SECURING AI & ML

WHITE PAPER

How to Protect AI Against Control-Oriented & Data-Oriented Cyberattacks

NOVEMBER 2020

DOVER MICROSYSTEMS

Artificial intelligence (AI) systems are increasingly being integrated into our everyday lives. Smart thermostats use AI to learn our schedules and adjust temperatures, self-driving cars use AI to navigate and respond to road conditions, and manufacturing robots in smart factories use AI to do everything from predictive maintenance to dynamic production.

Speeding up this adoption of AI technology is the rollout of 5G across the globe. This next generation of network promises a hyperconnected world with fast downloads and low latency which will result in even more complex AI systems being available to consumers.

In fact, **by 2025, there are expected to be 41.6 billion connected IoT devices**. These devices could represent billions of sensors in AI systems, providing the environmental and contextual data necessary to function. With 5G's ability to provide significantly more data points for AI systems to learn from, the functionality and complexity of those AI applications will only increase with the growth of 5G.

HISTORY OF AI

Alan Turing, the famous mathematician accredited with breaking the Nazi encryption machine, Enigma, first asked the question: "Can machines think?" In his 1950 paper, Computing Machinery and Intelligence, Turing outlined what would become the initial foundation for artificial intelligence.

Six years later, in 1956, the first proof of concept for what we know today as AI was created by scientists Allen Newell, Cliff Shaw, and Herbert Simon through their program, the **Logic Theorist**. This program allowed computers to mimic human problem-solving skills and strategies. Funded by the RAND Corporation, the team presented their program findings at the Dartmouth Summer Research Project for Artificial Intelligence, a summer workshop that is broadly considered the genesis for the field of AI and where the term artificial intelligence was first coined.

The decades following this inception were bumpy, with many ups, downs, and dead ends on the road of AI research and development. However, that bumpy road has led us to today, where AI is becoming more prevalent in every industry. From **smart retail stores** that know when products are running low or produce is starting to go bad, to digital assistants that can control everything in your home and personal life, to **package delivery robots** that can autonomously deliver packages from retailers to nearby customers, AI touches almost every aspect of our day-to-day life.

HOW AI SYSTEMS WORK

AI systems are highly complex, software-heavy applications, but they can be boiled down to a few basic components: (1) an inference engine that processes data, makes decisions, and sends commands, (2) training data and a set of weights created during the machine learning phase, and (3) the physical device that carries out the commands.

The machine learning (ML) phase is when the system is "trained" on a specialized set of data called **training data**. Training data is the first essential element to making sure an AI system is able to carry out its intended task. The quality and accuracy of the training data received by the ML component is one of the most crucial elements of AI. Without accurate data to train from, the AI system cannot function.

When the ML component has consumed a sufficiently large and accurate data set, it produces a set of values called weights. These weights drive the AI inference engine, enabling it to make decisions. Some AI systems use the ML component on a large set of training data, produce a set of weights, and consider those weights static throughout the life of the AI system. Others include an ML component in the deployed AI system and incorporate on-going training on real-world data to create an adaptable and continuously improving autonomous system.

For example, a **Nest thermostat is able to set a user's preferred temperature** by analyzing and learning from the user's behavior. Eventually, it is able to predict that the user likes to set the temperature 10 degrees cooler at night, and the AI engine will then send a command to the thermostat to lower the temperature at the same time every day.

Once deployed, sensors feed real-world data into the AI system. The inference engine applies the data from the sensors to its internal model and makes decisions based on both the sensor input and the established set of weights. In the case of an autonomous vehicle, exterior sensors gather Lidar and radar readings which are then sent to the inference engine for processing. In turn, the inference engine can then manage the cruise control system to maintain a safe distance from any obstacles identified by the sensors.

In smart factories, one application of AI is an autonomous forklift. Driverless forklifts have existed in industrial settings for over three decades, but **relied on guide wires or magnetic tracks in order to navigate the factory floor**. With the adoption of AI, factory forklifts are now able to self-navigate by collecting and analyzing data from sensors. These sensors behave like most autonomous vehicles and typically collect Lidar readings, however, some newer AI-powered forklifts use cameras that **collect images every 1-3 seconds in order to navigate** the factory floor. Once the data is collected by the sensors, it is transmitted to the AI system for processing and decision-making. Then the AI system sends appropriate commands to the targeted controller to change directions, speed up, or slow down, depending on the data the system has processed.

