

Attack the Glass: Introduction

Introduction to the Attack the Glass (ATG) Vulnerability

v2.4.7, September 20, 2026

TLP:CLEAR

Contact: Robert Beckett

Signal: ndiligence.99

Email: atg@nDiligence.com

Table of Contents

Introduction 4
Truth Is a Decision 4
How the Brain Automates Trust 5
With My Own Eyes 6
The Glass 7
Deception Works on Shortcuts 8
The Glass as an Attack Surface 9
Automation Bias 10
Decisions Have Consequences 11
The Decision Is the Target 12
The Vulnerability 13
Your Device Builds the Screen 13
A Supply Chain Nobody Sees 15
Forced Trust and Blind Trust 16
Swapping What You See 17
No Alarms, Because No One Is Watching 18
What About the Protections We Already Have? 19
No Trace and No Proof 20
Aimed at You 21
Artificial Intelligence Changes the Scale 23
Weapons of Mass Deception 24
Deceive, Disrupt, Degrade, Deny, Destroy 26
The Operations 29
The Internet Is a Supply Chain 29
Getting Into Position 29
Many Ways In, and Most Are Not Hacking 30
Propaganda Is Old, the Delivery Is New 31
Buying a Position of Trust 32

Anyone in Position 33
Digital Supernodes 34
The Impact 35
Losing Trust in the Glass 35
What Fails, and for Whom 35
What Is Known 37
The Risks It Implies 38
What Nobody Can See 38
What We Need to Do Now 38
The Work Still to Be Done 39

Introduction

This is about deception. It is about how people can be tricked, manipulated, and misled by the screens they trust most: the phone in their hand, the laptop on their desk, the display they work from every day. It is about a cybersecurity vulnerability that gives an attacker control over what your screen shows and what it does. The attacker can change the information in front of you. The attacker can add things that were never there. The attacker can take away what you needed to see: the warning, the fee, the alternative, the button that would have let you say no. A great many of the decisions we make each day rest on information shown to us on a screen, and the same screen often captures the decision and sets the action in motion: the payment is sent, the order is placed, the valve is opened. Whoever controls the screen can influence what people believe, the opinions they form, the decisions they make, and the actions they take. This is disinformation in its most direct form: a false experience, on a screen you had every reason to believe.

We call this cybersecurity vulnerability Attack the Glass (ATG). It exploits a weakness in our existing cybersecurity architecture through a supply chain injection on client-side rendering. Most of the screens around us are dynamic, created in the instant before they are shown to you. What appears is often shaped by who you are, where you are, what you are doing, why and when you are doing it, and even how you are interacting with the system. That real-time assembly of the user interface is accomplished using client-side rendering. The various companies and individuals that provide the data, media, components, code, and other assets are part of the vast, global internet supply chain. Anyone who controls one of those assets can quietly change what your screen shows and what it does, and every security tool in use today will report that all is well. An attacker can aim the change at everyone, at one group, or at one person, which makes the deception more persuasive and leaves almost nobody in a position to notice. Artificial intelligence can now write a different deception for every one of those people, in seconds, at almost no cost. The change happens in the memory of the device and is gone when the screen goes dark, so an attack that could not be detected while it ran cannot be proven afterward.

Truth Is a Decision

People act on what they believe to be true at the moment they act. A nurse gives a dose because she believes the chart. A trader sells because he believes the price. A customer pays, a driver brakes, an operator opens a valve, each acting on a picture of the world they did not build themselves and have no time to verify.

Nobody checks most of what they believe. Nobody could. A person cannot test the water before every glass, audit the bank before every deposit, or confirm each headline with a

second source. People do something else. They decide whom and what to trust, and then they take its word. These choices of who we trust anchor nearly everything a person holds to be true.

Truth, as people live it, is a decision. It is made quickly, many times a day, usually without noticing, and it is made about the source far more often than about the claim. My bank says this is my balance. The clinic says the appointment is Tuesday. The news says the bridge is closed. The person has verified none of it and acts on all of it, because of who said it.

How the Brain Automates Trust

Trust is one of the most common decisions a person makes, and the brain does with it what it does with anything repeated often enough: it turns the decision into a habit. The first time you used your bank's app, you may have looked it over with some care. By the hundredth time you were checking a balance while crossing the street. Psychologists call these mental shortcuts heuristics. They are rules the brain applies automatically so that it can save its attention for whatever is new. "This is my bank's app, and it has always been right" is a heuristic. So is "this newspaper checks its facts," and so is "this is the screen I use at work every day."

Shortcuts make daily life possible. Nobody could get through a morning while weighing every source from scratch. They also carry a cost. Once trust has become automatic, the brain stops inspecting. It ties the trust to something familiar, a name, a logo, a face, the look of a page, and it lets whatever arrives under that sign pass without a second look. Small things that are out of place go unseen, because nothing is looking for them.

Ordinary life is full of examples. Hardly anyone examines a twenty-dollar bill, which is the only reason counterfeit money works. A stranger in a high-visibility vest carrying a ladder walks past most front desks unchallenged, because the vest answers the question before anyone asks it. An email that appears to come from the chief executive, asking the finance team for an urgent wire transfer, succeeds often enough that the FBI counts the losses from that one trick in billions of dollars a year. In each case the victim checked the sign, a banknote's look, a uniform, a sender's name, and the sign was all the deceiver needed to supply.

Every con artist, forger, and propagandist in history has worked in that gap. The reliable way to deceive someone is to borrow a source they already trust and speak in its voice.

With My Own Eyes

Before we trust any person or any institution, we trust our own senses. "I saw it with my own eyes" is the strongest claim most people know how to make, and courts, newsrooms, and families all treat it that way.

Human beings take in the world through five senses, and vision dominates the other four. When sight and another sense disagree, the brain sides with sight, and it does so before we know there was a disagreement. In a classic experiment, people were shown a light and played a tone at the same instant. They responded to the light almost every time, and on a third of those trials they never noticed the tone at all. None of this is a choice. Nobody decides to weight their eyes over their ears, and nobody can decide to stop.

Sight and sound together are stronger still. When what we see and what we hear arrive at the same moment and agree, the brain fuses them into a single experience that is easier to understand, easier to remember, and harder to doubt than either one alone. Anyone who has sat in a cinema has felt it. The speakers are on the walls, and every voice seems to come from the mouth of the actor on the screen. The brain moves the sound to where the eyes say it belongs, and knowing how the effect works does nothing to stop it. A screen that delivers matched picture and sound is using the richest channel there is into the human brain.

People also learn by watching. We see something done and know how to do it. We watch something happen to another person and feel it ourselves. Human beings have handed down knowledge and experience this way since long before writing, and it has never required being in the room. Children who watch an adult on film imitate what they saw as readily as those who were there in person. Adults who watch a video of a stranger receiving electric shocks at the sight of a colored square display the physical signs of fear when it later appears in front of them, though nothing has ever happened to them. The brain files what it watches alongside what it lives through.

What we see shapes what we judge to be true. People rate the same statement as more likely to be true when it is printed in type that is easier to read. They find a claim more believable when an unrelated photograph sits beside it. They rate a statement as more accurate the second time they see it, even when they know it is false. None of these changes what is said. Each changes only how it appears, or how often. Each effect is small, and each can be repeated without limit.

What we see can also replace what we remember. People shown a doctored photograph of a public event come to remember the event the way the image shows it. When the truth

arrives later, it does not fully undo the damage: people go on reasoning from the first version they saw.

For nearly all of human history, everything a person saw or heard was physically present: a face, a fire, an animal, a voice across a room. Our senses, and the trust we place in them, were formed in that world over hundreds of thousands of years. Screens have existed for about a century, and the ones we carry for less than twenty years. The eyes and ears have made no adjustment. Light from a screen enters the same eye and is handled by the same machinery as light from a face, and the brain has no separate, more skeptical channel for things that are digital. We experience what is on a screen with senses that were built for what is physically there.

The gap between ancient senses and digital screens is about to be tested harder. Artificial intelligence can now produce a face, a voice, a document, or a video of something that never happened, and the number of screens around us grows every year. Some countries have started treating the public's ability to resist deception as a matter of national defense. Sweden set up its Psychological Defence Agency in 2022 to help a whole society detect and resist attempts to manipulate what it believes. A defense of that kind starts with knowing how deception reaches us, and today most of it arrives through a screen.

The Glass

Look around the room you are in and count the screens. A phone, almost certainly, within reach. A laptop or a monitor. A television. A watch. Beyond the room there is the dashboard of the car, the display on the thermostat and the refrigerator, the kiosk where you check in for a flight or order lunch, the card terminal at the counter, the pump at the gas station, the departure board, the menu above the register, the screen in the back of the seat in front of you. On the job there may be a point-of-sale system, a patient chart, a trading terminal, a dispatch board, or the control panel for a production line, a water plant, or a power grid.

There are billions of these surfaces, backlit and connected, and nearly all of them arrived within a single lifetime. Most arrived in the last twenty years. People spend a large part of their waking lives looking at them, and the screens call for attention when we look away, with alerts, notifications, vibrations, and chimes. Nearly every one of them can deliver picture and sound together.

We call all of them, together, **the Glass**. The Glass is any digital surface where a computer puts something in front of a human being to see, usually with sound to match. A browser window on a laptop is only the most familiar example. A phone app, a television menu, a vehicle display, an appliance panel, and an operator's console in a control room are the same thing under different housings, and they are built the same way.

Nothing in human history, or in the long evolution of the brain before it, resembles this. Never have so many eyes rested on so few kinds of surface for so many hours. Never has so much human attention been reachable so quickly, in every country at once, across every language and every culture. A story, an image, or an instruction can be in front of a billion people within minutes. The Glass is a direct channel, through sight and sound, to the place where people decide what is true.

