

Salinity Peer-to-Peer Botnet

Malware Analysis Project
MSc in Cybersecurity

John Linnane

Student ID: x20151292

School of Computing
National College of Ireland

Lecturer: Michael Pantridge

1. Executive Summary

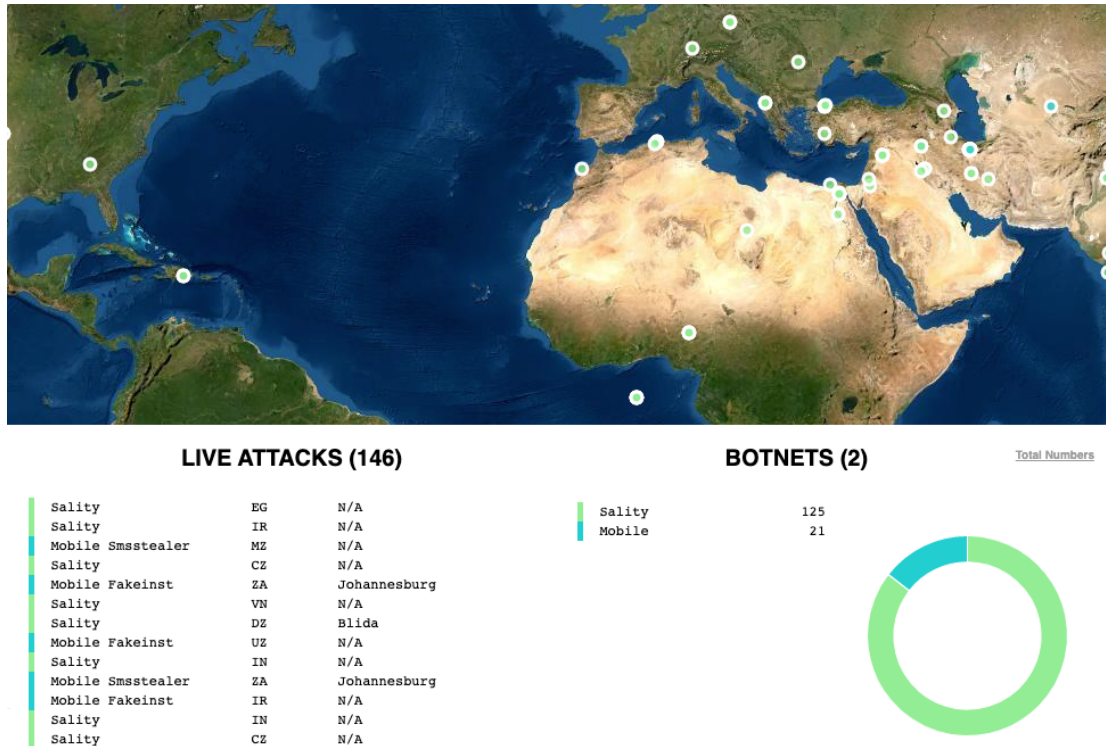
The objective of this report is to research the workings of a selected botnet. The Sality botnet has been highly active globally in various iterations for close to two decades and continues to be a significant threat to the present day. The significance of this botnet will be examined in terms of its impact on an operational level and on a global organisational level. Analysis of online research papers, datasets, and pcap files will be used to ascertain the botnet's workings from a software perspective. According to some malware threat maps, Sality is still one of the most pervasive botnets in terms of attack frequency. This report aims to investigate whether these map results are due to legacy mislabeling or whether Sality is still highly active.

Sality is noteworthy due to its persistence over time and its multifaceted mechanism of action. The central virus facilitated attack through a large selection of malware techniques, including file injection, polymorphic code, trojan downloaders and many more. Uses include sending spam, keylogging, data exfiltration, system manipulation and further infection of other systems [1]. Of particular interest is the peer-to-peer (P2P) control system employed. In contrast to a central command-and-control server, Sality creates a botnet through creation of a network of equipotent peers, with no central server [2]. The security implications of this will be examined and assessed. Sality can be found globally, with particular concentrations in Venezuela, Romania and India [3].

2. Methodology

2.1 Botnet Selection Strategy

A primary motivation for selecting this botnet was its prevalence on Checkpoint and LookingGlass threatmaps [4, 5]. The live feed on these maps suggest that Sality is highly active to this day.



[Lookingglass threat map showing high instance of Salinity] [5]

Another factor is choosing a botnet was to choose a more unusual botnet that may not be reported on as frequently. Botnets such as Mirai, Emotet and Mariposa have a significant amount of online resources available. The Emotet and Necurs botnets are well documented through online resources such as malware-traffic-analysis.net and would be assumed to be a popular research choice by other students [6].

Salinity was deemed worthy of investigation due to the question as to how a botnet that first became known in 2003 can still persist to this day. In addition, some preliminary research showed Salinity to have an interesting scope and many atypical behavioural attributes, such as the use of Peer-to-Peer networking (as opposed to a central C2 server), polymorphic code and array of attack methods including rootkits, backdoors and file infectors [14]. Salinity is considered significant due to the potential impact of some of its capabilities, including keylogging and other methods of sensitive data exfiltration. It is also significant in scale, with estimates of number of bots being in the hundreds of thousands [2]

Upon seeing the results in the Checkpoint and LookingGlass threatmaps the next step was to locate any reports that have been created about Salinity through trusted security vendors and research organisations. Of particular note is the very detailed but out-of-date 2011 report by Symantec [2] and a large dataset from VirusTotal which covered many aspects of Salinity's mechanism of action [1]. A sample of the malware itself was located from theZoo GitHub repository [7] and a sample .pcap file was located on the community website of RSA Security [8]

2.2 Available Data

Two resources that were vital to the understanding of this malware were the Hybrid Analysis 2018 report on Virus.Win32.Sality.ag [1] and Virus Total's 2016 report of a Sality sample based on Florian Roth's signature detection [24]. The RSA website .pcap sample was also useful for cross referencing report data with raw network traffic data [8].

