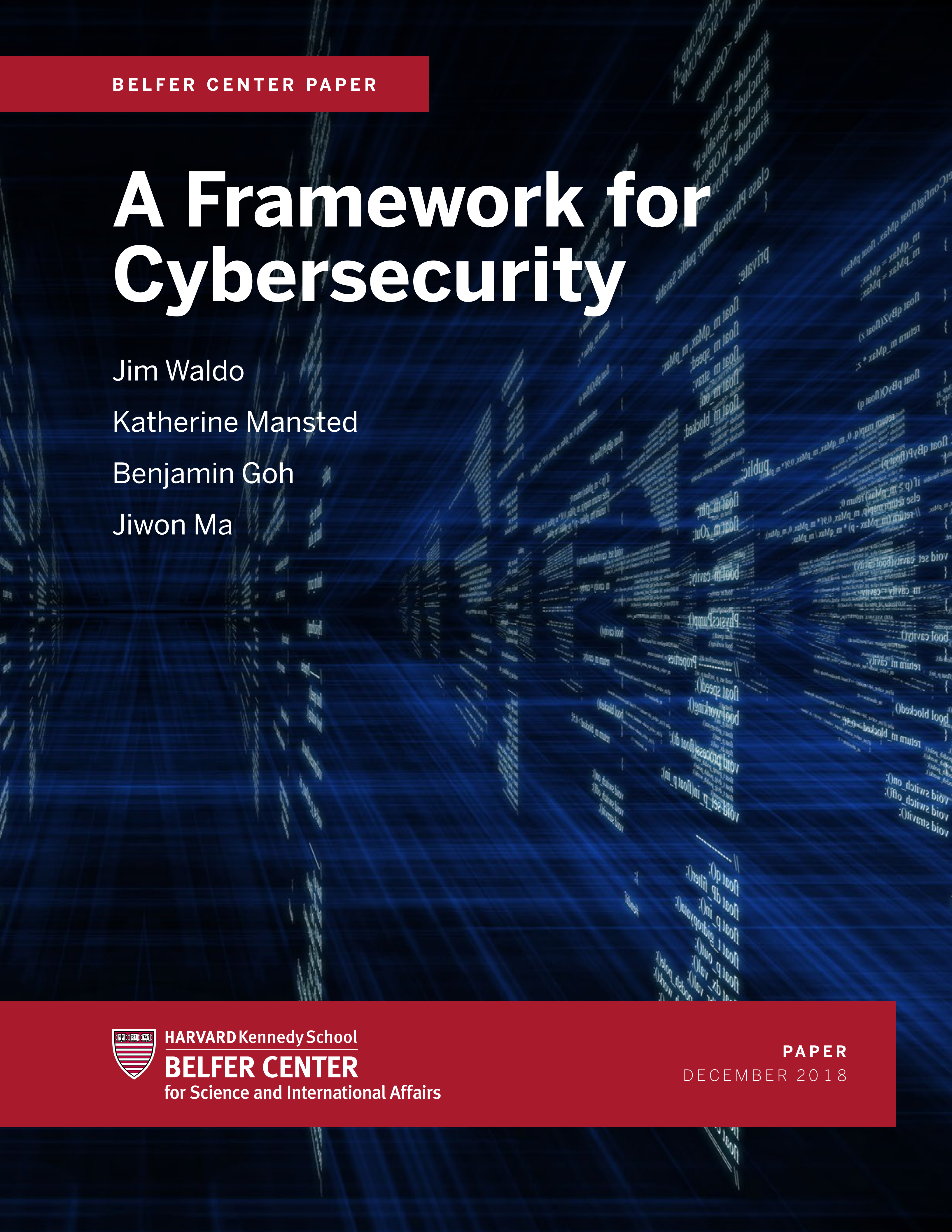


BELFER CENTER PAPER

A Framework for Cybersecurity

Jim Waldo

Katherine Mansted

Benjamin Goh

Jiwon Ma

HARVARD Kennedy School
BELFER CENTER
for Science and International Affairs

PAPER
DECEMBER 2018

Belfer Center for Science and International Affairs
Harvard Kennedy School
79 JFK Street
Cambridge, MA 02138

www.belfercenter.org

This work was made possible by a gift from Microsoft. Statements and views are those of the authors and do not imply endorsement by Microsoft, Harvard University, the Harvard Kennedy School, or the Belfer Center for Science and International Affairs

Layout by Andrew Facini

Cover photo: Adobe Stock.

BELFER CENTER PAPER

A Framework for Cybersecurity

Jim Waldo

Katherine Mansted

Benjamin Goh

Jiwon Ma

HARVARD Kennedy School
BELFER CENTER
for Science and International Affairs

PAPER
DECEMBER 2018

About the Authors

Jim Waldo is a Gordon McKay Professor of the Practice of Computer Science and the Chief Technology Officer of the John A. Paulson School of Engineering and Applied Sciences at Harvard University, and a Professor of Technology Policy at the Harvard Kennedy School. He teaches and does research in areas that are at the interface between technology, policy, the law, and ethics. He holds a Ph.D. in philosophy from the University of Massachusetts, Amherst.

Katherine Mansted is a Senior Research Officer at the Australian National University's National Security College, and a Nonresident Fellow at the Belfer Center. Her research focuses on emerging technologies and national security. Previously, Katherine practiced law as a commercial solicitor and served as a ministerial adviser in the Australian Government. She holds a Master of Public Policy from the Harvard Kennedy School.