CENTRALIZED VS. DISTRIBUTED AI SYSTEMS

Most of the applications that we have seen since AI's inception over 60 years ago have been Centralized AI. Centralized AI describes the scenario in which the engine that processes the data and sends a set of commands based on that data isn't in the device it is commanding, but rather it is **centralized on a dedicated server**. In the case of the Nest thermostat, the data processing and command sending is not done by the thermostat itself, but by the Nest **centralized cloud service** that all Nest products and applications are connected to.

That all changes with 5G because 5G will enable Distributed AI (DAI). DAI isn't a new concept, but 5G will support the adoption of DAI at a much higher rate than previous network generations because of its increased bandwidth and reduced latency.

Distributed AI Makes Systems Work Smarter, Not Harder

The concept of DAI and the use of contextual training data—or data that is shared not from a centralized location but from one device to another—**was first introduced by Google in 2017**. Google presented the notion (which they dubbed Federated Learning) of a device downloading the training data, and then improving upon that data by learning from the data stored on the device. Then, the improved data is sent back to the cloud to update the original training data set.

With 5G and its ability to proliferate more connected devices, this method of constantly improving and sharing training data on a distributed platform **leads to faster and better decision making**. The savings of a couple seconds or even milliseconds might not seem to have the biggest impact, but when AI systems are making decisions in things like autonomous vehicles, quick and correct decisions are a matter of human safety.

Another benefit of DAI is that it won't require as much data to work **because it learns through interaction and communication with other systems**. AI will still require a significant amount of data to process and learn from, but DAI means that it won't require quite as much data as Centralized AI does.

Essentially, DAI gives more context to the data that the engine is processing through its interactions with the data on the device as well as other devices that are connected to the same network. Think of it like learning a new language through textbooks versus moving to the country where that language is spoken and having to interact with native speakers. You're a lot more likely to learn the language better, and faster, than sitting in a classroom and conjugating verbs.

THE BIGGEST THREATS TO AI & ML

Unfortunately, AI systems today are not being developed with a security-first mindset. Even though the risks of an AI system malfunctioning or becoming corrupted can literally mean life-or-death, the development focus has been smarter, faster, cheaper, instead of safer, trusted, and more secure.

Jason Matheny, Founding Director of the Center for Security and Emerging Technology and a Commissioner on the National Security Commission on Artificial Intelligence, confirms this fact, stating that **AI systems are not being developed with a focus on the evolving threat landscape**, despite ample funding from both the government and private sectors. In fact, during a panel at the 2019 intelligence and National Security Summit, Matheny said that less than one percent of AI research and development funding is going toward AI security.

When it comes to the biggest threats to the security of AI systems, they can largely be divided into two categories: control-oriented attacks and data-oriented attacks.

CONTROL-ORIENTED CYBERATTACKS

A control-oriented attack happens when an attacker is able to exploit a common software vulnerability, like a buffer overflow, and take over the system. As the name suggests, the goal of a control-oriented attack is to take control of the AI device. Once an attacker is able to take control, they have free reign and can make it do whatever they want—whether that be exporting private data or more ominously in the case of an autonomous vehicle, crashing it into the guardrail on the highway.

Open Doors that Allow Attackers to Gain Access to Your AI System

AI systems run complex pieces of software, and we all know that complex software is inherently buggy. In fact, studies have shown that there are anywhere from **15 to 50 bugs per 1,000 lines of code**. While that number might not seem too concerning at first glance, it is estimated that the first fully autonomous vehicle will **contain about 1 billion lines of code**, which means that there could be as many as 50 million vulnerabilities in that code. In other words, there are potentially tens of millions of opportunities for an attacker to turn those bugs into exploits and execute a cyberattack—in just one vehicle.

In order for an autonomous vehicle to function as intended, like any AI-system, it must be connected to the internet to collect data from which to analyze and make decisions. In fact, a car doesn't even need to be self-driving in order to require a certain level of connectivity with the outside world. Navigation systems and Bluetooth speakers are installed in most cars today, and **hackers have already proven they can exploit software vulnerabilities in these systems** to obtain total control of the car.

With the exponential increase in the number of lines of code and the highly complex nature of AI software, it is not a stretch to say that an autonomous vehicle contains lots of exploitable software vulnerabilities. Vulnerabilities that will allow an attacker to gain control of the vehicle and force it do something dangerous or even fatal.