The Hybrid Analysis report gives a huge data set related to the sample's network activity, anti-detection capabilities, ransomware capabilities, installation mechanisms, anti reverse engineering systems and many other aspects. A sample of such information which illustrates the fidelity of the dataset is presented below:

Sends UDP traffic

details	"UDP connection to 58.40.150.204"
	"UDP connection to 105.62.38.78"
	"UDP connection to 37.68.182.68"
	"UDP connection to 48.0.73.23"
	"UDP connection to 200.60.57.62"
	"UDP connection to 86.153.176.84"
	"UDP connection to 83.82.19.124"
	"UDP connection to 165.1.101.249"
	"UDP connection to 89.149.236.171"
	"UDP connection to 189.68.58.176"
	"UDP connection to 81.190.94.112"
	"UDP connection to 77.122.85.173"
	"UDP connection to 201.45.100.171"
source	Network Traffic
relevance	7/10

[Virus Total dataset sample showing Sality's network traffic IP addresses]

Installation/Persistence

Drops executable files

details "AcroExt.exe" has type "PE32 executable (GUI) Intel 80386 for MS Windows"
"DW20.EXE" has type "PE32 executable (GUI) Intel 80386 for MS Windows"
"AdobeARM.exe" has type "PE32 executable (GUI) Intel 80386 for MS Windows"
"AcroRd32.exe" has type "PE32 executable (GUI) Intel 80386 for MS Windows"
"setup.exe" has type "PE32 executable (GUI) Intel 80386 for MS Windows"
"AcroBroker.exe" has type "PE32 executable (GUI) Intel 80386 for MS Windows"
"ose.exe" has type "PE32 executable (GUI) Intel 80386 for MS Windows"
"dwtrig20.exe" has type "PE32 executable (GUI) Intel 80386 for MS Windows"

source Extracted File

relevance 10/10

[Virus Total dataset sample showing executable files dropped by Sality]

Contains ability to download files from the internet

details URLDownloadToFileA@URLMON.DLL from [Virus.Win32.Sality.ag.exe](#) (PID: 2836) ([Show Stream](#))
InternetReadFileExA@WININET.dll ([Show Stream](#))

source Hybrid Analysis Technology

relevance 10/10

[Virus Total dataset sample showing Windows API call using DLL to download data from a URL]

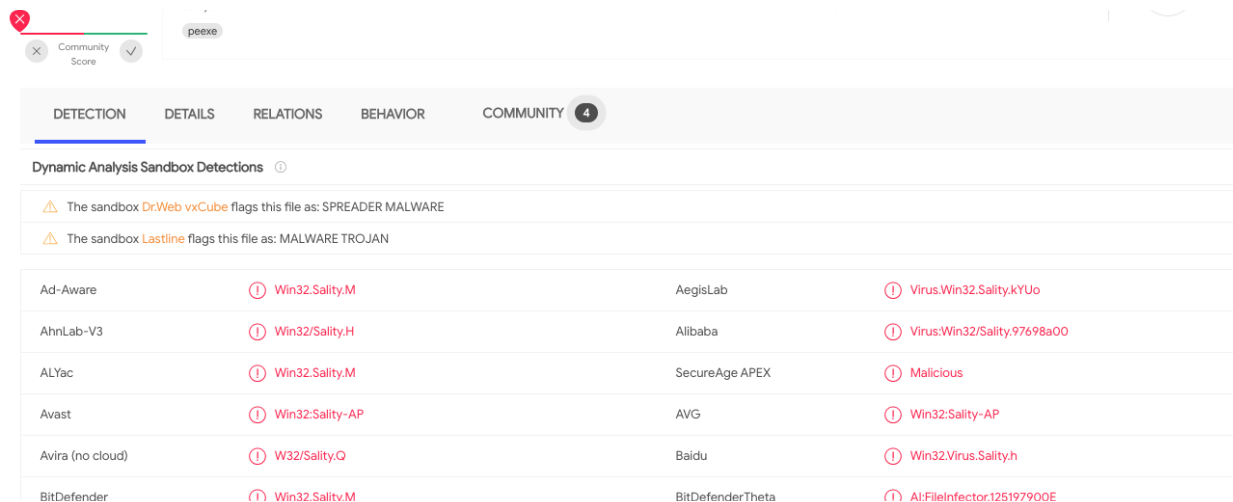
Numerous other reports and blog posts from security vendors were used during research. These are cited throughout this paper and are listed in the References section. The Symantec Sality report was instrumental in understanding Sality's mechanism of action throughout its various versions, in particular with respect to its evolution into a peer-to-peer botnet [2]

2.3 Analysis Methods

A sample of the Sality executable was located on theZoo GitHub repository. For added untraceability, the TOR network could be used for this download. An MD5 hash was calculated from the sample and this was searched in VirusTotal's database. This gave results from 61 security vendors which corroborated the legitimacy of the sample.

```
→ Downloads md5sum Sality_627B8095B1024A0DDFDA01BF9AFF803
627b8095b1024a0dddfa01bf9aff803 Sality_627B8095B1024A0DDFDA01BF9AFF803
```

[Obtaining MD5 hash of malware sample to verify authenticity]



Ad-Aware	Win32.Sality.M	AegisLab	Virus.Win32.Sality.kYUo
AhnLab-V3	Win32/Sality.H	Alibaba	Virus:Win32/Sality.97698a00
ALYac	Win32.Sality.M	SecureAge APEX	Malicious
Avast	Win32:Sality-AP	AVG	Win32:Sality-AP
Avira (no cloud)	W32/Sality.Q	Baidu	Win32.Virus.Sality.h
BitDefender	Win32.Sality.M	BitDefenderTheta	AI:FileInfecto.125197900E

[VirusTotal result from hashes of theZoo's Sality sample]

Analysis methods and tools used are detailed in Section 3 of this report.

2.4 Test Environment Setup

Analysis of the malware consisted of a combination of pre-existing research and static analysis of the sample. The extent of the datasets available through online resources, in particular the research of Hybrid Analysis, meant that a comprehensive understanding of the malware's structure and behaviour could be ascertained without excessive static analysis. VirusTotal's online database also provided additional information regarding vendor classifications and networking characteristics. Reports from security vendors also proved vital to building a complete picture of the botnet's diverse characteristics and evolution over time. Notwithstanding, certain scanning and analysis tools were utilized to corroborate research findings.

