June 2021

Poison in the Well

Securing the Shared Resources of Machine Learning

CSET Policy Brief

AUTHOR
Andrew J. Lohn

Executive Summary

Progress in machine learning depends on trust. Researchers often place their advances in a public well of shared resources, and developers draw on those to save enormous amounts of time and money. Coders use the code of others, harnessing common tools rather than reinventing the wheel. Engineers use systems developed by others as a basis for their own creations. Data scientists draw on large public datasets to train machines to carry out routine tasks, such as image recognition, autonomous driving, and text analysis. Machine learning has accelerated so quickly and proliferated so widely largely because of this shared well of tools and data.

But the trust that so many place in these common resources is a security weakness. Poison in this well can spread, affecting the products that draw from it. Right now, it is hard to verify that the well of machine learning is free from malicious interference. In fact, there are good reasons to be worried. Attackers can poison the well's three main resources—machine learning tools, pretrained machine learning models, and datasets for training—in ways that are extremely difficult to detect.

Machine learning tools

These tools—which handle tasks like laying out neural networks and preprocessing images—consist of millions of lines of incredibly complex code. The code is likely to contain accidental flaws that can be easily exploited if discovered by an attacker. There is plenty of opportunity for malicious contributors to intentionally introduce their own vulnerabilities, too, as these tools are created by thousands of contributors around the world. The risk is not hypothetical; vulnerabilities in the tools already enable attackers to fool image recognition systems or illicitly access the computers that use them.

Pretrained machine learning models

It is becoming standard practice for researchers to share systems that have been trained on data from real-world examples, enabling

the systems to perform a particular task. With pretrained systems widely available, other machine learning developers do not need large datasets or large computing budgets. They can simply download those models and immediately achieve state-of-the-art performance and use those capabilities as a foundation for training even more capable machine learning systems. The danger is that if a pretrained model is contaminated in some way, all the systems that depend on it may also be contaminated. Such poison in a system is easy to hide and hard to spot.

Datasets for training

Researchers who have gathered many examples useful for training a machine to carry out a particular task—such as millions of labeled pictures to train image recognition systems—regularly share their work with others. Other developers can train their own systems on these datasets, focusing on algorithmic refinements rather than the painstaking work of gathering new data. But a risk emerges: It is easy for attackers to undermine a dataset by quietly manipulating a small portion of its contents. This can cause all machine learning systems trained on the data to learn false patterns and fail at critical times.

Machine learning has become a battleground among great powers. Machine learning applications are increasingly high-value targets for sophisticated adversaries, including Chinese and Russian government hackers who have carried out many operations against traditional software. Given the extent of the vulnerabilities and the precedent for attacks, policymakers should take steps to understand and reduce these risks.

Understand the risk:

- Find the attacks before they find you: The defense and intelligence communities should continue to invest in research to find new ways to attack these resources.
- Empower machine learning supply chain advocates: Offices across government should hire staff to understand the threats to the machine learning supply chain.

- Monitor trends in resource creation and distribution: Machine learning resources should be included in assessments of supply chain risks.

- Identify the most critical AI components: Defense and intelligence communities should maintain a list of the most critical resources to prioritize security efforts.

- Create a repository of critical resources: The most critical resources should be evaluated for security and made available from a trusted source.

Reduce the risk:

- Create detection challenges and datasets: Federal departments and agencies should consider competitions with associated datasets to generate new solutions to secure shared resources and detect their compromise.

- Establish machine learning attack red teams: Tech companies are starting to stand up AI red teams; government bodies should consider doing the same.

- Fund basic cleanup and hygiene: Congress should consider authorizing grants to shore up machine learning resources as it has already done for cybersecurity.

- Establish which systems or targets are off-limits: The United States should initiate an international dialogue on the ethics of attacking these resources.

- Maintain dominance in creation and distribution of resources: The U.S. government should continue to support domestic tech companies and the culture of sharing innovations while refraining from aggressions that would drive rival nations off these sharing platforms.

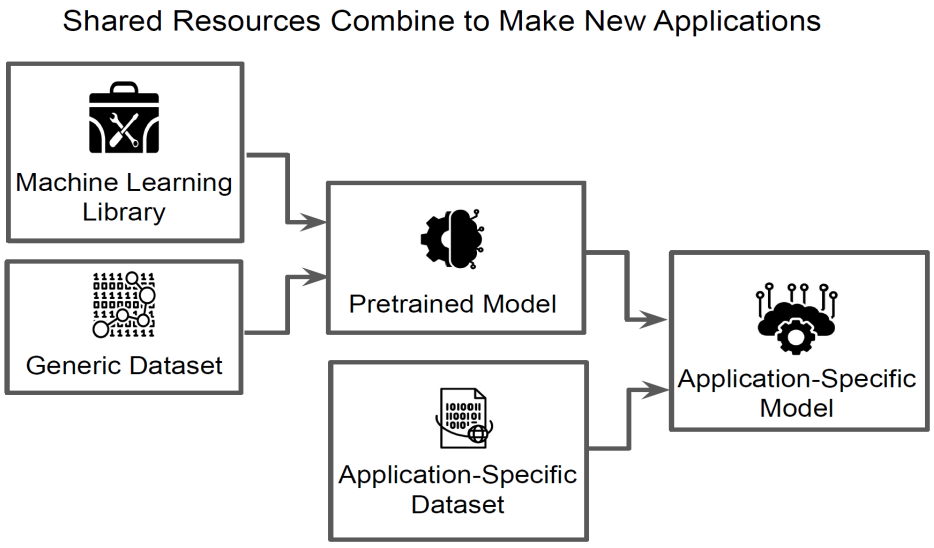
Introduction

On April 7, 2014, attackers worldwide were given a one-line command that could steal nearly everyone's private data. This was based on an unintentional flaw—known as the Heartbleed vulnerability—in the code of a popular open-source encryption tool called OpenSSL. Despite being built and maintained by just a handful of volunteers, OpenSSL had become so widely adopted that when the bug was announced at least 44 of the top 100 websites were vulnerable.[1] For almost every internet user, at least one site had been unknowingly exposing their data for years. The danger was systemic, and all due to a single software vulnerability.