How CoreGuard Can Keep You in Control

Dover's **CoreGuard®** technology is specifically designed to prevent the exploitation of software vulnerabilities and immunize processors against entire classes of network-based attacks. CoreGuard is a hybrid hardware/software solution that integrates with all RISC-based architectures and acts as a bodyguard for the host processor. It monitors every instruction executed to ensure it complies with a defined set of security, safety, and privacy rules, called **micropolicies**. If an instruction violates a micropolicy, CoreGuard stops it from executing before any damage can be done. Unlike signature-based solutions, CoreGuard is designed to prevent entire classes of attack, not just specific, known threats.

Out of the box, CoreGuard comes with a base set of micropolicies that protect against the most common and severe types of software vulnerabilities, including **100% of buffer overflows, buffer overreads, and code injection attacks**. This minimum level of protection will shut the door on the most common entrance paths for attackers to gain access to your AI system and take over.

Let's take buffer overflows as an example. CoreGuard's Heap micropolicy prevents buffer overflows and protects heap memory. A buffer overflow is the most common type of memory violation, accounting for over 12,000 known CVEs. It occurs when an instruction writes more data to a buffer than the buffer is designed to hold; data fills the intended buffer, and then overflows

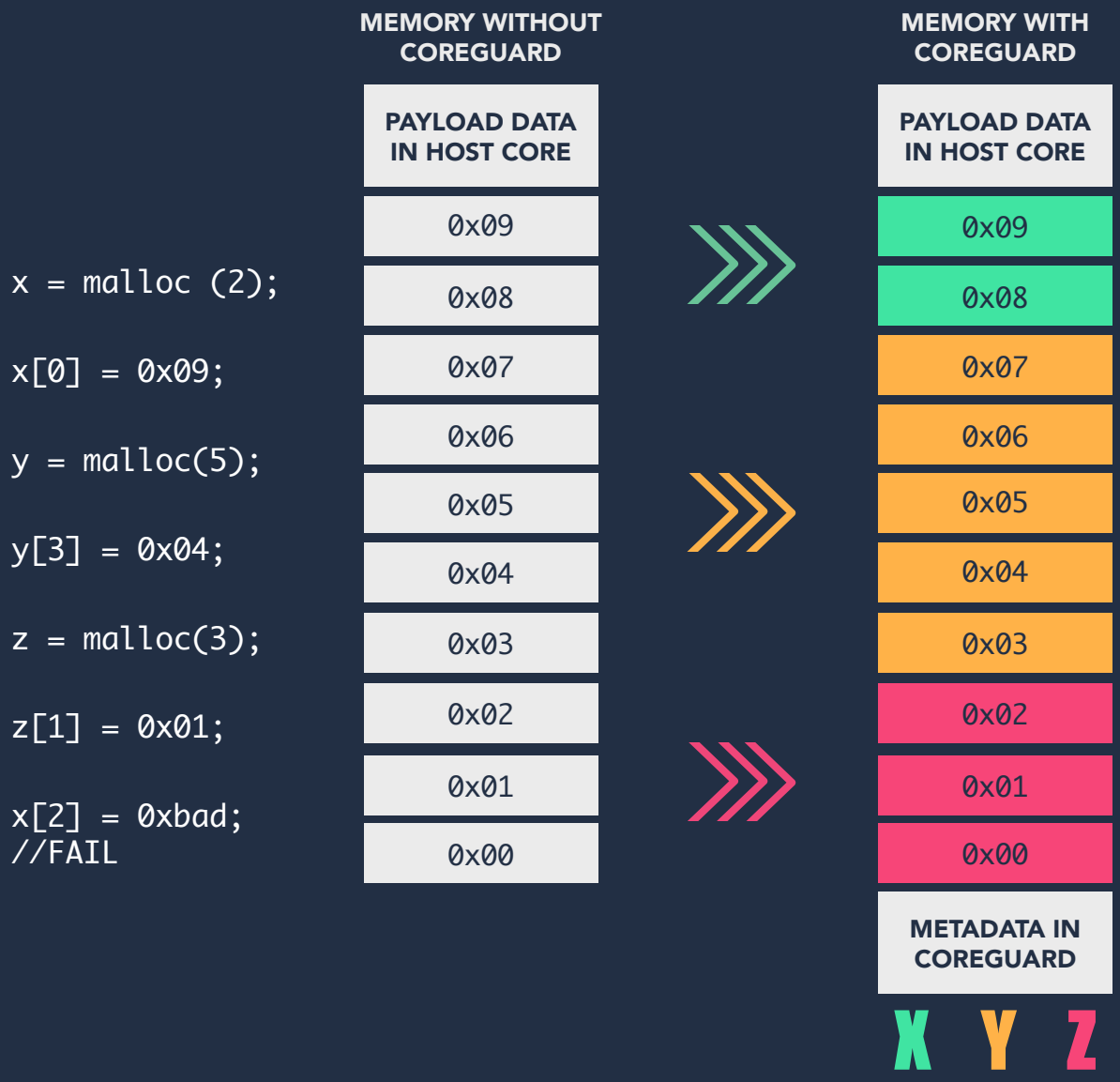
In figure 1, you see a picture of memory that has been "painted" with different colors, clearly showing each buffer. CoreGuard applied colors to the returned pointers (X, Y, or Z), as well as to all the words created by malloc.