A 2013 iMac running MojaveOS was used as the host computer. MacOS was beneficial in this instance due to Sality being built for Windows. This allowed a certain degree of freedom with respect to analysing the malware sample without sandboxing. A virtualbox instance of Ubuntu Server 16.04 running Lockheed Martin's Laikaboss object scanning system was utilised for more in depth scan results. Screenshots of terminal-based analysis results will be drawn from both MacOS and Ubuntu instances.

3. Botnet Investigation & Findings

3.1 Bots Identification

Information about the malware file sample itself was obtained from the *file* command. This confirms that the sample is a Windows PE32 executable.

```
→ Downloads file Sality_627B8095B1024A0DDFDFA01BF9AFF803  
Sality_627B8095B1024A0DDFDFA01BF9AFF803: PE32 executable (GUI) Intel 80386,  
for MS Windows
```

[Results from running the file command on the sample]

As described above, hashes were obtained from the sample using CLI tools. As there are many different iterations of Sality, this particular hash result was used as the basis for further identification.

```
→ Downloads md5sum Sality_627B8095B1024A0DDFDFA01BF9AFF803  
627b8095b1024a0ddfdfa01bf9aff803 Sality_627B8095B1024A0DDFDFA01BF9AFF803  
→ Downloads openssl dgst -sha256 Sality_627B8095B1024A0DDFDFA01BF9AFF803  
SHA256(Sality_627B8095B1024A0DDFDFA01BF9AFF803)= d1471ad5eb84ea711f65f5f579a  
af55aa5bec35d126e6158ea824e754fabb0a6
```

[Obtaining hashes of the malware sample]

Additional file information such as file size, file extension, modification time and machine type were obtained using the *exiftool* CLI tool.

```

→ Downloads exiftool Sality_627B8095B1024A0DDFDA01BF9AFF803
ExifTool Version Number      : 12.22
File Name                    : Sality_627B8095B1024A0DDFDA01BF9AFF803
Directory                   : .
File Size                    : 40 KiB
File Modification Date/Time   : 2014:11:26 19:43:54+00:00
File Access Date/Time        : 2021:04:28 21:38:20+01:00
File Inode Change Date/Time   : 2021:04:28 17:54:59+01:00
File Permissions              : -rw-r--r--
File Type                    : Win32 EXE
File Type Extension          : exe
MIME Type                    : application/octet-stream
Machine Type                 : Intel 386 or later, and compatibles
Time Stamp                   : 1997:01:25 02:45:02+00:00
Image File Characteristics    : Executable, No line numbers, No symbols, 32-bit
PE Type                      : PE32
Linker Version                : 3.10
Code Size                    : 10752
Initialized Data Size         : 10240
Uninitialized Data Size       : 0
Entry Point                   : 0x1c70
OS Version                   : 4.0
Image Version                 : 0.0
Subsystem Version             : 4.0
Subsystem                     : Windows GUI

```

[Addition sample information using exiftool]

These hashes were entered into the search facility on Virus Total's online database. This produced positive identifications from a large number of security vendors, as illustrated in part in Section 2.3 above. Complete VirusTotal results can be viewed in Appendix A.

The malware sample was then scanned with the Laikaboss python script to obtain a large amount of high-fidelity information on the PE package. This was executed on an VirtualBox instance of Ubuntu Server 16.04 dedicated to malware analysis. Python's SimpleHTTPServer module was used to download the sample from the host machine. The resulting text output contained information pertaining to the executable in addition to any sub-components contained within. Details displayed include MD5 and SHA265 hashes, spoken language used and details about positive YARA rule hits from the disposition.yara dispositioner file, a sample of this can be viewed in Appendix B. Also shown were lists of API calls from DLL libraries that sality has imported, a sample of which is illustrated in Appendix C.

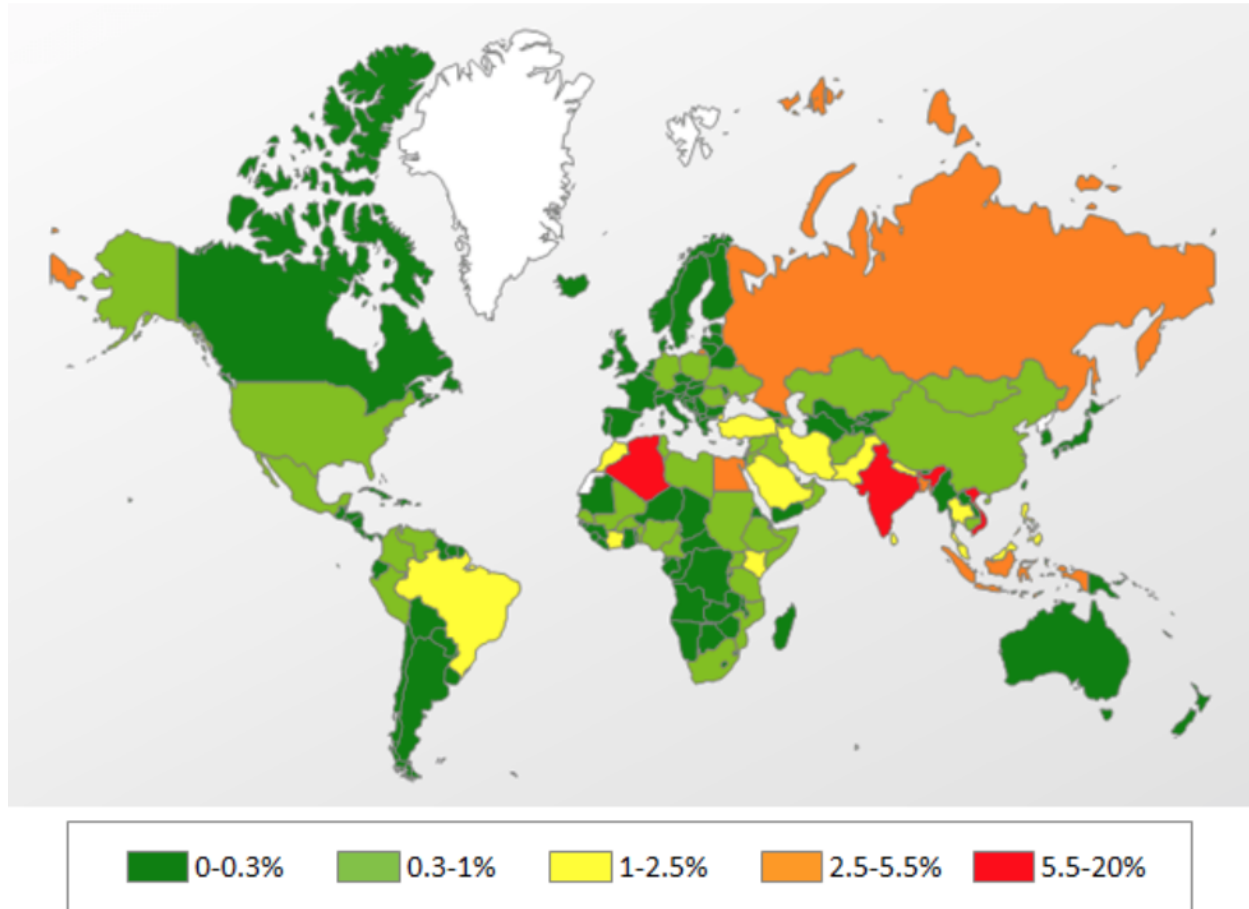
The *strings* command was then executed on the sample using the Ubuntu server in order to display the text strings inside the binary file. The first string “!This program cannot be run in DOS mode.” is indicative of Windows PEs. The subsequent strings are garbled, suggesting the use of a packer or other code obfuscation mechanism. A sample of the output is shown in Appendix D.