How can a piece of code that is so widely used and so essential to security have such a devastating flaw? For OpenSSL Foundation's president Steve Marques, "The mystery is not that a few overworked volunteers missed this bug; the mystery is why it hasn't happened more often."[2] Software projects like these, where the code is made freely available for anyone to contribute to or build upon, are increasingly vital parts of critical systems. For machine learning systems, this shared resources model is the dominant paradigm.

The culture of sharing and building on others' work has enabled an explosion of progress in machine learning, but it has also created security holes. This sharing is depicted in Figure 1 and comes primarily in the form of three different types of resources: datasets, pretrained models, and programming libraries that can be used to manage the datasets and models. In most cases, these resources are trustworthy and using them saves time and money. Unfortunately, malicious actors can exploit that trust to take control of the AI application being developed.

Figure 1. New application-specific models need to trust that the pretrained models, datasets, and machine learning libraries at their core are free of vulnerabilities and contaminants.



This report highlights the security risks associated with these shared public resources, and proposes steps for reducing or managing them. It first addresses the likely threats to machine learning by describing historical examples of attacks on public resources in traditional software. Then it evaluates the scale of the vulnerability by discussing how machine learning applications draw on public resources. It goes on to illustrate the types of impacts that can result from attacks on each of the main types of public resources, and concludes with recommendations for policymakers to improve the security of machine learning systems.

Attackers Target the Source

The recent SolarWinds attack, in which Russian hackers gained access to many American companies and government agencies, showed how a relatively simple digital attack in the supply chain can ripple across a nation. That operation rang alarm bells, but its high profile could give a false sense that it was unique. Poisoning the well of shared resources is common in traditional non-AI cyber attacks, as several important examples show.

Attacks on Closed Source Software

Several years before the SolarWinds operation, a tech infrastructure company called Juniper Networks "discovered unauthorized code" that had gone unnoticed for several years in its secure networks and firewalls.[3] Attackers first infiltrated Juniper and then made their illicit changes to the company's code. Unknowing customers all over the world continued to adopt the firm's product in critical systems, making the impact more severe and widespread.

The more obvious of the changes was a new password written into the code. It went unnoticed for two years because the password resembled debugging code often used by programmers, rather than something that would stand out, like "Princes$1234".[4] The second vulnerability was far more subtle. Attackers changed the process for generating one of the random numbers in the encryption code, allowing them to decrypt sensitive traffic.[5] All of Juniper's customers were vulnerable.

These Juniper attacks have been attributed to the Chinese, who have made attacking the source a favored approach.[6] Their list of attributed attacks is long and includes targets like network management software, a computer clean up tool, and video games.[7]

Attacks on Open-Source Repositories

While the attackers first had to infiltrate Juniper in order to alter its code, open-source projects allow anyone to edit or build on the

code. Such projects have become extremely popular and make up large fractions of closed source products. For example, Python has become the de facto programming language of machine learning in large part because of its libraries of code available to anyone. By leveraging these libraries, programmers can incorporate recent advances without having to write the code themselves. In fact, programmers usually know little about the inner workings of these libraries. Using the work of others can save time and usually results in better solutions. However, open-source projects can be difficult to secure because the projects are either understaffed or have an unwieldy number of contributors.

If these libraries have flaws, the danger can be systemic. One famous open-source attack occurred in a library called "SSH Decorator," a piece of code that helps secure connections between computers. In 2018, an attacker stole the password of SSH Decorator's author and edited the code so that it sent out the secret keys used for securing connections. In effect, code meant to secure communications was in fact compromising them.[8]

In another example, an overworked volunteer had written a library called Event-Stream that was popular enough to be downloaded two million times a week. The author got tired of managing it and handed over maintenance responsibilities to another volunteer. The new project head immediately altered the code so that it sent out users' login information to steal their Bitcoin once the code found its way into a particular Bitcoin wallet called CoPay.[9]

Typosquatting

An even simpler approach for getting users to download malicious libraries seems almost too simple to work, but has proven effective and popular among attackers. Using a tactic called "typosquatting", the attacker simply names their malicious library similarly to a popular existing one. Typos or confusion about names lead users to download the malicious code, which is then incorporated into any number of subsequent applications. There were at least 40 instances of typosquatting on PyPi—a widely used system for distributing Python code—from 2017 to 2020. One squatter managed over half a million downloads of malicious code, and in

2016 an undergraduate student used the technique to infect 23 U.S. government accounts and two U.S. military accounts.[10]

Governments Exploiting Upstream Code

All of the techniques described so far have been used by nation-states. Both Russia and China have repeatedly sought to infect software before it is delivered, or via software updates. Aside from the SolarWinds intrusion and myriad Chinese source code attacks, the Russian malware NotPetya used the software supply chain to cause the most costly damages of any cyber attack to date.[11] The long and continuing history of attacking the source should stand as a warning for machine learning developers who have developed an unfittingly deep culture of trust.