Jiwon Ma is an Executive Assistant at the Harvard John A. Paulson School of Engineering and Applied Sciences (SEAS). Prior to joining SEAS, Jiwon interned for the Principal of Small Planet Institute, researching Agricultural Economic Policy and managing an educational outreach program. Previously, she was as an education intern at the United Nations Association of Greater Boston, where her research primarily focused on International Human Rights Policy. She graduated from Lesley University in 2014 with a B.A. in Global Studies and a minor in Psychology.

Benjamin Goh was a Master in Public Policy student from the Harvard Kennedy School. He is fascinated by the way politics play out on the Internet, and spent his time at the Kennedy School researching on cybersecurity tactics and paradigms employed by governments, the military, and the private sector. He is co-author of the IEEE paper "A Principles-Based Approach to Govern the IoT Ecosystem", which lays out principles for societies to construct their IoT governance systems. His thesis advisor was Prof. Sheila Jasanoff, Pforzheimer Professor of Science and Technology Studies at the Kennedy School of Government.

Table of Contents

Introduction 1

The two dimensions of security threats 3

Targets of attack 3

Thread of control 5

Obtaining a thread of control 7

Thread of control repurposed from its intended use 7

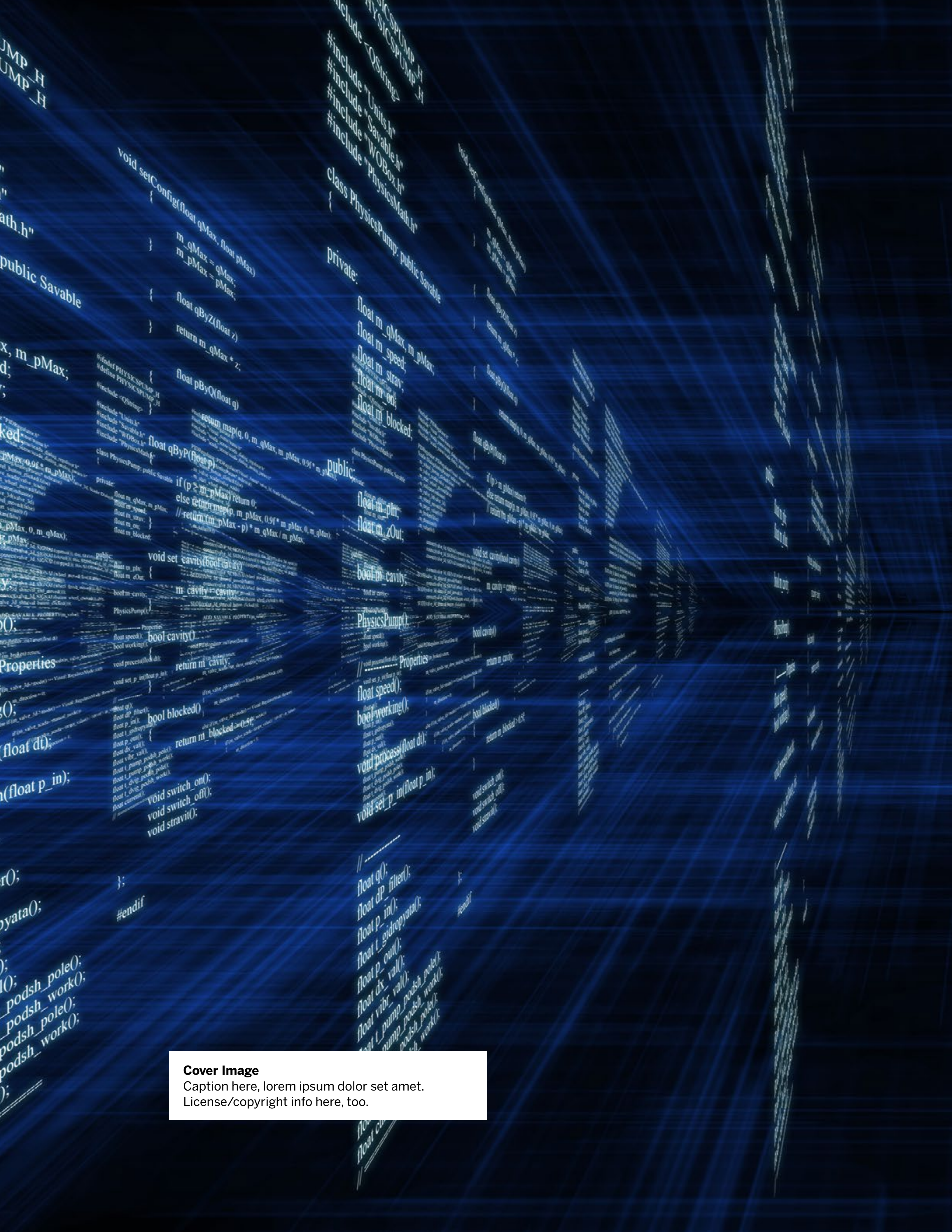
Thread of control obtained by standard means 9

Thread of control obtained by extraordinary means 10

A taxonomy of security threats 12

Implications of the taxonomy 13

Glossary of attacks 17



Cover Image
Caption here, lorem ipsum dolor set amet.
License/copyright info here, too.