3.2 Botnet Size and Damage

Attempting to build a complete picture of the pervasiveness of Sality was challenging due to its many iterations and lack of available up-to-date quantitative data.

A 2016 CheckPoint article puts sality at #2 in their list of “Most Wanted Malware” citing its complexity and willingness to adapt [9]. The above-mentioned Symantec report puts the number of infected bots as in the “hundreds of thousands” [2]. One news source from 2012 claims that Sality is using 3 million bots to scan the IPv4 address space of the entire internet [10]. This scanning campaign generated 20 million scans to 14.5 million addresses [11]. The size of the botnet has been claimed to have grown since 2017 due to its uptake of the Eternalblue vulnerability [3]. Recent reports indicate that Sality has recently become used for mining and theft of cryptocurrencies. This represents a move away from the traditionally lucrative applications of proxying and spamming towards theft of Bitcoin and Ethereum [3].

A 2016 report by Kaspersky illustrates the geographical distribution of the *Virus.Win32.Sality* family by country for a period between September 2014 to 2015. This shows heightened targeting of India and Algeria [12]. Another Kaspersky statement puts Sality as comprising 2.18% of all infections in India at the time of writing this report [13]. The Symantec report shows high attack rates in Brazil, Turkey and India [2]



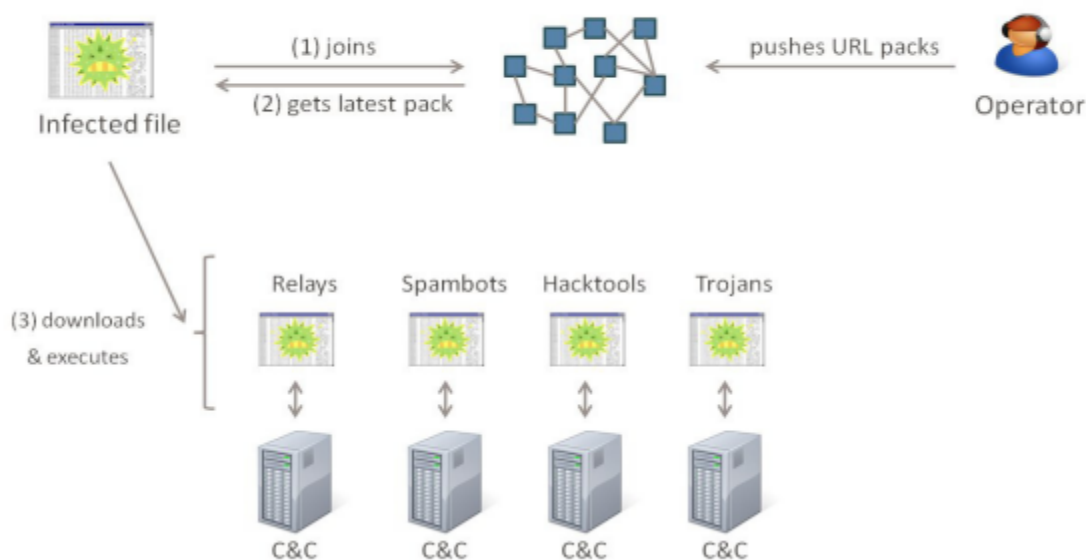
3.3 Target Devices

Salinity's primary target is PCs running various versions of Windows. It typically infects PE files on local, shared and removable drives. Its infector component is capable of propagating the virus into other removable drives and USB devices. In addition, its capabilities are reported to facilitate infection of routers [14, 15]. 2011 saw a large scale campaign targeting Session Initiation Protocol (SIP) servers in order to crack VoIP accounts [16, 2, 17]

3.4 Botnet Architecture

Sality's botnet capabilities are significant in that it does not rely on traditional C2 server structure. It instead incorporates a peer-to-peer distribution scheme whereby communication is not mediated by one central server, but rather through a multi-node network of equipotent infected computers where any peer is potentially a server. This makes botnet takedown

extremely difficult to fully accomplish. Botnet propagation takes place through a bootstrapped list of peers established upon machine infection [2].



