A study on the Morris Worm

Akshay Jajoo

December 15, 2021

Abstract

The Morris worm [10, 46] was one of the first worms spread via the internet. It was spread on November 2, 1988, and changed how computer security was viewed by computer professionals as well as general public [3]. Since its inception the Morris worm has been studied extensively from the security point of view [46, 37, 43, 10] and is still a point of interest [31, 36].

This paper summarizes the effects, impacts, and lessons learned from the episode. There are other copies of this paper present at [20, 21 and 18. However, I recommend using the arXiv version only.

Acknowledgement

ford guiding the paper. I would also like to thank my friend Ben Harsha for helping with proof reading the paper. Thanks to Lovepreet Singh for all his help.

Overview 1

On November 2, 1988 at around 6 PM a Cornell university grad-student, Robert Morris [39, 40], launched a computer worm (see §8 for explanation of worm) with the intention(he claims) of mapping the Internet. The worm was self-replicating and selfpropagating and took advantage of exploits in the Unix services, described in detail in §2.1. Though Robert claims that the worm was only intended for educational purposes, the worm disrupted the whole Internet [31] (This paper was started as part of CS 52600 more in §2.2). Being first of its kind, the inci-Information Security class at Purdue by prof. dent was very interesting for computer scien-Eugene H. Spafford. Thanks to prof. Spaf- tist. Many researchers at Berkely, MIT and it works and released some fixes and patches within a day. Technical details of the patches are discussed in §3. The worm also impacted the way computer security was perceived, right from the formation of CERT to people being more cautious and thoughtful about security. Some people even term the episode as the big bang of cybersecurity [35]. §4 and §5 talk more about learning and changes which followed the event. At that time there were roughly 60,000 machines on the Internet, and it is estimated that the episode resulted in a loss of \$98 million. Another estimate says that by the time incident was isolated around 5-10 % machines on the Internet were victimized [3]. How drastic could such an incident be with many companies and much of our day to day life activities solely relying on computers? §6 discusses this in detail. §7 is summary and §8 talks about some other but related interesting facts and terminologies.

2 Loopholes and Misfeatures

As soon as the worm was noticed individual researchers, University committees, and government agencies all started analyzing the

Purdue studied the worm and uncovered how vulnerabilities exploited by the worm. [12, it works and released some fixes and patches 46, 6, 43]. This section discusses what went within a day. Technical details of the patches wrong and what were the loopholes? Which are discussed in §3. The worm also impacted exploits of BSD UNIX the worm took advanthe way computer security was perceived, tage of to reach its target? Did Morris made some innocent mistakes in writing the replibeing more cautious and thoughtful about secating routine? Or was the worm intention-curity. Some people even term the episode as ally designed to spread like forest fire?

2.1 Vulnerabilities Exploited

Along with many other flaws, the episode exposed a few specific loopholes in services provided by BSD-derived versions of UNIX. Researchers, engineers and system administrators identified and reported [46, 43, 41] these bugs within 2 days of the outbreak. The following vulnerabilities were exploited.

1. Sendmail – Sendmail is a mailer program used to route emails on the Internet. The program was capable of running in various modes, one of them was being a background daemon. In this mode the program used to listen on a TCP connection for an incoming mail. Sendmail allowed mail to be delivered to a process (the background daemon) instead of the mailbox files, which was used for purpose like setting up automatic vacation responses. Also, while

fixing some other security bugs in the sendmail [43] accidentally a new misfeature was introduced in it. The new bug was that if the sendmail is compiled with DEBUG flag on and if the sender of a mail asks the daemon to go in debug mode by sending a debug command, then the sendmail allowed the sender to send a sequence of commands instead of a recipients address. A combination of these two features was exploited in the worm.