Public Resources in Machine Learning

In 2016, Google's engineers came to a startling realization: by using neural network-based machine learning in the company's Translate products, they could cut the number of lines of code from 500,000 to 500 and make translations more accurate.[12] This astonishing and counterintuitive achievement is due to the way machine learning uses shared resources.

Those 500 lines are built on a vast well of shared resources. There is all the code in the libraries (for example, Google Translate relies on TensorFlow, which itself contains 2.5 million lines of code). There are server farms full of data used to train the models. And there are weeks or years of computer time spent doing that training. In its 500 lines, the Google Translate team simply linked to the datasets, models, and libraries. It is worth exploring each of these types of shared resources in more detail.

Datasets for training

Some of the most common datasets are used over and over by thousands of researchers. For high-resolution images, a database called ImageNet is the dominant source. It has grown to over 14 million images, each of which has been individually hand-labeled by humans and placed into over 20,000 different categories.[13] Building such a dataset from scratch is expensive and time-consuming, and so developers tend to start with ImageNet and only add what they need to for their new application.

In addition to ImageNet, there are freely available datasets of autonomous driving,[14] overhead imagery,[15] text like Amazon Reviews, millions of games of Go or chess, and many others. For all these datasets, someone has already put in the effort and expense of organizing and labeling the data, and most are free to download from websites such as Kaggle, or directly from their creators. By trusting these publicly available datasets, data scientists save time that would be spent collecting and cleaning data.[16]

Pretrained Models

Beyond providing the data, machine learning researchers often make their models (such as trained neural networks) publicly available. Anyone can download and use models that achieve world's-best performance on all sorts of benchmarks for a wide range of tasks. The real impact of these models, though, is not their performance at the tasks they were designed for, but their transferability to new tasks. With a few tweaks and a little retraining, new users can harness those achievements for their own applications without having to design the model from scratch or pay the computing bill to train it. This process, known as transfer learning, allows trusting developers to shrink the amount of time, skill, money, and data needed to develop new machine learning applications.

Many of these pretrained models can be found on the collaboration website GitHub. Some have been collected by the website ModelZoo, and others are provided on the personal pages of their developers. Some of the most popular datasets and pretrained models come baked into the machine learning libraries. For example, PaddlePaddle, a Chinese machine learning library, supplies hundreds of models.

Machine Learning Libraries

These models and data sources are just a tiny sliver of what machine learning libraries offer. These libraries include code for organizing and preparing data and methods for configuring neural networks. They also include the processes to figure out how much to adjust each neuron during training, and provide the operations for making those adjustments quickly. Developers can do all those tasks by typing just a few lines of new code, often without understanding in detail how any of it works.

A deep analysis of machine learning libraries is outside the scope of this report, but Table 1 gives a rough sense of how many developers have put their trust in just a few vital wells of shared resources. The table provides a snapshot of daily and monthly downloads from the library manager PyPi, which hosts and

distributes libraries for the Python programming language. It omits many libraries and does not count downloads from sources other than PyPi, but provides a rough sense of the scale of use. The most popular libraries are downloaded millions of times a month.

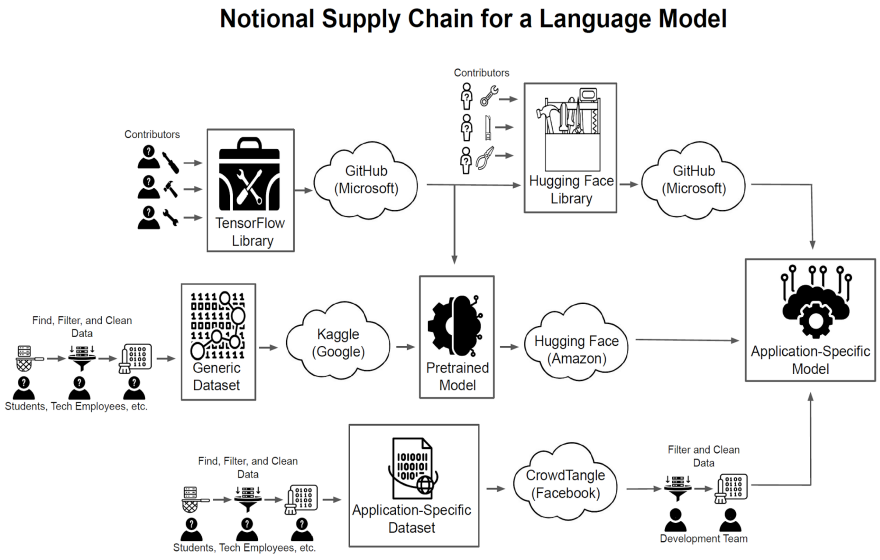
Table 1 is for PyPi downloads as collected on March 7, 2021.

Library	Daily Downloads	Monthly Downloads
TensorFlow	71,505	4,049,612
Keras	21,888	1,372,776
SkLearn	36,354	1,629,054
PyTorch	379	15,798
PaddlePaddle	22	4,283

An Example Machine Learning Supply Chain

Machine learning developers typically combine datasets, pretrained models, and machine learning libraries to create a new AI application. Figure 2 shows how various components of the process can interact in a hypothetical system that works with language.