[Sality's peer-to-peer distribution scheme] [2]

While peer-to-peer botnets are the exception, they have been known to exist in previous botnets, most notably the Storm botnet [18]. Sality takes full advantage of peer-to-peer architecture in combination with pseudo-random logic and a peer scoring system to create a robust web of control that is incredibly difficult to disable. A list of potential peers is bootstrapped to the infector virus, this can contain a maximum of 1000 potential peers to further infect. Each peer possesses an integer value score stored as the variable *goodcount* in the target's registry. As network requests are beamed out from the infected machine, this list is updated locally depending on the strength and success of the communication. The protocol used to find peers consists of three commands: Announcement & Promotion, Peer Exchange, Pack Exchange of URLs. This process is repeated automatically every 40 minutes using UDP protocol and a pseudo-randomly generated port number. Transmitted data is encrypted using an RC4 stream cipher. The key to decrypt is contained in the first 4 bytes of the payload.

61	06:05:38.625243	192.168.159.128	89.78.194.219	UDP	Source
62	06:05:39.136409	192.168.159.128	80.233.252.97	UDP	Source
63	06:05:39.651388	192.168.159.128	89.72.10.216	UDP	Source
64	06:05:40.166288	192.168.159.128	89.41.155.98	UDP	Source
65	06:05:40.683698	192.168.159.128	79.113.99.52	UDP	Source
66	06:05:41.198702	192.168.159.128	89.230.213.141	UDP	Source
67	06:05:41.714948	192.168.159.128	84.115.163.23	UDP	Source
68	06:05:42.232514	192.168.159.128	89.120.233.17	UDP	Source

0000	00 50 56 ea 65 5c 00 0c 29 61 f0 27 08 00 45 00	.pv.e\..)a.'..E.
0010	00 2f 05 30 00 00 80 11 e0 d9 c0 a8 9f 80 59 29	./0....Y)
0020	9b 62 04 ca 21 93 00 1b 6a bd ed 6e 0f 00 a9 5c	.b..!... j..n...\-. .. N.
0030	a1 07 e0 84 d0 af 2d a6 dd eb 20 4e f6	

[Network traffic .pcap file showing data decryption key] [2]

3.5 Botnet Behaviour

3.5.1 Mechanism of Action

An infected machine on the Sality botnet uses the P2P network to establish a dynamic list of possible other hosts. This list is stored in the machine's registry and is scored based on ability to connect directly with other hosts over UDP for network signalling. The TCP protocol is then used for data exchange of the file infector between peers, being known to be saved in the Device folder as `amsint32.sys` and in the `%SYSTEM%` folder as `wmdrtc32.dll` [19].

Once on a new machine this component replicates and saves itself in various locations throughout the system [20]. It is then used to infect other accessible machines, drives or network shares. The trojan downloader drops small infected executables such as Calculator or Minesweeper on the attacking machine using the bootstrapped list or URLs. The infection code is polymorphic and uses pseudorandom processes to alter file names, useful for evading detection [3, 21]. Other obfuscation methods are employed at this stage and are described below in Section 3.6. These infected executables are added to `explorer.exe` upon system restart [1]. The `autorun.inf` file on the newly infected machine is then modified to execute the infector software. This software payload then further infects files and enumerates network processes.

The bootstrapped URL list is used to download malware using RC4 encryption and a key hardcoded into the first bytes of the virus body. This key has been noted to contain the string "kukutrusted!.", suggesting a link to the Salty Spider threat actors [15]. The downloaded malware code style and signature also bears a resemblance to the Sality virus itself, suggesting a link between virus authors and malware authors. In addition, DLL files are loaded directly into memory during the dropping stage [19]. The infector also uses debug privileges to infect itself into restricted processes. A list of mutually exclusive (mutex) flags is maintained under the name "uxJLpe1m" to ensure that the same process is not infected twice [2].

3.5.1 Purposes and Use Cases

Once Sality has successfully infected a new machine using the peer-to-peer mechanism mentioned above, it then seeks to drop additional malware using a trojan downloader [19]. The list of potential malware behaviours is long and depends on the variant of Sality that is being examined at any one time. Earlier versions focused on generating spam, logging keystrokes using a `.dll` file, rootkits and establishing backdoors [19]. One early variant used the basic exfiltration method of sending a hardcoded email, revealing a connection to Salvat City in Russia. It is suggested that this explains the origin of the Sality name [2].

As Sality increased in complexity and sophistication so did the dropped malware. It was now possible to exfiltrate credentials and passwords, launch DDoS attacks, perform distributed

password cracking and to infect HTML files with malicious iframes using FTP [15, 10]. This era saw the 2011 attack on SIP VoIP servers mentioned earlier. Sality malware was also used to force enrollment to illegitimate Facebook apps and to facilitate manipulation of Google's autocomplete function. HTTP proxies were used to mask operations [2].

Recent iterations of Sality have seen malware being capable of opening a target machine's clipboard, query CPU info, download files from the internet, access cookie information from local browsers [1]. A table of some of these capabilities and their associated .dll files is shown below. Recently, Sality has been implicated in the illegitimate mining and theft of cryptocurrencies [3].

Capability	Associated DLL File
Keystroke Logging	GetAsyncKeyState@USER32.dll
Monitor Info	GetMonitorInfoA@USER32.dll
Get Clipboard Data	OpenClipboard@USER32.dll
Download File from Internet	URLDownloadToFileA@URLMON.DLL
Enumerate Processes	CreateToolhelp32Snapshot@KERNEL32.dll

[DLL libraries incorporated during malware attacks] [1]

3.6 Botnet Resilience

Sality's resilience centers upon its decentralised peer-to-peer architecture, polymorphic code and myriad process obfuscation measures. These have enabled Sality to persist in various iterations from 2003 until this day, having amassed a wide array of techniques both to evade detection and to reinstate itself should a removal occur. The P2P structure means that there is no single point of failure in the form of a central C2 server. Eradicating the Sality botnet would require cleaning every one of the hundreds of thousands of bots in the network, a practically impossible task. As Sality injects itself in numerous locations on a machine, it can simply reinstall itself if one process instance is killed [2].

Sality's polymorphic nature with respect to file naming means that AV systems cannot identify malicious processes. A kernel driver with a randomly generated name is generated, which creates an amsint32 service which kills security processes, filters security vendor IPs from network traffic and was known to block all SMTP traffic in earlier variants. Capabilities exist for blocking Safe-Mode from infected machines.