Introduction

One barrier to creating a systematic defense for a networked computing system is the broad range of ways in which adversaries attack such systems. While denial of service, phishing, and exploitation of zero-day vulnerabilities are all described as hacks or attacks, they exploit very different parts of a system, and defending against one does not connect to defending against another. Without a unified way of thinking about these attacks, defense becomes ad-hoc and reactive—doomed to playing catch-up to a constantly evolving, and unpredictable offense. There is a need for a framework which allows network defenders to systemize their thinking about what their adversaries will do and to proactively counter attacks before they happen.

Without such framework, defense can be a never-ending series of reactions to the latest attack. Successful ways of dealing with denial of service attacks appear to be unrelated to attempts to thwart phishing probes, both of which are completely different to keeping viruses and worms out of the network. While there are important differences in these various forms of attack, once we see the similarities in them, we can start thinking systematically about how to build a defense that is more unified and flexible than a collection of tools to defend against isolated attacks.

In this paper, we propose a way of thinking about cybersecurity that unifies the various forms of attack. The framework is two-dimensional, looking at both the goal of the attack and the mechanism for launching the attack. The first dimension looks at the goal of the attack by using the common "CIA" triad to categorize the target—that is, whether the attack affects a system's ***confidentiality***, ***integrity***, or ***availability*** (CIA). The second dimension is unique to our knowledge and differentiates attacks based on how the attacks obtain a thread of control. A thread of control is the mechanism that allows work to be done by a processor. It contains such information as what code is to run, the access privileges of that code, and what code is to run next. We show how existing known attacks can be categorized by how they obtain a thread of control. Some utilize an existing thread of control, while others create new threads of control by standard means. A third set of attacks exploit vulnerabilities in the system to create a new thread of control.

This framework allows thinking about the underlying mechanisms of attacks against computer networks, what they have in common, and how efforts to counter one sort of attack could be re-used or adapted to support efforts we have already expended on countering others. Concentrating on methods of control also redirects our attention concerning what should be monitored in a network, allowing defenders to better prioritize resources, and to more quickly and accurately detect attacks. In addition, we believe that this framework allows thinking about attacks that have not yet occurred, helping defenders to proactively detect those attacks and, perhaps, prevent them before they have been successful somewhere else. This, we hope, might help free defenders from the tyranny of always reacting to one attack after another, and may even allow defenders to proactively put into place defenses against attacks before they happen.

The two dimensions of security threats

Targets of attack

We begin our framework by thinking about the targets or effects of an attack against a networked system. These are traditionally categorized around the properties of confidentiality, integrity, and availability (CIA), a tradition we will adopt as well. Confidentiality ensures that the information stored in the system is only available to those who have been granted access to that information; compromises of confidentiality allow an attacker to access information they should not be able to see. Integrity means that the information in the system has not been changed or tampered with by those who are not allowed to make such changes. Availability means that the system and information within it is accessible to authorized users when they need it.

Violations of ***confidentiality*** occur when data can be read, copied, or exfiltrated by an adversary. Many of the attacks that have gained some measure of infamy over the past few years have been violations of confidentiality, from the Equifax exfiltration of millions of financial and identity records to the leaking and subsequent publication of emails from the Hillary Clinton presidential campaign. Attacks on confidentiality may not result in general leaking of information; the exfiltration of government records from the United States Office of Personnel Management was a confidentiality violation, even if none of those records ever make it into the public sphere.

An attack on ***integrity*** changes the data within the system, either making that data unreliable or inconsistent. A portion of the Stuxnet virus, which was used to attack Iranian nuclear centrifuges in 2010, was an integrity attack. Once that malware had gained access to the Natanz nuclear facility's industrial controllers, the injected code would misrepresent the speed of the centrifuges to other instruments in the plant, making it difficult or impossible to know the true speed of the centrifuges. (While the instruments could not see if anything was amiss, another part of the Stuxnet code controlled the spin rate of the centrifuges to damage them.)

Attacks on ***availability*** make it difficult or impossible for the system to be used, such as the “CryptoLocker” ransomware attacks that encrypted the contents of disks until a ransom was paid. The 2016 denial of service attack against the Dyn domain name server was also an accessibility attack, flooding Dyn’s servers with so much network traffic that it became unusable. The Shamoon attack that destroyed thousands of machines at Saudi Aramco was also, at base, an availability attack. By removing all data from the disks on the infected computers, those computers were rendered useless—the ultimate form of unavailability.

All kinds of attacks are, at base, attacks against one or more of these three properties of a system. Thinking in terms of which of these properties the attacker is attempting to compromise allows us to unify seemingly different attacks in terms of their end-goal, and also helps when we think of how to go about defending against the attack.

While it is important to understand which properties of a system are being targeted, it is also important to understand how the adversary is mounting the attack. Trojans, worms, phishing, or denial of service attacks are all attempts at compromising the confidentiality, integrity, or availability of a system, but are very different in the approach they take to attain that end-goal. Defending against one attack vector may have no effect on a different attack vector, even when both attacks have the same outcome. A useful taxonomy therefore requires a second dimension; one that allows us to talk about how an attack is being launched in addition to what the target of the attack is.