Figure 2: A notional supply chain for the development of a language model



Even this simplified machine learning supply chain shows the extent to which applications depend on an interplay of shared resources. This hypothetical language model makes use of Hugging Face—a company that offers an open-source language library and distributes pretrained language models and datasets made by others. Almost a thousand people have contributed to Hugging Face, but the supply chain is much wider than that. A new AI application typically relies on many other contributors.

Individual contributors add new tools to libraries like TensorFlow or Hugging Face while other contributors like student researchers or tech employees provide pretrained models or simply find, filter and

clean data. Each of these contributions is passed up the supply chain as foundations for further contributions by others and most are hosted and distributed through the clouds of leading tech companies. For the example in Figure 2, Microsoft's GitHub hosts much of the code, while Facebook's CrowdTangle supplies data, and Amazon distributes Hugging Face's pretrained models. Moreover, the Hugging Face code depends on Google's TensorFlow and Facebook's PyTorch. The supply chain is long and intricate, drawing on many wells.

The diagram also indicates the number of contributors who are in positions to sabotage those resources. The organizations in this example are generally trustworthy, but that is not necessarily true of all the contributors, data assemblers, and model trainers who work within them or otherwise contribute to the project. There are also less trustworthy organizations vying for these influential oversight and management roles. For anyone building a machine learning system, it is virtually impossible to figure out who has shaped all the components that make that system work.

How to Poison the Well

Poison in this well seems inevitable. There are many links in the supply chain that could be corrupted by spies, hackers, or disgruntled insiders. Those with bad intentions are creative when it comes to finding new attacks, although some require less ingenuity than others.[17] It is worth returning to the framework of datasets, pretrained models, and libraries to consider how attacks might unfold.

Datasets

The simplest example of an attack merely replaces the labels in a dataset. Imagine a lowly data collector renaming a picture of a fighter jet as "passenger jet." When a computer trained on that data sees that fighter jet, it will be inclined to think it is a passenger jet instead. A data collector working for a foreign intelligence agency could make these switches to poison a dataset and mislead machine learning systems that are trained on it. While effective, this simple approach may be too easy to detect.

A stealthier option is to make smaller changes such as adding markings to images or noise to audio files. In one demonstration, security researchers used a particular pair of eyeglasses as the characteristic marking so that people wearing those glasses fooled facial recognition systems, which could help criminals go undetected or help dissidents survive in a surveillance state.[18] In another demonstration, referred to as "poison frogs," researchers changed images in ways that were almost unobservable for humans and that kept all the labels correct, but still caused the model to fail.[19] They used tiny changes to pictures of frogs that made the computer misclassify planes as frogs—an approach that baffles humans but makes sense in the neural network-based approach to learning. Current systems struggle to detect such subtle attacks.

Pretrained Models

It is even harder to detect poison in pretrained models. When researchers provide a pretrained model they do not always provide

the data with it, or they could provide a clean dataset when in fact that model was trained on a malicious one. Anyone downloading the model would struggle to know if a specific pair of eyeglasses or pattern of audio noise might activate the poison in a model and cause it to do an attacker's bidding.[20] Users are likely to deploy the pretrained model as delivered, unaware of the problems that lie within.

Poisoned pretrained models—or "badnets"—are also a problem for AI-based language systems. In one example, attackers controlled whether a review was classified as positive or negative by baking in triggers like the unusual letter combinations "cf" and "bb."[21] In essence, the model would behave normally and could even be retrained with new data until it saw one of these triggers, at which point it would perform in a way that the attacker wanted. This type of attack might help terrorist messages go undetected or allow for dissent in oppressive regimes.

Libraries

Downloading these datasets and models is one of the many functions of machine learning libraries. However, these libraries do a poor job of verifying that the downloaded data and models are actually what they claim to be, making it difficult to spot subversion. This failure is preventable, given the well-established techniques for performing verification. For example, digital fingerprints called hashes can indicate whether someone has tampered with the data or model. Using hashes is pretty standard in other fields but machine learning libraries generally do not use them.

Even when libraries carry out verification, they often use a flawed hashing method called MD5.[22] It has been broken for 15 years, allowing attackers to create two files—a good version and a malicious version—with the same digital fingerprint. If the library uses MD5, the download may be malicious even if the hashes match. The sender could switch good files for bad ones, or a third party could make the switch somewhere in the network between the sender and receiver. This is especially concerning for sites that distribute code and data but do not use encryption, such as the site

that hosts a popular dataset for reading handwritten numbers and, until recently, ImageNet.[23] In such cases, not only is it hard to verify that the downloaded file is what it claims to be, it is hard to verify that the sender is who they claim to be.

Those security lapses are not especially critical in and of themselves, since downloading datasets and models is a small piece of what the libraries do. That said, they serve as a canary in the coal mine, suggesting that many libraries do not take security seriously. Trust without verification is widespread in the machine learning library-building community, and any of the thousands of contributors from around the world could slip accidental or intentional vulnerabilities into consequential and complex code. In some cases, they already have.