- 2. fingerd The fingerd the utility was used to obtain general user information like name, username and current login status of other users on the system. Like sendmail, fingerd also ran as a background daemon. The worm exploited *fingerd* by overrunning the buffer the process used as its input. The source of the bug was the *qets* routine in the standard C library. A call to *gets* writes input to a buffer and the flaw was that the function implementing gets assumed that the buffer passed to it is large enough for the input to be written. This fault was not only in *qets*, but also, exists in other input-output routines like *scanf* and *fscanf*.
- 3. rsh and rexec rsh and rexec are services which offer remote command interpreters over a network. For authentication purposes Rsh used permission files and a privileged source port whereas *Rexec* used passwords. The worm exploited the fact that there is a high possibility that a password for a local user for an account on a remote machine will be same as its local password. Another likelihood was that a remote account for a user will have rsh permission files for the local account of the user. The worm exploited above two ideas to penetrate into remote machines. Use of rsh was very simple – just look for an account on a remote machine for a user who is running the worm locally. Use of rexec was not that simpler. To use rexec for penetrating the worm used the local password to do a remote login. So for doing this the worm had to crack local password first. Following subsection discusses the password cracking.
- 4. Passwords One of the key requirements for the worm to be able to spread was to be able to break the password of its current host. This was done by

passwords were stored in a publically readable file. However, in his technical report [46] Spafford says that the passwords were not (he means in effect) encrypted as a block of zero bits were repeatedly encrypted using the user password and the result thus obtained was stored in the publically readable file. Interestingly Morris had done a case study [33] on this before the attack. So, breaking passwords was easy by guessing a list of passwords, then encrypting them and comparing the output with the stored value.

2.2 Innocence or not?

This subsection talks about the aspect that whether the intent was malicious or not.

Though the Cornell Commission [12] states that the worm did not harm any user data or system files, it did make infected systems slow. However, the commission also does not fail to mention that "given Morris's evident knowledge of systems and networks, he knew or clearly should have known that such a consequence was certain, given the design of the worm".

The commission report also states that Mor-

exploiting the fact that encrypted user ris made only minimal efforts to halt the worm once it started spreading and also accuses him of not informing any responsible authority about it. However, according to these press articles, Morris did try to talk to a friend, Graham, and Harvard University who informed Andy Sudduth [31, 17]. Morris did suggest Sudduth some steps to protect Harvard computers from the worm. Sudduth also says that after some time Morris again called him realizing that he had made a 'colossal' mistake, asking him to publicly publish an anonymous apology with instructions on how to fix things. However, Sudduth was also the first witness for the defense in the law-suit against the worm and, in response to a question by prosecutor Mark D. Rasch, he said [17] that "He (Morris) wanted that I should keep it quiet(a major security bug in a file transfer program in the BSD UNIX), yes". This fact gives a reason to doubt that Morris might have had notorious intents.

> With above contrasting information, it is not clear whether Morris was only performing an innocent educational experiment with no malicious intent. However, one thing is clear even if it was an innocent act the Internet had grown to such an extent that even innocence can cause great damage.

Another interesting fact is that Eric Allman, developer of the sendmail and delivermail, in a personal communication to Donn Seeley said, "The trap door resulted from two distinct 'features' that, although innocent by themselves, were deadly when combined (kind of like binary nerve gas)" [43]. Exploit of sendmail is another example of how innocent mistakes can be harmful.

3 Overview of the Worm and early actions

This section provides high-level overview of working of the worm *i.e.* answers the following question, "What exactly did the worm do that led it to cause an epidemic?". This section will also briefly discuss some of the early patches released to target this problem.

The worm can be considered as divided into two major parts a bootstrap(vector) program and a main program. The vector program is a 99-line C program and the main program is a large relocatable object file that is compatible with VAX and Sun-3 systems [43]. The bootstrap program is included in the appendix in [46].

3.1 Working of the worm

3.1.1 How does the worm spread?

Once the worm is established on a machine it then tries to locate a host and, most importantly, target accounts on the host to infect new machines which it then exploits via one of the loop holes discussed in §2 to pass a copy of the worm to the remote machine. The worm tries to obtain the address of potential target hosts by reading various system tables like /etc/hosts.equiv and /.rhosts and user files like .forward and .rhosts, ordered in such a way such that it reads files having the name of local machines first. It might be doing this as local machines are more likely to give access without authentication. For a fixed address the worm can try to penetrate in one of the following ways:

- Exploiting the bug in the *finger* server which lets the worm download its code instead of a *finger* request and then tricking the server to execute it.
- Using the bug in the debugging code of the *sendmail* SMTP mail service.
- Guessing passwords and then using *rsh* and *rexec*

3.1.2 Why was performance of the infected machines degraded?

Infected machines slowed down because of uncontrollable replication of the worm because the worm was only using the main memory (see §3.1.3) for its entire processing, this lead to memory clogging [12] and resulted in the machines slowing down.