This second dimension rests on the observation that, in order to compromise a computer system, the attacker will need to perform some action at the attacker’s direction. Doing something on a system requires what is called a **thread of control** on that system. Thus, the second dimension in our taxonomy concerns how the attacker is able to obtain or exploit a thread of control, which in turn allows the attacker to have the target system do something on the attacker’s behalf.

Thread of control

Modern computers work at unimaginable speeds. Even the most common of laptops or cell phones contain a processor that is capable of executing billions of instructions a second. At any time, a computer will be executing tens or hundreds of different programs. But all of these programs are actually taking turns, with only one at a time being run on the processor[1]. The operating system, the software underlying everything that gets done on the computer, arranges things so that it can appear that any computer is doing many things at once. The key to understanding how that is done is the thread of control.

A thread of control is the basic abstraction in an operating system that controls the activities performed by a computer. An active thread of control keeps track of what instructions to run next on the processor, and contains memory and other resources that allow those instructions to do the computing that is required. When the time allotted to the thread is finished, control of the processor is given to another thread, determined by the scheduler in the operating system. The thread of control that had been running is stored with the information that allows it to begin where it had left off when it is time to run that thread again.

Each thread of control contains information that is used by the computer to run the sequence of instructions associated with a particular thread. We have already mentioned that a thread of control has a section of memory that can be used by the instructions in the thread. A thread of control may also contain one or more network connections, or the ability to read or write files on the local machine. A thread of control also has an identity that determines the privileges of that thread of control. The identity of the thread can be derived from the user associated with the thread, the program being run by the thread, or the operating system itself. These rights determine the data that the thread of control can access and which actions the thread of control is permitted to perform.

1 Most modern computers contain multiple cores, which are each capable of running a sequence of instructions. However, the number of programs running at any one time will almost always exceed the number of cores. For example, at the time this was written, the computer on which it was being written had four cores but was running 230 different programs.

Threads of control can be created in many ways. When a computer is turned on or booted, a thread of control is created to load the operating system. This thread of control has the identity of the system itself and can carry out any function. As that operating system is loaded, the original thread of control creates new threads of control to enable the operating system to interact with peripherals, such as interacting with the mouse and display, reading attached storage, and reading from and writing to a network. When a user logs in to a computer, a thread of control is created for the user with the user's specific identity. When the user starts a program (by, say, double-clicking on an icon) a thread of control is created to run the program with the identity of the user, the program, or some combination of the two.

When a new thread of control is created, the thread gets its identity and the access rights associated with this specific identity from the thread of control that created it. The specification of the identity depends on the activities that the thread of control is supposed to execute. A thread that is being used by the operating system may have rights to do anything and access all information. A thread of control that is started when a user logs in is assigned the rights associated with that user, which means that it has a restricted set of data that can be accessed and a more limited set of activities that it can do. A thread of control started when a user begins a program may have even fewer rights than the user, or may be a trusted program that has more rights than the user that initiated the program.

All attacks on a computing system require that the attacker have a thread of control. Without a thread of control, nothing happens on the system. Malware will just be bytes sent to the machine, but cannot do anything. A phishing email will simply appear on the screen, but will not be able to request credentials or send the user to a web page. This need to obtain a thread of control unifies all of the kinds of attacks that we have discussed; how these attacks differ is the way in which they obtain the thread of control.

Obtaining a thread of control

Once we understand the need for an attacker to obtain a thread of control to do anything inside a system, we can ask how the attacker obtained that thread of control. There are three ways in which an attacker can obtain a thread of control:

- The attacker may exploit an existing thread of control by repurposing the thread of control from its intended use.
- The attacker may obtain the ability to create a thread of control by standard means.
- Finally, an attacker may exploit some flaw in a system to obtain a thread of control by extra-ordinary means.

We will now turn our attention to each of these.

Thread of control repurposed from its intended use

The first sort of exploit used by attackers is to use an existing thread of control. That is, the attacker uses a thread of control that they are authorized to use, but instead of using it in the way intended, they use it to advance some malicious purpose.

The clearest example of such an exploit is the denial of service attack. Here, the attacker exploits an existing thread of control on a system that is listening for messages over the network. In normal operation, these threads take up little to no time on a system. But when the number of requests coming in from the network increases, this thread takes up more and more of the time on the processor. A denial of service attack sends so many requests, either from a single system or (more likely) from a distributed group of multiple systems, that all of the time available on the processor is used up by the thread listening to the network. The combination of the existing thread of control and the surge of traffic to be handled results in the system no longer being available for the users.

The attack against Dyn, a Domain Name System (DNS) provider on October 21, 2016[2] was a distributed denial of service attack. DNS is the Internet equivalent of a phonebook—it connects the website name typed in by a user to its associated IP address, allowing users to access the website. The attackers, using a set of compromised machines, overwhelmed Dyn's DNS servers with floods of DNS lookup requests. The number of requests was huge; it is estimated that requests—each of which could have been legitimate—were coming in from tens of millions of IP addresses. Dyn's servers were spending so much time responding to requests that there was no time for any other thread of control, making them effectively unavailable to legitimate users. As a consequence of the attack, many major websites across the United States went "down" for several hours.