The Glass matters for a second reason, which is where it sits. It is the last step in every digital system. Whatever happened upstream, in the data centers, the databases, the networks, and the security tools, the final act is always the same: an interface is put together on a pane of glass in front of a person, with its information, its choices, and its buttons, and they read it and act. It is the point where data stops being data and becomes what someone believes, and then what they do. A bank runs its systems so that a customer can see a balance and move money. A hospital runs its systems so that a nurse can see a dose and give it. A utility runs its systems so that an operator can see the state of the grid and act on it. Years of engineering sit behind each of those screens, and all of it exists to produce the few square inches a person looks at.

The Glass is also where trust now arrives. Trust starts small, with family, friends, a teacher, a doctor. It scales up to institutions we will never meet in person: a bank, a newspaper, a hospital, an employer, a brand we have bought for twenty years. Nobody audits their bank before checking a balance. They trust the name, and the name has usually earned it. Until recently that trust was carried by things you could hold and people you could see: a printed statement, a signed letter, a teller behind a counter. Nearly all of it now arrives through the Glass. For practical purposes, your bank is the screen that says it is your bank.

The Glass is three things at once. It is the surface our senses trust most. It is the channel through which the institutions we rely on reach us. And it is the place where we decide: to pay, to treat, to buy, to sell, to open the valve.

Anything that carries this much of what people believe ought to be among the most carefully secured things ever built. Today, at the moment a screen is put together in front of a person, almost nothing protects it.

Deception Works on Shortcuts

People hold true what trusted sources tell them. The brain has automated that trust and tied it to familiar signs. It believes its eyes above all else, and almost everything it now sees arrives on the Glass.

The best-known deception on a screen is phishing: an email or a text message leads to a fake website dressed up as a real one, and the victim types a password or a card number

into it. People have been taught a few checks against it, and the careful ones use them. Look at the address. Look for the padlock. Make sure the logo and the page look right. Be suspicious of the email with the odd spelling and the link that goes somewhere strange. Those checks are shortcuts too, and they all ask one question: is this really my bank, my newspaper, my employer?

Phishing fails those checks when people remember to make them, because the fake site has to live at a fake address. Attack the Glass passes them, because the answer is yes. The site is genuine. The address is correct, the padlock is showing, the logo is real, and the person signed in the way they always do. The change is made to what that genuine site shows, after every one of those checks has passed. The altered paragraph arrives under the newspaper's masthead and the reporter's byline. The false notice arrives inside the credit union's own site, after sign-in. The attacker borrows all of the trust the institution spent decades earning, and the person's own shortcuts wave the deception through.

The Glass as an Attack Surface

Security professionals talk about an attack surface: every place where an attacker can get at something. A house has doors and windows. A company has its network, its email, and its staff.

For thirty years, security has meant protecting what computers hold. Keep strangers out of the network. Keep the data from being stolen or changed. Keep the systems running. Enormous effort has gone into that work, and much of it has succeeded. Almost none has gone into the last few inches, the point where a finished screen meets a person's eyes, because it was assumed to be a plain display of what the protected systems already held.

That assumption no longer holds. The screen has become a place where things are assembled, and anything assembled can be tampered with during assembly. Every organization now reaches people through the Glass. Customers, citizens, clinicians, traders, and operators make decisions based on what those screens show them. The organization is accountable for what appears there, and believes it controls it.

The profession has a shorthand for its goals, often called the CIA triad: confidentiality, integrity, and availability. Confidentiality means secrets stay secret. Availability means systems stay up. Integrity means information is not altered by anyone who should not alter it. In practice integrity has been defended where data is stored and while it travels across networks. Attack the Glass is an attack on integrity at a place those defenses were never extended to: the point of display.

An attacker who can change what appears on the Glass has no need to break into the bank. The bank's records stay correct. The customer sees something else, acts on it, and the

bank faithfully carries out the request. A network can be secure, its data can be intact, and the people making decisions on its screens can still be acting on a lie.

Automation Bias

It is tempting to believe that professionals are different, and that a trained operator watching a familiar display would catch what an ordinary person would miss. The evidence runs the other way.

In June 1995 the cruise ship *Royal Majesty* left Bermuda for Boston with more than 1,500 people aboard. Less than an hour out, the cable to its GPS antenna came loose. The navigation system switched, without any alarm the crew noticed, to estimating the ship's position from its speed and heading, and it went on drawing the vessel neatly on its planned track. For more than thirty hours, experienced officers stood their watches in front of a display that said everything was fine. One of them sighted a buoy in the wrong place and took it for the one he expected, because the screen agreed. Lookouts reported lights and breaking water that should not have been there. The ship ran aground off Nantucket, seventeen miles from where its screen placed it. Investigators put the grounding down to the officers' overreliance on the automated system. Years of accurate readings had taught them to stop questioning it.

Researchers call this automation bias: people who work with an automated display accept what it tells them over evidence that contradicts it, and stop looking for any. It has been reproduced under controlled conditions many times. In a flight simulation, every airline pilot tested shut down an engine because an automated warning reported a fire that no other instrument showed. Afterward, two thirds of them remembered seeing other indications of fire. There were none. The false display drove the decision, and then it wrote the memory that justified it. In hospitals, physicians reading heart tracings have been pulled toward a computer's interpretation when the computer was wrong, and specialists examining mammograms have missed cancers that the detection software failed to mark.

Automation bias fails in two directions. People act on what the display wrongly shows, and they miss what it leaves out. It appears in experts as much as in novices. Neither an instruction to double-check nor a second person in the room removes it. It grows with reliability, which means the best-run systems produce the most trusting operators. After ten thousand correct readings, nobody checks the display.

All of those displays were wrong by accident. Attack the Glass describes one that is wrong on purpose, at a chosen moment, for a chosen person. The usual advice, to check against a second source, runs into a problem of its own: on a modern screen the second source is often another panel on the same Glass, put together the same way.

Decisions Have Consequences

A false screen does its damage through what a person then does. The harm lands in the world: money sent to the wrong account, a drug given at the wrong dose, a valve opened, a vote cast on a false story, a family that stays home because the evacuation notice never appeared.

Two well-known incidents show how fast a false message on a trusted channel turns into action. On April 23, 2013, hackers took over the Twitter account of the Associated Press and posted that explosions at the White House had injured the president. The message was false and was corrected within minutes. In those minutes the Dow Jones Industrial Average fell more than 140 points and roughly 136 billion dollars of stock market value vanished, because traders and automated trading programs acted on a trusted name before anyone could check. The market recovered. The lesson stayed: one false sentence from a trusted source briefly erased more value than any bank robbery in history has taken.

On January 13, 2018, phones across Hawaii displayed an official emergency alert: a ballistic missile was inbound, seek immediate shelter, this is not a drill. It was a mistake made by a state employee during a routine exercise, and it took 38 minutes to send a correction. In that time parents lowered children into storm drains, drivers abandoned cars on the highway, and people called their families to say goodbye. Nobody who received that alert did anything foolish. They believed an official message on their own screen, and they acted on it.

Neither incident was an Attack the Glass operation. One was a stolen password and the other was human error. Both show what the target of such an operation would be: the short distance between what a trusted screen says and what people do.

How much damage depends on who is looking. A forged memo handed to a receptionist cannot move money, because nobody at the front desk has authority over the accounts. The same memo handed to the treasurer can. The forgery is identical in both cases. What differs is the authority of the person who believes it.

Screens work the same way. Someone reading a news page can only read, so an attacker who alters that page misleads a reader and goes no further. A bank customer can move their own money, so an altered payment screen can send one person's savings to the wrong place. A payments clerk can move an employer's funds. A nurse can give a drug. A plant operator can open a valve. An IT administrator can delete every account the organization has. In each case the attacker never breaks into anything. The attacker changes what an authorized person sees, or acts through the access already granted, and that individual's own authority does the rest.

The most powerful screens are also the ones fewest people use and the hardest to reach. An attacker who goes after an administrator's console takes on a smaller, better-defended target in exchange for a larger result. An attacker who goes after a news page takes on an easy target and reaches millions of readers, each of whom can only be misled. Organizations usually protect the first kind of screen far more carefully than the second, and neither kind is protected against a change that arrives from a source the organization chose to trust.

Some of the damage is physical and immediate. Some of it is social. A rumor that a bank is failing can start a run on it, and screens have made runs faster than any in history: when doubts about Silicon Valley Bank spread through social media and group chats in March 2023, customers used their phones to pull 42 billion dollars out in a single day, and the bank was closed the next morning. False stories shown to one community about another can end in violence. In India in 2017 and 2018, rumors about child kidnappers passed from phone to phone on WhatsApp, and mobs killed about thirty innocent people, many of them strangers passing through a village. And a public that has been fooled once believes less of what it is told afterward, including what is true.

The Decision Is the Target

Most cyberattacks go after what a computer holds. They steal data, lock it up for ransom, or shut a system down. Attack the Glass goes after what a person decides. It is a technical vulnerability, and its target is human: the moment when someone reads a screen, believes it, and acts. It reaches that moment through the two things people trust most, the institutions they have chosen to rely on and the evidence of their own eyes.

Security professionals have a name for attacks that work on a person instead of a machine: social engineering. The con artist on the phone who claims to be from the bank is a social engineer, and so is the author of a phishing email. Attack the Glass is social engineering carried out by technical means, with one difference that matters a great deal. There is no stranger in the conversation for the victim to be suspicious of. The message comes from the institution's own screen.

