

to retrieve, via the WHOIS web service, registration data associated with the domain name, such as information of an owner of the domain name. In additional non-limiting embodiments of the present technology, the processor may further be configured to verify the availability of the domain name by submitting requests for random subdomain names associated with the domain name and analyzing respective responses.

[54] In yet other non-limiting embodiments of the present technology, where the associated network architecture element type is the website, the processor may be configured to verify availability thereof by establishing a connection therewith. In these embodiments, the connection may be established directly or via a proxy server, as an example.

[55] Finally, in those non-limiting embodiments of the present technology, where associated network architecture element type is the URL, the processor may be configured to verify availability thereof by submitting a request to a webserver associated with the URL. Further, upon receiving a response from the webserver associated with the URL, the processor may be configured to analyze at least some of the following information associated with the response in order to verify the availability of the URL: a response code of the response; a content of the response; and a header of the response. Finally, the processor may be configured to determine a type of the URL, which, according to certain non-limiting embodiments of the present technology, may be at least one of: a file, a directory, and undetermined. Additionally, the processor may be configured to determine whether or not the URL responds to requests for non-existing files; whether the URL is associated with an predetermined exclusion task template; and whether or not the webserver associated with the URL is configured to redirect requests submitted by the processor, and if this is the case – the processor may further be configured to determine a URL associated with the redirection.

[56] The method 100 thus proceeds to step ~~104~~106.

STEP ~~104~~106: DETERMINING A RESPECTIVE PREDETERMINED TASK TEMPLATE ASSOCIATED WITH THE GIVEN NETWORK ARCHITECTURE ELEMENT

[57] According to certain non-limiting embodiments of the present technology, at step ~~104~~106, the processor may be configured to determine tasks for scanning the given network architecture element for determining potential vulnerabilities associated therewith.

[58] To that end, in some non-limiting embodiments of the present technology, the processor may be configured to apply a respective predetermined task template including tasks configured for scanning the given network architecture element for detecting vulnerabilities associated therewith.

manager module 302 may be configured to arrange tasks therein, for example, according to their priority, thereby forming a task queue 304.

[64] Finally, as it can be appreciated, the task manager module 302, may be configured to transmit the tasks from the task queue 304 to one of a plurality of scanning modules, such as a given scanning module ~~304~~306.

[65] According to certain non-limiting embodiments of the present technology, the given scanning module ~~304~~306 may be scaled up for different networks and operating frameworks; it also may be scaled up for different ranges of network architecture elements.

[66] In some non-limiting embodiments of the present technology, scalability of the scanning system 300 may be controlled by the task manager module 302 and by tasks running on scanning modules. For example, the scalability of the scanning system 300 may be achieved by isolating the task manager module 302 from each of the scanning modules. Besides, the scanning modules may be associated with a plurality scanning tools, therefore it would be possible to use a large number of commercial, open source, proprietary, and other tools to scan the network for vulnerabilities. By doing so, certain non-limiting embodiments of the present technology may allow improving flexibility of the scanning system 300, as well as reducing scanning time, thereby saving computational resources of the scanning system 300 and the processor.

[67] Thus, according to certain non-limiting embodiments of the present technology, in order to execute one or more tasks received from the task manager module 302, the given scanning module ~~304~~306 may be configured, by the processor, to execute (or otherwise, have access to) one or more network security tools.

[68] Below, there is provided an example list of network security tools executable by the given scanning module ~~304~~306 for executing one or more tasks, according to certain non-limiting embodiments of the present technology. It should be expressly understood that the present example list of network security tools is not exhaustive, and in additional non-limiting embodiments of the present technology, may include other network security tools currently used for executing tasks associated with scanning networks for vulnerabilities.

[69] Network Mapper (Nmap) may be used at the data collection stage to scan in real time for currently active systems within a given network. Nmap also allows locating a access control list (ACL) of a router or firewall rules by checking an associated acknowledgment flag (ACK) and using other methods. At the binding and evaluation stages, Nmap may be used to scan network

can be stored in Docker containers on a remote server (not depicted) and may be accessed by the processor as needed.