Not all attacks that exploit an existing thread of control target availability. The Heartbleed vulnerability[3], first publicly revealed in 2014, used an existing thread to compromise the confidentiality of systems. This exploit used a bug in the OpenSSL cryptographic library—a commonly used suite of open-source software that secures browser connections by encrypting the traffic between the browser and a web server. As part of this functionality, OpenSSL requires the machines connecting to each other to do an "are you there" check in the Transport Layer Security (TLS) protocol. A client machine would ask a server to return a value sent by the client to the server to prove that the server was still responding. The client would also specify the number of bytes to be returned by the server. However, the server would not check that the size of the requested return was the same as the size of the value originally supplied by the client. An attacker acting as a client could therefore specify a much larger number of bytes be returned than what was needed. The server would then return whatever was in the memory after the value supplied, leaking information that was stored there. An attacker did not need to create a thread of control to get this information; they instead used an existing thread of control that was granted to them to do the "are you there check" to breach the confidentiality of the server.

2 See https://dyn.com/blog/dyn-statement-on-10212016-ddos-attack/ for details of this attack, or see the glossary at the end of this paper.

3 See http://heartbleed.com/ for details of this vulnerability.

Indeed, one of the most common forms of confidentiality leak on the World Wide Web is the SQL injection attack[4], in which an attacker places Standard Query Language (SQL) commands as input to some part of the web server in such a way that the web server receiving the command executes not only the intended SQL query, but other SQL code that can expose parts of the database used by the web server. This form of attack uses an existing thread of control that is open to a set of users, including the attacker, but in a way designed to furnish the objectives of the attacker by exposing information in the database. For example, it can be used to leak information such as account names and passwords. This was the first part of the campaign run by vigilante activists Anonymous against the cybersecurity firm H.B. Gary Federal[5], which led the firm into bankruptcy. It should be noted that this form of attack could also be used to change existing data (and thus, be an integrity attack), or to delete the data entirely (and thus, be an availability attack).

Thread of control obtained by standard means

A second mechanism used by attackers is to gain a thread of control using one of the existing mechanisms for creating such threads. Attempts to get the login credentials of a system user, whether by compromising a password file or phishing, are really attempts at getting credentials so that the attacker can log in to the system in order to create the thread of control that such a login provides. The phishing attack is just a means to an end. When an attacker obtains the credentials of an authorized user of the system, the attacker can log in as this user—enabling the attacker to gain a thread of control with the rights given to the intended user.

Perhaps the most infamous attack of such a kind was the phishing attack that allowed the attacker to obtain the login credentials of John Podesta, Chairman of Hillary Clinton's 2016 presidential campaign, leading to the leaking of over 60,000 of his personal emails to WikiLeaks.[6]

4 See https://www.incapsula.com/web-application-security/sql-injection.html for details of such attacks.

5 See https://arstechnica.com/tech-policy/2011/02/anonymous-speaks-the-inside-story-of-the-hb-gary-hack/

6 See https://www.apnews.com/dea73efc01594839957c3c9a6c962b8a

Similarly, many attacks embed malware within an email attachment. Such malware cannot do anything without obtaining a thread of control. However, most email programs allow the user to double-click on an attachment in order to create a thread of control that is used to launch the application that is used to display or work with the attachment. This is convenient for users when the attachment is a document or a spreadsheet. But if the attachment contains malware, the thread of control, with the rights and privileges of the recipient, will become available to the malware.

Often threads of control obtained in this fashion are used to compromise the confidentiality of the system that is breached. However, there is an increasing number of cases in which this mechanism has been used to compromise the availability of systems, either after the data has been exfiltrated by the attacker or as the primary goal of the attack. Ransomware, such as CryptoLocker[7] and WannaCry[8], obtains the thread of control needed to encrypt the victim's information by getting the victim to click on an email attachment that contains the malware. Once the thread of control has been obtained, the data on the computer (and any data that the computer can access on other computers) is encrypted and becomes unavailable until a ransom is paid. Even Stuxnet, the malware used to attack Iranian nuclear centrifuges, had some code that used this approach— Stuxnet initially spread itself by using the thread created when a USB-device was plugged into a computer.

Thread of control obtained by extraordinary means

A final mechanism that can be used by an attacker to obtain a thread of control is by exploiting a vulnerability within the target system itself. Bugs within the target system may allow the attacker to take control of an existing thread, or create a new thread in a way that was never intended by the system.

An example of such attack is The Morris Worm—the first worm to spread over the Internet. The Morris Worm[9] exploited a vulnerability in the email

7 See https://usa.kaspersky.com/resource-center/definitions/cryptolocker

8 See https://www.symantec.com/security-center/writeup/2017-051310-3522-99

9 See https://limn.it/articles/the-morris-worm/

program used on the target computers. The email program did not check the size of the data that was received by the network—allowing a very long message to overflow the area in memory allocated to hold incoming messages. In a sense, the Heartbleed attack is a mirror-image of the Morris Worm. In the case of Heartbleed, the client would request for more data from a buffer than the client had initially supplied; whereas in the case of the Morris Worm, the client provided more data than was allocated by the server. The Morris Worm exploited this vulnerability to write into an area of memory used to indicate which program should run next, replacing the value that had been written into this area by the operating system with a value that would give control to the Morris Worm itself. This meant that when the thread running the mail program was finished, the operating system gave a thread of control to the Morris Worm. The ability to overflow the buffer was a bug in the email system code, unknown before the Morris Worm attack, and led to an attack that shut down a significant portion of the Internet of the time.