The Vulnerability

Attack the Glass is a weakness in the way modern screens are put together. It lets someone other than the company you are dealing with change what appears on your screen, and what your screen does, at the moment it is assembled in front of you. The company cannot see what appeared on your device. You cannot see what the company meant to send. Neither of you has any way to compare the two, so neither of you learns that anything was changed.

Your Device Builds the Screen

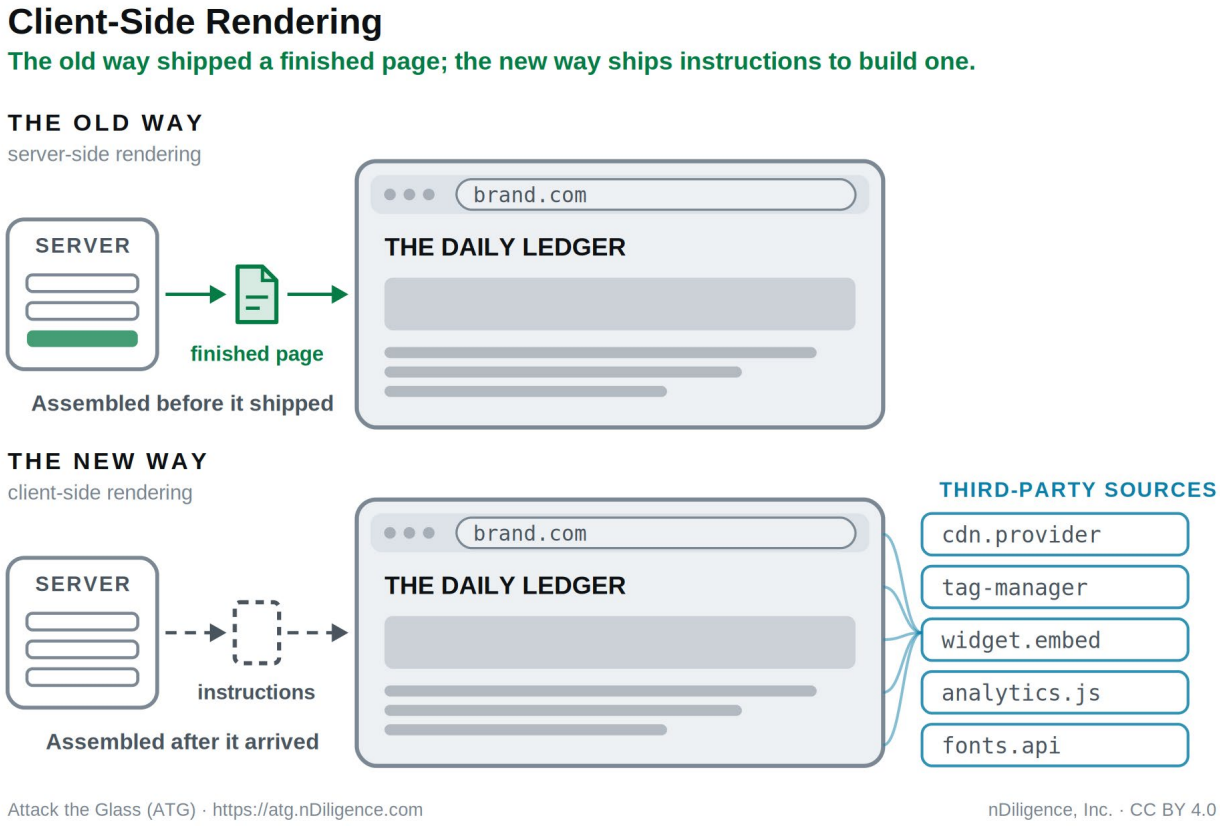


***Figure 1**: Server-side rendering delivers a finished page. Client-side rendering delivers the parts and the instructions to assemble them.*

Most people picture a web page or an app screen as a finished thing that a company sends them, the way a printer sends out a newspaper. In the early years of the web that picture was accurate. A company's own computer, called a server, put the whole page together and sent it out complete, and your browser simply displayed what it received. Engineers call that server-side rendering, because the work of drawing the page was done there.

For more than a decade it has worked differently. When you open a modern site or app, the company sends a set of instructions, and your phone or laptop builds the screen itself, on

the spot. The instructions are written in a programming language called JavaScript, which every web browser can run, and a modern page is less a document than a small program that starts up each time you open it. Following those instructions, your device collects dozens or hundreds of separate resources, most of them from other companies: program code, fonts, images, video players, advertising, maps, chat windows, sign-in and payment services, tools that measure what visitors do, and increasingly AI services. Engineers call this client-side rendering. The client is your device, and rendering is the work of drawing the screen. The period while that program is running on your device, fetching and assembling and redrawing, is called runtime.

Building a screen this way is the difference between buying a finished table and receiving a sheet of instructions that tells you to collect the legs from one supplier, the top from another, and the screws from a third.

Take an ordinary article on a news site. The words of the article come from the publisher. The typeface comes from a font company. The video at the top plays in a player supplied by a video company. Each advertisement is sold in an automated auction that runs in the fraction of a second while the page loads, and the winning advertiser, whom the publisher has never met, supplies its own code. A measurement service counts the visit. Another tool tests two versions of the headline against each other. The comment box, the share buttons, the newsletter sign-up, and the cookie consent banner each come from a different firm. Surveys of the web find that more than nine pages in ten load resources from outside companies, and a typical page makes about eighty such requests.

The pattern is not limited to web pages in a browser. Most phone apps are built with kits of ready-made code from other companies, and many display web content inside the app. Smart televisions, the screens in cars, check-in kiosks, and a growing share of industrial and medical displays are built on the same web technology, because it is cheap, flexible, and familiar to every programmer. A page built the older way, on the company's server, does not escape either. It still loads outside resources for advertising, measurement, payments, and sign-in, and those still run on your device with the full run of the page.

Companies build screens this way for good reasons. It is faster, it is cheaper, and it lets them use excellent services they could never afford to create themselves. It also moves the final act of assembly out of the company's hands. The screen you look at is put together on your device, in the moment before you see it, from resources the company does not make, does not store, and in most cases never sees.

That moment of assembly, on your device, is where Attack the Glass takes place. The attack does not happen on the company's computers, and nothing crosses the defenses the company has built around them. It happens on your phone, your laptop, your television,

the display in your car, in the instant the screen is being assembled. Nobody is guarding that instant. The company's security team cannot see into your device, and you were never told there was anything to guard.

A Supply Chain Nobody Sees

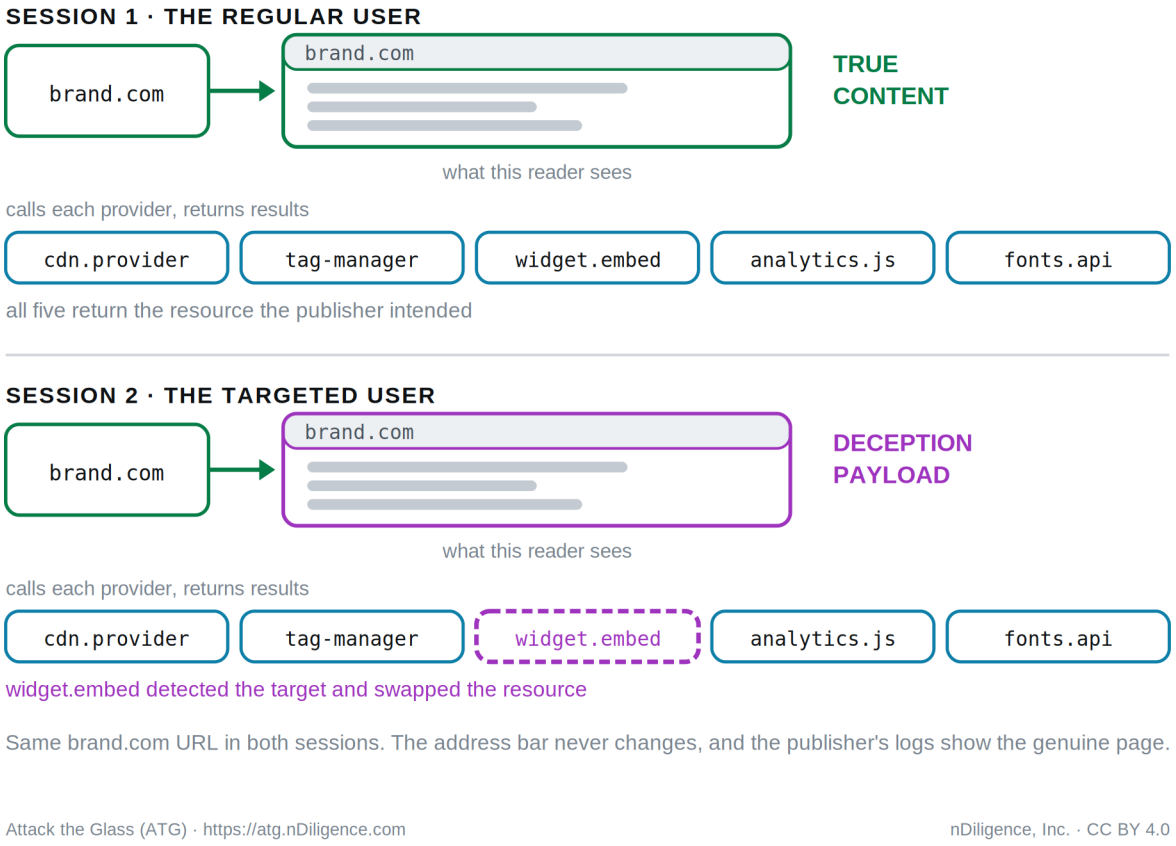


***Figure 2**: The page the publisher serves is not the page the reader receives, and neither of them can tell.*

Every one of those resources comes from a supplier, and most suppliers have suppliers of their own. The mechanism is almost absurdly simple. A developer adds a single line to the page that says, in effect, "fetch whatever is at this web address and run it." That line is called a script tag. The trust it grants belongs to the address. It says nothing about the contents, and the supplier can change what sits at that address at any time, which is how suppliers deliver updates and the reason companies like the arrangement.