3.1.3 How does it try to hide itself?

Following steps summarize some of the masquerading steps taken by the worm described in [43]

- On startup, the worm used to delete its argument list and set the very first argument as sh in an attempt to pretend to be a shell command interpreter.
- Used to *fork* itself so that it doesn't have to stay with the same process *i.d.* for very long.
- It read all the files which are part of the worm program into memory and deletes them so that no evidence is left
- Turns off *core-dump* generation, so that if the worm program crashes no dump files, leaving evidences behind are generated. This also helps in preventing analysis of the worm.

 While loading the worm file most of the non-essential symbol table entries were deleted to ensure that even if the worm file is caught before deleting, it will be harder to guess what the routines are doing.

3.1.4 What does it not do?

The program was a worm, and not a virus (see §8.1 and §8.2 for difference in a worm and a virus). It did not attempt to modify any other program or files on the system. It also, didn't delete any system files [43, 12], and in-fact did not attempt to incapacitate the system by deleting any of the already existing files, it only deleted files created by itself. It, also neither installed *Trojan Horses* nor transmitted decrypted passwords anywhere. It didn't try to get *superuser* privileges.

3.2 Early Actions and Patches

Scientist and engineers at major institutions like MIT, NASA, Purdue University, UC Berkeley, and many others started realizing that something was wrong late night on Nov. 2, 1988, or early morning Nov. 3, 1988, and started taking immediate action. The first formal public posting about the virus was sent by Peter Yee of NASA Ames at 2:28 am,

timeline of the major events and actions from the assumed "beginning" of the worm until its full decompiled code was installed at Berkeley [43, 41].

On Thursday, Nov. 3 morning at 5:58 am several patches to fix the worm were released by Keith Bostic of UC Berkeley [30], one of the key people in the history of BSD UNIX. via the tcp-ip mailing list which was also forwarded by several others [43, 41]. E.H. Spafford [13] from Purdue University analyzed the worm and released some patches in his detailed technical report on the worm[46]. Patches for fingerd and sendmail are discussed in detail in this section.

Nov. 8, 1988, on the Tuesday following the event, the National Computer Security Center(NCSC) called a meeting of scientists, officers, faculty members from institutions including the National Institute of Science and Technology, the Defence Communication Agency, the DARPA, the Department of Energy, the Ballistics Research Laboratory, the Lawrence Livermore National Laboratory, the CIA, the UC Berkeley, the MIT, SRI International, the FBI, and various other

on Nov. 2, 1988, via the mailing list "tcp-stake holders. The three fourth of the day ip@sri-nic.arpa" [41]. Reports gave a detailed was spent in analyzing the event from the perspective of all the participants and the remaining time was used to discuss learnings from the event and what actions to be taken [41]. Some of the actions taken by NCSC are discussed in section §5.2.

3.2.1Patches for sendmail

On the Thursday following the attack, Keith Bostic sent following two fixes or suggestions for sendmail:

- 1. At 5:58 a.m. on the list tcp-ip@srinic.arpa which provided the compile without the debug command fix to sendmail [41]. This posting also suggested to rename the UNIX C compiler(cc) and loader(ld), this worked as the worm needed the path to them to compile itself and helped in protecting against the non-sendmail attack.
- 11:12 2. At the list a.m. on comp.4bsd.ucb-fixes. This fix suggested to use θxff instead of $\theta x\theta\theta$ in the binary patch to the *sendmail*. This was needed to support the previous patch. The previous patch was effective however would have fallen to debug mode if an empty command line was sent. He

also asked to look for string "debug" in the sendmail binary using the UNIX strings command and mentioned that if there is no presence of the string then the version is definitely safe.

The patch for *sendmail* can be found in the appendix of this [46] report.

3.2.2 Patches for fingerd

On the Thursday night following the attack at 10:18 p.m., Bostic sent out a fix for fingerd. Overall, this was the third fix posted by him. This fix had new source code for fingerd which used fgets instead of gets and instead of return used exit. This bug-fix post also had another version of sendmail which totally removed the debug command. See §8.4 to know more about why gets cannot be fixed. A patch for *fingerd* can be found in the appendix of this [46] report.

On Friday, following the attack at 5:05 p.m., Bostic released his fourth bug-fix. This was a fix to the worm [41].

