

What Happens to Trust When the File Leaves?

Why More Cybersecurity Does Not Necessarily Create More Certainty

A question about digital integrity, provenance and Connected Trust

Author: Chris Els

Founder & CEO: Resolut

Research paper | September 2026

Every morning I seem to wake up to another collection of cybersecurity infographics. Protect your identity. Secure your data. Know your suppliers. Manage your vulnerabilities. Most of the advice is perfectly reasonable, although remarkably similar.

Recently, I found myself thinking about something slightly different. We spend enormous amounts of time and money protecting data while it sits inside systems. We protect the people who have access to it. We secure networks, endpoints and applications. We authenticate users and monitor behaviour.

But modern business depends upon data continually leaving those protected environments. It becomes a document, attachment, report, certificate, spreadsheet, image, software package or another digital object.

Then somebody sends it somewhere.

What happens to trust when the journey begins and the file leaves?

Imagine an ordinary business document.

At its point of creation, we may know a great deal about it. We know who authored it. We know which version was approved. We know who is authorised to access it. It may sit inside an environment protected by sophisticated cybersecurity controls.

Then it starts travelling.

It might pass through email systems, security gateways, cloud platforms, collaboration tools and third-party environments. It may be downloaded, copied, forwarded, stored, processed and shared again.

The sender protects its environment.

The recipient protects theirs.

The intermediaries have their own security controls.

At the destination, another sophisticated collection of technologies may inspect what arrives.

That should make us considerably safer. But it raises a question that is subtly different from threat detection:

Can the recipient prove that what arrived is the same trusted digital object that originally left its source?

“Is this dangerous?” may not be the only question

Modern cybersecurity has become very good at looking for malicious activity.

Files can be scanned against known threats. Suspicious objects can be analysed. Behaviour can be monitored. Sandboxing can allow an object to execute in an isolated environment before it is trusted.

These capabilities continue to improve. But sophisticated adversaries also design attacks around the controls intended to stop them.

MITRE ATT&CK documents obfuscation techniques in which ***payloads can be split*** into separate, seemingly benign files that only reveal malicious functionality when they are reassembled.

Microsoft has documented another example: ***HTML smuggling***.

Rather than sending a conventional malicious executable through the network perimeter, an attacker can use legitimate HTML and JavaScript capabilities to construct the malicious file locally on the target device. Microsoft describes how some protection systems may initially see benign HTML and JavaScript traffic because the malicious file is only created after the HTML reaches the endpoint.

Microsoft's research also makes an important point that should prevent us from drawing the wrong conclusion. Modern defences are not limited to examining an individual file. Microsoft's own response combines email protection, machine learning, sandboxing, endpoint detection and correlation across different security domains.

The issue therefore isn't that cybersecurity has somehow forgotten to inspect files.

The more interesting question is whether **detection and integrity are answering different questions.**

Detection asks:

Is what I received dangerous?

Integrity asks:

Is what I received actually what was sent?

Imagine that **A** becomes **B**

Suppose an organisation creates an important digital object. Call it **A**.

Its source is known. Its digital state can be established. Its authorised creator is known.

The organisation sends A to another party. Somewhere during its journey, A changes.

The recipient receives **B**.

B has the expected filename. It appears to have come from the expected organisation. It looks exactly as the recipient expects. The recipient's security environment examines B and finds nothing that warrants blocking it.

Perhaps B is entirely harmless.

But sender and recipient have established something else: **A is not B**.

Neither party authorised the change. Should the recipient trust B?

I think that question is worth separating from whether B can currently be shown to contain something malicious.

Why could an apparently harmless change matter?

An intelligent adversary doesn't necessarily need every individual component of an attack to be independently malicious.

MITRE's documentation of split payloads makes that clear. Components can appear benign until they are brought together.

HTML smuggling shows another variation. Legitimate technologies can carry information across a boundary from which a malicious object is subsequently assembled inside the destination environment.

A more recent Microsoft investigation shows how attack chains can combine several different mechanisms. In one 2026 campaign, a downloaded archive led through a shortcut, PowerShell execution and DLL side-loading before a final payload executed in memory and established external command-and-control communication.

These examples are not evidence that an intercepted legitimate business document was secretly modified during transit. That distinction matters.

They demonstrate something broader.

Sophisticated attacks don't have to reveal their intent in every individual object or at every stage of their journey.

The malicious outcome may only become apparent when components, actions and conditions eventually come together.

This changes the significance of our unexplained A-to-B transformation. We don't need to claim that B is malicious. Perhaps the more useful response is:

We don't know why B is different, therefore trust has been interrupted.

The consequence may be much greater than a corrupted document

This is where the question stops being about document management.

A persistent adversary may have objectives that extend far beyond changing a file.

The objective could ultimately be access to another system, theft of information, disruption of operations or preparation for some later action.

The initial anomaly may appear insignificant when examined alone. That is precisely why provenance interests me.

If an object was trusted at its source and its digital identity subsequently changes without authorisation, why should we need to understand the attacker's eventual intention before questioning that object?

Perhaps an unexplained integrity failure is itself useful information.