A company adds a chat window from another firm. The chat window brings in a tool from a third firm, which pulls in something from a fourth. Many companies also use a tag manager, a tool that lets the marketing department add new outside scripts to the site without asking a programmer, and often before anyone in security hears about it. Your device fetches all of

it and runs all of it as though it came from the company whose name is at the top of the screen.

Browsers do have a rule for keeping websites apart, called the same-origin policy. A page from one site cannot read or change a page from another, which is why a shopping site open in one tab cannot look at your bank in the next. The rule stops at the door, though. An outside script that a page has invited in runs as part of that page, with the page's full powers. It can read everything you type, including passwords and card numbers, and it can rewrite anything you see. Programmers call the live, editable structure of a page the Document Object Model, or DOM, and any script on the page may change any part of it at any moment, including long after the page has finished loading.

Whoever controls any one of those resources can change what it delivers. When they do, the change is injected into the screen as your device builds it, which is why the technical name for this kind of attack is a supply chain injection. One altered resource is enough. It can rewrite the words on the page, replace a number, add a form, remove a button, or change what happens when you press one. All of it appears under the brand of the company you thought you were dealing with.

This has been done, many times, for theft. In 2018 attackers altered the code of a chat service that Ticketmaster had placed on its payment pages, and it quietly copied customers' card details as they typed. The same year, 22 lines of added code on the British Airways site sent the personal and payment details of more than 400,000 people to a server the thieves controlled, and the United Kingdom's data protection regulator later fined the airline 20 million pounds. The security industry calls this web skimming, or Magecart after the criminal groups that pioneered it, and it has hit thousands of online shops. In March 2016, visitors to the websites of the New York Times, the BBC, and other major publishers were served advertisements that tried to install ransomware. None of those publishers had been hacked. The poisoned ads arrived through the advertising supply chain, and the industry gave that a name too: malvertising.

Those cases were about stealing, which leaves victims who eventually notice missing money. The thieves used the very access Attack the Glass describes. Code that can copy a card number off a page can just as easily change a paragraph on it.

Forced Trust and Blind Trust

Two kinds of trust hold this arrangement together, and neither involves anybody checking anything.

The first is forced on you. The company's developers decided which outside suppliers your device would fetch from and run. Your device does as it is told. You were never shown the

list, never asked to agree to it, and cannot look at what any supplier on it actually sends. You cannot turn one of them down and keep using the app. Tools such as ad blockers can refuse a supplier outright, but no tool can inspect what an accepted one delivered. The trust is real and binding, and someone else entered into it on your behalf.

The second is blind, and it belongs to the company. Most of those suppliers were chosen because they are popular, well known, and often free. Millions of other sites use them, which feels like safety in numbers. Hardly anyone verified what the code does, who owns the firm behind it, or who that firm depends on in turn. Ask most organizations to name every company whose resources load on their screens and they cannot, because no such list has ever been written down. Nobody approved the full set, because it has never been seen in one place.

A small episode from 2016 shows how deep the dependence runs and how little of it anyone sees. A programmer, annoyed by a dispute over a name, deleted a tiny piece of free code he had published for others to use. It was eleven lines long and did one trivial job: padding out the left side of a line of text. Within hours, software projects around the world stopped building, including tools relied on by companies such as Facebook and Netflix and by thousands of smaller ones. Almost none of them knew the code existed, because they had never chosen it. A tool they relied on had pulled in another, which depended on those eleven lines.

The person at the screen trusts the company. The company trusts its suppliers on reputation, and they in turn trust theirs the same way. Everyone in the chain is trusting, and nobody in it is verifying.

Swapping What You See

What an attacker does from that position is simple to describe: swap one thing on the screen for another. We call it Content Swapping. The account number on a payment page becomes a different one. The telephone number for the bank's fraud line becomes the attacker's own. One paragraph of a news story is replaced. A lab result, a dose, a price, an interest rate, a delivery address, a chart, or a download link is exchanged for another. A warning is taken off the page, or the button that would have let you cancel.

Follow one case through. A bookkeeper at a small construction firm signs in to a supplier's online portal to pay an invoice, as she does every month. The address is right, the padlock is showing, her password works, and the invoice on the screen has the right amount and reference number. The bank account number printed under "pay to" belongs to someone else. It was swapped on her screen by an altered resource from one of the portal's outside suppliers. She types the number into her bank, exactly as the screen shows it, and

approves the payment. The supplier's records still hold the true account number. The bank processes a valid payment from an authorized customer. Three weeks later the supplier asks why it has not been paid, and each side can show the other records proving it did everything correctly.

Criminals have already shown that this kind of swap deceives people at scale. From 2007, families of bank-fraud software, first Zeus and later SpyEye, infected millions of home computers and then waited for the owner to visit a bank. When the bank's page loaded, the software changed it inside the victim's browser, adding fields, hiding transactions, and replacing payee account numbers, and customers approved transfers they believed they had written themselves. Banks called the technique a man-in-the-browser attack. It needed malicious software installed on each victim's computer, which antivirus companies eventually learned to find. Attack the Glass produces the same altered page with nothing installed on anybody's computer.

The swap can happen after the page has loaded, while you are already reading it. It can last for a few seconds and then be put back. With generative AI running on the device, the replacement can be written on the spot for the one person looking at it, in their language, about their town, their employer, or their money.

Everything else on the screen stays exactly as it should be, which is what makes the altered part believable.

No Alarms, Because No One Is Watching

A company hit this way does not have a failing security team. Its defenses were built to protect what the company runs, its own computers, networks, and data, and this attack never touches them.

Nearly every security check in use today inspects a thing before it is used. Code is reviewed before it is published. Software is scanned before it is installed. A supplier is approved before it is added to a list. Once a resource has passed and started running on your device, nothing looks at it again. The kitchen passed its inspection. The plate was changed between the kitchen and the table.

Detection is the usual answer when prevention fails, and here detection has nothing to work with. There is no malicious file to find, because nothing resembling harmful software is ever installed. There is no fake website to block, because the source is one the company deliberately chose. There is nothing odd in the company's records, because every computer the company owns behaved normally. Security tools are built to catch bad software from a bad source. This is harm done with clean software from a trusted one.

In the common case today, nothing on the device is watching either. No mainstream browser, app platform, or device checks, while a screen is running, that the resources it fetched are the ones it was supposed to receive. The deeper cause is in the design of the platforms themselves, which were built to fetch whatever a trusted address returns and run it.

What About the Protections We Already Have?

People who work in security will point to a list of protections at this stage. Every one of them is real and worth having, and each stops short of the moment this attack uses.

The padlock. The padlock in the address bar stands for HTTPS, the encryption that protects a web connection, which engineers also call TLS. It guarantees two things: that you are connected to the genuine address, and that nobody altered the traffic on its way to you. It makes no promise about what the owner of that address chose to send. If a supplier's address is under an attacker's control, or has changed hands, encryption faithfully protects the delivery of the altered content.

Firewalls and the company's own monitoring. A firewall, its web application cousin, and the monitoring systems that collect security records all watch the company's own computers and networks. The exchange between your device and an outside supplier never passes through them.

Antivirus. Antivirus software, and the corporate version usually called endpoint detection, looks for malicious programs installed on a device. Nothing is installed here. A script running inside a browser tab is what browsers are for.

Code review, scanning, and ingredient lists. Careful companies review their code, scan it for known flaws, and increasingly keep a software bill of materials, a list of the ingredients built into a product before it ships. All of that examines software at rest, before release. It does not cover what a running page fetches from other companies afterward.

App stores and code signing. Phone apps are reviewed by the app store and carry a digital signature proving who made them. The review and the signature cover the app as it was submitted. Many apps then fetch content, settings, and sometimes program code from the internet every time they run, and none of that passes back through the store.

Content Security Policy. Content Security Policy, or CSP, lets a site give the browser a list of the addresses it may load from. It is the most useful control available in a browser today, and it is a list of approved senders. It approves the sender, not the package. When an approved address sends altered content, the policy has done its job and the change runs.

Subresource Integrity. Subresource Integrity, or SRI, is the closest thing the web has to a tamper-evident seal. The site records a digital fingerprint of a resource, and the browser refuses anything that does not match. It is worth using, and it has three limits. It is optional, and most sites do not use it. It cannot be applied to a resource that is supposed to change from one visit to the next, which describes a large share of what modern pages load, advertising and measurement scripts among them. And it checks a resource once, as it arrives. A script that passes the check can go on to fetch fresh instructions while it runs, so the code is genuine and the harm arrives in the data.

Zero Trust. Zero Trust is the security model most large organizations are now adopting, and the name suggests it should cover this. It is a model for deciding who and what is allowed to reach a company's systems, checked every time, before access is granted. It has nothing to say about whether content is still correct after it has been delivered to someone's screen.

Ad blockers. Ad blockers, script blockers, and corporate filtering can refuse a supplier altogether, and they work. None of them can inspect what a permitted supplier sent, or accept one on conditions.

Cross-site scripting. Security professionals will notice a family resemblance to cross-site scripting, or XSS, one of the oldest attacks on websites. XSS exploits a mistake in one particular site, which that site's owner can find and fix. Attack the Glass needs no mistake anywhere. The site correctly fetches from an address it was set up to trust, and the device correctly runs what that address returned. Nothing is broken, so there is nothing to patch.

