein.

Ex. 1003.

Finjan's expert Dr. Medvidovic ignores Khazan's broad disclosure



Q Let's turn to paragraph 118, please. This is on page 26 of Exhibit 5. It states,

"It should be noted that although the foregoing description instruments libraries, such as DLL's, other bodies of code, such as different types of libraries (memory loaded, ROM- or flash-resident and disk), shared objects, and even the application or other customized routine

Page 43

used by the particular application, may also be instrumented and used in connection with the techniques described herein."

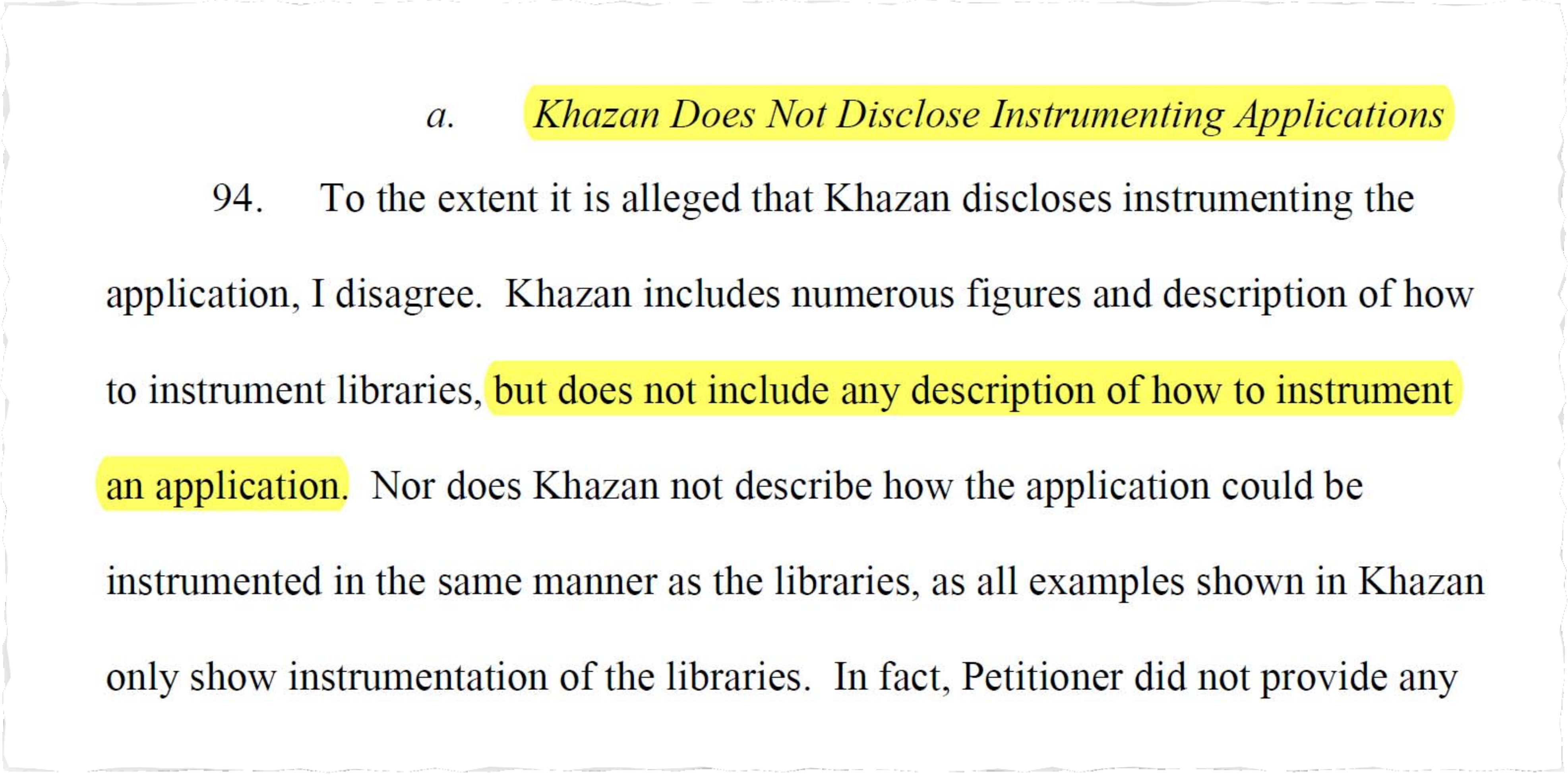
You agree with this paragraph, sir?

MR. HANNAH: Objection; form.

THE WITNESS: This paragraph, I have to say, makes no sense.

Ex. 1039, Medvidovic Tr. at 42:17-43:8.

Medvidovic complains that Khazan does not teach *how* to instrument applications, but enablement is not at issue in this IPR

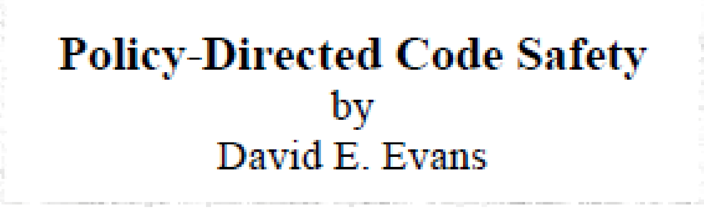


a. *Khazan Does Not Disclose Instrumenting Applications*

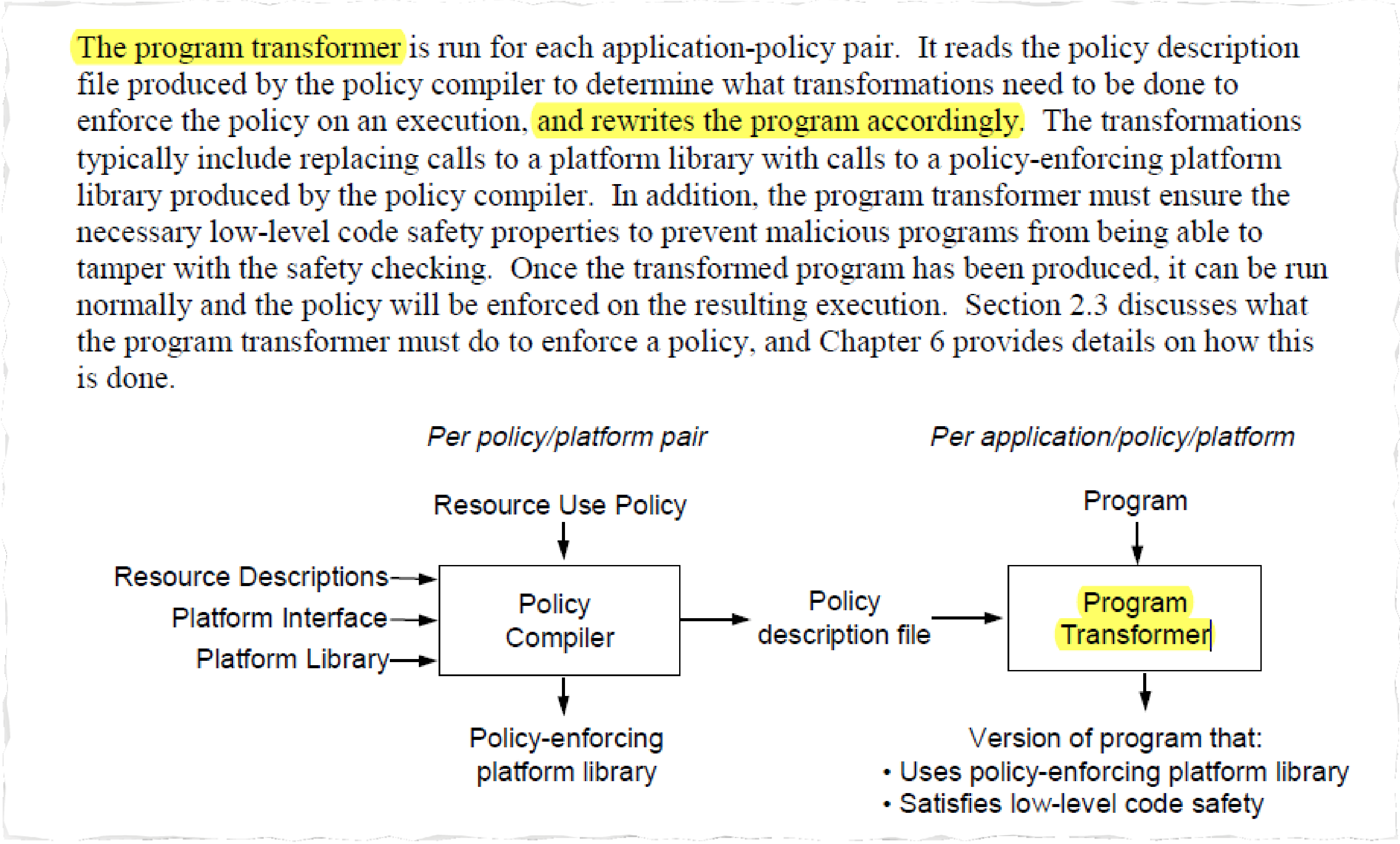
94. To the extent it is alleged that Khazan discloses instrumenting the application, I disagree. Khazan includes numerous figures and description of how to instrument libraries, but does not include any description of how to instrument an application. Nor does Khazan not describe how the application could be instrumented in the same manner as the libraries, as all examples shown in Khazan only show instrumentation of the libraries. In fact, Petitioner did not provide any

Ex. 2002, Medvidovic Decl.

Dr. Medvidovic also ignores Finjan's admissions in this IPR that instrumenting applications was well known in the prior art



Policy-Directed Code Safety
by
David E. Evans



The program transformer is run for each application-policy pair. It reads the policy description file produced by the policy compiler to determine what transformations need to be done to enforce the policy on an execution, and rewrites the program accordingly. The transformations typically include replacing calls to a platform library with calls to a policy-enforcing platform library produced by the policy compiler. In addition, the program transformer must ensure the necessary low-level code safety properties to prevent malicious programs from being able to tamper with the safety checking. Once the transformed program has been produced, it can be run normally and the policy will be enforced on the resulting execution. Section 2.3 discusses what the program transformer must do to enforce a policy, and Chapter 6 provides details on how this is done.