[74] In specific non-limiting embodiments of the present technology, the network security tools may be stored on the remote server based on their functionality. In other non-limiting embodiments of the present technology, the network security tools may be stored on the remote server based on associated predetermined task templates.

[75] Needless to say that, in various non-limiting embodiments of the present technology, the given scanning module 306 may be configured to access more than one network security tools simultaneously – that is, a predetermined combination thereof.

[76] With continued reference to Figure 3, according to certain non-limiting embodiments of the present technology, the task manager module 302 processes the predetermined task template 200 to select scanning modules corresponding to the tasks included therein. Further, the task manager module 302 may be configured to transmit the tasks to each of the associated scanning modules – such as the given scanning module 306.

[77] In some non-limiting embodiments of the present technology, the task manager module 302 may store a list of scanning modules coupled therewith as well as data about their current utilization. The data about utilization of the scanning modules can be, in turn, provided by the scanning modules. According to certain non-limiting embodiments of the present technology, if there is at least one scanning module with less than 100% utilization, the task manager module 302 would add tasks to the task queue 304. Then, available scanning modules may be configured to request the tasks from the task queue 304 based on the “first come, first served” principle, and execute the task by using one or more network security tools as described above.

[78] As also mentioned above, the predetermined task template 200 may include one or more tasks depending on an associated type of the given network architecture element to be scanned. Below, there is provided an example non-exhaustive list of types of tasks that may be associated with the predetermined task template 200.

[79] 1. Tasks for identifying subdomain names. According to certain non-limiting embodiments of the present technology, these tasks may include tasks for dictionary-based search for subdomains (sets of names as text lists) and for wildcard-based search (all possible combinations of certain groups of characters, with certain length e. g. a-z + 0-9, 1 to 4 characters long).

[80] Typically, in certain non-limiting embodiments of the present technology, there are two ways to run this scanning:

- A regular query for subdomains on DNS servers. This method may be used when a DNS zone sends a negative response to non-existing names.
- A query to a DNS server followed by a HTTP request to a returned IP address. This method may be used for wildcard zones when any subdomain query returns a positive response.

[81] 2. Tasks for identifying URLs. According to certain non-limiting embodiments of the present technology, these tasks may include tasks for dictionary-based search for URLs (sets of names as text lists) and for wildcard-based search (all possible combinations of certain groups of characters, with certain length e. g. a-z + 0-9, 1 to 4 characters long).

[82] Also, the tasks for identifying URLs may be based on the sets which are generated using the known URLs of a website (e.g. if the last character of a URL is a digit, all its versions with the modified last digit are going to be checked).

[83] 3. Tasks for identifying open network ports. According to certain non-limiting embodiments of the present technology, to execute such tasks, the processor may be configured to use a standard TCP connection.

[84] 4. Tasks for retrieving information of at least one web service associated with the open network ports. According to certain non-limiting embodiments of the present technology, to run these tasks, the processor may be configured to query each of the services to determine their respective versions and other data.

[85] 5. Tasks for brute-force password mining (similar to the aforementioned). In some non-limiting embodiments of the present technology, if a given service has been previously identified, the processor may be configured to attempt to connect with it and authorize therewith using a pre-determined list of usernames and passwords.

[86] 46. Tasks for identifying vulnerabilities associated with the at least one web service. These tasks may be used once a version of the at least one web service and the other data are received. According to certain non-limiting embodiments of the present technology, this data may further be verified against databases of the web resource vulners.com, which may return information about potential vulnerabilities associated with the at least one web service. Finally, according to certain

non-limiting embodiments of the present technology, the information about the vulnerabilities may be recorded in the database of network architecture elements.

[87] The method 100 thus proceeds to step 110.

[88] STEP 110: IS AN OTHER NETWORK ARCHITECTURE ELEMENT IS DETECTED?