No Trace and No Proof

A burglar leaves a broken lock. A forger leaves the forged check. Most cyberattacks leave something behind as well: a malicious file on a disk, an unfamiliar address in a log, a record that an investigator can find later and hold up as evidence.

An altered screen leaves nothing. The changed content lives in the memory of the device for as long as that one session lasts. Close the tab, quit the app, switch off the television, park the car, and it is gone. The attacker also decides whether the device keeps a copy at all. Browsers save copies of what they download, in a store called the cache, so that pages load faster next time, and the server that sends each resource also sends an instruction saying how long to keep it: a year, an hour, or, with the instruction "no-store," not at all. An attacker sends the altered version once, marked to be stored nowhere, and answers every later request with the genuine one, including the investigator's.

The traffic that carried it was encrypted, as it should be, so anything watching the network saw a trusted device talking to a trusted address, which it does thousands of times a day.

The publisher's own computers recorded an ordinary visit, because from their side it was one.

When a customer says "your site told me to send the money there," the company has nothing to check the claim against. It cannot reproduce what that person saw. Neither can the customer, a regulator, a court, or a forensic examiner.

A screenshot would help, and almost nobody takes one of a page that looks normal. One taken afterward shows the genuine page. And because a screenshot is trivial to fake, it proves little even when it exists.

When a person's account and a computer's records disagree, institutions side with the machine. A notorious modern example is the British Post Office scandal. Between 1999 and 2015 more than 900 people who ran local post offices were prosecuted for theft and false accounting because a bookkeeping system called Horizon showed money missing from their branches. The system was faulty. It took two decades, a public inquiry, and an act of Parliament to clear them, and the courts had begun from a legal presumption that a computer's records are correct. A customer who says "the screen told me to" starts from the same disadvantage, with the added problem that in this case the company's records really are accurate. The false version never touched them.

It is nearly impossible to defend against an attack you cannot detect. It is harder still when you cannot prove afterward that it ever happened. An attack that is delivered to a chosen few, passes every security check, exists only in memory, and vanishes when the screen goes dark is silent, invisible, unprovable, and unattributable. Nobody can say who did it, because nobody can show that anything was done.

Aimed at You

An attacker never has to change a screen for everybody. Someone who controls one advertising or visitor-tracking resource on a news site can, within the same hour and from the same web address readers have always used, send the real page to most readers, an altered paragraph to readers in one country, a different alteration to readers whose browsing marks them as likely voters or likely buyers, and the real page again to anyone who looks like a security researcher, a fact-checker, or a rival newsroom. Each reader sees one version and cannot see anyone else's. The publisher's own computers handle none of it.

Showing one face to the inspector and another to everyone else has a name, cloaking, and a long record. Volkswagen's diesel cars, exposed in 2015, carried software that recognized the conditions of an emissions test and ran clean for the test, then polluted at up to forty times the legal limit on the road. In 2017 Uber was found to have used a tool called

Greyball, which identified city officials trying to catch its drivers operating illegally and showed them a fake version of the app, populated with ghost cars that never arrived. In 2024 altered code served from a widely used web address, polyfill.io, did the same thing. It checked what kind of device was asking, checked again inside the browser, looked at which site the visitor had come from and at the time on the device's clock, and acted on only a sample of visits, all to stay out of sight of the people most likely to investigate.

None of this is exotic. Every advertisement slot on a modern page is sold in an automated auction, known as real-time bidding, in which advertisers bid for one particular viewer based on what is known about that person. An advertising platform already sells targeting by country, device, time of day, past purchases, and membership of a commercial or political segment, and those same controls decide who receives the altered page. Reaching a hundred thousand people who share one characteristic costs an attacker no more effort than reaching one person, because advertising companies built the infrastructure to do exactly that.

The political use of that machinery is a matter of record. In 2018 it emerged that the consulting firm Cambridge Analytica had obtained the Facebook profile data of up to 87 million people and used it to sort voters by personality and target them with tailored political messages. Whatever those messages achieved, the episode showed the public how precisely an audience can be sliced, and everything involved in it was ordinary commercial advertising technology.

Targeting does two things for an attacker, and both make the attack worse for everyone else. It makes the deception more effective, because a message shaped for one group, in their language, about their town, their employer, their money, or their fears, persuades better than one written for everybody. And it makes the deception harder to catch, because the people most likely to notice are exactly the ones an attacker leaves out: the publisher's own staff, a security researcher, a regulator, a journalist checking a claim. They load the same address and see the genuine page. The only people who could report the altered version are the ones who received it, and each of them has every reason to believe that what they saw is what everyone saw.

A head of government reads from one screen. A bank's chief executive reads from another. A commander directing a force reads from a third. An attacker who reaches only one of those screens has picked the smallest audience the mechanism allows and, in the same move, the largest consequence it can produce, because the authority of the person reading sets the ceiling. Targeting one named individual on one device is the **precision extreme**, the finest aim available, and a decision any of those three takes from a false screen needs no second recipient to matter.

How many people to reach and whose screen to reach are two separate choices, and neither limits the other. Narrowing the audience to one narrows nothing except the number of people who could have noticed.

Artificial Intelligence Changes the Scale

Everything described so far could be done by hand, and hand work has been the limit on it. Writing a convincing false paragraph for one reader takes someone who knows that reader: the language, the town, the employer, the worries. Doing it for a million readers used to mean a million hours of skilled labor, which is why precise deception belonged to intelligence services and was aimed at a few people at a time.

Generative artificial intelligence removes that limit. These are the systems, ChatGPT being the best known, that produce text, images, voices, video, and computer code on request. A model can write a paragraph in the style of a particular newspaper, rewrite it for a retired teacher in Ohio and again for a dockworker in Marseille, translate both, and supply a matching photograph, all in seconds and for a fraction of a cent. It can copy a person's voice from a few seconds of recorded speech. In 2024 a video call built with tools of this kind, in which every face and voice was fake, cost one company in Hong Kong 25 million dollars.

Attack the Glass and AI each supply what the other lacks. AI can manufacture a convincing falsehood for every person on earth, and it still needs a channel those people trust, which is why the Doppelganger campaign had to build imitation news sites and hope nobody looked at the address. Attack the Glass is that channel: delivery inside the genuine site, to a chosen audience, with no trace. Without AI, an operator has to write every alteration by hand. With it, the operator describes the goal and the model writes a separate version for each reader.

Three further developments tie the technologies together more tightly.

AI now runs on the device. New phones, laptops, and web browsers carry AI models that work on the device itself, without calling out to a distant data center. A model running inside the same program that draws the screen can compose the replacement text on the spot, using whatever that program is permitted to know about the person in front of it: their language, their location, the page they are reading, and often a good deal more. Nothing crosses the network, so nothing watching the network sees it happen. No two people receive the same words, and that removes the last handhold defenders have. The filters that catch spam and scam websites work because the same fake reaches thousands of inboxes and someone reports it. A deception written once, for one reader, on that reader's own device, belongs to a population of one.

AI products are built on the Glass. Chat assistants, the AI answers that now sit above search results, and the AI helpers inside email and office software all reach people as screens assembled on the device from outside resources, like any other modern application. The answer itself is fetched while the screen is running, which makes it exactly the kind of ever-changing resource that no fingerprint check can cover. People are quickly learning to trust these answers, and a new mental shortcut is forming in real time: the assistant said so. An answer swapped on the Glass carries all of the assistant's authority.

AI reads the Glass too. AI agents now browse websites, compare prices, summarize documents, and fill in forms on behalf of people and companies. A model does not judge whether what it reads is true. It works with what it is given. Text on a page can even carry instructions aimed at the AI and invisible to a human reader, a technique known as prompt injection. A person might notice that a figure looks wrong. An automated reader will not, it acts at machine speed, and the people downstream receive its output as analysis.

AI is also a proliferating technology. Capable models can be downloaded free of charge, run on an ordinary computer, and stripped of the safety limits their makers built in. No earlier technology with this much potential for misuse has spread so far, so quickly, to so many hands.

Weapons of Mass Deception

Put the pieces together at the scale of a population. Attack the Glass delivers inside the sources people already trust. Commercial advertising technology chooses who receives what. AI writes a separate version for every audience, or every person. At population scale this is the delivery layer for an influence operation: different segments receive different manipulated versions in the same hour, generative AI tailors each one, and no shared version is left for anyone to compare against or to fact-check.

We call the upper bound of this weapons of mass deception, and we name it to mark the stakes, not to claim it is happening today.

Picture the final days before an election, the first hours of a bank panic, or the middle of a public health emergency. Each group of citizens sees its own version of events on the news sites, government pages, and apps it already relies on. Older voters in one district read that their polling place has moved. Customers of one bank see a notice that withdrawals are suspended. Parents in one city find that the official guidance on a vaccine has changed. Every one of those pages sits at its genuine address, under its genuine name.

Four things set this apart from every influence operation on record. It arrives inside trusted sources, where earlier campaigns had to compete with them from outside using fake accounts and imitation sites. It leaves no single false story for anyone to debunk, because

no two groups saw the same one, and the journalists and fact-checkers who load those addresses find nothing wrong. It leaves no evidence, so nobody can show afterward what was done, to whom, or by whom. And it is cheap enough to be within reach of far more actors than a government.

The damage would go beyond whatever each false message achieved. A society argues, votes, and governs itself on the assumption that its members are looking at the same record, even when they disagree about what it means. An operation of this kind removes the common record without anyone seeing it go.