Per policy/platform pair

Per application/policy/platform

Resource Use Policy

Resource Descriptions

Platform Interface

Platform Library

Policy Compiler

Policy description file

Program

Program Transformer

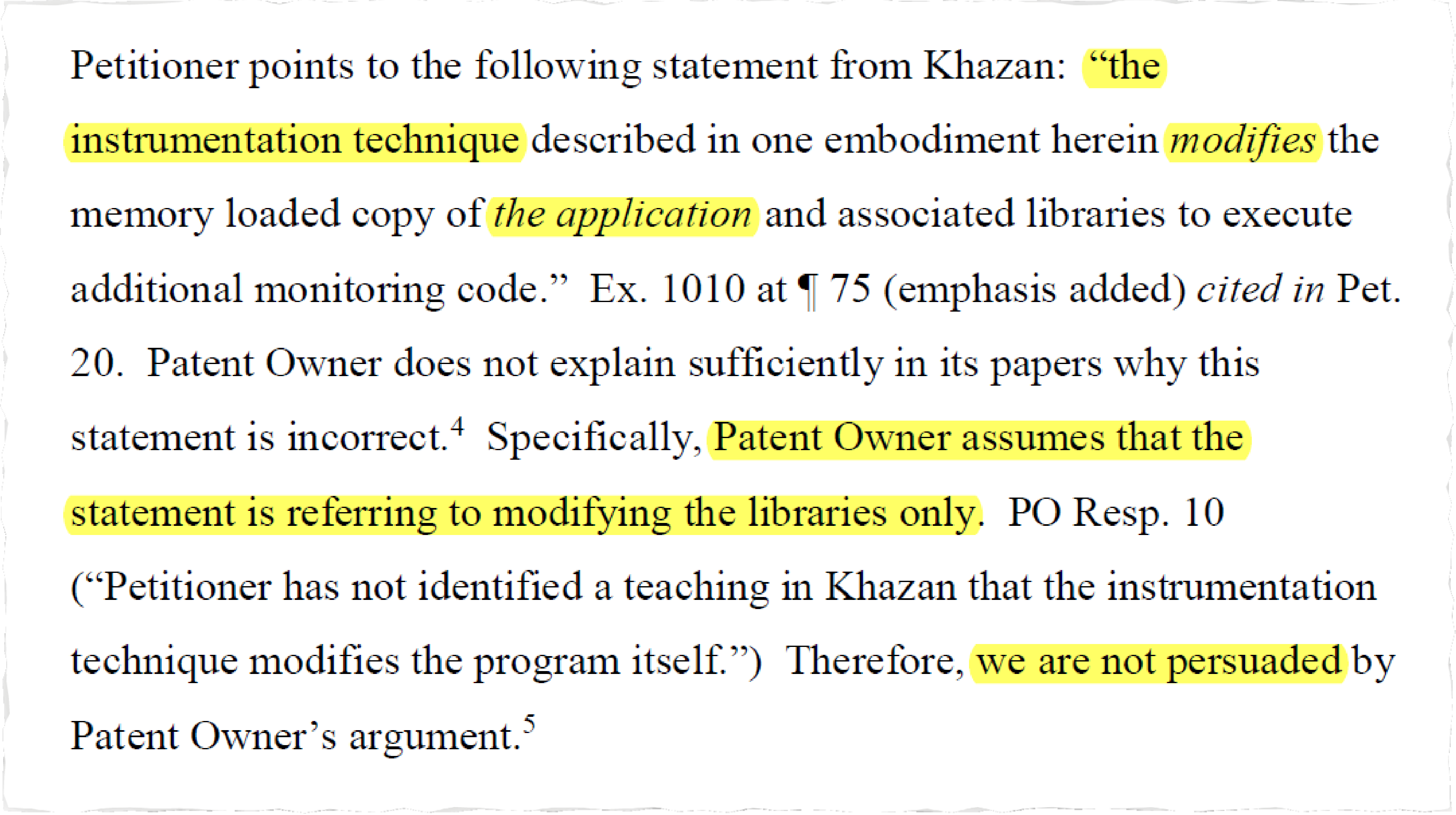
Policy-enforcing platform library

Version of program that:
- Uses policy-enforcing platform library
- Satisfies low-level code safety

Paper 2 at 6-7; Ex. 1002, Rubin Decl. ¶ 40, Ex. 1015 at 22-23.

Another Board panel already found what Dr. Medvidovic refuses to acknowledge: Khazan teaches instrumenting applications

- *Symantec Corp. v. The Trustees of Columbia Univ.*, IPR2015-00375, Paper 47 at 15-16 (P.T.A.B. June 30, 2016).

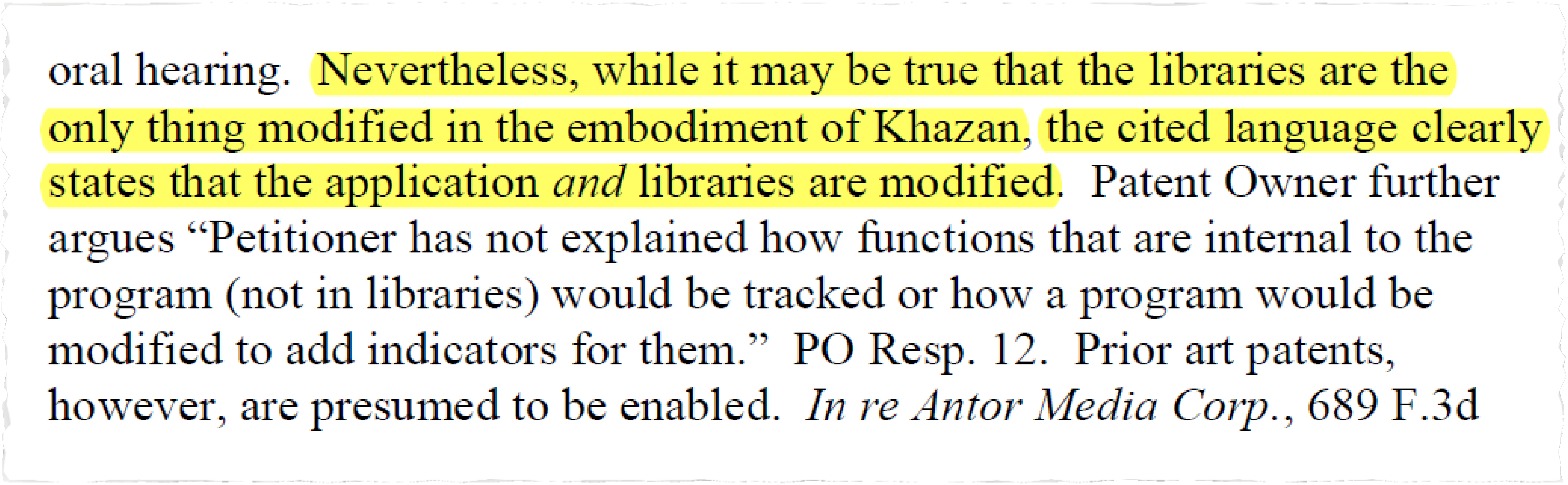


Petitioner points to the following statement from Khazan: "the instrumentation technique described in one embodiment herein *modifies* the memory loaded copy of *the application* and associated libraries to execute additional monitoring code." Ex. 1010 at ¶ 75 (emphasis added) *cited in* Pet. 20. Patent Owner does not explain sufficiently in its papers why this statement is incorrect.[4] Specifically, Patent Owner assumes that the statement is referring to modifying the libraries only. PO Resp. 10 ("Petitioner has not identified a teaching in Khazan that the instrumentation technique modifies the program itself.") Therefore, we are not persuaded by Patent Owner's argument.[5]

Ex. 1043 at 15-16.

Another Board panel already found what Dr. Medvidovic refuses to acknowledge: Khazan teaches instrumenting applications

- *Symantec Corp. v. The Trustees of Columbia Univ.*, IPR2015-00375, Paper 47 at 15-16 (P.T.A.B. June 30, 2016).

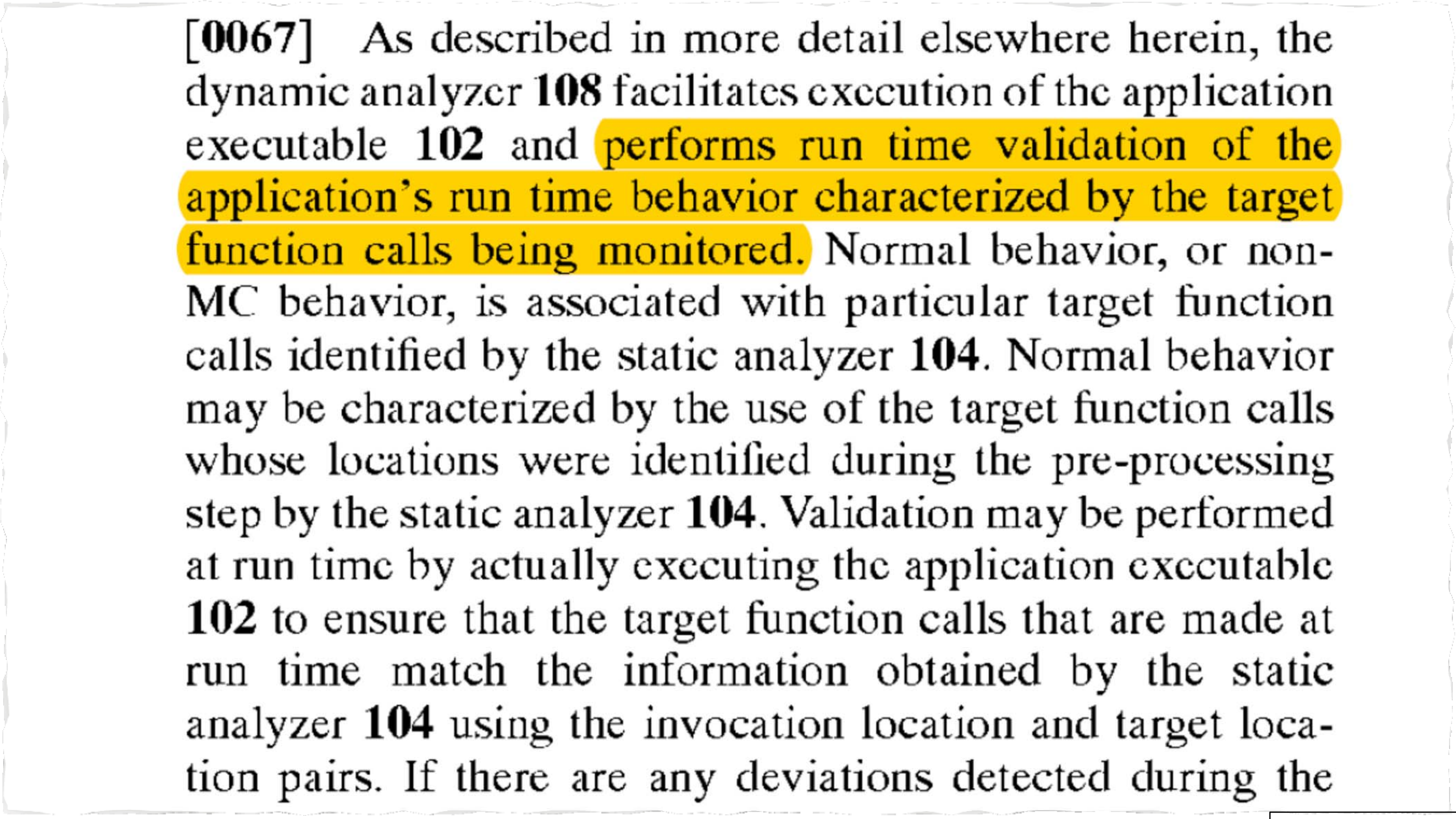


oral hearing. Nevertheless, while it may be true that the libraries are the only thing modified in the embodiment of Khazan, the cited language clearly states that the application *and* libraries are modified. Patent Owner further argues "Petitioner has not explained how functions that are internal to the program (not in libraries) would be tracked or how a program would be modified to add indicators for them." PO Resp. 12. Prior art patents, however, are presumed to be enabled. *In re Antor Media Corp.*, 689 F.3d

Ex. 1043 at 16 n.4.

Khazan's broad disclosures render the "content including a call to a first function" limitations obvious

- The purpose of Khazan's malicious code analysis is to verify calls from an application