The most complex known attack to date, the Stuxnet virus, also used extraordinary means to create the new threads of control needed to compromise the integrity of the targeted Iranian centrifuges. In fact, Stuxnet used several "zero-day exploits"– so named because they exploit bugs unknown to the writers of the software being exploited. The day the exploit is discovered is "day zero" in the race to fix the bug.

Attacks like Stuxnet, which exploit an unknown (or unpatched) bug in a software to give the attacker a thread of control, are perhaps the most frightening since there is little that can be done to counter the attacks until after they happen. These types of attacks are the cyber equivalent of "unknown unknowns" (or, in the case of unpatched bugs, known unknowns where the vulnerability might be known but the solution is either unknown or the impact of applying that solution is unknown). However, these exploits appear to be rare (at least if we determine how common they are by how often we experience attacks based on them), and when found, are often held by attackers in reserve for particularly important or valuable targets.

A taxonomy of security threats

Some of the best-known attacks can be assessed against each of the two dimensions discussed above, in order to develop a taxonomy of security threats (figure 1).

Note, some of the attacks utilize more than one approach to gain a thread of control. Increasingly, attacks are "modularized" so that different components of the code perform different steps to carry out the attacker's malicious intent. A cyber attack can be broken up into separate stages similar to how a belligerent employs separate but related campaigns or battles to achieve their strategic intent in war. Additionally, note that some of the attacks also had multiple targets. So for example, the attack on Sony Pictures first exfiltrated data from the company, and then made most of the computers on the network unavailable for use.

	Confidentiality	**Integrity**	**Availability**
Existing thread of control	Heartbleed SQL injection		Estonia banking attack DynDNS attack
Threat of control created by extraordinary means	DNC email leak Sony Pictures email leak OPM hack		Sony Pictures computer destruction Saudi Aramco attack CryptoLocker
Thread of control created by vulnerability		Stuxnet	Morris Worm WannaCry

Figure 1. A taxonomy of security threats, categorized by target (columns) and how the thread of control is obtained (rows). For each of the named attacks, refer to the Glossary for a brief description.

The use of multiple mechanisms for a single attack points out a little discussed fact about cybersecurity. What is often discussed is a particular attack, which makes the process sound like it is short-lived, occurs very quickly, and has to be defended against at machine rather than human speeds. While this may be true of some attacks, it is more often true only of the end part of a much longer sequence of events that culminate in what we hear about as the attack.

It is more accurate to think of cyber-security as defending against long-term campaigns where the adversary slowly and methodically builds up information and infrastructure that allows them to compromise the confidentiality, integrity, or availability of a system. Even a supposedly short-term attack, such as the Dyn DNS distributed denial of service attack was preceded by a much longer building of a botnet, that is, a set of machines that were used to make the flood of requests that made the Dyn service unavailable. These machines, which were low-end internet devices like webcams and baby monitors, were compromised by creating a thread of control using a standard means — default administrative passwords. Other well-known incidents, like the exfiltration of data from the United States Office of Personnel Management[10], were preceded by months of the attackers having access to the OPM network through credentials obtained by phishing attacks. With these credentials, the attackers were able to probe the network to find the information of interest, and to exploit other weaknesses to give the attackers enhanced privileges so that they could access that information.

Implications of the taxonomy

The taxonomy presented above simplifies the way of categorizing the attacks that have been seen on computer systems and networks, which by itself would be useful. We can now concentrate the defense on keeping the adversary from misusing an existing thread of control or obtaining the ability to create a thread of control rather than see cyber defense as an endless task of protecting against unrelated attacks that are completely unique.

This taxonomy also allows us to concentrate our efforts in areas that may be most useful.

First, considering the three CIA properties enables an organization to efficiently prioritize its resources, in order to more effectively manage its cyber risk. This is an example of what tools, like the NIST Cybersecurity Framework,[11] already do to some extent: they invite organizations to consider

10 https://www.wired.com/2016/10/inside-cyber attack-shocked-us-government/

11 See https://www.nist.gov/cyberframework

what is most important to them, and to anticipate what an adversary might be most motivated to attack. For example, a political office might be most concerned about confidentiality attacks against emails; while an electricity grid might be most concerned about availability and integrity attacks.

Second, by considering a taxonomy with the dimension of the thread of control that is exploited by the attack, we can begin to see not only the effects of the attack but also the duration of the attack and some mechanisms that might be employed to counter or blunt the attack.

Knowing where in our systems attackers may be listening to the network allows us to monitor such points for excess traffic, and throttle spikes in that traffic to avoid distributed denial of service attacks. This type of monitoring can be put in place before the attack occurs, and can be far more effective than reacting to an attack that is already in progress.

When we combine the taxonomy with the insight that most attacks are the end-point of much longer and more complex cyber campaigns, we can start to think of places where the campaign can be interrupted or disrupted most easily. For example, instead of trying to train our users to distinguish between "good" emails and "bad" emails to avoid phishing attacks that try to harvest their credentials, we can see such attacks as part of a chain that is attempting to gain a thread of control. While we can combat the campaign by trying to train our users, we can also interrupt the attack chain at a later point by, for example, requiring two-factor authentication or running certain emails in a sandbox.

We can also monitor logins closely in much the same way that credit card agencies monitor charges; if we see a login from an unusual location, we can check with the user (by email) to see if the login is valid, or simply deny the login. Such approach leverages the fact that the phishing is just a means to an end; any interruption of the chain that leads to that end is useful.