The name deliberately echoes weapons of mass destruction, for three reasons: the effect lands on whole populations at once, there is no practical defense at the moment of use, and the capability is spreading. Every ingredient exists now. The targeting machinery is a commercial product. The AI is free. The delivery was demonstrated on a hundred thousand websites by polyfill.io. What nobody can say is whether anyone has yet put them together, because an operation of this kind would leave nothing behind to find.

Deceive, Disrupt, Degrade, Deny, Destroy

An attacker in position can do five different things to a screen. Each one works on its own, each suits a different purpose, and nothing stops an attacker combining them.

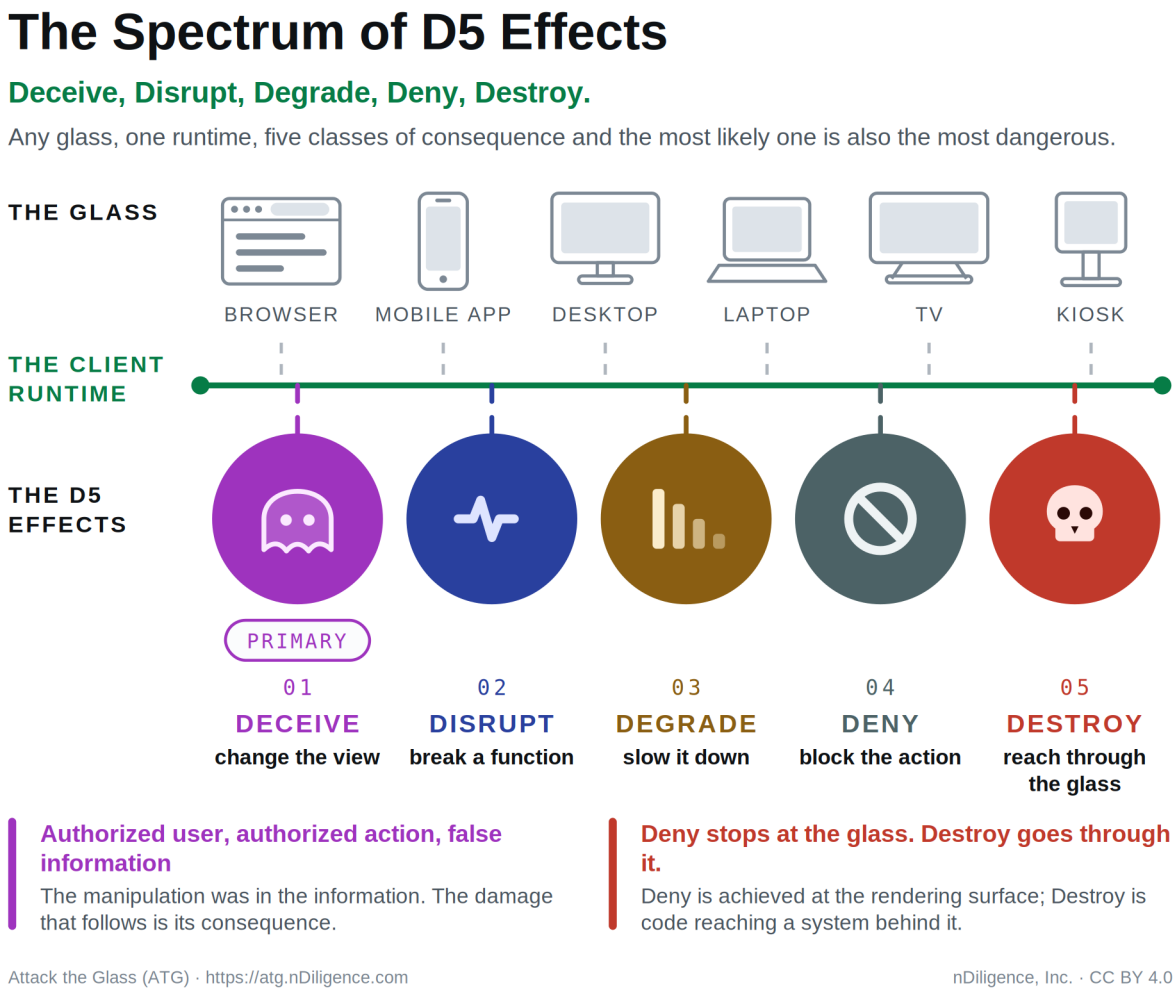


***Figure 3**: Five effects an injected resource can produce. The set is an index, not an escalation.*

Deceive. The screen shows something false, and everything else works perfectly. A trader sees holdings the firm does not have. A nurse sees a dose that nobody prescribed. A shopper sees a price, a voter sees a headline, a loan applicant sees terms, and each of them then does exactly what a sensible person would do with that information. The person is who they say they are, the action is one they are allowed to take, and every record behind the screen shows a normal transaction, because from that side it was one. Machines can be deceived the same way. An alarm that watches a temperature does not judge whether the number it receives is true. If the number says normal, it stays silent. The best-known demonstration is Stuxnet, an earlier attack on industrial control equipment that used the same logic: the monitoring screens replayed recorded normal readings while the

machinery was driven to failure. Deception matters most because it turns an organization's own trusted people, and its own safety systems, into the instrument.

Disrupt. A feature is still on the screen, still looks like it works, and quietly does nothing. A password reset completes and sends no email. A button for reporting fraud returns a confirmation and files nothing. An employee submits a safety concern, sees "thank you, your report has been received," and it goes nowhere. What makes disruption dangerous is that both sides believe the job was done. The person thinks they have acted, so they do not try again or pick up the phone. The organization receives no complaints, so it concludes nothing is wrong. On a web page that exists mainly to be read, there are only so many functions worth breaking. On a control panel, where nearly everything on the screen is a function, there are a great many: the acknowledge button, the alarm, the emergency stop.

Degrade. Everything still works, and works worse. Pages load a few seconds slower. A search takes long enough that people give up. A checkout drags until customers abandon their carts, and the company books the loss as a marketing problem, because nothing is broken and no alarm has any reason to fire. Online retailers have known for years that small delays cost real sales, which makes degradation a quiet weapon against a competitor: aimed at one company's checkout during its busiest week, it costs that company money and customers and never looks like an attack. Where timing is the whole point, the stakes are higher. An order that reaches the stock market two seconds late is filled at a different price. A control command that must arrive within a fixed window, and arrives after it, might as well not have been sent.

Deny. Something the person needs is not there. The button to dispute a charge is missing from the page. The link to appeal a decision does not appear. A sign-in is refused, a route does not load, a notice never shows. Anyone who has hunted through an account page for a cancel button knows how effective simple absence is, and that was only poor design. Done on purpose and aimed at the right moment, denial locks a customer out while a deadline passes, hides an evacuation notice from one district, or keeps an operator from the one control the situation calls for. Denial can reach a single person or every user of a service, and everything behind the screen stays healthy throughout. Support lines hear "I can't find it," and the company's own staff, looking at their unaltered screens, see it right where it should be.

Destroy. The attacker's code reaches through the screen and damages what is behind it. The first four effects all happen on the Glass. This one goes through it. Code running inside a screen can do whatever the signed-in person is allowed to do, and when that person is a system administrator, that includes deleting stored files, erasing databases, wiping the list of user accounts, or sending a bad update to every device an organization owns.

Organizations that have lived through ransomware know what it means to lose systems for weeks. Here the damage arrives through a page the administrator opens every morning, from a source the organization chose to trust. The administrator signs in, reads the dashboard, and sees nothing unusual. Behind that ordinary screen, the code is already deleting the backups.

The effects combine naturally. A benefits decision is changed from approved to denied, and the button to appeal it is removed from the same page. The applicant reads the decision, looks for a way to contest it, finds none, and gives up. In every case, what sits behind the screen sets the limit, far more than the attacker's skill.

The Operations

The Internet Is a Supply Chain

Every physical product reaches us through a supply chain, and people understand what that means. Food passes from farm to processor to truck to warehouse to shelf, and at each step somebody could tamper with it. In 1982, seven people in the Chicago area died after someone put cyanide into Tylenol capsules on store shelves. The manufacturer had done nothing wrong. The product was altered after it left the factory and before it reached the customer. Within months, medicine was being sold in tamper-evident packaging, and everyone learned to check the seal.

The internet is a supply chain too, and a far longer one than most people imagine. The screen in front of you was assembled from resources that passed through software developers, volunteer programmers, hosting companies, delivery networks, advertising exchanges, and the firms that register web addresses. Each of them is a place where a resource can be altered after the publisher approved it and before your device receives it. For most of what travels along that chain, there is no seal.

Attacks through the software supply chain are already among the most damaging on record. In 2017, attackers broke into the maker of one of only two tax programs approved for use in Ukraine, installed at most of the country's businesses, and hid destructive code in a routine update. The malware, known as NotPetya, spread around the world in hours, crippled the shipping company Maersk, the drug maker Merck, and many others. A former White House security adviser put the damage at more than ten billion dollars. In 2020, attackers hid a back door in an update to network management software made by a company called SolarWinds. As many as 18,000 customers installed it, among them several departments of the United States government, and it went undetected for most of a year before a security firm discovered it while investigating a break-in at its own offices.

Both of those attacks worked by getting customers to install poisoned software, and anything installed can eventually be found, taken apart, and traced. Attack the Glass uses the same route, a trusted supplier, and installs nothing. What it delivers exists for the length of one visit.

Getting Into Position

An operation of this kind begins long before anything appears on a screen. It begins with getting into position. Intelligence services have a phrase for it, placement and access: being placed somewhere inside the chain, with access to something that flows toward the

target. A person with placement and access has no lock to defeat, because they are already inside, doing what looks like their job.

The better the position, the more it allows. A place on one small site reaches that site's visitors. A place inside a resource that thousands of sites use reaches all of them at once. What an attacker can do follows from where they sit, far more than from how large or sophisticated they are.