In previous iterations any anti-virus software detected was simply deleted by the virus. More sophisticated and stealthy techniques have since emerged for disabling anti-virus activity. These involve overwriting entry point instruction to anti-virus execution [2]. Windows Registry Editor and Windows Task Manager are capable of being disabled, along with modification of firewall and notification settings [1]. The Windows Defender Service file has also been known to have been infected [20].

Entry-point manipulation is also used to evade malware detection by signature. This Entry Point Obscuration (EPO) technique involves overwriting the first 327 bytes from the file's entry point and replaced with deobfuscation code [14].

As dropped malware is signed using an ever changing encryption key hardcoded into the Sality instance, it is difficult to replace harmful processes in order to block their activity. In a similar vein, the bootstrapped list cannot be replaced on the client-side with junk URLs to block malicious traffic. This is due to any changes in the peerlist needing to be validated through an external server on the P2P network [2].

3.7 Botnet Takedown

A report by Cloudstrike has indicated that the threat actors responsible for Sality are known as Salty Spider. They are assumed to be of Russian origin, specifically the Republic of Bashkortostan, close to the border with Kazakhstan [3]. This correlates with the address of Salavat City found on early emails used for data exfiltration, as shown in Symantec's Sality report [2].

AVG has released a bespoke tool for removing Sality known as Sality Fix [22] and Kaspersky released their SalityKiller utility in 2015 [23]. However, Semantec report suggests that Version 4 of Sality has no known vulnerability and is so well coded that it is immune to traditional attacks such as buffer overflow [23]. However, the disparity between some of the extremely high threatmap results and the lack of up-to-date reporting leaves a large question mark as to the severity of Sality's threat today.

3.8 Botnet Evolution

The evolution of Sality can be divided roughly into four periods:

3.8.1 Early Versions

First iterations of the Sality virus began to appear in 2003. This version was relatively primitive and consisted of a file infector which prepended UPX packed code onto Windows PE files. One primary malware payload in operation was the keylogger activated through the `GetAsyncKeyState@USER32.dll` DLL library. The data exfiltration mechanism was very basic, transmitting data through a hardcoded email sent through a hijacked SMTP mail protocol. These emails even included the location of Salavat City in Russia. Other tell-tale strings (*KuKu* and *Sector*) were left in the code, enabling easy detection. At this stage it was not possible for the author to update the virus and payload after infection [2].

3.8.2 Improved Version

The period of 2004 to 2008 saw the emergence of multiple variants. Polymorphic code and entry-point obscuring was introduced during this period. The payload was now separated from the virus and malware dropping occurred through a list of hardcoded URLs. It was now possible for the virus to block firewalls and anti-virus processes. Some tell-tale strings were removed from the code base, however one string (*KuKu*) remained [2].

3.8.3 Peer-to-Peer Implementation

The period between 2008 and 2011 saw a significant development through the incorporation of a peer-to-peer networking architecture. It was at this stage that the virus attained botnet functionality. The details of this mechanism is described in Section 3.4. This period also saw numerous developments with respect to detection evasion. Anti-virus processes were evaded in a stealthier manner, malware downloads became encrypted and signed, and the scope of the downloaded malware itself was significantly increased, notably through the targeting of Facebook apps, Google autocomplete and 2011's large-scale SIP server which enabled access to VoIP accounts [2].

3.8.4 Present Day

The period of 2011 to the present day saw the separation of UDP for network signalling and TCP for data exchange. The encryption standard was upgraded from RC4 to RSA-2046 [2]. 2017 saw the introduction of the Eternalblue exploit into Sality's toolkit. This period also saw use cases shift from traditional spambot and proxying functionality into the mining and theft of cryptocurrencies [3]. Exploitation of the use of Windows DLLs for malicious purposes also increased during this time [1].

4. Recommendations

While research suggests that Sality has very little to no vulnerabilities, some behavioural recommendations and best practices should help prevent initial infection from both Sality and other malware. Organisations should provide staff with anti-phishing training so that email attachments are only downloaded from trusted sources [15]. Intrusion Detection Systems and anti-virus software should be properly configured and rigorously updated.

Software should only be downloaded from legitimate sources and downloaded packages should have their hashes checked through online resources such as VirusTotal or other automated means. The VirusTotal API can be used to automate this process. An effective password policy should be introduced in order to prevent email hijacking. Remote workers should use a VPN when connecting to the enterprise network in any way.

Should infection occur AVG's Sality Fix or Kaspersky's Salitykiller can be executed to attempt removal. Firewalls can be configured to only allow traffic on predefined ports, this can assist in mitigating against Sality random port P2P communication.

5. Conclusions

Sality represents an interesting case study due to its unique P2P networking capabilities, large array of downloadable malware tools and substantial anti-detection mechanisms. It has a noteworthy evolution due its transition from a relatively simple virus into a globally dispersed botnet. It was challenging to obtain a clear picture of how pervasive Sality is today. On one hand there are high instances in certain threatmaps and numerous security blog posts written in the last 5 years about how devastating a threat it poses for businesses. Conversely, most of the in-depth reports and analysis seems to stop about 2014. It is possible that the threatmaps are mislabelled, or that Sality has morphed into something else with a different name. The research of Symantec and Hybrid Analysis proved invaluable through their in-depth examinations of Sality's operation, in addition to their substantial datasets.