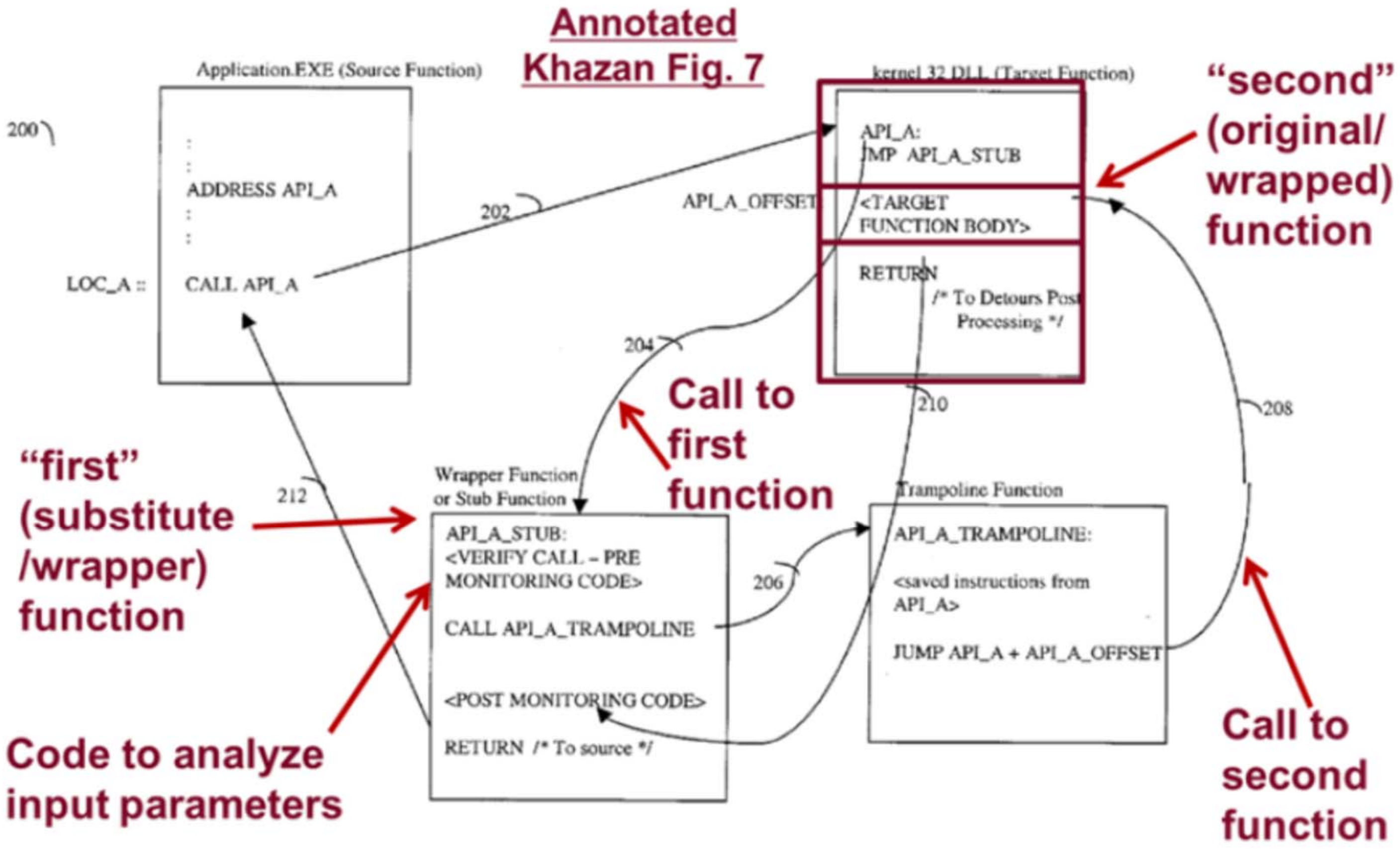


[0067] As described in more detail elsewhere herein, the dynamic analyzcr **108** facilitatcs cxccution of thc application executable **102** and performs run time validation of the application's run time behavior characterized by the target function calls being monitored. Normal behavior, or non-MC behavior, is associated with particular target function calls identified by the static analyzer **104**. Normal behavior may be characterized by the use of the target function calls whose locations were identified during the pre-processing step by the static analyzer **104**. Validation may be performed at run timc by actually cxccuting thc application cxccutable **102** to ensure that the target function calls that are made at run time match the information obtained by the static analyzer **104** using the invocation location and target location pairs. If there are any deviations detected during the

Ex. 1003 at 20 [0067].

The “content including a call to a first function” limitations are obvious in view of Khazan’s call 202 and jump 204

Petition Figure 1



Annotated Khazan Fig. 7

“second” (original/ wrapped) function

Call to first function

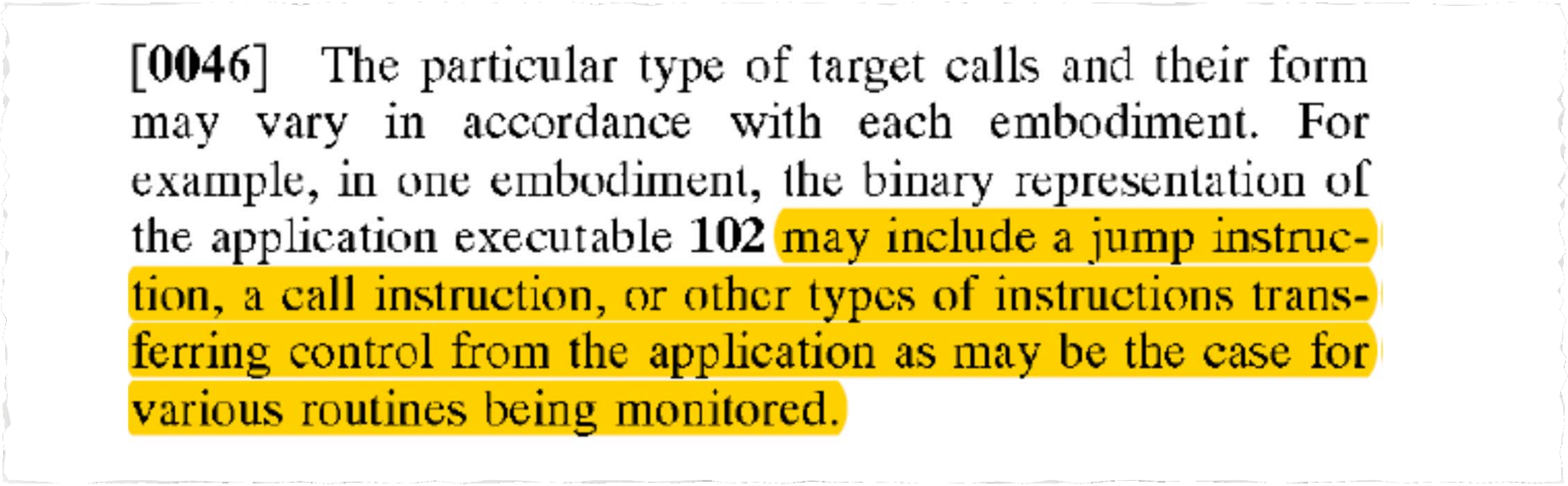
“first” (substitute /wrapper) function

Code to analyze input parameters

Call to second function

Petition at 24.

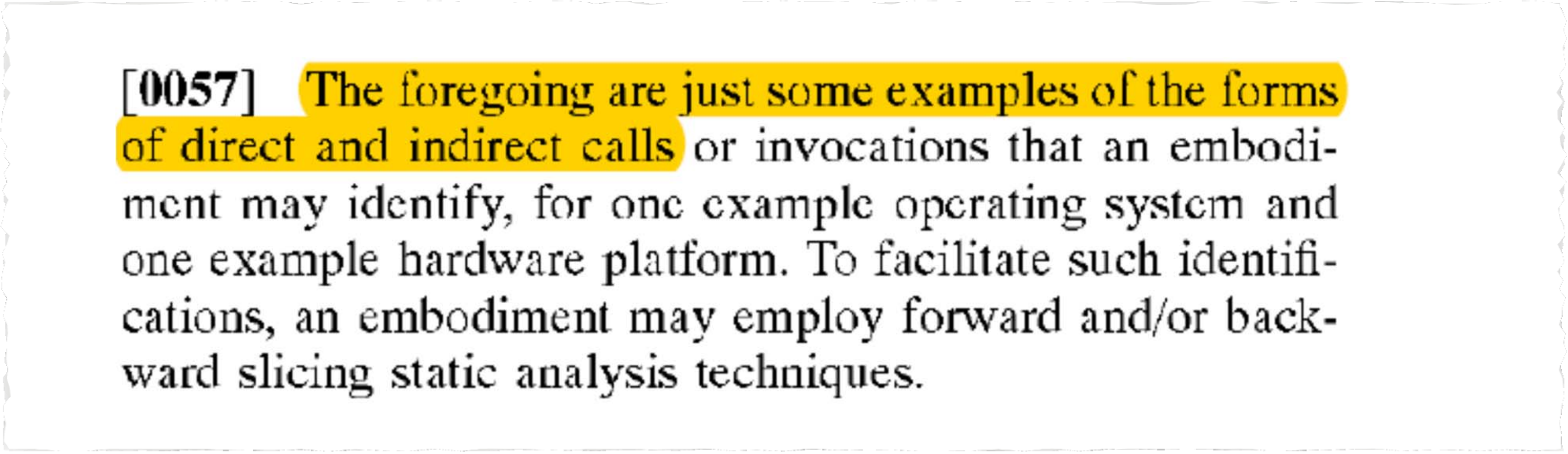
Kazan teaches that jumps, calls, and transfers are interchangeable



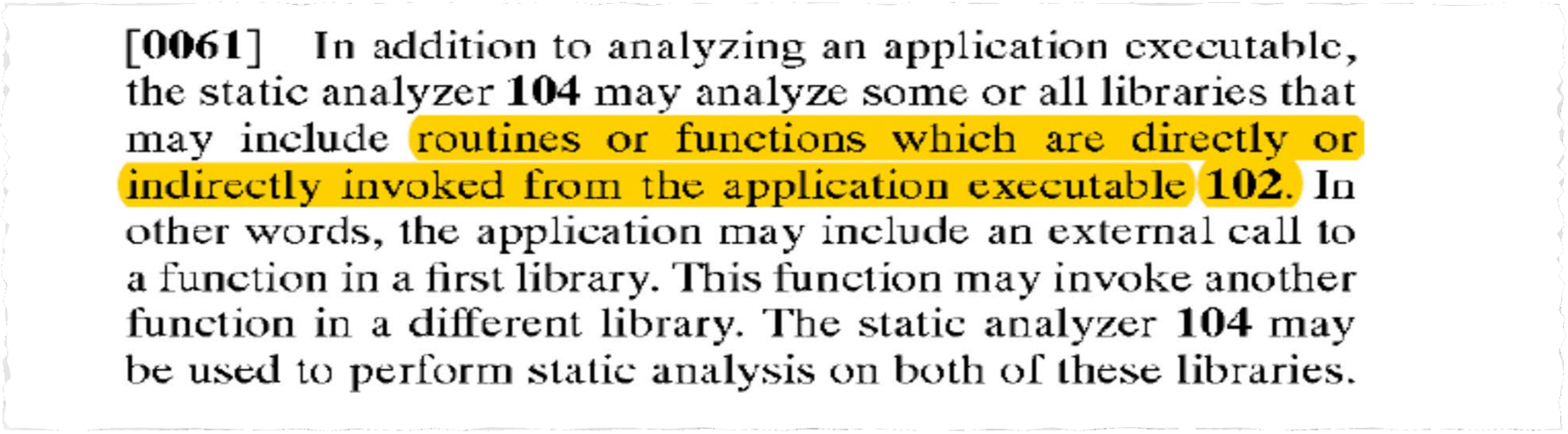
[0046] The particular type of target calls and their form may vary in accordance with each embodiment. For example, in one embodiment, the binary representation of the application executable **102** may include a jump instruction, a call instruction, or other types of instructions transferring control from the application as may be the case for various routines being monitored.

Ex. 1003 at 18 [0046].