The Heap micropolicy dictates that an instruction cannot write data to a buffer with a color that doesn't match the color of the buffer pointer. This means if an attacker attempted to use the X pointer (green) to overflow and write data into the orange buffer, CoreGuard would issue a violation and stop any memory writes from executing—thus shutting the door on the attacker.

FIGURE 1

CoreGuard Heap Micropolicy

Example Memory Coloring



MICROPOLICY RULE:

```
storeGrp(addr == color, mem == color > mem = color)
```

Metadata on pointer to: x y z

Metadata on address to STORE

Memory stays the same

DATA-ORIENTED CYBERATTACKS

A data-oriented attack is when an attacker is able to successfully manipulate either the training data of an AI system, or the real-world data that the system uses to make decisions, causing the AI device to malfunction and do something that it shouldn't, based on false data.

Data is a necessity for any AI system. Without data, AI wouldn't exist. There are several different types of attacks that target the data used by AI, including evasion, membership inference, and data poisoning attacks.

EVASION

An evasion attack, also known as an adversarial attack, is an attack that attempts to deceive the training models used by the AI system with a deceptive input—like an optical illusion, but for a machine. This false input can look like anything, from filling an email with certain words in order to **subvert an email spam filter**, to placing a small piece of tape on a stop sign to prevent a self-driving car from recognizing it as a stop sign.

MEMBERSHIP INFERENCE

A membership inference attack is when **an attacker tries to determine whether or not sample data is part of the training data set**. The attacker accomplishes this by constructing shadow models that can recognize differences in the target model's behavior and uses them to identify whether data is a part of training data based on the target model's output. If the sample data is linked to a specific person, such as medical or financial information, determining whether samples were part of the training data could result in a privacy threat.

POISONING

A poisoning attack happens when the attacker is able to **inject bad data into the AI system's training data** and over time, negatively impacting the AI system's ability to make correct decisions. Poisoning attacks are easier for attackers to carry out because they don't require them to breach the security systems of a third party and historically, they've been more difficult to detect.

Authenticity of the Data is a Major Problem for AI

As mentioned earlier, AI systems rely on sensor data to make decisions. In order to ensure that the sensor data being sent to an AI system is protected, it should first be signed so that the authenticity of the data is proven. As it is received, the data undergoes digital signature authentication. This proves the data comes from a trusted source (a sensor) and has not been altered. Once verified, the data can safely be used by the AI inference engine.

In the case of an autonomous forklift, once the sensors collect data either in the form of Lidar readings or images, that data is signed by the sensor and is then sent to the AI system, which authenticates the digital signature on the data before processing. Once processed, the AI system sends instructions to the controller of the forklift in the form of data, which is signed again before leaving the AI system. When that data is received, it is signature verified by the navigation controller

before the instruction is carried out.

When the AI system sends a command to the controller, the data must again be signed before sending. The SoC of the controller would perform the same signature authentication of the data being sent from the AI system before allowing it to take action.

It is during the signing and signature verification process when data that feeds AI systems is the most vulnerable. An attacker can gain access to a system that incorporates AI by finding and exploiting a software vulnerability, like a buffer overflow. Once they have gained access to the system, they can then execute a code injection attack which would allow them to intercept the data going to and from the AI system by monitoring a specific memory location or function that is part of the data transmission process. The attacker can then alter or replace the data being fed into the AI system, making the AI-powered device do something it was not intended to do. When you consider the heavy-duty machinery that uses AI, **the consequences of an attack could be extremely dangerous and severe**.