6. References

1. "Hybrid Analysis - Virus.Win32.Sality.ag"
<https://hybrid-analysis.com/sample/b962405951204905dfd57cab9723d647ea9c5466aefcd5c5a2bac4ddf83a4318?environmentId=100>. Accessed 29 Apr. 2021.
2. "Sality: Story of a Peer- to-Peer Viral Network - vx-underground."
<https://vx-underground.org/archive/Symantec/sality-story-of-peer-to-peer-11-en.pdf>. Accessed 29 Apr. 2021.
3. "Who is SALTY SPIDER (Sality)?| Threat Actor Profile | CrowdStrike." 6 Sep. 2019,
<https://www.crowdstrike.com/blog/who-is-salty-spider/>. Accessed 29 Apr. 2021.
4. "Live Cyber Threat Map." <https://threatmap.checkpoint.com/>. Accessed 29 Apr. 2021.
5. "THREAT MAP by LookingGlass - LookingGlass Cyber Solutions."
<https://map.lookingglasscyber.com/>. Accessed 29 Apr. 2021.
6. "2021-01-15 - Emotet infection from ... - Malware-Traffic-Analysis.net." 15 Jan. 2021,
<https://www.malware-traffic-analysis.net/2021/01/15/index.html>. Accessed 29 Apr. 2021.
7. "GiHub - theZoo"
<https://github.com/ytisf/theZoo/tree/master/malwares/Binaries/Win32.Sality>. Accessed 29 Apr. 2021.
8. "Sality Botnet Beacons Change- How to Detect It - RSA Link - 519442"
<https://community.rsa.com/t5/rsa-netwitness-platform-blog/sality-botnet-beacons-change-how-to-detect-it/ba-p/519442>. Accessed 29 Apr. 2021.
9. "Top 10 Most Wanted Malware - Check Point ... - Check Point Blog." 18 Jul. 2016,
<https://blog.checkpoint.com/2016/07/18/top-10-most-wanted-malware-2/>. Accessed 29 Apr. 2021.
10. "Sality Botnet Scans Entire Internet in Search for ... - Softpedia News." 9 Oct. 2012,
<https://news.softpedia.com/news/Sality-Botnet-Scans-Entire-Internet-in-Search-for-Vulnerable-VoIP-Servers-Video-298049.shtml>. Accessed 29 Apr. 2021.
11. "Darknet as a Source of Cyber Intelligence: Survey, Taxonomy, and" 4 Nov. 2015,
<https://ieeexplore.ieee.org/document/7317717>. Accessed 29 Apr. 2021.
12. "Virus.Win32.Sality - Kaspersky Threats."
<https://threats.kaspersky.com/en/threat/Virus.Win32.Sality/>. Accessed 29 Apr. 2021.

13. "Cyberthreat statistics by Kaspersky Lab - Securelist."
<https://statistics.securelist.com/country/india/on-access-scan/month>. Accessed 29 Apr. 2021.
14. "Virus:W32/Sality Description | F-Secure Labs."
https://www.f-secure.com/v-descs/virus_w32_sality.shtml. Accessed 29 Apr. 2021.
15. "Salty Sality: How Sality malware can affect your business - Reason"
<https://blog.reasonsecurity.com/2020/11/03/salty-sality-how-sality-malware-can-affect-your-business/>. Accessed 29 Apr. 2021.
16. "Analysis of a ``/0" Stealth Scan from a Botnet - CAIDA."
https://www.caida.org/publications/papers/2012/analysis_slash_zero/analysis_slash_zero.pdf. Accessed 29 Apr. 2021.
17. "Endpoint Protection - Symantec Enterprise - Broadcom Community." 16 Feb. 2011,
<https://community.broadcom.com/symantecenterprise/viewdocument?DocumentKey=533f51a1-1ee0-4175-8a64-e62e7b2f2216&CommunityKey=1ecf5f55-9545-44d6-b0f4-4e4a7f5f5e68&tab=librarydocuments>. Accessed 29 Apr. 2021.
18. "Peer-to-Peer Botnets - Department of Computer Science."
<https://www.cs.ucf.edu/~czou/research/P2PBotnets-bookChapter.pdf>. Accessed 29 Apr. 2021.
19. "Remove Sality virus (Virus Removal Instructions ... - 2-Spyware.com." 8 Oct. 2019,
<https://www.2-spyware.com/remove-sality-virus.html>. Accessed 29 Apr. 2021.
20. "Cylance vs. Sality Malware - BlackBerry Blog." 20 Dec. 2018,
<https://blogs.blackberry.com/en/2018/12/cylance-vs-sality-malware>. Accessed 29 Apr. 2021.
21. "What is polymorphic virus? - Definition from WhatIs ... - SearchSecurity."
<https://searchsecurity.techtarget.com/definition/polymorphic-malware>. Accessed 29 Apr. 2021.
22. "How to Remove Win32/Sality in 3 Easy Steps | AVG."
<https://www.avg.com/en-ww/remove-win32-sality>. Accessed 29 Apr. 2021.
23. "Kaspersky SalityKiller Utility 1.3.7.0 Download | TechSpot." 16 Dec. 2015,
<https://www.techspot.com/downloads/6780-kaspersky-salitykiller-utility.html>. Accessed 29 Apr. 2021.
24. "VirusTotal: Sality_Malware_Oct16".
<https://www.virustotal.com/gui/file/5c423b6d21f10dbf02a40e642bfd94f35da0a1aa85ae2b1ee2ecb161c7e0ea9/detection>. Accessed 29 Apr. 2021.