Kazan teaches that calls can be direct or indirect



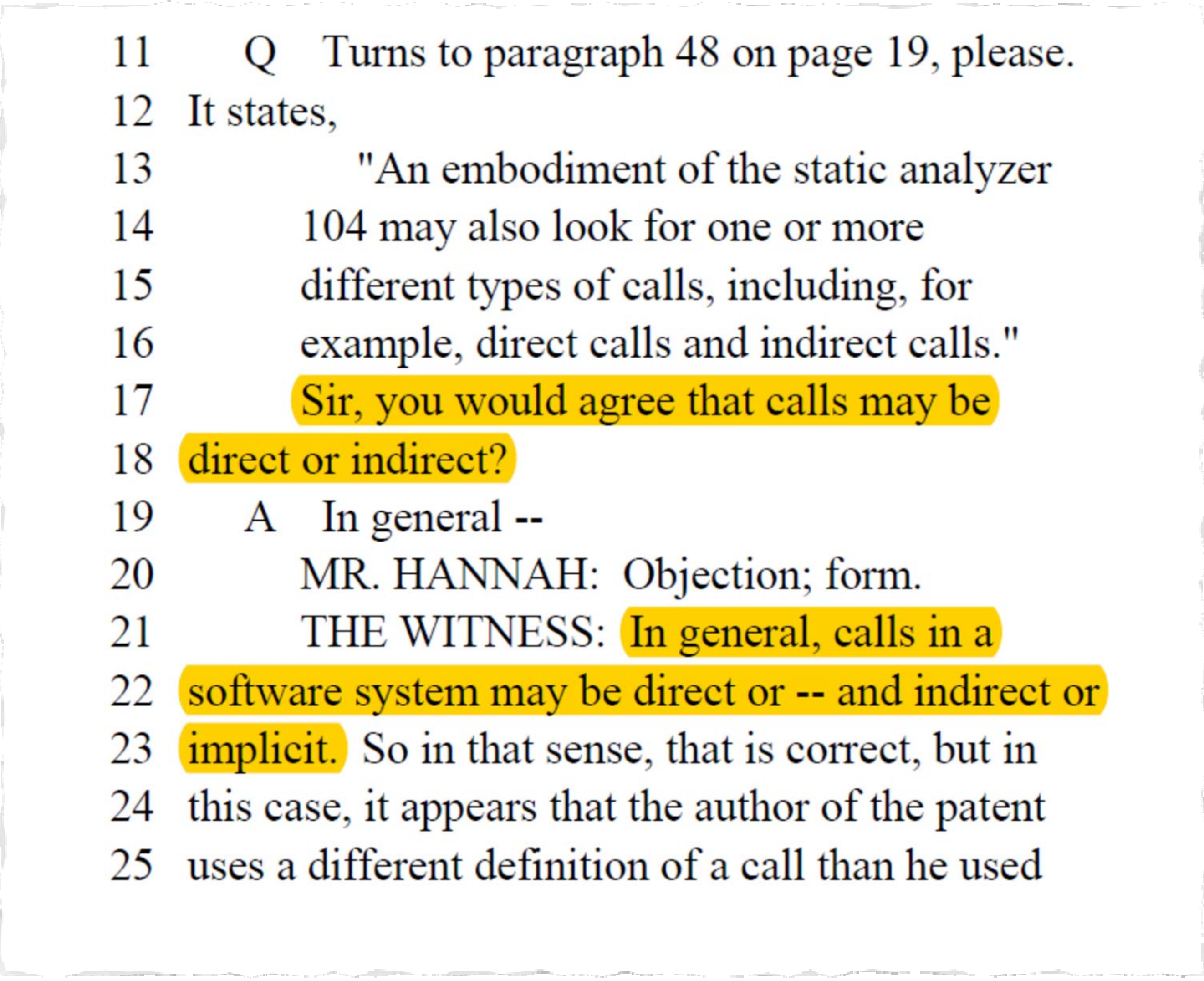
[0057] The foregoing are just some examples of the forms of direct and indirect calls or invocations that an embodiment may identify, for one example operating system and one example hardware platform. To facilitate such identifications, an embodiment may employ forward and/or backward slicing static analysis techniques.



[0061] In addition to analyzing an application executable, the static analyzer **104** may analyze some or all libraries that may include routines or functions which are directly or indirectly invoked from the application executable **102**. In other words, the application may include an external call to a function in a first library. This function may invoke another function in a different library. The static analyzer **104** may be used to perform static analysis on both of these libraries.

Ex. 1003 at 19 [0057], [0061].

Finjan's expert Dr. Medvidovic agrees that calls can be direct or indirect



Q Turns to paragraph 48 on page 19, please. It states,

"An embodiment of the static analyzer 104 may also look for one or more different types of calls, including, for example, direct calls and indirect calls."

Sir, you would agree that calls may be direct or indirect?

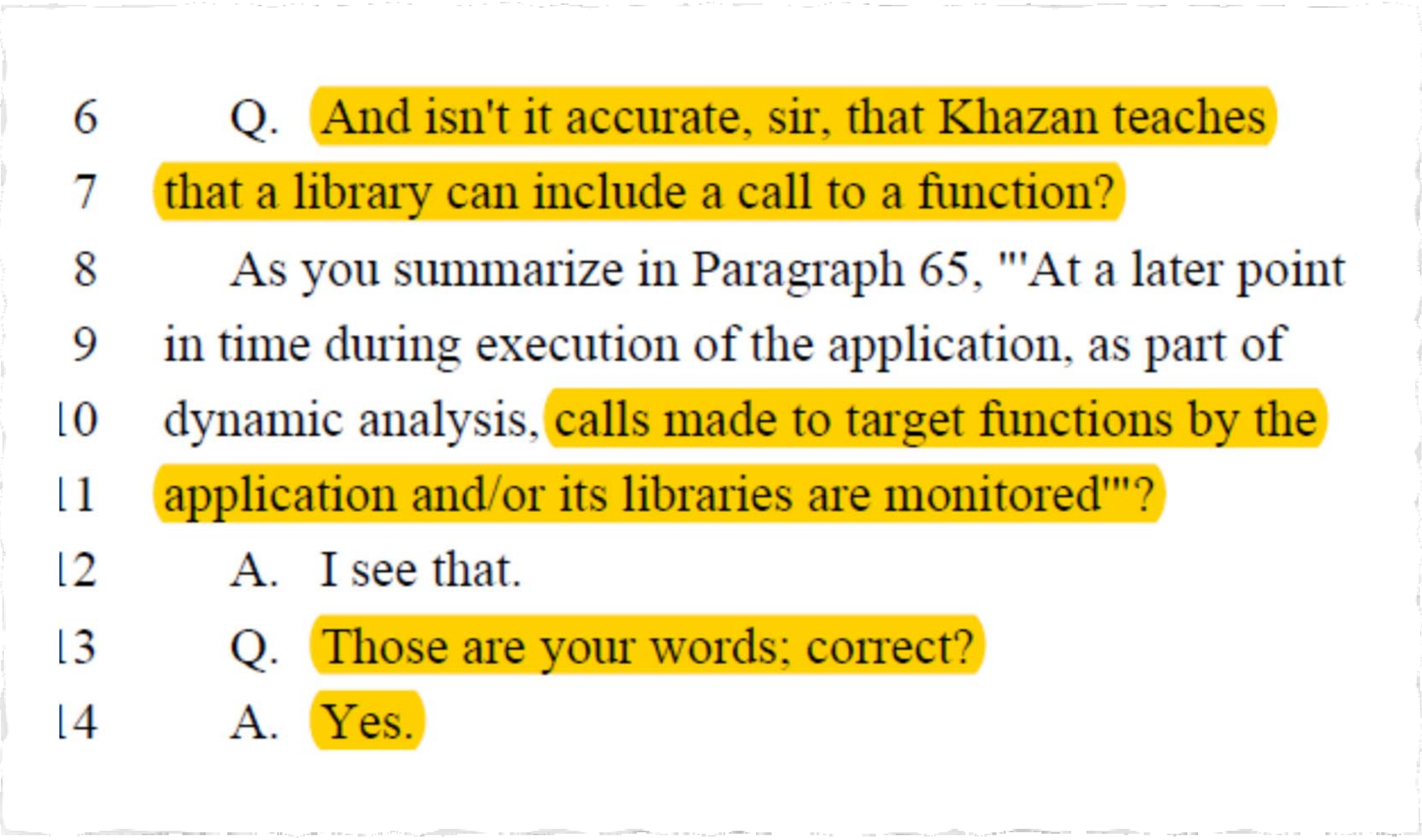
A In general --

MR. HANNAH: Objection; form.

THE WITNESS: In general, calls in a software system may be direct or -- and indirect or implicit. So in that sense, that is correct, but in this case, it appears that the author of the patent uses a different definition of a call than he used

Ex. 1038, Medvidovic Dep. Tr. at 32:11-23.

Finjan's expert Dr. Goodrich testified that Khazan teaches that a library can include a call to a function



Q. And isn't it accurate, sir, that Khazan teaches that a library can include a call to a function?

As you summarize in Paragraph 65, "'At a later point in time during execution of the application, as part of dynamic analysis, calls made to target functions by the application and/or its libraries are monitored'"?

A. I see that.

Q. Those are your words; correct?

A. Yes.

Ex. 1047, Goodrich Dep. Tr., at 40:6-14

Estoppel Arguments

35 U.S.C. § 315(e)(1)

There is no need for the Board to reach the estoppel issue

- No need for the Board to reach the estoppel issue
- Schedules allow for issuance of final written decisions at the same time
- Oral argument dates
  - IPR2015-01979: December 15, 2016
  - IPR2016-00151: January 24, 2017
- Final written decision deadlines
  - IPR2015-01979: March 21, 2017 (Paper 9)
  - IPR2016-00151: April 20, 2017 (Paper 11)
- No Board decision has applied estoppel to cases only weeks apart
- Normal practice is to decide cases at the same time
  - *See* IPR2014-00052 and IPR2014-00053 (decided same day)

The estoppel statute does not apply to these proceedings

- Estoppel statute will not apply to these proceedings
  - Oral argument in IPR2016-00151 will occur on January 24, 2017
  - Upon completion of oral argument, Board takes parties' arguments under submission and decides IPR without action by Petitioner
  - Prohibition against "maintaining" a proceeding refers only to actions parties must take
  - *CBS Interactive Inc. v. Helferich Patent Licensing, LLC*, IPR2013-00033, Paper 118 at 2-3 (Oct. 23, 2013) (holding that by time proceeding reaches final oral hearing, trial is complete)

Even if Board concludes PAN may be estopped, the Board should enter final written decisions in both IPRs

- The Board is not subject to estoppel
  - *Progressive Casualty Insurance Co. v. Liberty Mutual Insurance Co.*, No. 2014-1466, 2015 WL 5004949, at *2 (Fed. Cir. Aug. 24, 2015) ("by its terms [§ 325(e)(1)] does not prohibit the Board from reaching decisions. It limits only certain (requesting or maintaining) actions by a Petitioner.")

- The public interest favors decision of fully argued, submitted cases
  - *Apple Inc. v. Smartflash LLC*, CBM2015-00015, Paper 49 (Nov. 4, 2015) (finding petitioner estopped, but choosing to issue FWD anyway despite six month gap between final written decisions because record was already fully developed)

The Board should enter final written decisions in both IPRs because Symantec will remain a petitioner

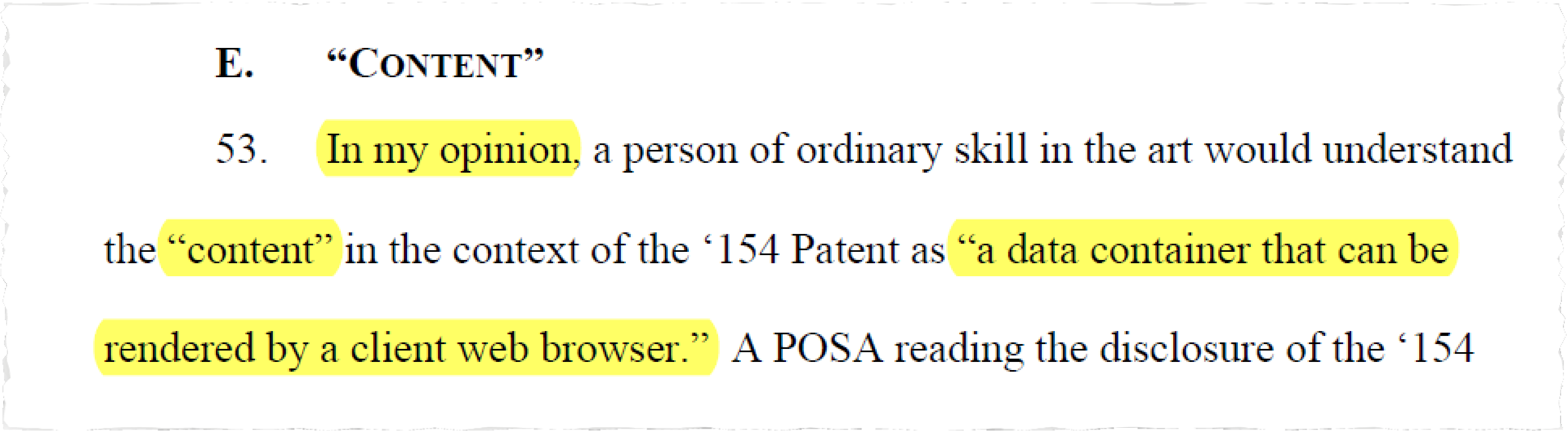
- Even if estoppel were found to apply to PAN, IPR2016-00151 will live on
- IPR cannot be terminated unless “no Petitioner remains in the *inter partes* review.” 35 U.S.C. § 317(a)
- Symantec need not take any action to remain a joinder party
- Symantec could not have raised the grounds instituted in IPR2016-00151 when it moved to join this case
    - *Harmonix Music Systems, Inc. v. Princeton Digital Image Corp.*, IPR2015-00271, Paper 15 at 4 (“the Petition includes a new challenge to both a claim not instituted in the ’635 IPR, claim 14, and claims instituted in the ’635 IPR, claims 5–7 and 16–18, based on a new combination of references considered in the ’635 IPR”)