Once in position, an attacker can wait. Nothing about the resource changes until the day it is used, so a search in the meantime comes up empty.

The most patient example yet discovered came to light in March 2024. A small, unglamorous piece of free software called xz, which compresses data, is built into most of the Linux systems that run the internet's servers. For years it had been maintained by one unpaid volunteer. Around 2021 a helpful newcomer using the name Jia Tan began contributing improvements, while other accounts, since judged to be fake, pressured the tired maintainer to share the load. Over more than two years the newcomer earned full control of the project, and then slipped in a carefully hidden back door that would have opened a vast number of servers to whoever held the key. It was caught by luck. An engineer at Microsoft noticed that logging in to a test machine was taking half a second longer than it should, went looking for the reason, and found it days before the altered version would have spread worldwide. Nobody has publicly established who Jia Tan was. The operation took years, cost little, broke into nothing, and came within days of succeeding.

Many Ways In, and Most Are Not Hacking

People hear "cyberattack" and imagine someone breaking through defenses. That is one way into a supply chain, and often the hardest. The others are the ways anyone has ever gotten into any supply chain.

An insider can do it. An employee at one of the suppliers, with ordinary access and few resources, can hold more power over what millions of screens show than most outside attackers could ever obtain. Insiders can also be recruited. In 2019 the United States charged two former Twitter employees with using their staff access to look up the private account details of critics of the Saudi government on behalf of Saudi officials, and one of them was later convicted.

A supplier can be bought, openly and legally, and the new owner inherits its place on every screen it serves. This happens regularly with browser extensions, the small add-ons people install to block ads or manage tabs. A popular one called The Great Suspender, with around two million users, was sold by its developer in 2020. The new owner added hidden code,

and in early 2021 Google removed the extension as malware. Nothing about it had changed from the users' side except who controlled it.

A piece of free software maintained by one tired volunteer can be handed over to a helpful stranger who offers to take on the work. In 2018 the unpaid author of a widely used piece of code called event-stream, downloaded about two million times a week, did exactly that. The new maintainer added code designed to steal from one particular brand of bitcoin wallet, and it went unnoticed for about two and a half months after the handover.

A web address can be allowed to lapse and be registered by someone else, who then receives every request still being sent to it. A distributor or reseller in the middle can alter what passes through. Passwords can be stolen, and systems can be hacked, which is how the Ticketmaster and British Airways thefts began.

Most of these involve no break-in, which means most of them are not things a cybersecurity team is placed to stop. They are matters of ownership, hiring, contracts, and money. No public registry records who owns the companies and addresses that deliver resources to our screens, and when control of one changes hands, nobody who depends on it is told. The address stays the same. The padlock stays the same. The party deciding what gets delivered is someone new.

Propaganda Is Old, the Delivery Is New

Deceiving a population is as old as politics. Four days before the British general election of 1924, a newspaper published a letter, supposedly from a Soviet leader named Zinoviev, urging British communists to prepare for revolution. It was a forgery, and it dominated the final days of the campaign. During the Second World War, Britain ran a radio station called Soldatensender Calais that posed as a German military broadcaster, mixing accurate news and popular music with invented stories meant to wear down German morale. In the 1980s the Soviet KGB spent years spreading the claim that the United States had created the AIDS virus as a weapon, a story that is still in circulation today.

The practice goes by many names. Propaganda is the oldest. Militaries speak of psychological operations and information operations. Governments and researchers speak of influence operations, and the European Union uses the term foreign information manipulation and interference, or FIMI. Disinformation is falsehood spread on purpose, and misinformation is the same falsehood passed along by people who believe it. NATO has begun to describe the contest over what populations believe as cognitive warfare. Every one of these names points at a single aim: changing what people hold to be true so that they decide differently. What the internet changed is the cost and the reach of delivery.

The record of the last ten years is public, and four episodes from it show the range.

Election interference. Before the 2016 United States election, a Russian organization called the Internet Research Agency ran thousands of fake social media accounts posing as American citizens and community groups. Its staff organized real rallies from St. Petersburg by pretending to be local activists. Facebook later estimated that the agency's posts reached 126 million Americans.

Conspiracy theory. In late 2016 a story spread from anonymous message boards to social media claiming that a Washington pizzeria was the center of a child-abuse ring run by politicians. It was entirely invented. On December 4 a man who believed it drove from North Carolina, walked into the restaurant with a rifle, and fired it, intending to free the children. There were none. Nobody was hurt, and the man went to prison. A lie assembled on screens had put an armed man in a room full of families.

Cloned news. Since 2022 a Russia-attributed campaign known as Doppelganger has built copies of real Western news sites, including major German, French, British, and American outlets, and used them to publish fabricated stories under trusted names. In 2024 the United States government seized 32 web addresses used in the campaign.

Deepfakes. In early 2024 an employee of the engineering firm Arup, in Hong Kong, sent about 25 million dollars to criminals after joining a video call in which every colleague on the screen, including the chief financial officer, was an AI-generated fake. The company's computer systems were never breached. The attack was on what one employee saw and heard.

Every one of those operations had to build its own stage. The fake accounts had to gather followers. The cloned news sites lived at slightly wrong addresses, which is how they were eventually caught. The fake executives had to get their victim onto a call. Each left something for a careful person or an investigator to notice.

Attack the Glass removes that burden. The false content is delivered inside the real site, at the real address, under the real name, to the people the attacker selects. A campaign like Doppelganger carried this way would need no cloned sites, because it would be speaking from inside the genuine ones. Nobody can say whether that has been done, because it would leave nothing to find.

Buying a Position of Trust

The clearest public case of Attack the Glass is polyfill.io. A polyfill is a small piece of helper code that lets older web browsers display newer websites properly. For years a free service at the address polyfill.io supplied that code to any site that asked, and more than a hundred thousand websites added the one line that fetched it. Most of them then forgot it was there.

In early 2024 the address and the service were sold to a company called Funnull. The service's original creator, who no longer controlled it, publicly warned every website to remove it at once, and two large internet companies set up safe replacement copies for anyone who wanted to switch. Few did. In June 2024 security researchers reported that the address had begun serving altered code, through the same channel those websites had always used, and that it sent some visitors on mobile phones to gambling and scam sites. Google began blocking advertisements that pointed to affected sites, and within days the company that registered the address suspended it.

Nobody broke in, and nothing a security alarm is built to catch would have fired. The US Treasury sanctioned Funnull in 2025, finding that the company had bought a repository of code used by web developers and altered it to send visitors of legitimate websites to scam websites. The address never changed. The owner did.

This operation was crude. It redirected people to other sites, which is about the most noticeable thing altered code can do, and it was still running on a hundred thousand sites before anyone raised the alarm. A quieter operator with the same position could have changed what those sites said.

Anyone in Position

It would be comforting to file all of this under espionage and leave it to governments. State services can certainly launch an attack of this kind. So can anyone else who holds placement and access. A single developer who controls a popular code library or a source of data can do it alone. So can a corporation moving against a rival, an industry protecting itself, an organized crime group, a terrorist or violent extremist organization, a political party, a lobbyist, or one super-empowered individual with money and a cause.

The record already holds examples at each scale. In January 2022 the author of two free code libraries used by thousands of companies, angry that large firms profited from his unpaid work, deliberately broke them, and applications around the world started printing gibberish. Two months later the maintainer of another popular library added code that erased files on computers located in Russia and Belarus, as a protest against the invasion of Ukraine. Each was one person acting alone with a keyboard. At the corporate scale, Volkswagen and Uber both built software to show inspectors a false version of events. Web skimming is the work of organized criminal groups. During the 2016 election, more than a hundred websites pushing invented political stories to American readers turned out to be run by teenagers in one small town in Macedonia, who had no interest in politics and were doing it for the advertising money.

A precision operation once needed a state intelligence service behind it, because a human analyst had to research every target by hand. Generative AI does that research now, so content shaped for one recipient, or for one segment, costs whatever the computing costs. Criminal groups, commercial rivals, insiders, and state services all reach the same capability at the same price.

Whoever it is works without detection, without evidence, and without attribution: a ghost in the machine.

Digital Supernodes

The internet's supply chain is not spread evenly. A small number of suppliers sit at hubs that millions of sites, apps, and organizations all depend on: a dominant content delivery network, a nearly universal piece of shared code, a sign-in service, an advertising exchange. A content delivery network, or CDN, is a company that keeps copies of websites' resources on computers all over the world, so that each visitor is served from somewhere nearby. A handful of CDNs carry a large share of everything on the web. They reached that position by being good, cheap or free, and popular, and each new customer made the next one feel safer in choosing them. We call these hubs Digital Supernodes.

The world has had two accidental demonstrations of how much rests on these hubs. On June 8, 2021, a software fault at a CDN called Fastly, set off by one customer changing a setting, took a long list of major sites offline at once for about an hour, among them Amazon, Reddit, the New York Times, and the British government's main website. On July 19, 2024, the security company CrowdStrike sent out a faulty update that crashed about 8.5 million Windows computers in a morning. Airlines grounded flights, hospitals cancelled operations, and banks and broadcasters went dark. Neither event was an attack. Each showed that a single supplier, trusted by everyone for good reasons, can reach a large part of the world's screens in minutes.

For an attacker a supernode is the most valuable position there is. Control of one gives a place on millions of screens at once, and the ability to choose among them. Polyfill.io was a small example, one shared piece of code on a hundred thousand sites. There are hubs far larger.

The Impact

Losing Trust in the Glass

A false screen costs someone a decision first. It costs more later. Once people learn that a trusted screen showed them something false, they start doubting screens that are telling the truth.