Take for example, the autonomous forklift in a smart factory—its distance readings could be manipulated by an attacker, causing it to crash. In a situation such as this, the destruction of this expensive piece of machinery is a best-case scenario. In a worst-case scenario, the people working in the factory side-by-side with that forklift are at risk of serious injury or even death.

Poisoning Machine Learning

Manipulating the data being sent to or coming from an AI system is not the only type of data-oriented attack threatening AI today. As previously highlighted, a particularly AI-savvy attacker could corrupt the machine learning process—via a poisoning attack—by feeding incorrect training data into the system over a longer period of time so that the system learns from false data, rather than legitimate sensor readings.

This type of attack is particularly relevant in situations where predictive maintenance is used. **Predictive maintenance** is a common application that uses AI to predict when a machine will need maintenance, in order to prevent the breakdown of the machine. It is used across industries to predict anything from mechanical issues on aircrafts which will result in flight delays, to predicting when an industrial refrigerator will need maintenance to continue functioning properly at a food production facility.

For predictive maintenance to work, it relies on high-quality, accurate data readings. If an attacker is able to manipulate those data readings, the AI system will not have a clear or accurate picture with which to make predictions. This could mean that aircrafts requiring maintenance will go unnoticed and a malfunction could occur mid-flight or thousands of dollars' worth of inventory spoiling because the refrigerator storing it did not maintain a safe temperature.

How CoreGuard Can Ensure Data Authenticity

For AI systems which can have such high stakes if compromised, shutting down the most common weak points are not enough. A sophisticated and determined attacker could find another way in, so a defense-in-depth approach is the only way to truly protect your AI system.

As mentioned earlier, the data that powers our AI systems is most vulnerable just before the signing and just after the signature verification process. If an attacker can intercept and alter that data, your AI system could be severely compromised.

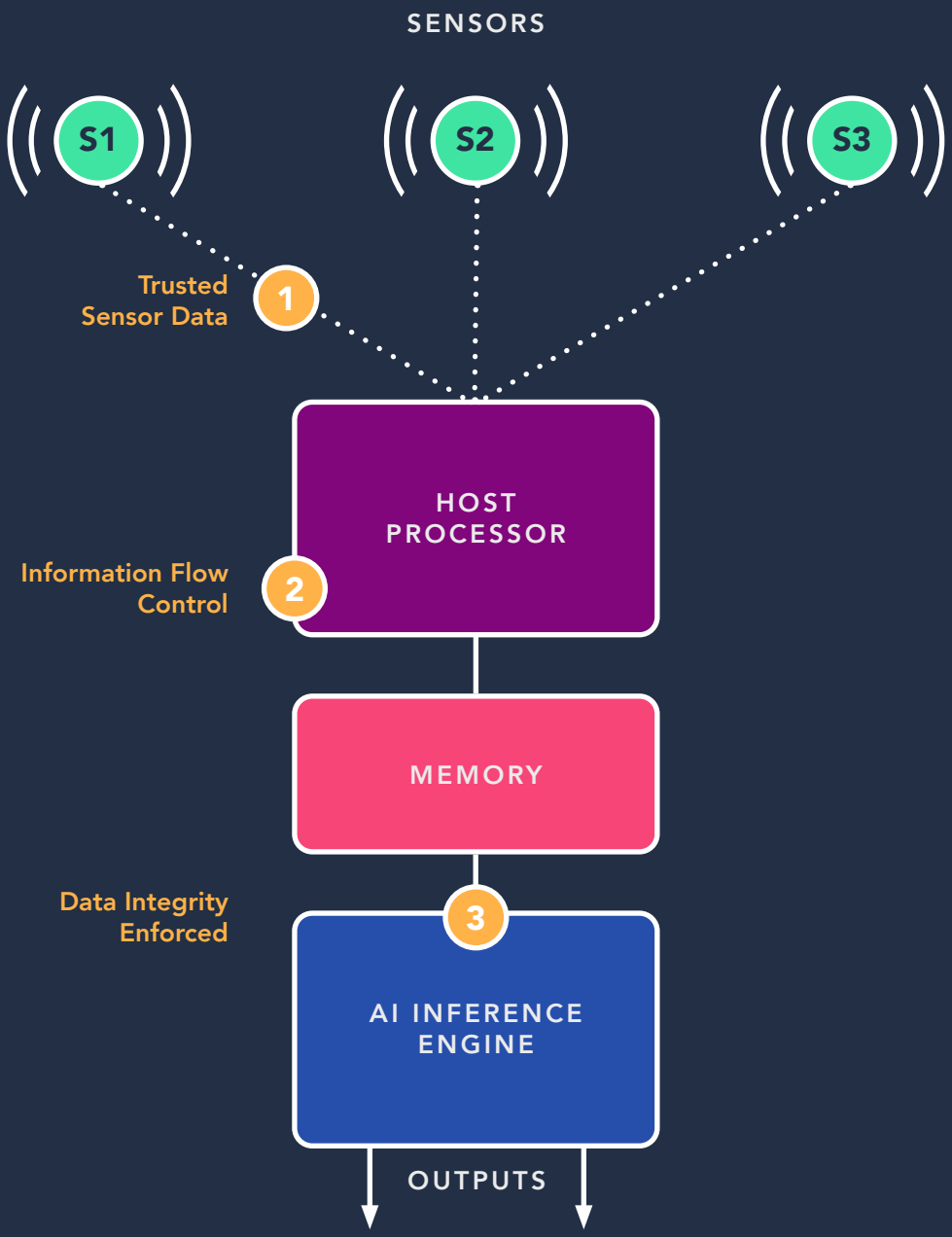
CoreGuard can ensure data authenticity with our AI Data Integrity micropolicy. This micropolicy prevents the modification of data between digital signature authentication and the AI system, ensuring only trusted and secure data is fed into the system. So, how does the AI Data Integrity micropolicy work?