Finjan's arguments for estoppel are unpersuasive

- Finjan fails to address the timing, public interest, or joinder issues central to estoppel considerations in this case
- Finjan admits "petitioner estoppel is not yet ripe" (Paper 42 at 4)
  - But seeks premature application of estoppel by stay or termination
  - Estoppel only accrues *after* written decision (35 U.S.C. **§** 315(e)(1))
- Finjan's § 315(d) argument is procedurally improper and too late
  - Not raised in Preliminary Response, Response, or by motion

Finjan’s Motion to Exclude Should Be Denied

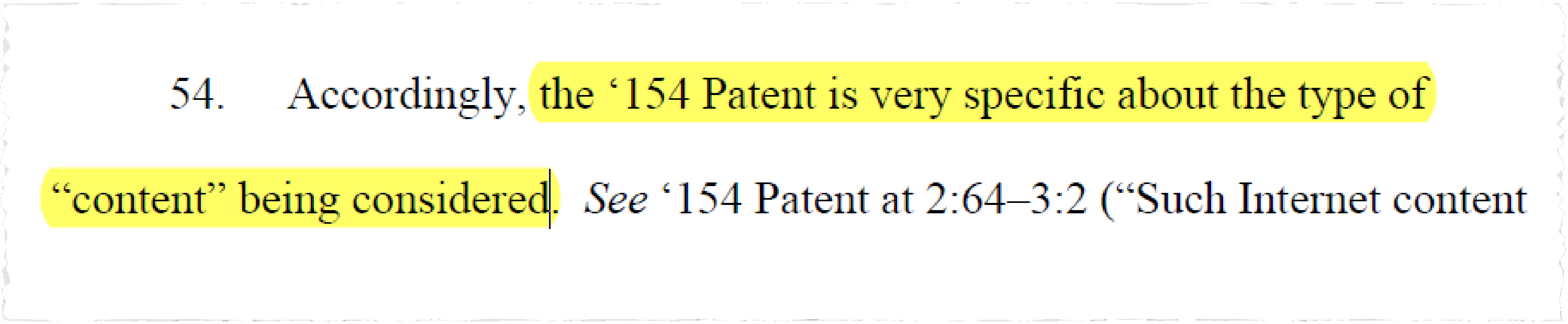
Finjan’s new construction of “content” in its Patent Owner Response relied on Dr. Medvidovic



E. “CONTENT”

53. In my opinion, a person of ordinary skill in the art would understand the “content” in the context of the ‘154 Patent as “a data container that can be rendered by a client web browser.” A POSA reading the disclosure of the ‘154

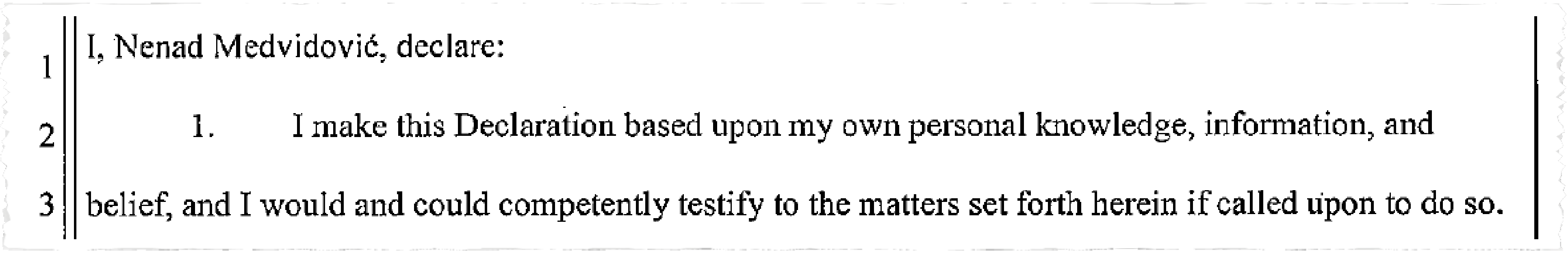
Ex. 2002, Medvidovic Decl. ¶ 53.



54. Accordingly, the ‘154 Patent is very specific about the type of “content” being considered. *See* ‘154 Patent at 2:64–3:2 (“Such Internet content

Ex. 2002, Medvidovic Decl. ¶ 54.

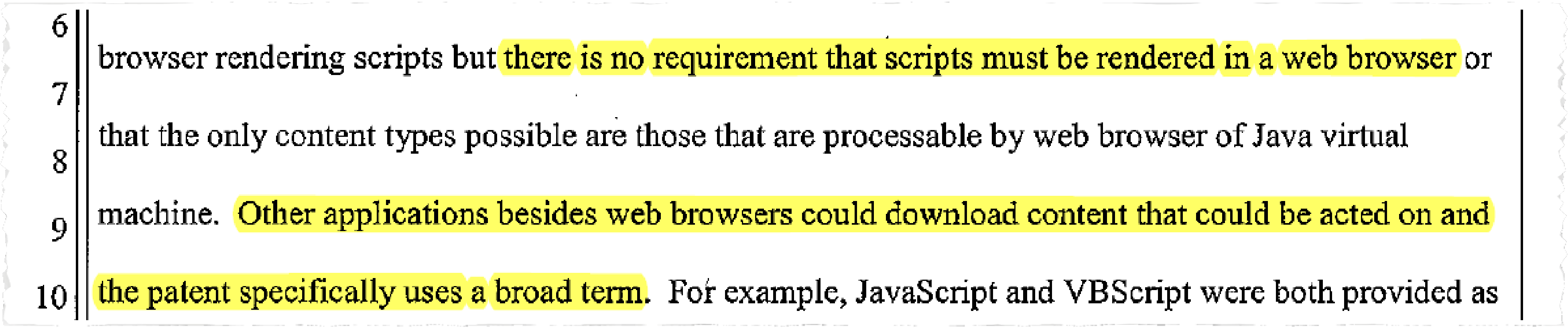
Dr. Medvidovic's Declarations (Exs. 1039-1041) in district court contradict Finjan's new construction of "content"



I, Nenad Medvidović, declare:

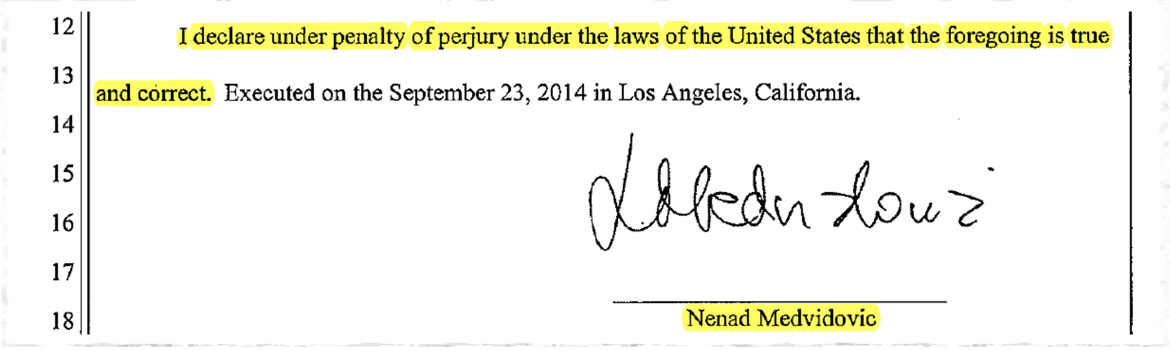
1. I make this Declaration based upon my own personal knowledge, information, and belief, and I would and could competently testify to the matters set forth herein if called upon to do so.

Ex. 1039, Medvidovic Decl. at 1.



browser rendering scripts but there is no requirement that scripts must be rendered in a web browser or that the only content types possible are those that are processable by web browser of Java virtual machine. Other applications besides web browsers could download content that could be acted on and the patent specifically uses a broad term. For example, JavaScript and VBScript were both provided as

Ex. 1039, Medvidovic Decl. at 16.

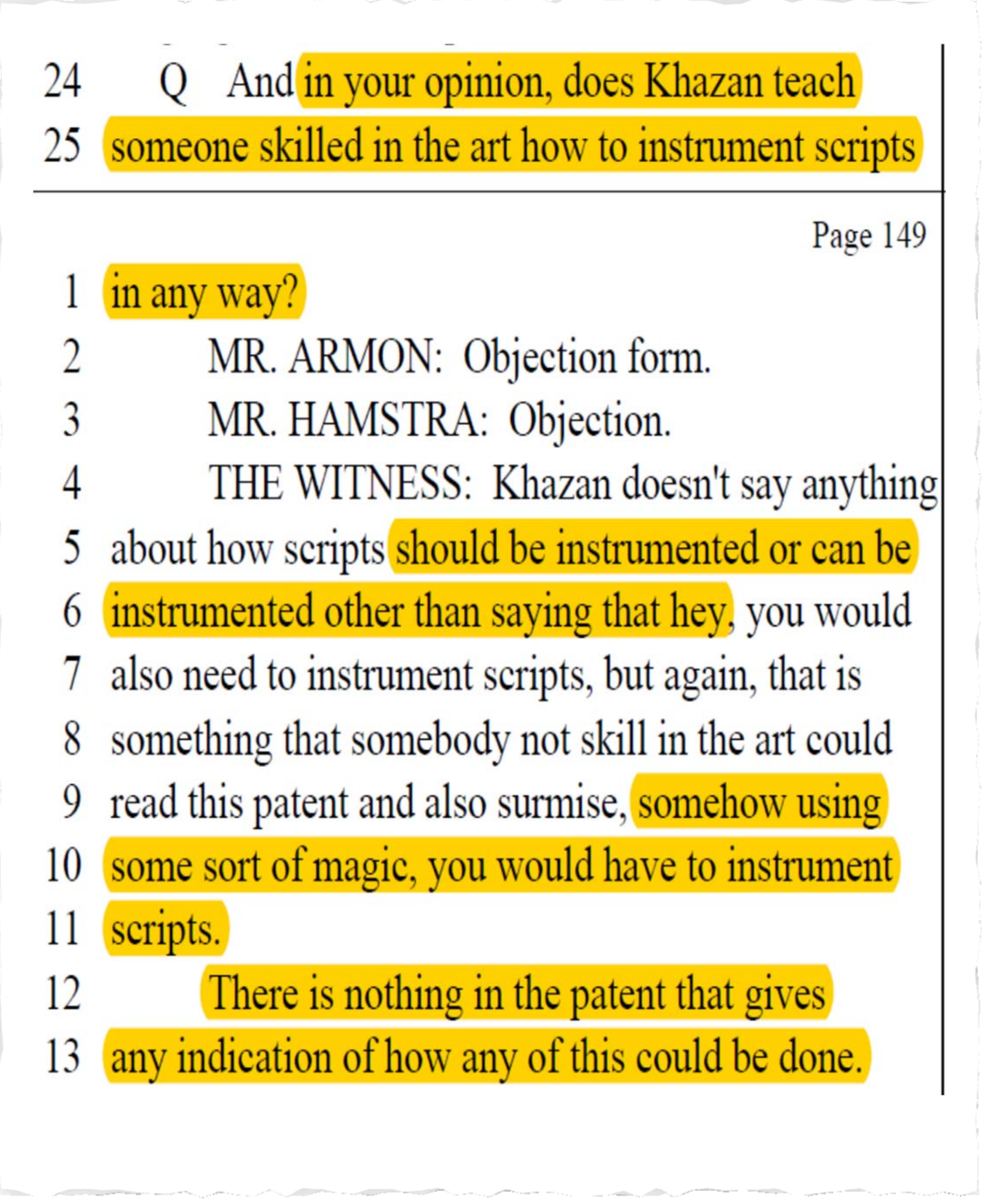


I declare under penalty of perjury under the laws of the United States that the foregoing is true and correct. Executed on the September 23, 2014 in Los Angeles, California.

Nenad Medvidovic

Ex. 1039, Medvidovic Decl. at 23.