[89] At step 110, according to certain non-limiting embodiments of the present technology, the processor may be configured retrieve results from scanning modules (such as the given scanning module 306) indicative of the identified vulnerabilities associated with the given network architecture element. Further, the processor may be configured to determine whether the results include any data indicative of at least one another network architecture element associated with the given one. In other words, the processor may be configured to determine if at least one identified vulnerability associated with the given network architecture element provides access to the at least one another network architecture element.

[90] Further, according to certain non-limiting embodiments of the present technology, in response to receiving, ~~from~~ the given scanning module ~~304~~306, the data indicative of the at least one another network architecture element, the processor may be configured to identify further vulnerabilities associated therewith. To that end, in some non-limiting embodiments of the present technology, the processor may be configured to store the at least one another network architecture element in the database of network architecture elements. Further, the processor may be configured to repeat at least some of the steps 102, 104, ~~104~~106, 108, and 110 of the method 100 in respect of the newly identified, the at least one another network architecture element, in order to identify further vulnerabilities associated therewith. By doing so, the processor may be configured to loop the method 100 back to the beginning thereof every time a new network architecture element is uncovered.

[91] To that end, based on a type of the at least one another network architecture element, in some non-limiting embodiments of the present technology, the processor may further be configured to apply a same predetermine task template, that is, the predetermined task template 200. However, in other non-limiting embodiments of the present technology, in order to determine further vulnerabilities associated with the at least one another network architecture element, the processor may be configured to apply at least partially different predetermined task template (that is, another predetermined task template) from the predetermined task template 200.

CLAIMS

What is claimed is:

1. A method for determining vulnerabilities on a network, the method being executable by a processor, the method comprising:

receiving, by the processor, data indicative of a first network architecture element;

determining, by the processor, based on the data indicative of the first network architecture element, a respective one of a plurality of predetermined task templates,

the respective one of the plurality of predetermined task templates including one or more tasks executable for identifying respective vulnerabilities associated with the first network architecture element;

in response to identifying, based on the respective one of the plurality of predetermined task templates, at least one vulnerability associated with the first network architecture element, ~~the at least one vulnerability providing access to a second network architecture element associated with the first network architecture element:~~

determining, by the processor, based on the at least one vulnerability, a second network architecture element associated with the first network architecture element, the at least one vulnerability providing access to the second network architecture element;

determining, by the processor, data indicative of the second network architecture element;

storing, by the processor, the data indicative of the second network architecture element in a database; and

using, by the processor, the data indicative of the second network architecture element for identifying further vulnerabilities on the network.

2. The method of claim 1, wherein the first network architecture element is associated with a first network architecture element type comprising at least one of: an IP address, a website, a Uniform Resource Locator(URL), a domain name, and a network port.

3. The method of claim 2, wherein, if the first network architecture element type is the domain name, the method further comprises verifying, by the processor, availability of the domain name using a WHOIS web service.
4. The method of claim 3, wherein the method further comprises determining, by the processor, using the WHOIS web service, registration data associated with an owner of the domain name.
5. The method of claim 3, the verifying the availability of the domain name further comprises analyzing, by the processor, data responsive to requests for random subdomain names associated with the domain name.
6. The method of claim 2, wherein, if the first network architecture element type is the website, the method further comprises verifying, by the processor, availability of the website, the verifying comprising establishing a connection with the website, the connection being one of a direct connection and a connection via a proxy server.
7. The method of claim 2, wherein if the first network architecture element type is the URL, the method further comprises verifying availability of the URL, the verifying comprising transmitting, by the processor, at least one request to a webserver associated with the URL, and the verifying further comprising at least one of: analyzing a response code of a response received from the webserver associated with the URL; a content associated with the response; a header of the response; and a type of the URL, wherein the type of the URL is associated with one of a file, a directory, and underdetermined.
8. The method of claim 7, wherein the verifying the availability of the URL further comprises determining at least one of:
 - if the URL is responsive to requests for non-existing files;
 - if the URL is associated with a predetermined exclusion template;
 - if the webserver associated with the URL is configured to redirect the at least one request to at least one another URL, and determining the at least one another URL.
9. The method of claim 1, wherein the data indicative of the first network architecture elements are received from one of: a user input and the database.