Lessons Learnt 4

Use of worm-like capability was not new in this case researchers have tried it to enable automated operating system patches across multiple networks[35]. This incident showed that, with technology advancement, unintended or undesirable consequences can follow. Being the first of its kind this event gave many lessons to scientists, engineers, technical agencies as well as general public. If we look at the broader timescale around this event, this was the time when the use of the Internet was growing rapidly outside of research. Press reporting of the event (like this [17] and many others) made the general public (non-research community) aware of computer networking and, most importantly, it made many people aware of the fact that malicious computer programs can be written [32] and also made the community aware of computer and network security issues and a widespread concern grew out among people that whether the network no which many essential things like transportation, commerce, and high-risk services like national defense and space missions rely upon is secure. The shocking slowdown of the fix was different than the previous ones, It system and the awareness created made government agencies, universities, and all other and think about computer and network sefaults were in the application programs. curity from all perspectives, and figure out Need of a Response and coordinating ever, based on my readings I found that there threats was more or less a general consensus among No limited connectivity - Researchers outall the stakeholders regarding the following right denied suggestions of limiting the conpoints.

analogy with biological fact, biological genetic diversity makes species more robust. tively impact the progress of the research. Having diversity in computer programs will make the network more robust as an attack exploiting flaws of any particular software will have its effect limited only to machines running or using that software, and will prevent it from bringing the whole network down.

Least Privilege - The basic security principle of least privilege says that any entity should only be given just enough privilege to carry out the work they are supposed to. There was a consensus among people that the principle of *least privilege* should not be ignored for computer security. Ignoring this might lead to disasters.

Defense Mechanism - Defense mechanisms should be installed at end-hosts, in this par-

stakeholders analyze the incident carefully ticular event the network performed well and

learnings from the episode for future safety team - Just like for any other critical situa-[38], [41] etc. Some of the views put forward—tion there should be an emergency response by different people were contrasting. How- team for responding to computer security

nectivity and agreed to the argument that Diversifying the options - If we make an "the cure shouldn't be worse than the disease" [41]. Limiting connectivity will nega-

Have we really learned? 4.1 Status after 29 years

In that period, 1988 - 1990, for spreading the worm which slowed down some machines Morris was sentenced to 3 years of Probation and \$3276 to cover the cost of the probation, a fine of \$10,500 and 400 hours of community service. There was a sense of urgency among people – many technologies and security-related (including non-computer security) agencies met within days of the event[38]. Teams like *CERT* were created and many other actions described in §5 were taken, public awareness on network security

where hackers are mostly known as someone who does money related fraud or steals digital information and uses it for some unintended purpose. The Morris worm has become a history lesson, a forerunner that put everyone on guard from a demon (However, if executed even now the worm will still need daemons for carrying out its task! Pun intended).

As discussed earlier in this section, the event gave many immediate lessons but have we been able to keep up with learnings the forerunner gave us. Have we learned a hidden lesson, that if things can then they will go wrong? Have we been able to scale up the learnings and concern for computer and network security with growing use of digital devices and computer networks? In my opinion, the answer is yes and no. To me, it seems that in many cases we are on the "best effort" mode for computer and network security. Many core-technical companies do understand the importance of security and are working towards making systems and designs They have dedicated teams more secure. working towards product security [7]. System designers are taking care of security issues in advance while designing new systems or architecture. Security considerations in the de-

spiked. Now, 29 years later, we are in a world amples. Anti-virus and anti-malware have a big market, and many products with varying prices, features and options available. They are also being widely used by people.