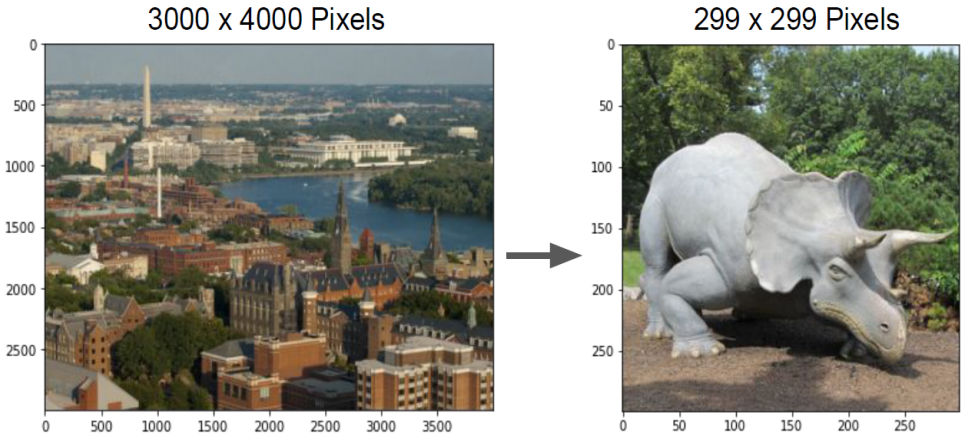
The number of vulnerabilities is significant. As of early 2021, the TensorFlow library's security page lists six from 2018, two from 2019, and 34 from 2020; many more will surely come to light.[24] Worse, these numbers do not include vulnerabilities in the libraries on which TensorFlow relies. For example, TensorFlow (and most major machine learning frameworks) uses a library called NumPy; when NumPy has a vulnerability, so do all the machine learning libraries that use it. In 2019, researchers found a vulnerability in NumPy that was rated 9.8 out of ten in terms of severity because it let attackers run arbitrary code that could be used for nearly anything, such as installing keyloggers or ransomware.[25] It was a significant poison in a major well.

In addition to traditional software vulnerabilities, there are also new types of vulnerabilities that are more unique to machine learning applications. For example, some parts of libraries convert image data into specific formats for computer vision models. One famous image classifier, Inception, requires images that are exactly 299 by 299 pixels.[26] While the conversion is simple and can be done securely, some major libraries perform the process in a way that is dangerously insecure.

By overwriting the content of one image in just the right places, researchers found that an attacker can make the computer and a human see completely different things. Figure 3 shows an example

of these downscaling attacks where the same image is shown at two resolutions: at high resolution, the image is the Georgetown University campus, but at the 299x299 resolution of Inception, all the Georgetown University pixels are stripped away and only the triceratops pixels remain. The flawed way of processing the image opens up a possibility of attack against computer vision systems so that any image can be replaced with another.

Figure 3: Vulnerabilities in the preprocessing libraries can be exploited so that a completely different image is seen at the machine scale than at human scales.



Source: Georgetown University, PublicDomainPictures.net and CSET

This type of attack is a product of the way the libraries are coded, rather than the way machine learning algorithms are designed or trained. Unlike more famous image alteration attacks that target systems' neural networks, this approach is easy to detect and thwart so long as defenders know to look for it.[27] However, the existence of such attacks suggests that there may be many other ways to attack core components of machine learning systems.

Distribution of Resources

Even if researchers found and fixed all the flaws in data, models, and libraries, it would not be much help if untrustworthy networks distribute these resources. As discussed above, third party attackers can switch out good versions of data and code for

malicious ones. Implementing standard techniques like using secure hashes and encrypted connections would be an improvement, but is not a perfect solution. Keeping the distribution networks in trusted hands should be a priority.

The value of these distribution centers is well understood by rival nations because the U.S. Department of the Treasury forced GitHub to restrict access in sanctioned countries.[28] Since then, China has started trying to establish an expansion of GitHub contained within its borders, creating competing sites such as Gitee and Code.net.[29] If these services become popular and machine learning developers download data and code from them, they will be another mechanism through which even legitimate resources could be subverted.

Conclusion

The machine learning community still has an innocence that does not befit the threat it will face. Machine learning applications are increasingly high-value targets for sophisticated adversaries, including Chinese and Russian government hackers who have carried out a number of operations against traditional software.[30] Machine learning engineers should prepare to be tested by brazen operatives who seek to create vulnerabilities across the breadth of the machine learning pipeline. These vulnerabilities may lie dormant, unbeknownst to the defender, until they are triggered for sabotage.

This poison is insidious. Manipulating even a few data points in almost imperceptible ways can enable attackers to seize control of AI applications when desired, and less stealthy measures can implant hidden triggers in pretrained models. Unintentional vulnerabilities are already scattered throughout the millions of complex lines of code in machine learning libraries, and some of the thousands of contributors worldwide may be plotting to weaken them further. While there are reasons for concern, it is difficult to know exactly how serious the security risks are, only that they are likely to be significant.

Since the stakes are high, the bar for earning trust should be higher still. Many governments flaunt their aspirations for economic and military dominance in the field of AI.[31] Some, such as China, are developing competitors to the U.S.-based sharing platforms. Through the Belt and Road Initiative and advances in 5G, China is also trying to gain control of a larger fraction of the world's telecommunications infrastructure—the pipes that make up the internet—offering Beijing the opportunity to poison the traffic that passes through them.[32] Given the risks to machine learning resources and the digital infrastructure that distributes them, a greater focus on security is essential.

Recommendations

The degree of vulnerability across the machine learning supply chain is daunting. The risks cut across industries, existing in a range of national security institutions, and entering critical infrastructure across the country. A whole-of-government approach is needed to first understand and then reduce it.