Perhaps the most valuable aid the taxonomy provides is the ability to think about attacks that might occur in the future. Attackers move to other forms of exploitation as defenses against current forms of attack are strengthened. We can look at those parts of the taxonomy that are not occupied by historic attacks as those areas where adversaries could strike next.

Looking at our taxonomy, it is striking that there are no entries in the categories of confidentiality attacks using unknown exploits to obtain a thread of control, or for integrity attacks using either an existing thread of control or a thread of control that is obtained in a standard way. Because we have not seen attacks of this sort, it is unsurprising that we have few, if any, defenses against such an attack. But this is just the sort of circumstance that will make these attacks attractive to an adversary.

Indeed, in the time between the first population of this taxonomy and the present, reports came out about a data breach from Equifax—one of the largest credit-reporting companies in the world. The breach was staggering in its size, reportedly including social security numbers, credit card numbers, and other financial information on over 145 million people[12]. While the cause of the breach is still unclear, it appears to have been caused by a known but unpatched vulnerability in the Struts framework, an open-source library widely used to create web servers, which Equifax employed in their web page[13]. The attackers used the vulnerability to inject their own code into the Struts framework, which allowed them to run their own programs on a newly-created thread of control. (See figure 2)

	Confidentiality	**Integrity**	**Availability**
Existing thread of control	Heartbleed SQL injection		Estonia banking attack DynDNS attack
Thread of control created by standard means	DNC email leak Sony Pictures email leak OPM hack		Sony Pictures computer destruction Saudi Aramco attack CryptoLlocker
Thread of control created by vulnerability	Equifax data leak	Stuxnet	Morris Worm WannaCry

Figure 2. Updated taxonomy of security threats, post Equifax data breach.

12 See https://www.wsj.com/articles/equifax-hack-might-be-worse-than-you-think-1518191370

13 See https://www.wired.com/story/equifax-breach-no-excuse/

This leaves just two forms of attack that have not been seen (or recognized) in the wild, both of which are attacks against the integrity of information. Such an attack may never occur since the business model for the attacker is unclear. But if an attacker solely wished to cause chaos in an organization, such an attack might well be a way to accomplish this goal. Imagine, as security expert Dan Geer has[14], that a virus was launched inside an organization that silently found all of the spreadsheets within the organization and changed some of the values in some of the cells by some random amount. The results of such a virus could be considerable—calling into question the integrity of any of the spreadsheets used within an organization. A similar attack on the results of an election could be catastrophic, not only for the outcome of the election whose integrity was compromised, but in the trust the electorate has with elections in general[15].

However, there are immediate steps that could be taken to avoid the consequences of such an attack. Automated mechanisms that would cross-check the values in important documents can be developed, run as needed, and flag when the changes in values are not compatible. This would allow an organization to discover such an attack. Developing a backup strategy that would allow uncorrupted files to replace those that had been tampered with would allow recovery once the attack had been discovered. Putting such measures in place before any such attack occurs may incur an expense that will never pay off. But if such an attack does occur, having these measures in place will pay off in a dramatic fashion.

This framework also suggests a new direction for research into defending against cyber attacks. Current defenses attempt to keep the malware out of a computing system by recognizing the malware (anti-virus), blocking network traffic that might contain malware (firewalls), or detecting malware from the network activity generated by that malware (network monitoring). This framework suggests that monitoring the creation of threads of control on individual machines and flagging anomalous thread creation for further investigation may well be a profitable avenue for future research in cyber defense.

14 Private correspondence with the author.

15 See https://www.esecurityplanet.com/hackers/researchers-find-russian-hacker-selling-access-to-u.s.-election-assistance-commission.html

Glossary of attacks

CryptoLocker: A type of availability attack, most often spread through an email attachment. The attachment obtains a thread of control through standard means—once the user clicks on the attachment. Once it has a thread of control, the malware within the attachment encrypts all of the data on the computer being attacked, along with any data it can access on shared storage. Once the encryption is complete, the user is informed that they must pay (often through the transfer of some amount in bitcoin) to obtain the key that will decrypt the data, and that if that payment is not made in a certain amount of time, the decryption key will be deleted.

Dyn DNS DDoS: A particular distributed denial of service attack against a provider of a core Internet service. On Friday 21 October 2016, Internet users on the East Coast of the U.S. experienced something highly unusual: no, or disrupted, access to many of their favorite Internet sites. Subsequent outages affected the U.S. West Coast, and Europe. The outages were caused by a distributed denial of service (DDoS) attack against Dyn, a major Domain Name System provider. The DDoS attack was an availability attack, which exploited the existing thread of control in Dyn's servers that listens for requests for a name-to-IP-address lookup. Demonstrating one of the complexities of defending against an attack that uses an existing thread of control, Dyn struggled to distinguish legitimate requests from malicious requests, as it tried to contain the attack.[16] The Dyn DDoS attack was carried out by around 100,000 consumer-level devices such as baby monitors and DVD players. These devices were in turn compromised by the Mirai virus, which used a built-in vulnerability[17] that allowed the attackers to create a thread of control to administer those devices using existing means.