First, the micropolicy taints all sensor data as "trusted." Then, all data is tracked through computations. If the tainted data is modified in any way, the "trusted" label is removed. This ability to trace the flow of specific words throughout their life cycle and to know each and every instruction that operates on those words is a unique and extremely powerful capability of CoreGuard called Information Flow Control.

It means that if an attacker tries to add or subtract some value from any data words in the stream, CoreGuard detects that and instantly removes the taint which removes the "trusted" property from that data. CoreGuard then enforces that only tainted data with the "trusted" label is fed into the AI system. This means that a system with CoreGuard installed will prevent an attacker from writing any incorrect or manipulated data to the AI system, as it will not be tainted with the "trusted" label.

FIGURE 2

CoreGuard-Protected AI Data Flow



MICROPOLICY RULE FORMAT: (InstrType, InstrTag, SourceTag) > (NewDestTag)

1. "Taint" data as trusted from sensor:
   (Load, —, SensorMem) > (trusted)
2. Remove taint if data is modified
   (Add, —, trusted) > (untrusted)
3. Require trusted data before use by IE
   (Load, IEEntry, untrusted) > (violation)

AI POSES A SERIOUS RISK WITHOUT BETTER SECURITY

The increasing adoption of AI is changing lives, but it's also posing a very serious—and potentially dangerous—security problem.

A report by the World Economic Forum warned that for too many device manufacturers, "security-by-design" is still a secondary concern to simply getting the products to market. This strategy is leaving systems, particularly those with AI capabilities, vulnerable to cyberattacks. The adoption of AI is creating a cyber-physical world, and cyber-physical attacks have serious, even life-threatening consequences.