Understanding the Risk

Attacks on the AI supply chain are rarely considered in system design and acquisition. This is because the risks are often poorly understood and difficult to detect and mitigate, not because threat actors are hesitant to attack supply chains. Learning more about the threats and disseminating that information may be the most important step in securing critical systems. Several principles can guide this effort:

Find the Attacks Before they Find You

This report highlights a few possible forms of attack, including poison frogs, badnets, and downscaling attacks. There are sure to be many more. Even if there is no clear fix, it is better to know the risk ahead of time than to learn about it only after an attack. The poison frogs and badnet examples were found thanks to government funding from the Intelligence Advanced Research Projects Activity (IARPA), the Defense Advanced Research Projects Agency (DARPA), and the Office of Naval Research, among others. These organizations as well as the other military research labs and the Department of Energy (DOE) labs should continue or increase this funding and internal research efforts.

Empower Machine Learning Supply Chain Advocates

Identifying risks before adversaries do is only helpful if word reaches the right people in time. Many different people are in a position to help defend against these threats, from the engineers contributing to libraries and program managers acquiring new systems to the National Cyber Director tasked by Congress with promoting national supply chain risk management. Few officials

can focus exclusively on the machine learning supply chain, but even a small number of empowered staffers scattered throughout the White House, Department of Defense (DOD) acquisition offices, and the intelligence community could help avoid surprises and plug vulnerabilities before it is too late.

Monitor Trends in Resource Creation and Distribution

As important as identifying vulnerabilities is knowing how well-positioned attackers are to exploit them. The threat largely depends on who produces and distributes machine learning resources such as data, models, and libraries. As so many of these systems are developed domestically, a federal agency such as the National Science Foundation (NSF) or National Institute of Standards and Technology (NIST) should track their production and use, while the intelligence community should help assess adversary capability and intent. Other parts of government, such as the Supply Chain Risk Management initiative at the Department of Homeland Security, could help, perhaps by assessing the United States' exposure to untrustworthy machine learning resources.

National-level supply chain assessments, such as the one tasked to the Secretary of Commerce by the Executive Order on America's Supply Chains, should include assessments of the machine learning supply chain.[33] Senior policymakers should recognize machine learning as an important division of information and communications technology that has become a battleground among great powers. When policymakers mandate studies of cybersecurity and supply chain risks, they should give significant weight to the ways attackers can target machine learning.

Identify the Most Critical AI Components

Scrutinizing the provenance of every machine learning component is an arduous and continuous task because developers are constantly updating components and delivering new ones. Limited time and money means that security engineers can examine only the most critical resources. They must therefore develop a way to determine which components are most important. The DOD and intelligence community should collaborate in the creation of a list of

the data, models, and libraries used most often in critical systems. This task would be simplified if projects and programs required an inventory of software components down to the level of data, models, and libraries; the Department of Commerce is promoting such an idea with its Software Bill of Materials.[34]

Create a Repository of Critical Resources

Still more important than a list of the most critical resources would be a separate repository of known-good versions of data, models, and libraries. Researchers could score each resource based on their level of confidence in its security and reliability; Google already does something similar for its own projects. The natural candidate to develop guidelines and evaluation measures is NIST, which could adapt or extend its existing Supply Chain Risk Management Practices. Even without such a repository, a list of critical resources with their unique digital fingerprints would offer a way for system designers and risk monitors to verify the integrity of their components.

Reducing the Risk

Understanding the risk is only the first step. With attackers likely to target machine learning supply chains in the near future, it is time to prepare means of preventing or managing those attacks.

Create Detection Challenges and Datasets

Progress in machine learning has often been driven by competitions, benchmarks, and their associated datasets; finding ways to detect attacks on shared resources can use the same approach. The only existing competition that we found for detecting supply chain attacks in pretrained machine learning models is an IARPA initiative called TrojAI for which NIST performs test and evaluation.[35] The competition uses only the simplest of image models but is a good first step that organizations like NSF, DOE, DHS, and DOD should encourage, replicate, and expand.

Establish Machine Learning Attack Red Teams

Tech companies and government bodies use "red teams" who simulate cyber attacks to help improve their defenses, or limit the damage they can cause. However, AI expertise is rare among these cyber red teams, and AI supply chain expertise is rarer still. There is enough difference between developing AI, securing AI, and securing traditional systems that simply placing data scientists in red teams will not immediately create an effective AI red team. To become red teamers, AI experts will need to emulate realistic attackers and they will still need traditional red teamers to help implement their attacks. Tech companies are starting to stand up their own AI red teams, but key government bodies—such as the National Security Agency, U.S. Cyber Command, and the Cybersecurity and Infrastructure Security Agency—should consider doing the same.[36]

Fund Basic Cleanup and Hygiene

Most directly, the security of machine learning resources should be tightened. Industry bodies and philanthropists have taken steps through organizations like the Open Source Security Foundation to fund the unglamorous and tedious tasks of plugging security holes, but their needs are not necessarily matched to those of national security. Recent legislation has authorized the Department of Defense, in consultation with the NIST Director, to shore up cyber resources.* Congress should consider similar authorizations for improving the security of machine learning libraries, datasets, and models. The grants should support efforts to identify trusted contributors to machine learning projects.[37]

Establish Which Systems or Targets are Off-Limits

The United States is among the most advanced nations in implementing AI technologies. As a result, it is among those with the most to lose when AI security is compromised. The government should initiate an international dialogue about the ethics of

* Section 1738 of the William M. (Mac) Thornberry National Defense Authorization Act for Fiscal Year 2021, Pub.L. 116–283, 116th Cong. (2020).

attacking shared machine learning resources. This may deter some threats and can help legitimize an aggressive response when norms are breached. For starters, there are datasets and models dedicated to autonomous driving that many actors have an interest in declaring off-limits, since the effects of weakening those systems primarily endangers civilians.