Researchers have measured this loss of trust. People who watch an automated aid make a single mistake stop relying on it, even when it is still more accurate than they are. In an experiment with about two thousand adults, a fabricated video was more likely to leave viewers unsure what to believe than to fool them, and the people left unsure trusted news less afterward. Once fakes are known to exist, anyone caught by a true record can call it fake. Legal scholars have named this the liar's dividend: the more the public hears about convincing forgeries, the easier it becomes for a dishonest person to wave away genuine evidence.

A bank whose customers question their balances, a hospital whose staff second-guess the chart, a newsroom whose readers wonder whether the article they read is the one that was published: none of them needs a second attack. One well-publicized false screen is enough. People build trust in a screen slowly and drop it quickly, and the institutions that reach us through the Glass have no other way to reach most of us.

What Fails, and for Whom

A hospital, a trading floor, a newsroom, and a power substation all assemble their screens the same way. They differ in what sits behind the screen and who acts on what it shows.

News and media. A publisher's name is its most valuable asset, and an attacker reaches it directly. Readers can be shown an altered version of a story under the publisher's name and the reporter's byline, while the editors and fact-checkers see the correct one. The alteration can be as small as one number in a report on a company's earnings, one name in a story about a crime, or one quotation put into a politician's mouth. The false Associated Press message of 2013 moved the stock market in minutes, and that was a single sentence on one social media account. The publisher cannot retrieve what any individual reader was shown, because no copy exists anywhere the publisher controls, and so it cannot issue a correction to the people who need one.

Banking and **payments**. Criminals have used this method for a decade to steal card numbers from checkout pages. In the Ticketmaster case the attacker never touched the company itself. The break-in was at another firm whose chat window sat on the payment page, and a United Kingdom regulator recorded it in a penalty notice. The same method

that copies a card number can change the account a payment is sent to, or the balance a customer sees. Fraud that works by changing the bank details on an invoice already costs businesses billions of dollars a year when it is done by email, where the victim at least has a suspicious message to look back at. Inside a bank, a trader or treasury officer who acts on a false summary of the firm's holdings is an authorized person taking an authorized action on information they had every reason to trust.

Healthcare. Medication charts, test results, appointment systems, and the displays on medical devices are all screens assembled in software, and patients now read their own on portals and phone apps. A nurse reading a value usually has nowhere else to check it. A decimal point moved one place in a dose, a blood type changed, an allergy warning that fails to appear: each is a small change on a screen and a large one for a patient. Often nobody reads the value at all. An automatic alert watches the number instead, and if the reading looks normal, it stays silent.

Power, water, and industry. Elsewhere an attacker changes what a person believes. Here an attacker can change what a machine does. An operator's control screen commands pumps, breakers, valves, and alarms, so interfering with what it does matters more than changing what it says. Timing matters too. A delay that would merely irritate someone shopping online can make a control command arrive too late to work. In December 2015, operators at three regional electricity distribution companies in Ukraine watched the pointers on their control screens move by themselves as remote attackers opened circuit breakers one after another, cutting power to about 225,000 customers in midwinter. That attack took over the operators' computers, which is a different method. It is a reminder that a control screen moves real equipment. Well-run plants restrict what reaches those screens, and many keep their control networks cut off from the internet altogether. That helps a great deal. It shortens the supply chain without ending it, because the software on an isolated network was still built from outside parts, by other companies, and is still updated from time to time.

Government and public services. Benefits, taxes, public health guidance, election information, and emergency alerts all reach citizens through the Glass. The Hawaii missile alert showed what one false official message does to a population in 38 minutes. Election results are an obvious target. During Ukraine's presidential vote in May 2014, attackers broke into the election commission's systems and prepared a fake results page showing a fringe candidate winning. Officials found and removed it less than an hour before it was due to appear, and a Russian state television channel reported the fake result that evening anyway. The officials who run public services also make their own decisions from screens assembled the same way, so the exposure runs in both directions.

Inside every organization. The people who administer an organization's technology work from screens too: the consoles that manage user accounts, databases, cloud storage, and the building of the organization's own software. Those consoles are assembled on the device from outside resources like any other modern screen. An altered shopping page can mislead a customer. An altered administrator's console can delete the accounts, the data, and the backups.

What Is Known

We separate what we can prove from what we reason, what we presume, and what nobody can measure.

The way screens are assembled is established, and anyone can confirm it. Incidents that used it are part of the public record, including the polyfill.io and Ticketmaster cases. The research on how people respond to what they see, and how far they trust an automated display, is established in its own fields.

None of the parts is new, and we do not claim to have discovered an unprecedented trick. Security teams already worry about the risk from outside scripts. Web skimming has stolen from checkout pages for a decade. Cloaking is a documented practice. Polyfill.io happened in public. What has been missing is one account of the weakness underneath all of them: modern screens are assembled at the last moment from resources trusted by their address and never checked for their content, the change can be made after every security check has passed, and nothing records it. We also follow that weakness beyond the web browser, where almost all of the existing attention has gone, to the apps, vehicles, appliances, medical devices, and control rooms that are built the same way.

This also explains why Attack the Glass has no entry in the public catalog of software vulnerabilities, known as CVE. That catalog lists specific flaws in specific products, each of which a vendor can fix. Attack the Glass is a property of how thousands of products are assembled. There is no single product to list and no single fix to ship.

We evaluated 50 kinds of device and platform that build screens this way, in 10 categories from web browsers and phones to vehicles, medical devices, and industrial control systems. All 50 are exposed, and we rated 46 of them Critical or High. Protection runs opposite to consequence. The web browser, which has the most safeguards, is where people read the news. The operator's station, the medical display, and the command screen, where the gravest decisions are made, have almost none of the browser's safeguards.

The Risks It Implies

Some conclusions are reasoned from how these systems are built, since no public case yet documents them in every field. The damage an attack can do grows with what sits behind the screen and with the authority of the person reading it. A false screen, once publicized, weakens trust in true ones, which follows from research on automated aids and fabricated video. That generative AI can produce tailored content for any number of individual readers is established, and anyone can test it. That it multiplies the reach of an Attack the Glass operation is reasoned from how the two fit together.

One conclusion we presume**.** Capable actors already hold the positions an operation would need. Our threat-intelligence work places them there, but those findings sit in restricted channels a reader cannot check, which is why we mark the conclusion as presumed and rest no other claim on it.

What Nobody Can See

Nobody can establish how much this is being used today, because the instruments that would measure it do not exist. We cannot tell you an operation is running against you, and no tool in standard use can tell you one is not. Absence of evidence is not evidence of absence, and here the absence is a property of the attack rather than a sign of safety. That is the finding, and it is why awareness is the defense available now.

What We Need to Do Now

Awareness comes first. A person who knows that a trusted screen can be altered treats a surprising one differently: they stop, and they check.

For anything consequential and hard to reverse, a payment, a change of bank details, a password reset, a command to a physical system, verify through a second channel before acting on a screen alone. The second channel has to be one the screen did not supply: a phone number you already had, or a person you can ask directly. A telephone number shown on the same screen is part of what could have been altered. Banks and companies that handle large payments have long used a version of this rule, calling back on a known number before changing where money is sent, and the rule protects against this attack just as it does against a forged email.

When a screen surprises you, with an unexpected demand, a missing option, a number that seems wrong, treat the surprise as information. Look at the same thing on another device or network if you can, or ask someone else what they see. If the two differ, keep a record of both. It may be the only evidence that ever exists.

Organizations can do a good deal more, and none of it waits on new technology. They can write down the list that does not exist, every outside company whose resources load on their screens, including the suppliers of their suppliers. They can remove the ones they do not need, keep their own copies of the ones they do instead of fetching them from someone else's address, and switch on the protections that are already available, Content Security Policy and Subresource Integrity among them, and then confirm that those are truly in force. They can ask the same questions of the vendors whose consoles their own staff sign in to every day. The Remediation Playbook sets it out in three horizons, beginning with what an engineering team can change in the first ninety days.

The Work Still to Be Done

Awareness and good housekeeping reduce the risk. They do not remove it, because the weakness is built into how screens are assembled on nearly every platform, and no single company can fix that alone. We do not have a solution for Attack the Glass, and we are not selling one.

Several things do not exist yet and need to. Devices need a way to confirm, while a screen is running, that what they assembled is what the publisher intended, which requires a trusted record of that intention to compare against. Engineers call the general idea runtime monitoring: watching how a screen behaves as it is drawn, where today's checks inspect resources only before they arrive. It would also take measurement across many users at once, because comparing what one person was served with what everyone else received is the only way a change aimed at a few can be noticed at all. Deception written by AI for a single reader defeats even that comparison, so detection will also have to work without a second copy to compare against, by recognizing that a screen is behaving in a way its publisher never intended. None of this ships as a standard part of any browser, phone, or device today. That is an engineering problem, and those get solved when someone pays for the work.

Someone needs to be able to establish afterward what a screen showed a given person at a given time. The ownership of the companies and addresses that deliver resources to billions of screens needs to be visible, and changes of control need to be disclosed, as they are in other industries the public depends on. And people need better preparation for a world in which a trusted screen can be wrong on purpose, which is work for educators, psychologists, and institutions as much as for engineers.

The standards that organizations are audited against, and the questions buyers ask their suppliers before signing a contract, need to reach this layer too. At present none of them asks what a product loads onto the customer's screen while it runs, or who controls it.

None of that is the work of one company or one country. The builders of browsers, phones, and other platforms, the industries that depend on them, researchers, and governments each hold a piece. It starts with enough people understanding the problem to insist that it be solved.

The engineering behind everything said here is in the Technical Primer and the Technical Whitepaper.