Equifax data leak: A confidentiality attack initiated by a vulnerability in the Struts web framework that allowed an arbitrary program to be run on the Equifax servers. Once their program was running inside the servers, the attackers were able to search for and exfiltrate millions of records

16 Scott Hilton. October 21, 2016. "Dyn Analysis Summary of Friday October 21 Attack" Dyn Company News https://dyn.com/blog/dyn-analysis-summary-of-friday-october-21-attack/

17 The vulnerability exploited was a hard-wired administrative account name and password that was in the chip-set used for networking these devices.

concerning American, UK, and Canadian consumers, including credit card numbers and social security numbers.

Estonia banking attack: The 2007 distributed denial of service attack on Estonian infrastructure swamped the servers of banks, the Estonian parliament, newspapers and broadcasters. The attack is widely believed to have been state sponsored.[18]

Morris Worm: An availability attack that made use of an unknown vulnerability to create a thread of control that ran the attack's code. Generally regarded as the first Internet worm, this attack made use of a buffer overflow to obtain a thread of control from the operating system. Once the thread of control was obtained, the Morris Worm would send itself to anyone in the contact lists on the infected system, which allowed it to spread over much of the Internet in 1984. So many copies of the worm would be running on a system that the system became unusable, thus turning the worm into an availability attack.

Office of Personnel Management Hack: A confidentiality attack initiated when the attackers obtained credentials through a phishing attack, allowing them to create a thread of control by standard means. Once they gained access to the system, a back door was installed that allowed the attackers to gather and exfiltrate the personnel records of as many as four million U.S. government employees. The attack has unofficially been attributed to the People's Republic of China.

Saudi Aramco: An availability attack that obtained a thread of control by exploiting a vulnerability in the Windows NT kernel, the attack would spread itself to other computers that were on the network of an infected computer, and then over-write the boot sector of the infected computer, making it unusable. It gained initial entry into the network by being placed in an email attachment, that obtained a thread of control when an employee receiving the email double-clicked on the attachment, thus giving the entry vector a thread of control by standard means. The attack disabled more than 30,000 computers.[19]

18 See https://www.wired.com/2007/08/ff-estonia/

19 See https://www.symantec.com/connect/blogs/shamoon-back-dead-and-destructive-ever

Sony Pictures email link: A confidentiality compromise triggered by obtaining login credentials via phishing and gaining a thread of control by a standard mechanism. Attackers were in the Sony Pictures Entertainment network for an unknown period of time, but for at least two months, collecting email and other information. On November 24 of 2014 this information, including highly confidential and embarrassing emails, was leaked, causing considerable embarrassment and unspecified financial losses to the company. The attack, and the subsequent destruction of much of the Sony computing infrastructure discussed below, is believed to have been the work of North Korea, in response to the planned release of the movie *The Interview*.

Sony Pictures computer destruction: Related to the email leak discussed above, this was an availability compromise that was also enabled by obtaining credentials that allowed getting a thread of control by standard means. After the exfiltration of information that was subsequently leaked, the campaign finished by removing all data and programs from a large number of servers and other computers used by Sony Pictures, making their computer infrastructure unusable.[20]

Stuxnet: Stuxnet was the world's first confirmed use of a purely digital tool to achieve destruction of physical equipment.[21] While it has never been officially acknowledged, the authorship of the malware is generally ascribed to the United States and Israel. The technical view of Stuxnet separates the malware into two distinct sections.[22] One section of the code enabled the malware to move from computer to computer, attempting to find computers that might be physically carried into the Natanz nuclear facility that was the ultimate target. This code, which relied on up to two zero-day vulnerabilities in the Microsoft Windows operating system, utilized networks, thumb drives, shared disk stores, and even network printers as a mechanism to infect any computer running the Windows operating system with the malware. All of the attacks used a standard mechanism to obtain a thread of control for the exploit. The malware would then attempt to use

20 See https://www.wired.com/2016/02/sony-hackers-causing-mayhem-years-hit-company/

21 https://www.wired.com/2014/11/countdown-to-zero-day-stuxnet/

22 Much of the technical analysis of Stuxnet is taken from Nicolas Falliere, Lian O Murchu, and Eric Chien, W.32 Stuxnet Dossier, Version 1.4 (February 2011), Semantec Security Response, https://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/w32_stuxnet_dossier.pdf

two additional zero-day vulnerabilities found in the Microsoft operating system in an attempt to escalate privileges of the code to allow the malware to insert code into the "Step 7" system used to program a particular set of Siemens programmable logic controllers (PLC). When the infected system was attached to such controllers, the malware injected code into the PLCs that sped up and slowed down the centrifuges that were controlled by the PLC, while failing to alert any monitoring software of the disruption. It is estimated that the Stuxnet malware temporarily disrupted the operation of up to one fifth of centrifuges at the Natanz nuclear facility.[23]

WannaCry: An availability attack that obtained a thread of control by exploiting a vulnerability in the Microsoft Windows operating system. In 2017, more than 200,000 computers in some 150 countries were hit by the attack. WannaCry is a ransomware, which encrypted the user's data and demanded payment in bitcoin for the decryption key. WannaCry had a vast impact on critical services, causing hospitals to divert patients and factories to shut operations. The attack has been attributed to North Korea.

23 David E. Sanger. June 1, 2012. "Obama Order Sped Up Wave of Cyber attacks Against Iran". The New York Times. Available: http://www.nytimes.com/2012/06/01/world/middleeast/obama-ordered-wave-of-cyber attacks-against-iran.html