However, there are still many attacks happening and the number is increasing day by day. If we take a look at some of the top data breach attacks of 21^{st} century [47] more than 70% of the victim companies are not core technical companies and are just using technology as a tool for their business. Attacks like Wanna Cry ransomeware [49], which are of much greater impact and severity as compared to the *Morris Worm*, are occurring. In many cases, we are not even able to track the attacker, a conviction is out of the ques-Very few digital products (excluding tion. those which are specifically designed for security) highlight security features in the product description. The following could be the potential reasons behind the above problems that either leadership of the non-core tech companies and other regulatory authorities are not aware of or are not able to judge or willfully ignore to acknowledge the severity of risks involved in the computer security. Legal policies are not strong enough or detailed and precise enough to capture the computer security requirements [16, 11]. Another posign of ICN and NDN [48, 5] are great ex-tential reason could be that people are not involved with the digital world as it is still very new. People do not use security features as one of the selection criteria for digital products. Or maybe agencies still believe and agree with the statement that "It may be more expensive to prevent such attacks than it is to clean up after them" [41]. When the worm incident happened most of the people using computers and networks were professionals and had significant knowledge of design and working of the system. However, today this not at all the case computers are just like any other equipment and many people with little, if any, knowledge about how system works are using it. There has been no significant step taken yet to educate such people. Schools and other public and private institutions do arrange time-to-time fire drills, and drills to handle other disaster events. However, I am not aware of or have heard of computer security drills. Even many Computer Science departments do not have a single mandatory course on computer security [1]. However there are a large number of vacancies for cybersecurity experts, as per [29] every year in the United States., 40,000 jobs for information security analysts go unfilled, and institutions are struggling to fill 200,000 other cyber-security related positions. I am

able to correctly sense and measure the risk not sure about lack of awareness but there involved with the digital world as it is still definitely is lack of attention to computer severy new. People do not use security feacurity. Computer security is not getting the tures as one of the selection criteria for digi-importance it should.

5 What changed?

Before the incident the Internet was a closed community, and ethical and responsible behavior and good intent were assumed. The worm episode changed people's behavior and attitude towards computer security. Various actions were taken by different organizations [10, 6], government agencies, and universities, and of course, an event with such huge impact will definitely bring some changes in practice and perspective of people. This section discusses these changes following the event.

5.1 Changes in perspective

Post worm people had this sense that the Internet is no longer a closed community. As published in an interview by Intel news room [32] – The day after the worm an awful lot of people were shocked that such abuse could happen. This is evident from the following statement in [12] "A community of scholars should not have to build walls as high as the sky to protect reasonable expectation of privacy, particularly when such walls

tion." The statement clearly shows the dis- work protection. appointment of the community on observing In a report to the Chairman of Subcomthe breach of trust. This incident made a mittee on Telecommunications and Finance, lot of people outside the academic commu- Committee on Energy and Commerce, House nity aware of the fact that malicious software of Representatives US General Accounting could be written. From being a closed and Office recommended formation of an interatrusted community of researchers the Intergency group including agencies funding renet started becoming a large community, and search networks on the Internet, under the it had to accept uncontrollable sociopaths as coordination of President's Science Advisor, it members. Computer science departments with following goals (extracted from the reall across the world started defining the apport [6]). propriate and inappropriate usage of the Internet resources.

5.2 Changes in system

This subsection discusses more concrete and administrative changes and other objective changes made in response to the episode. There were many administrative actions suggested and taken [10, 6, 38]. Like the formation of Computer Emergency Response Team(CERT) at Carnegie Mellon University with funding support from DARPA. CERT was formed by organizing computer scientists with aim of isolating such problems and preventing them from happening in the future. Though the firewall technology already existed before the event, it saw a surge after the disturbing event [37]. Some people at DEC

will equally impede the free flow of informa- started putting an effort into corporate net-

- 1. provide Internet-wide security policy, direction and coordination.
- 2. support ongoing efforts to enhance Internet security.
- 3. obtain the involvement of Internet users, software vendors, technical advisory groups, and federal agencies regarding security issues.
- 4. become an integral part of the structure that emerges to manage the National Research Network.

Cost **Analysis:** 6 and now

This section subjectively discusses the direct or indirect costs incurred due to the worm and what it would cost for such an event in today's scenario. With approximately only 6000 (10% of total machines) [12] machines being affected and the size of the network being only 60,000 machines, use of computers was esoteric and primarily for research, with high chances of the attacker not having criminal intent the losses were in millions of dollars. Experts estimated that the per machine loss could be up to \$53,000 [9] and another estimate in [12] said around 6000 machines were infected, so the total loss could be as high as \$318 million.

In contrast today there are 1) billions of digital devices in use and most of them are connected to the Internet, 2) more than 51% of total world population 3) Computers are being used widely ranging trivial day to day work like managing traffic, listening to music, and education to high-risk tasks like medical

then an attack involving 10% of total machines on the Internet. It could simply break traffic in an entire city, state or maybe of whole nation. It could bring down several businesses or can even lead to large number of deaths if medical systems or security systems get into ambit of the attack.