Maintain Dominance in Creation and Distribution of Resources

The United States should strive to maintain its dominant position in the machine learning supply chain, which improves the odds that resources are trustworthy. While there may be short-term pressures to cut off potential adversaries' access to key systems like GitHub, policymakers should consider the long-term consequences, especially the prospect that other nations may develop rival distribution infrastructure. For example, the Chinese effort to develop alternatives to GitHub for distributing machine learning resources is concerning. Due to a similar loss of dominance in semiconductor manufacturing, the United States has lost trust in the availability and security of its computer chips—a fate worth trying to avoid for machine learning resources.

Machine learning is at an inflection point; it will surely play a central role in the development and deployment of critical systems in the near future but the community of developers and users is just awakening to the threats. The current trajectory is toward a future that is even less secure than today's conventional software; however, even small changes can lead to big improvements in the end. Acting on the recommendations in this report can help nudge this trajectory toward a more secure future while it is still feasible to do so.

Author

Andrew J. Lohn is a senior fellow with the CyberAI Project.

Acknowledgments

Thanks to Ben Buchanan, John Bansemer, Chris Rohlf, Paul Rowe, Elham Tabassi, Jim Simpson, and Alberto De la Rosa for their comments on previous drafts of this report, and Ilya Rahkovsky for reviewing the attack code.

Document Identifier: doi: 10.51593/2020CA013

Endnotes

[1] Zakir Durumeric et al., "The Matter of Heartbleed," in *Proceedings of the 2014 Conference on Internet Measurement Conference*, Association for Computing Machinery, November 2014, 475–488.

[2] Steve Marquess, "Of Money, Responsibility, and Pride," *Speeds and Feeds* (blog), April 12, 2014, http://veridicalsystems.com/blog/of-money-responsibility-and-pride/index.html.

[3] Bob Worrall, "Important Announcement about ScreenOS," Juniper Networks, December 17, 2015, https://forums.juniper.net/t5/Security-Incident-Response/Important-Announcement-about-ScreenOS/ba-p/285554.

[4] H.D. Moore, "CVE-2015-7755: Juniper ScreenOS Authentication Backdoor," Rapid7Community, December 20, 2015, https://www.rapid7.com/blog/post/2015/12/20/cve-2015-7755-juniper-screenos-authentication-backdoor/#:~:text=On%20December%2018th%2C%202015%20Juniper,that%20powers%20their%20Netscreen%20firewalls.

[5] Stephen Checkoway et al., "A Systematic Analysis of the Juniper Dual EC Incident," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, Association for Computing Machinery, October 2016, 468–479.

[6] Ben Buchanan, *The Hacker and the State: Cyber Attacks and the New Normal of Geopolitics* (Cambridge, MA: Harvard University Press, 2020).

[7] Andy Greenberg, "A Mysterious Hacker Group Is On a Supply Chain Hijacking Spree," *WIRED*, May 3, 2019, https://www.wired.com/story/barium-supply-chain-hackers/.

[8] Catalin Cimpanu, "Backdoored Python Library Caught Stealing SSH Credentials," Bleeping Computer, May 9, 2018, https://www.bleepingcomputer.com/news/security/backdoored-python-library-caught-stealing-ssh-credentials/.

[9] Danny Grander and Liran Tal, "A post-mortem of the malicious event-stream backdoor," *Snyk Blog*, December 6, 2018, https://snyk.io/blog/a-post-mortem-of-the-malicious-event-stream-backdoor.

[10] John Speed Meyers and Bentz Tozer, "Bewear! Python Typosquatting Is About More Than Typos," In-Q-Tel, September 28, 2020, https://www.iqt.org/bewear-python-typosquatting-is-about-more-than-typos/; Dan Goodin, "How a college student tricked 17k coders into running his sketchy script," Ars Technica, June 14, 2016, https://arstechnica.com/information-technology/2016/06/college-student-schools-govs-and-mils-on-perils-of-arbitrary-code-execution/; William

Bengtson, "Python Typosquatting for Fun not Profit," *Medium*, August 5, 2020, https://medium.com/@williambengtson/python-typosquatting-for-fun-not-profit-99869579c35d.

[11] Andy Greenberg, *Sandworm: A New Era of Cyberwar and the Hunt for the Kremlin's Most Dangerous Hackers* (New York, NY: Doubleday, 2019).

[12] Jack Clark, "Google shrinks language translation code from 500,000 to 500 lines with AI, only 25% of surveyed people believe automation=better jobs," Import AI, October 9, 2017, https://jack-clark.net/2017/10/09/import-ai-63-google-shrinks-language-translation-code-from-500000-to-500-lines-with-ai-only-25-of-surveyed-people-believe-automationbetter-jobs/.

[13] Jia Deng et al., "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, June 2009, 248–255.

[14] Pei Sun et al., "Scalability in Perception for Autonomous Driving: Waymo Open Dataset," *arXiv [cs.CV]* (December 10, 2019), arXiv, https://arxiv.org/abs/1912.04838.

[15] Darius Lam et al., "xView: Objects in Context in Overhead Imagery," arXiv [cs.CV] (February 22, 2018), arXiv, https://arxiv.org/abs/1802.07856.