Dr. Medvidovic testified that Khazan did not enable instrumenting scripts or applications



Q And in your opinion, does Khazan teach someone skilled in the art how to instrument scripts

Page 149

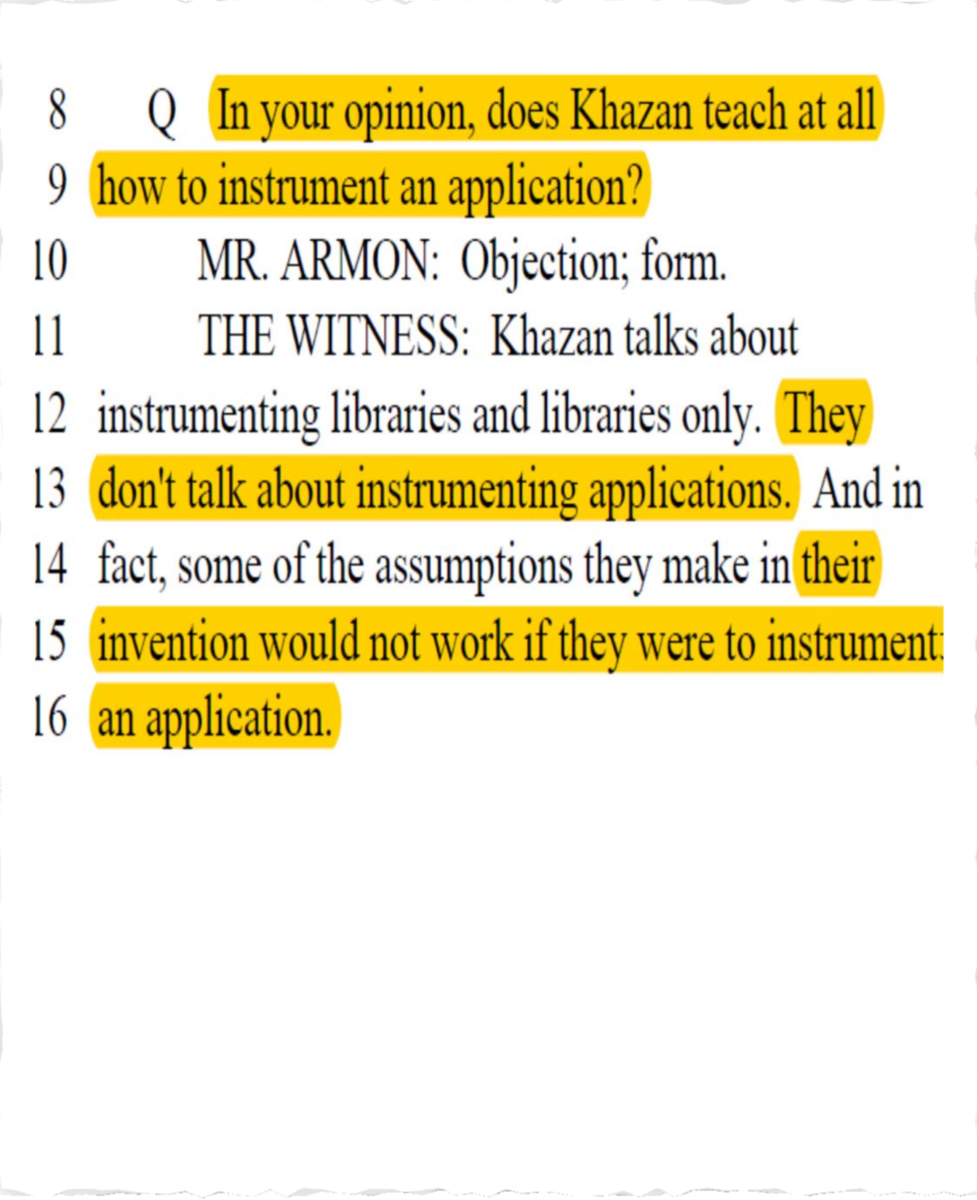
in any way?

MR. ARMON: Objection form.

MR. HAMSTRA: Objection.

THE WITNESS: Khazan doesn't say anything about how scripts should be instrumented or can be instrumented other than saying that hey, you would also need to instrument scripts, but again, that is something that somebody not skill in the art could read this patent and also surmise, somehow using some sort of magic, you would have to instrument scripts.

There is nothing in the patent that gives any indication of how any of this could be done.



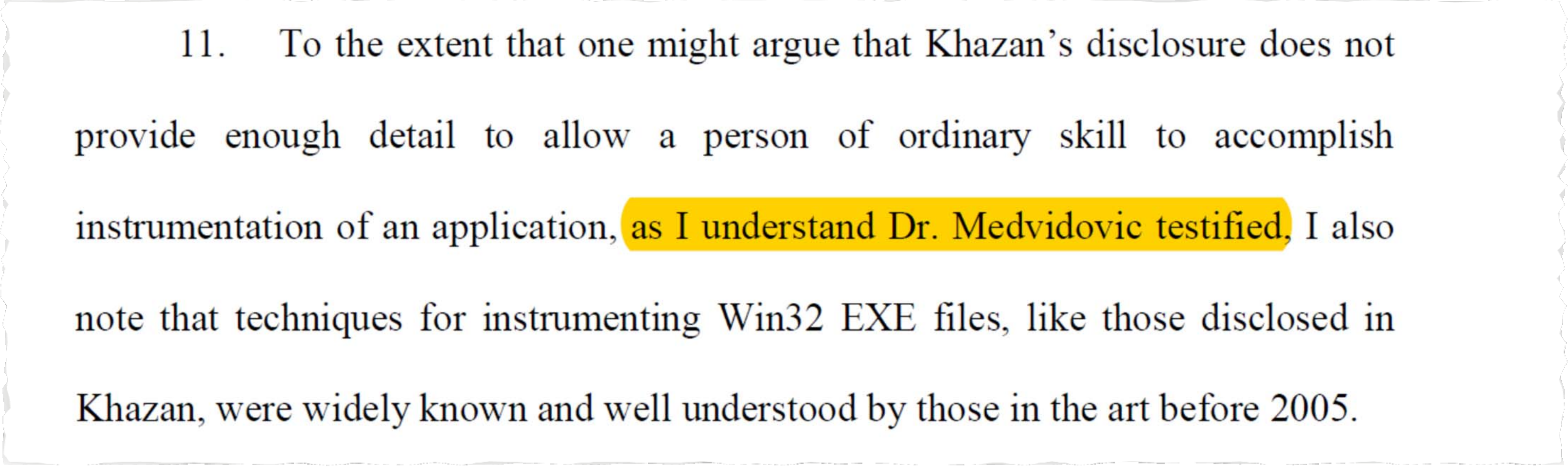
Q In your opinion, does Khazan teach at all how to instrument an application?

MR. ARMON: Objection; form.

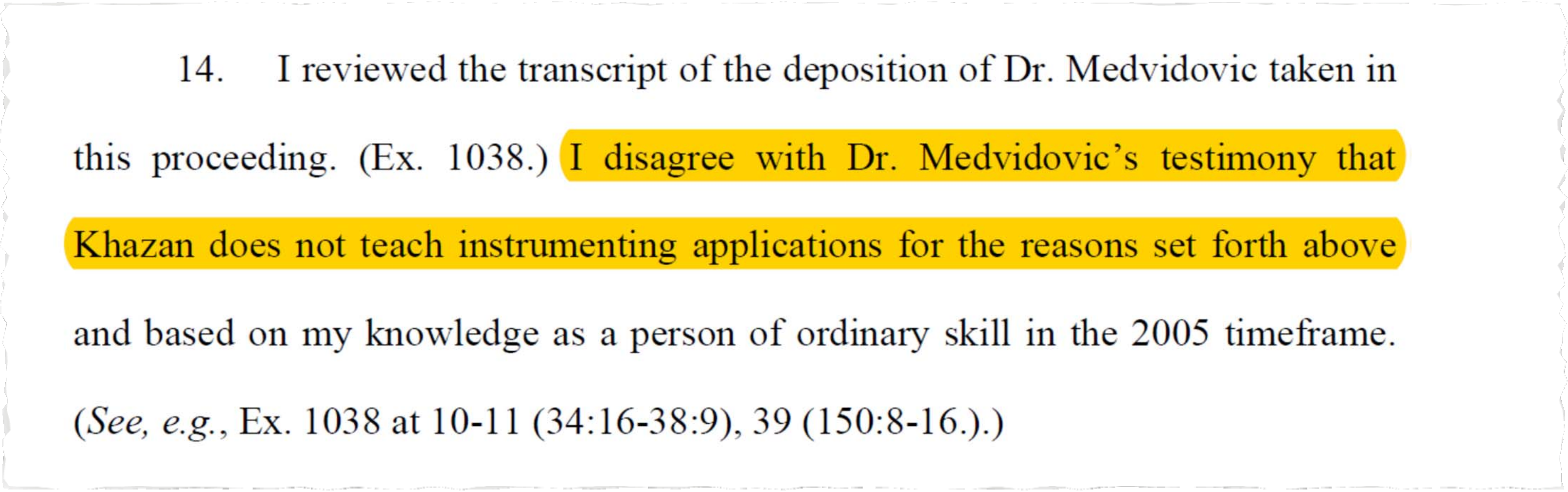
THE WITNESS: Khazan talks about instrumenting libraries and libraries only. They don't talk about instrumenting applications. And in fact, some of the assumptions they make in their invention would not work if they were to instrument an application.

Ex. 1038, Medvidovic Dep. Tr. at 148:24-149:13, 151:8-16.

Dr. Rubin's Supplemental Declaration and the Nebenzahl Article respond to Dr. Medvidovic's testimony



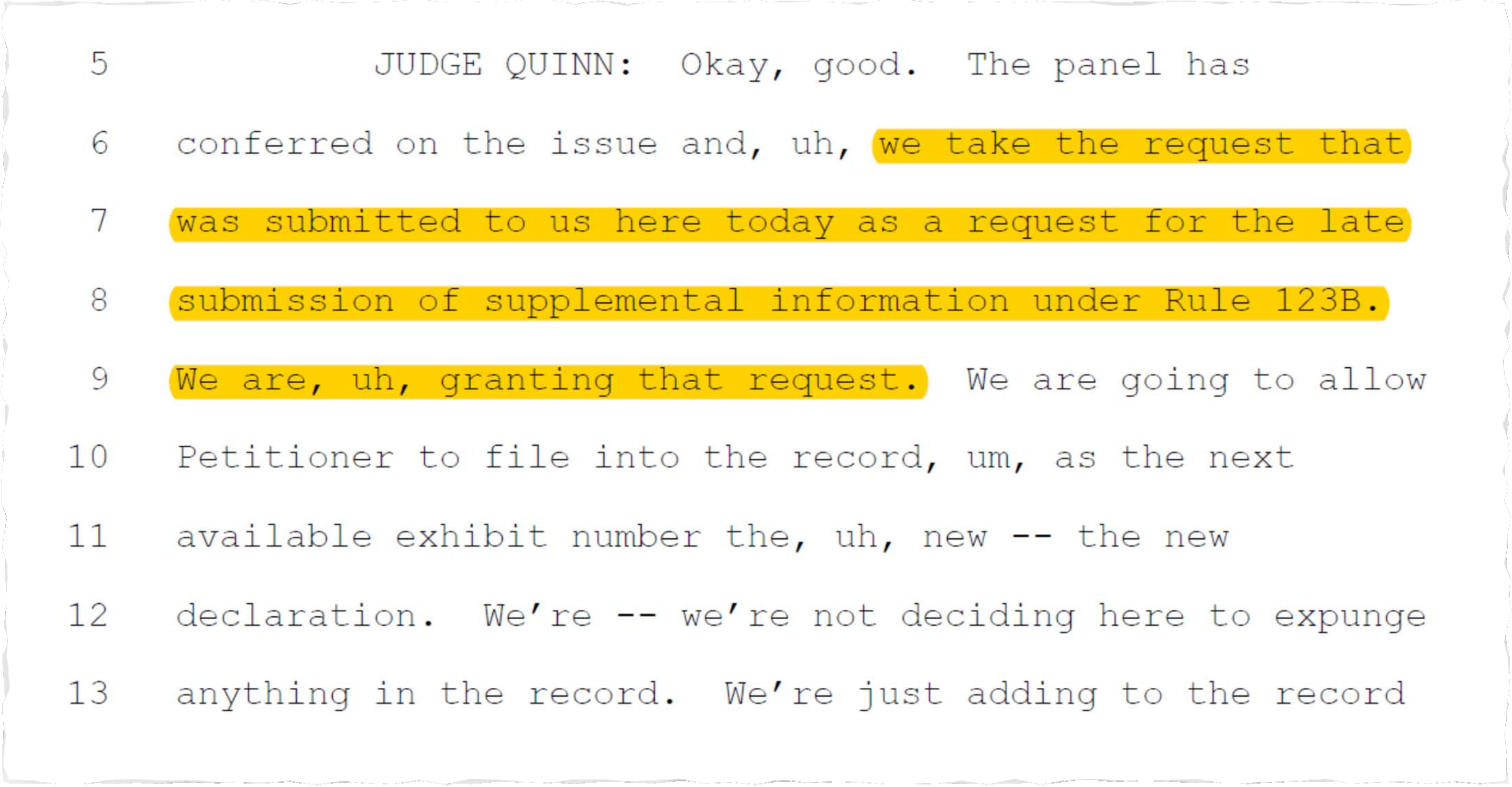
11. To the extent that one might argue that Khazan's disclosure does not provide enough detail to allow a person of ordinary skill to accomplish instrumentation of an application, as I understand Dr. Medvidovic testified, I also note that techniques for instrumenting Win32 EXE files, like those disclosed in Khazan, were widely known and well understood by those in the art before 2005.