> According to cyber risk modeling from Cyence [2], economic losses from a recent cyber attack, WannaCry ransomeware, [49], could reach \$4 billion. The attack began on 12^{th} May 2017 and impacted just 300,000 devices (less than 0.01% of total digital devices) and 200,000 people (less than .003%) of world population [49]. Even if extrapolate linearly the losses due to an attack involving 10% of total devices on the network could be as high as \$4 trillion.

Summary

This study compels me to agree to the statement (or its variations) by Prof. Spafford that "The only truly secure system is one that is powered off, cast in a block of contreatment, national and internal security. A crete and sealed in a lead-lined room with large number of businesses entirely depend on armed guards — and even then I have my computers and network 4) cyber-attacks be-doubts" [32]. Any new system starting with ing made with criminal intent [49, 47]. It is a closed group will have to deal with secuhard to imagine what will be the impact of rity issues as it expands. This is precisely what the Morris worm showed. An attack wrote the worm to take revenge. More about like this was inevitable, had Morris not done it someone else would have. New upcoming systems are now more cautious about security, ICN is an example. However, that is not enough as discussed in §4.1 we are not yet fully ready to efficiently tackle cyber attacks like [49]. Other than specialized training, there is pressing need to make general masses and policy makers realize the importance of computer and Internet security. Demo citesjajooSLearn, jajooPhilae, jajooGraviton, jajooSLearn, saathTechReport, phdthesis, philaeTechReport, slearnTechReport

Appendix 8

8.1 What is a computer worm?

The concept of self-propagating worm program was first described by John Brunner in his fictional novel The Shockwave Rider [4]. Worm in the novel was used as a tool for taking revenge! [8]. A computer worm is a complete piece of program that replicates itself in order to spread to other machines. Certainly, other than the very weak claim made worms here [50].

8.2 What is a computer virus?

A computer virus is a malicious software. On execution, it replicates itself by "infecting" other programs i.e. by inserting its own code into other computer programs. fected programs can include data files or system programs or even boot sector of the hard drive.

8.3 Robert Morris's father was chief for computer security!

Robert Tapan Morris's father, Robert Morris, was a computer scientist at Bell labs and helped in designing Multics and Unix and later he became chief scientist at National Computer Security Center, a division of the NSA. Some people claim [3] that Junior Robert was trying to get away from his father's image and have one of his own.

8.4 gets cannot be fixed.

We can say that functions implementing APIs like *qets* will be insecure-by-design. in [45] that warm might have been written these function will only get a pointer to a to take subconscious revenge on his father. I buffer (char*) and not the size of the buffer didn't find any source claiming that Morris and since there is no implicit size bound to a buffer in C, it will be impossible for the function to bound the size of data being read. This point is very clearly explained in the linux man page of gets [14].

Never use gets(). Because it is impossible to tell without knowing the data in advance how many characters gets() will read, and because gets() will continue to store characters past the end of the buffer, it is extremely dangerous to use. It has been used to break computer security. Use fgets() instead.

References

- [1] Bachelor of Science Degree Requirements. https://www.cs.purdue.edu/undergraduate/curriculum/bachelor.html.
- [2] Jonathan Berr. "WannaCry" ransomware attack losses could reach dollar 4 billion. https://www.cbsnews.com/news/wannacry-ransomware-attacks-wannacry-virus-losses/.

 May 2017.
- [3] Larry Boettger. The Morris Worm:

 How it Affected Computer Security and

- Lessons Learned by it. https://www.giac.org/paper/gsec/405/morris-worm-affected-computer-security-lessons-learned/100954. 2000.
- [4] John Brunner. The Shockwave Rider. 1975.
- [5] Jeff Burke et al. "Securing instrumented environments over content-centric networking: the case of lighting control and NDN". In: Computer Communications Workshops (INFO-COM WKSHPS), 2013 IEEE Conference on. IEEE. 2013, pp. 394–398.
- [6] Ralph V. Carlone. Computer Security: Virus highlights Need for Improves Internet Management. Tech. rep. United States General Accounting Office, Washington, D.C. 20548, 1989.
- [7] Google Cloud. Google's Security Culture. https://gsuite.google.com/learn-more/security/security-whitepaper/page-2.html.
- [8] Computer Worm History. https:// en.wikipedia.org/wiki/Computer_ worm\#History.
- [9] Computing's 11 Smartest Super-Viruses-And The Damage They Wrought. https://www.fastcompany. com / 3015224 / computings - 11 -

