Attorney's Docket No.: FIN0009CIP1 *PATENT*

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In Re Patent Application of:	)	
	)	Examiner: Gary Gracia
Yuval Ben-Itzhak	)	
Golan Yosef	)	Art Unit: 2491
Israel Taub	)	
	)	
Application No: 12/174,592	)	
	)	
Filed: July 16, 2008	)	
	)	
For: COMPUTER SECURITY METHOD	)	
AND SYSTEM WITH INPUT	)	
PARAMETER VALIDATION	)	
	)	

Mail Stop AMENDMENT
Commissioner for Patents
P. O. Box 1450
Alexandria, VA 22313-1450

AMENDMENT AND RESPONSE TO OFFICE ACTION

UNDER 37 C.F.R. §1.111

Sir:

In response to the Office Action dated June 1, 2011, applicants respectfully request that the above-identified application be amended as follows.

IN THE SPECIFICATION:

Please amend the specification as follows.

[0026] Downloadables may be inter alia in the form of source code, such as JavaScript, or in the form of ~~complied~~ compiled code, such as Java applets, that is de-compiled in order to derive its source code.

[0035] At step **230** a determination is made whether or not suspicious computer operations have been detected in the downloadable. If not, then the downloadable is deemed safe and is forwarded to its destination at step **240**. Otherwise, if one or more suspicious computer operations have been detected, then at step **250** monitoring program code is appended to the original downloadable. Referring to the example above, the monitoring code includes lines 11 – 36, and has two functions; namely, ~~makeVulnDict(Arr)~~ makeVulnObjDict(arr) and checkAcx(acxId).

[0036] At line 21 the function ~~makeVulnDict()~~ makeVulnObjDict() is called with array parameter VulnAcxStruct[], to build a dictionary, Vuln_Obj_Dict, of potentially malicious function calls. As seen at lines 6 – 9, VulnAcxStruct[] is an array of three-element arrays, each three-element array corresponding to a potentially malicious function. For purposes of clarity, only one three-element array is defined in lines 6 – 9, corresponding to the method setRequestHeader() of object Msxml2.XMLHTTP, but it will be appreciated by those skilled in the art that additional three-element arrays may be defined. The first element of the three-element array in VulnAcxStruct[] is the name of the object containing the potentially malicious function; i.e. "Msxml2.XMLHTTP".

The second element of this array is the name of the suspicious method, setRequestHeader(), together with the function to be used for input validation of the method; namely,

```
function()
{
        allow =         ["GET", "POST", "HEAD", "DELETE", "PUT",
                        "CONNECT", "OPTIONS"];
        for (i in allow){
                if (arguments[0]==allow[i] return;
        }
        alert("malicious!")
}
```

Thus to validate input parameters for the method setRequestHeader(), the input parameter is matched against six expected non-malicious parameter values GET, POST, HEAD, DELETE, PUT, CONNECT and OPTIONS. If no match is found then an alert is made. It will be appreciated by those skilled in the art that the function given above is but one of many methods for validating input parameters. Other such methods to validate input parameters and to issue a notification when input parameters are not validated, are also within the scope of the present invention.

[0042] At step **280** a determination is made whether or not the input parameters to each of the suspicious computer operations have been validated. If so, then the downloadable is deemed safe and is forwarded to its destination at step **240**. Otherwise, the downloadable is deemed suspicious, an alert is made, and various preventive actions may be taken. One such action, at step **291**, is simply not to forward the downloadable to the destination computer. Another such action, at step **292**, is to neutralize the input parameters that were not validated, by replacing them with valid input parameters, and then forwarding the

remedied downloadable to the destination ~~computers~~ computer. Another such action, at step **293**, is to consult a computer security policy to determine whether or not to forward the downloadable to the destination computer, based on the suspicious computer operations that were detected.

[0045] Subsequent to step **350** the modified downloadable is executed. At step **355** suspicious computer operations are identified at run-time. Step **355** may be performed by referencing a structure, such as the VulnAcxStruct[] structure in the example JavaScript, that lists pre-designated suspicious computer operations. Alternatively, step **355** may be performed by referencing ~~structure~~ a structure that lists pre-designated ~~safe computer~~ non-malicious computer operations.

[0048] At step **380** a determination is made whether or not the input parameters to each of the suspicious computer operations have been validated. If so, then the downloadable is deemed safe and is forwarded to its destination at step **340**. Otherwise, the downloadable is deemed malicious, an alert is made, and various preventive actions may be taken. One such action, at step **391**, is simply not to forward the downloadable to the destination computer. Another such action, at step **392**, is to neutralize the input parameters that were not validated, by replacing them with valid input parameters, and then forwarding the remedied downloadable to the destination ~~computers~~ computer. Another such action, at step **393**, is to consult a computer security policy to determine whether or not to forward the downloadable to the destination computer, based on the suspicious computer operations that were detected.

IN THE DRAWINGS:

Please replace FIG. 2 with the attached replacement sheet. FIG. 2 has been amended to insert the word "TO" in box 250.

Please replace FIG. 3 with the attached replacement sheet. FIG. 3 has been amended to insert the word "TO" in box 350.

The amendments to FIGS. 2 and 3 are shown as mark ups in the attached annotated sheets.

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS

Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS

Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS

Sync your system to PACER to automate legal marketing.