10. The method of claim 1, wherein the ~~the~~ respective one of the plurality of predetermined task templates is associated with a plurality of network architecture element types, to which the respective one of the plurality of predetermined task templates is applicable.

11. The method of claim 10, wherein the one or more tasks comprise at least one of:

tasks for identifying subdomain names;

tasks for identifying URLs;

tasks for identifying open network ports;

tasks for retrieving information of at least one web service associated with the open network ports;

tasks for password mining for accessing the at least one web service associated with the open network ports; and

tasks for identifying vulnerabilities associated with the at least one web service.

12. The method of claim 1, further comprising:

receiving, by the processor, the data indicative of the second network architecture element;

determining, by the processor, based on the data indicative of the second network architecture element, another one of the plurality of predetermined task templates,

the other one of the plurality of predetermined task templates including one or more tasks executable for identifying respective vulnerabilities associated with the second network architecture element;

in response to identifying, based on the other one of the plurality of predetermined task templates, at least one vulnerability associated with the second network architecture element, the at least one vulnerability providing access to a third network architecture element associated with the second network architecture element:

determining, by the processor, data indicative of the third network architecture element;

storing, by the processor, the data indicative of the third network architecture element in the database; and

using, by the processor, the data indicative of the third network architecture element for identifying further vulnerabilities on the network.

13. The method of claim 12, further comprising iteratively repeating at least the steps of the receiving data and the determining a predetermined task template until no further vulnerability is identified.
14. The method of claim 13, further comprising generating, at each iteration, an intermediate report including data of vulnerabilities having been identified on the network.
15. The method of claim 12, wherein the other one of the plurality of predetermined task templates is a same as the respective one of the plurality of predetermined task templates.
16. The method of claim 12, wherein the other one of the plurality of predetermined task templates is at least partially different from the respective one of a plurality of predetermined task templates.
17. A system for determining vulnerabilities on a network, the system comprising a computing device, the computing device further comprising:

a processor;

a non-transitory computer-readable medium comprising instructions;

the processor, upon executing the instructions, being configured to:

receive data indicative of a first network architecture element;

determine based on the data indicative of the first network architecture element, a respective one of a plurality of predetermined task templates,

the respective one of the plurality of predetermined task templates including one or more tasks executable for identifying respective vulnerabilities associated with the first network architecture element;

in response to identifying, based on the respective one of the plurality of predetermined task templates, at least one vulnerability associated with the first network architecture

element, ~~the at least one vulnerability providing access to a second network architecture element associated with the first network architecture element:~~

~~determine, based on the at least one vulnerability, a second network architecture element associated with the first network architecture element, the at least one vulnerability providing access to the second network architecture element;~~

determine data indicative of the second network architecture element;

store the data indicative of the second network architecture element in a database; and

use the data indicative of the second network architecture element for identifying further vulnerabilities on the network.

18. The system of claim ~~15~~17, wherein the processor is further configured to:

receive the data indicative of the second network architecture element;

determine, based on the data indicative of the second network architecture element, another one of the plurality of predetermined task templates,

the other one of the plurality of predetermined task templates including one or more tasks executable for identifying respective vulnerabilities associated with the second network architecture element;

in response to identifying, based on the other one of the plurality of predetermined task templates, at least one vulnerability associated with the second network architecture element, the at least one vulnerability providing access to a third network architecture element associated with the second network architecture element:

determine data indicative of the third network architecture element;

store the data indicative of the third network architecture element in the database; and

use the data indicative of the third network architecture element for identifying further vulnerabilities on the network.

19. The system of claim 18, wherein the processor is further configured to iteratively repeat at least the steps of the receiving data and the determining a predetermined task template until no further vulnerability is identified.
20. The system of claim 19, wherein the processor is further configured to generate, at each iteration, an intermediate report including data of vulnerabilities having been identified on the network.