- smartest-super-viruses-and-the-damage-they-wrought. Feb. 2013.
- [10] Peter J. Denning. The Internet Worm. Tech. rep. Research Institute for Advanced Computer Science, NASA Ames Research Center, 1989.
- [11] Myriam Dunn Cavelty. "From Cyber-Bombs to Political Fallout: Threat Representations with an Impact in the Cyber-Security Discourse". In: International Studies Review 15.1 (2013), pp. 105-122. DOI: 10.1111/misr. 12023. eprint: /oup / backfile / content_public/journal/isr/15/1/10.1111/misr.12023/2/15-1-105.pdf. URL: +http://dx.doi.org/10.1111/misr.12023.
- [12] T. Eisenberg et al. "The Cornell Commission: On Morris and the Worm".
 In: Commun. ACM 32.6 (June 1989),
 pp. 706-709. ISSN: 0001-0782. DOI: 10.
 1145/63526.63530. URL: http://doi.acm.org/10.1145/63526.63530.
- [13] Eugene Howard Spafford Wikipedia page. https://en.wikipedia.org/wiki/Gene_Spafford.
- [14] gets(3) Linux man page. https://linux.die.net/man/3/gets.

- [15] Global Internet Usage. https://en.wikipedia.org/wiki/Global_ Internet_usage.
- [16] Lene Hansen and Helen Nissenbaum.

 "Digital disaster, cyber security, and the Copenhagen School". In: International studies quarterly 53.4 (2009), pp. 1155–1175.
- [17] Herald-Journal. http://media. syracuse.com/vintage/other/2016/ 01/20/Testifies\%20merged.pdf. 1990.
- [18] Akshay Jajoo. A study on the Morris Worm. Tech. rep. 2018.
- [19] Akshay Jajoo. "EXPLOITING THE SPATIAL DIMENSION OF BIG DATA JOBS FOR EFFICIENT CLUSTER JOB SCHEDULING". PhD thesis. Purdue University Graduate School, 2020.
- [20] Akshay Jajoo. "Term Paper on Morris Worm". In: (2017).
- [21] Akshay Jajoo. "Term Paper on Morris Worm". In: (2017).
- [22] Akshay Jajoo, Rohan Gandhi, and Y. Charlie Hu. "Graviton: Twisting Space and Time to Speed-up CoFlows". In: 8th USENIX Workshop on Hot Top-

- ics in Cloud Computing (HotCloud 16).

 Denver, CO: USENIX, 2016.
- [23] Akshay Jajoo, Y. Charlie Hu, and Xiaojun Lin. "A Case for Sampling Based Learning Techniques in Coflow Scheduling". In: CoRR abs/2108.11255 (2021). http://arxiv.org/abs/2108.11255. arXiv: 2108.11255.
- [24] Akshay Jajoo, Y. Charlie Hu, and Xiaojun Lin. "Your Coflow has Many Flows: Sampling them for Fun and Speed".
 In: 2019 USENIX Annual Technical Conference (USENIX ATC 19). Renton, WA: USENIX Association, 2019, pp. 833–848. ISBN: 978-1-939133-03-8.
- [25] Akshay Jajoo et al. "A Case for Task Sampling based Learning for Cluster Job Scheduling". In: Proceedings of the 19th USENIX Conference on Networked Systems Design and Implementation. NSDI'22. Renton, WA, USA: USENIX Association, 2022.
- [26] Akshay Jajoo et al. "Saath: Speeding Up CoFlows by Exploiting the Spatial Dimension". In: Proceedings of the 13th International Conference on Emerging Networking Experiments and Technologies. CoNEXT '17. Incheon, Republic of Korea: ACM, 2017, pp. 439—