14. I reviewed the transcript of the deposition of Dr. Medvidovic taken in this proceeding. (Ex. 1038.) I disagree with Dr. Medvidovic's testimony that Khazan does not teach instrumenting applications for the reasons set forth above and based on my knowledge as a person of ordinary skill in the 2005 timeframe. (*See, e.g.*, Ex. 1038 at 10-11 (34:16-38:9), 39 (150:8-16.).)

Ex. 1045, Supplemental Rubin Decl. at ¶¶ 11, 14 citing Ex. 1044, Nebenzahl.

The Board granted PAN's request to submit the DeSart Decl. (Ex. 1036) as supplemental information and it should not be excluded

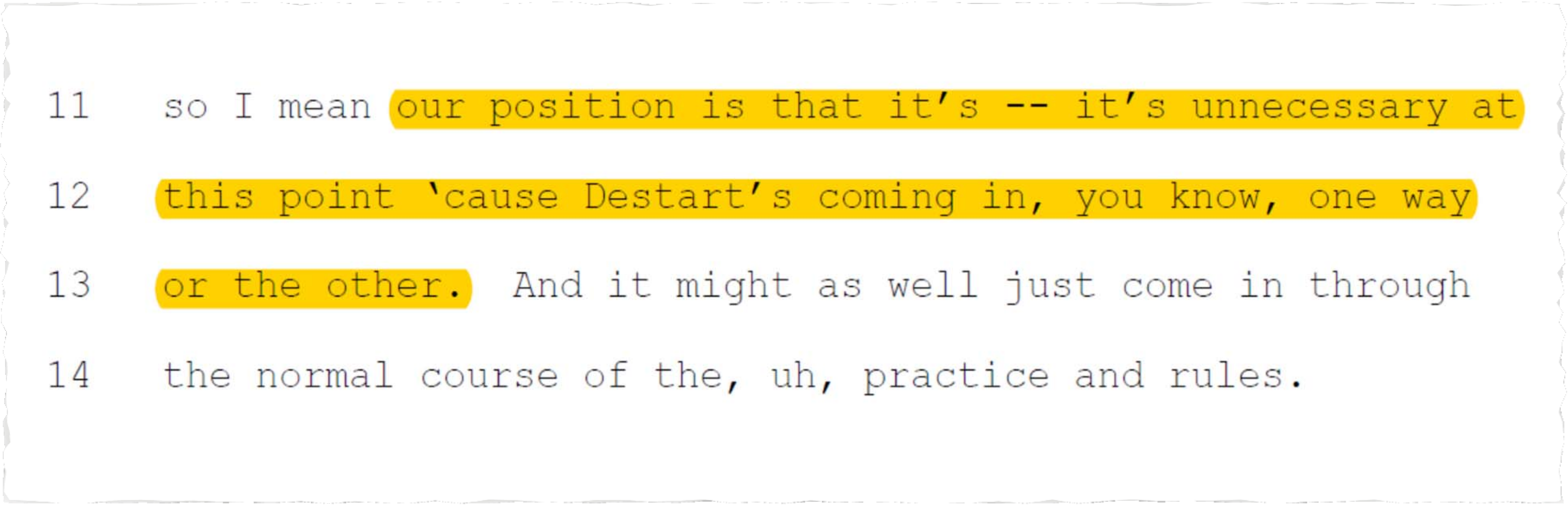


JUDGE QUINN: Okay, good. The panel has conferred on the issue and, uh, we take the request that was submitted to us here today as a request for the late submission of supplemental information under Rule 123B. We are, uh, granting that request. We are going to allow Petitioner to file into the record, um, as the next available exhibit number the, uh, new -- the new declaration. We're -- we're not deciding here to expunge anything in the record. We're just adding to the record

Ex. 1037, June 14, 2016 Board Call Tr. at 24:5-13.

Finjan waived its argument to exclude the DeSart Declaration as improper new evidence

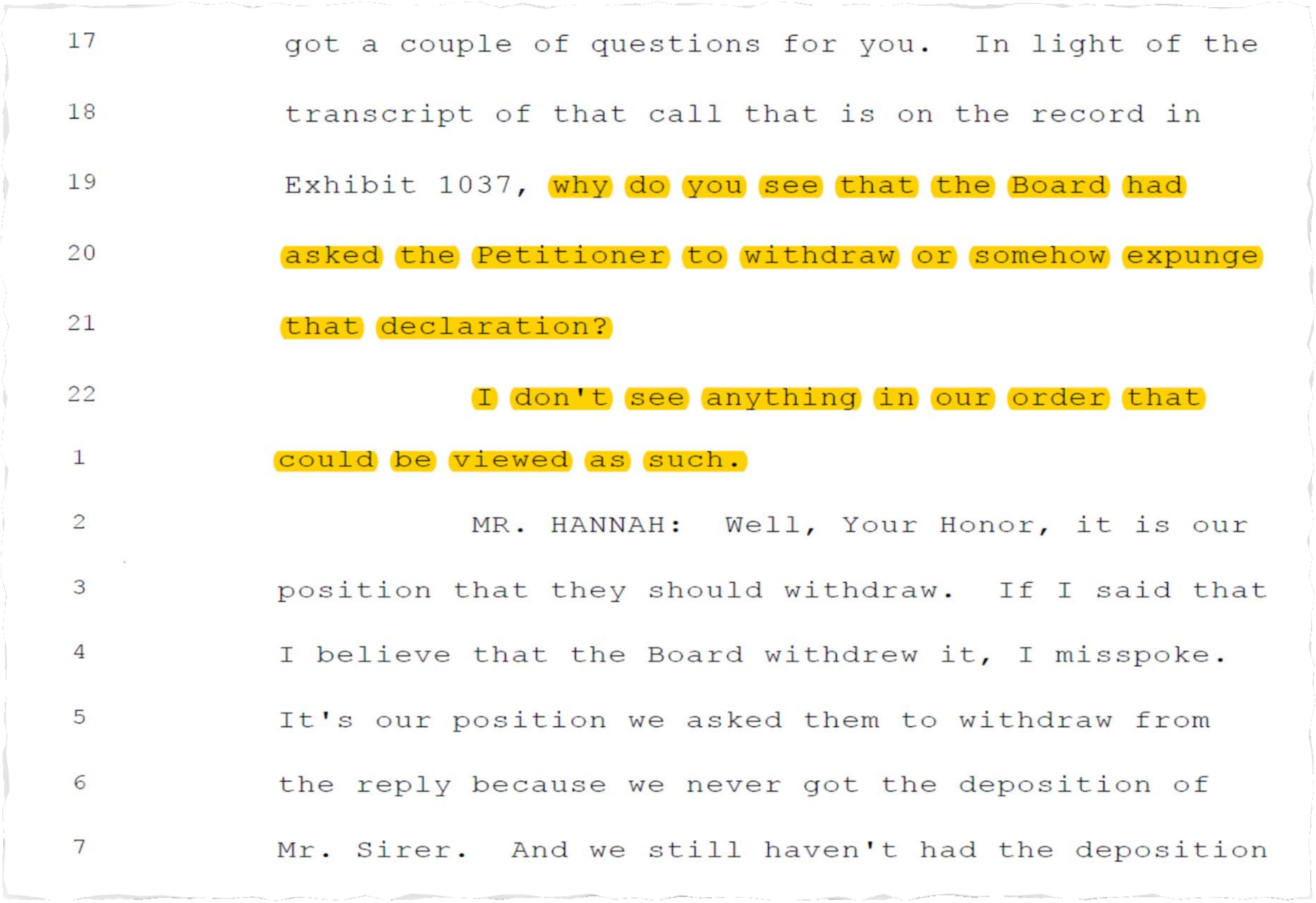
- Finjan did not object to the Board’s order granting PAN’s motion to admit the DeSart Declaration as supplemental information, nor did Finjan ask the Board to reconsider its decision.



so I mean our position is that it’s -- it’s unnecessary at this point ‘cause Destart’s coming in, you know, one way or the other. And it might as well just come in through the normal course of the, uh, practice and rules.

Ex. 1037, June 14, 2016 Board Call Tr. at 20:11-14.

Finjan abandoned its pursuit of Dr. Sirer's deposition even though the Board decided not to exclude it



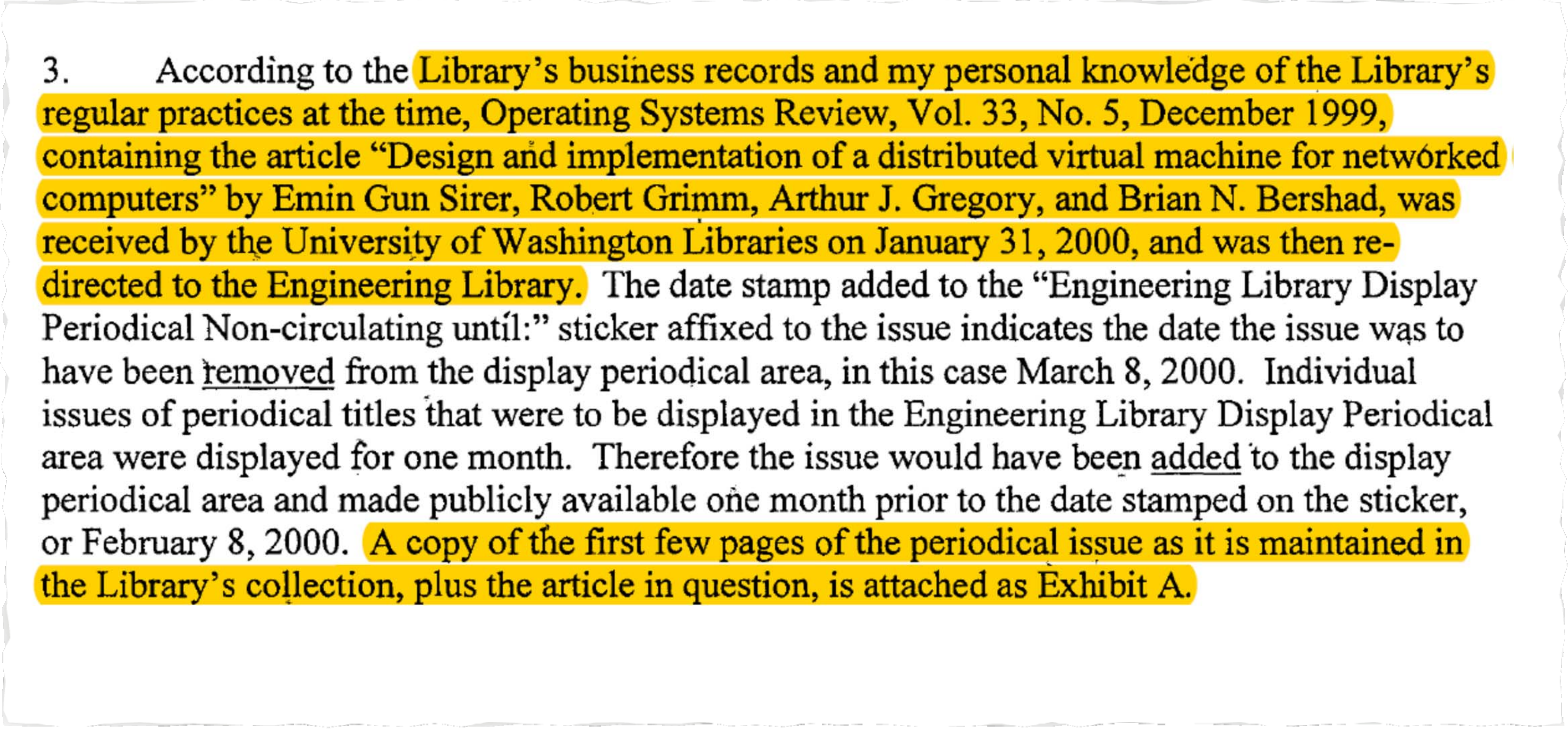
got a couple of questions for you. In light of the transcript of that call that is on the record in Exhibit 1037, why do you see that the Board had asked the Petitioner to withdraw or somehow expunge that declaration?