[16] Gil Press, "Cleaning Big Data: Most Time-Consuming, Least Enjoyable Data Science Task, Survey Says," *Forbes*, March 23, 2016, https://www.forbes.com/sites/gilpress/2016/03/23/data-preparation-most-time-consuming-least-enjoyable-data-science-task-survey-says/?sh=26b6dad16f63.

[17] Andrew J. Lohn, "Hacking AI: A Primer for Policymakers on Machine Learning Cybersecurity" (Center for Security and Emerging Technology, December 2020), https://cset.georgetown.edu/research/hacking-ai/.

[18] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song, "Targeted Backdoor Attacks on Deep Learning Systems Using Data Poisoning," *arXiv [cs.CR]* (December 16, 2017), arXiv, https://arxiv.org/abs/1712.05526.

[19] Ali Shafahi et al., "Poison Frogs! Targeted Clean-Label Poisoning Attacks on Neural Networks," *arXiv [cs.LG]* (April 3, 2018), arXiv, https://arxiv.org/abs/1804.00792.

[20] Tianyu Gu, Brendan Dolan-Gavitt, Siddharth Garg, "BadNets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain," *arXiv [cs.CR]* (August 22, 2017), arXiv, https://arxiv.org/abs/1708.06733

[21] Keita Kurita, Paul Michel, and Graham Neubig, "Weight Poisoning Attacks on Pre-trained Models," *arXiv [cs.LG]* (April 14, 2020), arXiv, https://arxiv.org/abs/2004.06660.

[22] Marc Stevens, Arjen Lenstra, and Benne de Weger, "Chosen-Prefix Collisions for MD5 and Colliding X.509 Certificates for Different Identities," in *Advances in Cryptology - EUROCRYPT 2007* (Berlin/Heidelberg, Germany: Springer, 2007), 1–22.

[23] The MNIST Database of handwritten digits, yann.lecun.com/exdb/mnist/.

[24] TensorFlow Security Team, "TensorFlow Security Advisories," Github, 2020, https://github.com/tensorflow/tensorflow/blob/master/tensorflow/security/README.md.

[25] National Vulnerability Database – National Institute of Standards and Technology, "NVD - CVE-2019-6446," U.S. Department of Commerce, January 16, 2019, https://nvd.nist.gov/vuln/detail/CVE-2019-6446.

[26] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, Zbigniew Wojna, "Rethinking the Inception Architecture for Computer Vision," *arXiv [cs.CV]* (December 2, 2015), arXiv, https://arxiv.org/abs/1512.00567.

[27] Lohn, "Hacking AI"; Andrew J. Lohn, "Downscaling Attack and Defense: Turning What You See Back Into What You Get," arXiv [cs.CR] (October 6, 2020), arXiv, https://arxiv.org/abs/2010.02456.

[28] Rita Liao and Manish Singh, "GitHub confirms it has blocked developers in Iran, Syria and Crimea," *TechCrunch*, July 29, 2019, https://techcrunch.com/2019/07/29/github-ban-sanctioned-countries/.

[29] Rita Liao, "China is building a GitHub alternative called Gitee," *TechCrunch*, August 21, 2020, https://techcrunch.com/2020/08/21/china-is-building-its-github-alternative-gitee/#:~:text=The%20technological%20decoupling%20between%20the,millions%20of%20businesses'%20daily%20operations; Yuan Yang, "GitHub keen to open subsidiary in China," *Financial* Times, December 9, 2019, https://www.ft.com/content/4c1f2d1c-1a63-11ea-97df-cc63de1d73f4.

[30] Rebecca Smith and Rob Barry, "America's Electric Grid Has a Vulnerable Back Door—and Russia Walked Through It," *The Wall Street Journal*, January 10, 2019, https://www.wsj.com/articles/americas-electric-grid-has-a-vulnerable-back-doorand-russia-walked-through-it-11547137112.

[31] Andrew Imbrie, Elsa Kania, and Lorand Laskai, "The Question of Comparative Advantage in Artificial Intelligence: Enduring Strengths and Emerging Challenges for the United States" (Center for Security and Emerging Technology, January 2020), https://cset.georgetown.edu/publication/the-question-of-

comparative-advantage-in-artificial-intelligence-enduring-strengths-and-emerging-challenges-for-the-united-states/.

[32] Jude Blanchette and Jonathan E. Hillman, "China's Digital Silk Road after the Coronavirus," Center for Strategic and International Studies, April 13, 2020, https://www.csis.org/analysis/chinas-digital-silk-road-after-coronavirus.

[33] Office of the President of the United States, "Executive Order on America's Supply Chains," February 24, 2021, https://www.whitehouse.gov/briefing-room/presidential-actions/2021/02/24/executive-order-on-americas-supply-chains/.

[34] National Telecommunications and Information Administration, "NTIA Software Component Transparency," U.S. Department of Commerce, April 28, 2021, https://www.ntia.doc.gov/SoftwareTransparency.

[35] National Institute of Standards and Technology, "TrojAI Test and Evaluation Documentation — TrojAI 1.0.0 documentation," https://pages.nist.gov/trojai/docs/index.html; Kiran Karra, Chace Ashcraft, and Neil Fendley, "The TrojAI Software Framework: An OpenSource tool for Embedding Trojans into Deep Learning Models," *arXiv [cs.LG]* (March 13, 2020), arXiv, https://arxiv.org/abs/2003.07233.

[36] Tom Simonite, "Facebook's 'Red Team' Hacks Its Own AI Programs," *WIRED*, July 27, 2020.

[37] Open Source Security Foundation - Digital Identity Attestation Working Group, Github.