- 450. ISBN: 978-1-4503-5422-6. DOI: 10. 1145/3143361.3143364.
- [27] Akshay Jajoo et al. "Saath: Speeding up CoFlows by Exploiting the Spatial Dimension". In: CoRR abs/2111.08572 (2021). arXiv: 2111.08572.
- [28] Akshay Jajoo et al. "The Case for Task Sampling based Learning for Cluster Job Scheduling". In: CoRR abs/2108.10464 (2021). http://arxiv.org/abs/2108.10464. arXiv: 2108.10464.
- [29] Jeff Kauflin. The Fast-Growing Job
 With A Huge Skills Gap: Cyber Security. https://www.forbes.com/
 sites/jeffkauflin/2017/03/16/
 the-fast-growing-job-with-ahuge-skills-gap-cyber-security.
 Mar. 2017.
- [30] Keith Bostic Wikipedia page. https: //en.wikipedia.org/wiki/Keith_ Bostic.
- [31] Timothy B. Lee. How a grad student trying to build the first botnet brought the internet to its knee. https://www.washingtonpost.com/news/the-switch/wp/2013/11/01/how-a-grad-student-trying-to-build-

- the-first-botnet-brought-theinternet-to-its-knees/. 2000.
- [32] Lessons from the 1st major computer virus. https://newsroom.intel.

 com/editorials/lessons-fromthe-first-computer-virus-themorris-worm/. 2013.
- [33] Robert Morris and Ken Thompson.

 "Password Security: A Case History".

 In: Commun. ACM 22.11 (Nov. 1979),

 pp. 594-597. ISSN: 0001-0782. DOI: 10.

 1145/359168.359172. URL: http://

 doi.acm.org/10.1145/359168.

 359172.
- [34] Morris worm. https://en.wikipedia.org/wiki/Morris_worm.
- [35] Morris Worm Turned on Cyber Security 25 years Ago. http://www.njvc.com/blogs/morris-worm-turned-cyber-security-25-years-ago. 2013.
- [36] Morris Worm Turns 25. https://www.kaspersky.com/blog/morris-worm-turns-25/3065/. 2003.
- [37] Hilarie Orman. "The Morris worm: A fifteen-year perspective". In: *IEEE Security & Privacy* 99.5 (2003), pp. 35–43.

- [38] Proceedings of the virus post-mortem meeting. Ft. George Meade, MD, Nov. 1988.
- [39] Robert Morris Homepage. https://pdos.csail.mit.edu/archive/rtm/.
- [40] Robert Morris Wikipedia page. https: //en.wikipedia.org/wiki/Robert_ Tappan_Morris.
- [41] Jon A. Rochlis and Mark W. Eichin. "With Microscope and Tweezers: The Worm from MIT's Perspective". In: Commun. ACM 32.6 (June 1989), pp. 689-698. ISSN: 0001-0782. DOI: 10. 1145/63526.63528. URL: http://doi.acm.org/10.1145/63526.63528.
- [42] Donn Seeley. A tour of the worm. http:
 // www . cs . unc . edu / ~jeffay /
 courses / nidsS05 / attacks / seely RTMworm-89.html.
- [43] Donn Seeley. A tour of the worm. Tech. rep. 1989.
- [44] E. H. Spafford. "Crisis and Aftermath".
 In: Commun. ACM 32.6 (June 1989),
 pp. 678–687. ISSN: 0001-0782. DOI: 10.
 1145/63526.63527. URL: http://doi.acm.org/10.1145/63526.63527.
- [45] Eugene H Spafford. "A Failure to Learn from the Past". In: Computer Security Applications Conference, 2003.

- *Proceedings. 19th Annual.* IEEE. 2003, pp. 217–231.
- [46] Eugene H. Spafford. The Internet Worm Program: An analysis. Tech. rep. Purdue University West Lafayette IN USA, 1988.
- [47] The 16 biggest data breaches of the 21st century. https://www.csoonline.com/article/2130877/data-breach/the-16-biggest-data-breaches-of-the-21st-century.html. 2017.
- [48] Reza Tourani et al. "Security, privacy, and access control in information-

- centric networking: A survey". In: *IEEE Communications Surveys & Tutorials* (2017).
- [49] Wanna Cry ransomeware attack.

 https://en.wikipedia.org/wiki/
 WannaCry_ransomware_attack.
- [50] Nicholas Weaver et al. "A Taxonomy of Computer Worms". In: Proceedings of the 2003 ACM Workshop on Rapid Malcode. WORM '03. Washington, DC, USA: ACM, 2003, pp. 11–18. ISBN: 1-58113-785-0. DOI: 10.1145/948187.
 948190. URL: http://doi.acm.org/10.1145/948187.948190.