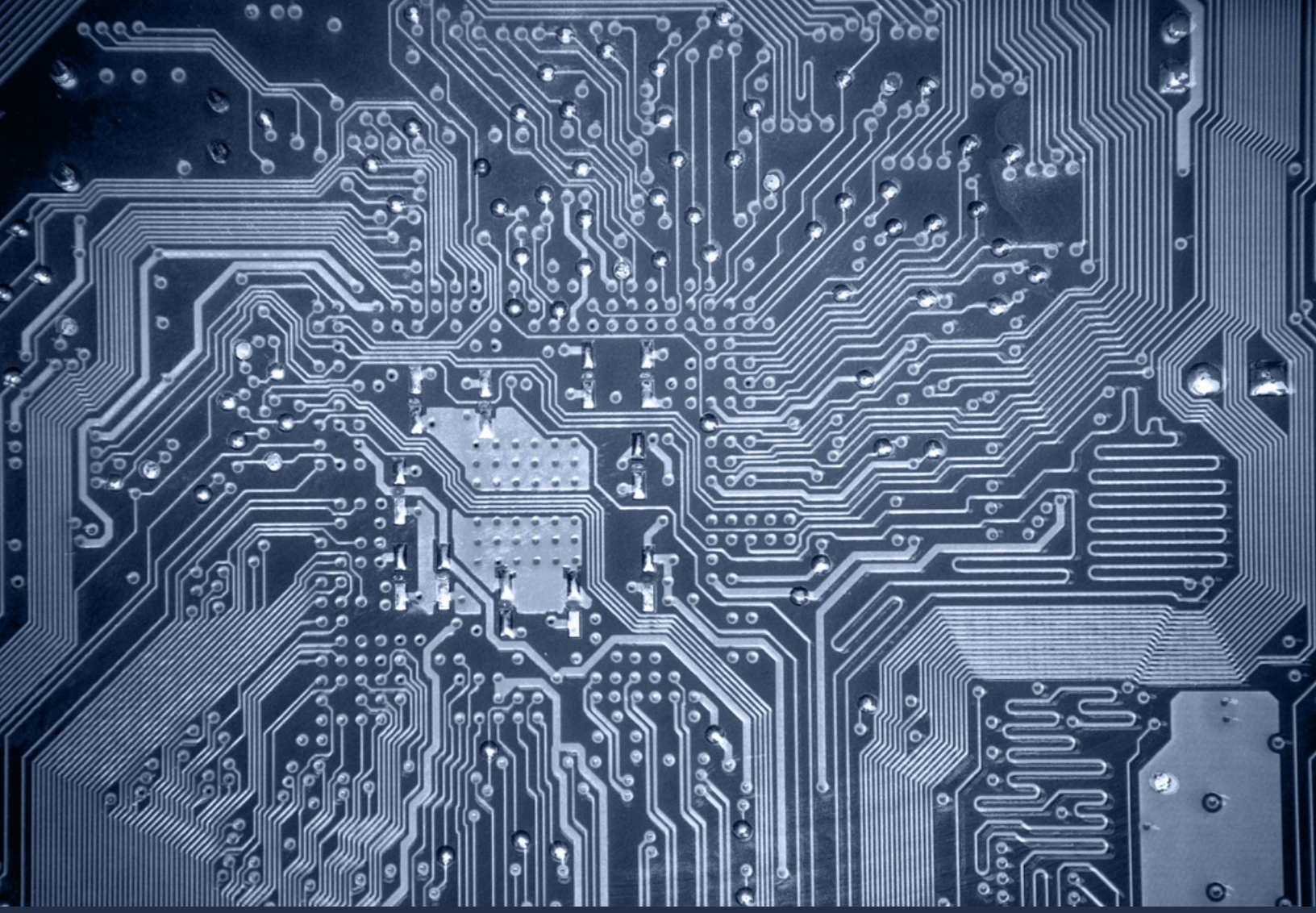
All of these risks are only amplified by roll out of 5G, which will support the connection of billions of devices, many of which will come equipped with AI applications. Of course, these attacks are not just hypothetical. **Attackers have already proven they can exploit software vulnerabilities to take control of systems with a vehicle**, and they didn't even need the potentially 50 million vulnerabilities present in an autonomous vehicle to do it.

AI is some of the largest and most complex software out there, making it even more vulnerable to control-oriented attacks. On top of that, when you take complex software and put it in charge of making decisions for things like self-driving cars, the consequences of a "wrong" decision could be dire.

The only way to address this problem is with a cybersecurity solution that can stop both control-oriented and data-oriented attacks. CoreGuard is that solution.

See CoreGuard in Action

REQUEST A DEMO



Learn More: info@dovermicrosystems.com

DOVER MICROSYSTEMS

About Dover Microsystems

Dover's lineage began in 2010 as the largest performer on the DARPA CRASH program. In 2015, Dover began development inside Draper before spinning out in 2017.

Based in Boston, Dover is the first company to fill the Enforcement layer of the cybersecurity stack. Dover's CoreGuard is the only solution for embedded systems that prevents the exploitation of software vulnerabilities and immunizes processors against entire classes of network-based attacks.

www.dovermicrosystems.com