7. Appendices

Appendix A: Complete VirusTotal results on malware sample

Community Score

peeee

d1471ad5eb84aa71f65f5f79aaf55aa5bec35d126e158aa824e754fab0a6

salty.exe

40.00 KB

Size

2021-03-24 13:28:42 UTC

1 month ago

DETECTION

DETAILS

RELATIONS

BEHAVIOR

COMMUNITY

Dynamic Analysis Sandbox Detections

The sandbox **DrWeb_vnCube** flags this file as: SPREADER MALWARE

The sandbox **Lostkline** flags this file as: MALWARE TROJAN

Ad-Aware	Win32.Salty.M	AegisLab	Virus.Win32.Salty.kYUo
AhnLab-V3	Win32/Salty.H	Alibaba	Virus:Win32/Salty.97e98a00
ALYac	Win32.Salty.M	SecureAge APEX	Malicious
Avast	Win32:Salty-AP	AVG	Win32:Salty-AP
Avira (no cloud)	W32/Salty.Q	Baidu	Win32.Virus.Salty.h
BitDefender	Win32.Salty.M	BitDefender Theta	AI.FinInfecter.125197900E
Bkav Pro	W32.Salty.U.PE	CAT-QuickHeal	W32.Salty.K
ClamAV	Win.Trojan.Salty-1020	Comodo	Virus.Win32.Salty.Q@27fjm
CrowdStrike Falcon	Win/malicious_confidence_100% (W)	Cybereason	Malicious.Sb1024
Cylance	Unsafe	Cynet	Malicious (score: 100)
Cyren	W32/Salty.AD	DrWeb	Win32.Sector.28480
Elastic	Malicious (high Confidence)	Emsisoft	Win32.Salty.M (B)
eScan	Win32.Salty.M	ESET-NOD32	Win32/Salty.NAJ
FireEye	Generic.mg.627b8095b1024a0d	Fortinet	W32/Salty.P
GData	Win32.Salty.M	Gridinsoft	Trojan.Hear.L03012301
Ikarus	Virus.Win32.Salty	Jiangmin	Win32/HLLP.Kuku.g
K7AntiVirus	Virus (00001b731)	K7GW	Virus (00001b731)
Kaspersky	Virus.Win32.Salty.q	Kingsoft	Win32.Infected.Salty.ar(kcloud)
Malwarebytes	Malware.Heuristic.1001	MAX	Malware (ai Score=100)
MaxSecure	Virus.W32.Salty.Q	McAfee	W32/Salty.j.x
McAfee-GW-Edition	BehavesLike.Win32.PWSZbot.pc	Microsoft	Virus:Win32/Salty.R
NANO-Antivirus	Virus.Win32.Salty.kozu	Palo Alto Networks	Generic.mf
Panda	W32/Salty.T	Qihoo-360	Virus.Win32.Salty.B
Rising	Virus.Saltyft.A51B (CLOUD)	Sangfor Engine Zero	Virus.Win32.Salty.R
SentinelOne (Static ML)	Static AI - Suspicious PE	Sophos	Mal/Generic-R + W32/Salty-AA
Symantec	W32.Salty.U	Tencent	Virus.Win32.HanKu.h
TotalDefense	Win32/Salty.P	TrendMicro	PE_SALTY.AS
TrendMicro-HouseCall	PE_SALTY.AS	VBA32	Virus.Salty.309
ViRE	Virus.Win32.Salty.r (v)	VRobot	Win32.Salty.C
Webroot	W32.Salty	Yandex	Win32.Salty.X

Appendix B: Results from Laikaboss scan detailing hashes and YARA hits from a sub-component of the Salty malware sample

```
"DISPOSITIONER": {
  "Disposition": {
    "Matches": [],
    "Disposition_File": "etc/modules/disposition.yara",
    "Input_Flags": [],
    "Result": "Accept"
  }
},
"DISPATCH": {
  "Rules": [
    "scan_yara (9)",
    "meta_hash (9)"
  ],
  "Conditional Rules": "DISPOSITION_FILE (10)"
},
"META_HASH": {
  "HASHES": {
    "ssdeep": "24:pjkgWqissssAlmtzPPAtpLBuezhhBSqJRK734:NkhssssAmBPIT/kaKr",
    "SHA256": "3171662f70d4357ccc9803f6508fcb7bc5a7ff5b3dbb11eddb996cc91942b015",
    "SHA512": "05fc743adb19e620d0e581d984333693caa78566affa4bf2d7b87dab0fcb3b4988a0abe28ac6819e7a5ba585728191a6d1dd53a074b55bc0321b61b5b674f87c",
    "SHA1": "6741e21c6f695480b126739b8d4fb8a18bee97f3",
    "md5": "2c7f4739eeddbc8e91f96040ef60c52c"
  }
},
"sourceModule": "META_PE",
"level": 2,
"scanTime": 1619700600,
"depth": 1,
"flags": [],
"rootUID": "784dc4fe-cafb-478d-afdc-d0a73829d56d",
"order": 7
}
1
}
laikaboss@ubuntu:~/laikaboss-master$ python laika.py ~/sal | jq ' _
```

Appendix C: Results from Laikaboss scan showing API calls from imported DLL libraries

```
"Imphash": "cf044153c898d84c13ae1557f40337f7",
```

```
"Imports": {
```

```
  "ADVAPI32.dll": [
```

```
    "RegCloseKey",
```

```
    "RegQueryValueA",
```

```
    "RegOpenKeyA"
```

```
  ],
```

```
  "KERNEL32.dll": [
```

```
    "GetSystemDirectoryA",
```

```
    "LoadLibraryA",
```

```
    "GetModuleFileNameA",
```

```
    "_lclose",
```

```
    "_lopen",
```

```
    "_lread",
```

```
    "_llseek",
```

```
    "GetLastError",
```

```
    "GetFileAttributesA",
```

```
    "GetModuleHandleA",
```

```
    "GetStartupInfoA",
```

```
    "GetCommandLineA",
```

```
    "GetVersion",
```

```
    "ExitProcess",
```

```
    "TerminateProcess",
```

```
    "GetCurrentProcess",
```

```
    "GetStringTypeA",
```

```
    "WideCharToMultiByte",
```

```
    "GetStringTypeW",
```

```
    "GetCurrentDirectoryA",
```

```
    "GetProcAddress",
```

```
    "GetACP",
```

```
    "lstrcpyA",
```

```
    "RtlUnwind",
```

```
    "UnhandledExceptionFilter",
```

```
    "FreeEnvironmentStringsA",
```

```
    "GetEnvironmentStrings",
```

```
    "FreeEnvironmentStringsW",
```

Appendix D: Results from using the strings command on the malware sample showing typical PE warning and some jumbled text.

```
!This program cannot be run in DOS mode.
.text
.rdata
.data
.idata
.rsrc
.reloc
rdata
=\\e
=U\\e
=H[e
SUW3
=U\\e
=U\\e
QhePe
PhU\\e
=U\\e
hDPe
h(Xe
j\\heZe
hePe
heZe
= Pe
</t)<-t\\W
<Et}<ety<Dt[<dtWhlPe
^hpPe
WhU\\e
5H[e
5$Pe
tUhdPe
hXPe
SheZe
heZe
heZe
= Pe
J#D$
```