I don't see anything in our order that could be viewed as such.

MR. HANNAH: Well, Your Honor, it is our position that they should withdraw. If I said that I believe that the Board withdrew it, I misspoke. It's our position we asked them to withdraw from the reply because we never got the deposition of Mr. Sirer. And we still haven't had the deposition

Ex. 2037, November 16, 2016 Board Call Tr. at 7:17-8:7.

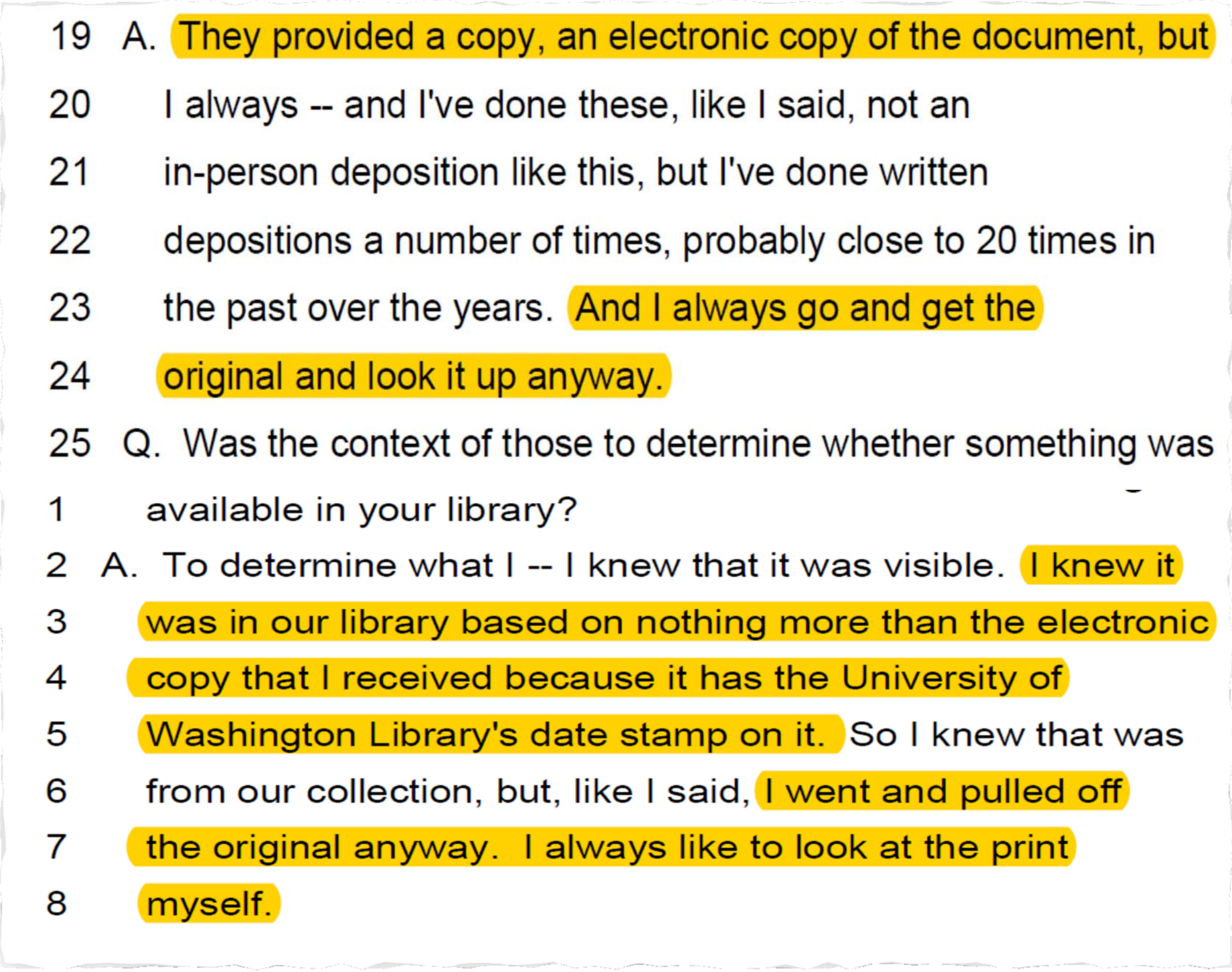
Mr. DeSart authenticated the Sirer Article (Ex. 1004)



3. According to the Library's business records and my personal knowledge of the Library's regular practices at the time, Operating Systems Review, Vol. 33, No. 5, December 1999, containing the article "Design and implementation of a distributed virtual machine for networked computers" by Emin Gun Sirer, Robert Grimm, Arthur J. Gregory, and Brian N. Bershad, was received by the University of Washington Libraries on January 31, 2000, and was then re-directed to the Engineering Library. The date stamp added to the "Engineering Library Display Periodical Non-circulating until:" sticker affixed to the issue indicates the date the issue was to have been removed from the display periodical area, in this case March 8, 2000. Individual issues of periodical titles that were to be displayed in the Engineering Library Display Periodical area were displayed for one month. Therefore the issue would have been added to the display periodical area and made publicly available one month prior to the date stamped on the sticker, or February 8, 2000. A copy of the first few pages of the periodical issue as it is maintained in the Library's collection, plus the article in question, is attached as Exhibit A.

DeSart Decl. Exhibit 1036.

Mr. DeSart's testimony confirms the authenticity of the Sirer Article



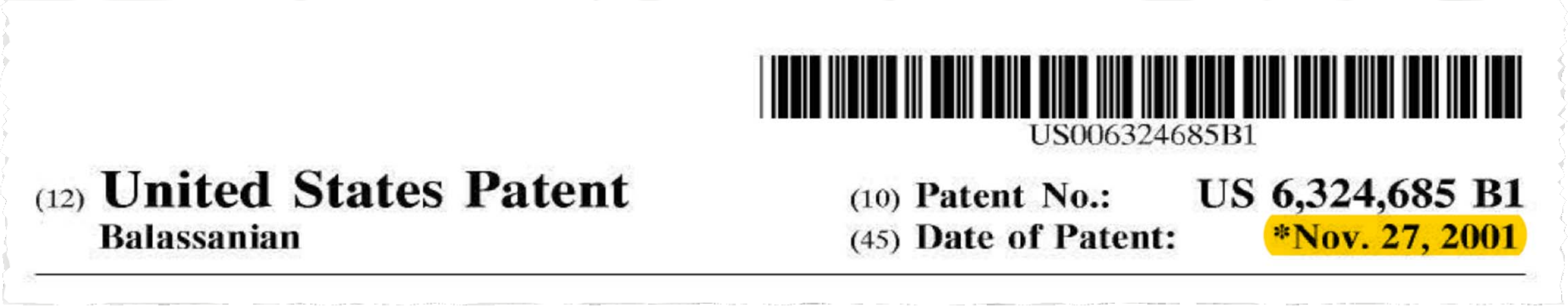
A. They provided a copy, an electronic copy of the document, but I always -- and I've done these, like I said, not an in-person deposition like this, but I've done written depositions a number of times, probably close to 20 times in the past over the years. And I always go and get the original and look it up anyway.

Q. Was the context of those to determine whether something was available in your library?

A. To determine what I -- I knew that it was visible. I knew it was in our library based on nothing more than the electronic copy that I received because it has the University of Washington Library's date stamp on it. So I knew that was from our collection, but, like I said, I went and pulled off the original anyway. I always like to look at the print myself.

Ex. 2006, DeSart Tr. At 6:19-7:8.

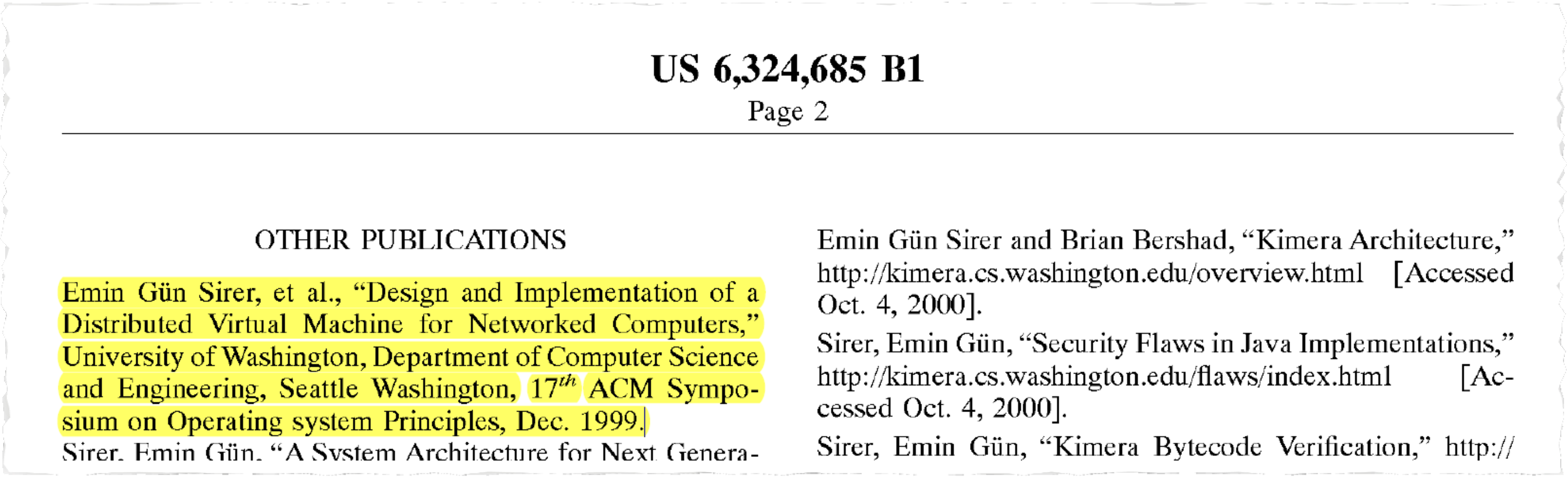
U.S. Patent 6,324,685—issued in 2001 and cited Sirer—is relevant to show public availability and accessibility



US006324685B1

(12) **United States Patent**
Balassanian

(10) **Patent No.:** **US 6,324,685 B1**
(45) **Date of Patent:** ***Nov. 27, 2001**



US 6,324,685 B1
Page 2

OTHER PUBLICATIONS

Emin Gün Sirer, et al., "Design and Implementation of a Distributed Virtual Machine for Networked Computers," University of Washington, Department of Computer Science and Engineering, Seattle Washington, 17^{th} ACM Symposium on Operating system Principles, Dec. 1999.

Sirer, Emin Gün, "A System Architecture for Next Genera-

Emin Gün Sirer and Brian Bershad, "Kimera Architecture," http://kimera.cs.washington.edu/overview.html [Accessed Oct. 4, 2000].

Sirer, Emin Gün, "Security Flaws in Java Implementations," http://kimera.cs.washington.edu/flaws/index.html [Accessed Oct. 4, 2000].

Sirer, Emin Gün, "Kimera Bytecode Verification," http://

Ex. 1024

Questions?