Instead of: **“We found something malicious, so we stopped it.”**

the principle becomes: **“Something changed that should not have changed, so trust has stopped until we understand why.”**

That is a different security proposition. We already have much of the technology. None of this requires us to pretend that digital integrity is a new idea. Cryptographic hashes can establish whether two digital objects are byte-for-byte identical.

Digital signatures go further by linking integrity with authentication. NIST describes digital signatures as providing assurance both that the claimed signatory signed the information and that the information was not modified after the signature was generated. Its Digital Signature Standard specifically identifies detecting unauthorised modification as one of their purposes.

- Encryption protects confidentiality and, depending on the protocol used, can also provide integrity protections.
- Secure file-transfer technologies exist.
- Provenance technologies exist.
- Chain-of-custody systems exist.
- Threat detection exists.

The question I am interested in is therefore not:

Can we invent technology capable of determining whether a file changed?

Clearly we can.

It is this:

Do we adequately connect these capabilities across the complete journey of important digital objects between independent organisations, platforms and intermediaries?

From file integrity to provenance:

Suppose A leaves its authenticated source with a verifiable digital identity.

At the next participating boundary, that identity is verified.

Then again.

And again.

If the object remains unchanged, its integrity can continue to be established.

If an authorised transformation takes place, that event can become part of its history.

If an unexplained transformation occurs, the chain of trust breaks.

Now we know something we didn't know before.

We don't merely know that the final object is different.

We may be able to establish **where the difference first appeared**.

That creates provenance.

The concept is familiar in the physical world. Evidence has a chain of custody.

Regulated products can be traced through supply chains. Components carry identifiers and histories.

We do this because inspecting something at its destination doesn't necessarily tell us everything we need to know about it.

- Origin matters.
- Custody matters.
- Unauthorised alteration matters.
- Why should an important digital object be fundamentally different?

A file can tell us something about the network

There is another consequence that interests me even more. Imagine an object moving through:

A → B → C → D → E

At each participating point its integrity is independently verified. Now suppose repeated unexplained integrity failures begin appearing between C and D. One anomalous file tells us something about that file.

A pattern might tell us something about **C, D or the relationship between them**. That starts to turn digital-object provenance into a source of network intelligence. The object becomes evidence about the systems and relationships through which it travels.

This is where the idea connects with something I have been researching for some time:
Connected Trust.

Modern organisations don't operate as isolated entities.

People connect to organisations. Organisations depend upon suppliers. Suppliers depend upon other suppliers. Systems connect to systems. Software depends upon other software.

Digital objects continually travel across those relationships.

We can assess the organisations independently.

We can secure individual environments.

We can authenticate the participants.

But trust ultimately has to survive the relationship between them.

Trusting the sender isn't the same as trusting the object

There is an important distinction here.

If I know that an organisation is trustworthy, that doesn't automatically prove that every object apparently originating from it should be trusted.

If I authenticate the sender, I have established something about the sender.

If I encrypt the transmission, I have established protections around that transmission.

If I scan the received file, I have established something about the security assessment of the object I received.

If I establish provenance and integrity, I can ask another question:

Is this demonstrably the object that the trusted sender intended me to receive?

These controls are complementary. One needn't replace another. Perhaps that is the mistake in thinking about this solely as another cybersecurity product. The opportunity may lie in connecting assertions that currently exist separately.

What would Connected Trust for a digital object require?

At this stage, I think the questions are more useful than a proposed architecture.

- Could an important digital object have a persistent verifiable identity from the moment it is created?
- Could that identity survive movement between independent organisations without requiring every participant to use the same document-management system?
- Could authorised transformations become part of the object's provenance rather than simply destroying its original identity?
- Could unexpected transformations automatically interrupt trust?
- Could a recipient establish where an object's integrity was last verified?

- Could patterns of integrity failure provide evidence about third-party or intermediary risk?
- And could this work at sufficient scale to become infrastructure rather than another isolated security control?

Those are much harder questions than whether we can calculate a cryptographic hash.

There are limits to the idea

Provenance is not malware detection. A malicious object created by an attacker can maintain perfect cryptographic integrity throughout its journey. An authorised source could itself be compromised and a participating checkpoint could be compromised.

Not every system through which an object travels will necessarily participate in a common trust framework.

Legitimate processing can also alter digital objects, sometimes substantially.

Continuous object integrity therefore cannot replace endpoint protection, behavioural detection, identity security, sandboxing, network monitoring or the other layers organisations already depend upon.

Nor should it try. The proposition is narrower.

Could persistent integrity and provenance provide an additional trust signal that existing detection models do not always provide?

And if so, what should happen when that signal fails? **Perhaps trust should fail before detection succeeds**

This may be the part of the idea I find most interesting. Security controls often need some basis for deciding that an object, behaviour or event is suspicious.

Integrity verification can potentially tell us something simpler.

- The object changed.
- We don't yet know why.
- That may be enough.

An organisation doesn't necessarily have to accuse an intermediary of compromise or identify malicious code. It doesn't need to know the attacker's objective.

It can simply say:

This is no longer the object whose integrity we previously established. Until we understand why, we will not trust it.

That reverses an assumption worth questioning.

Instead of continuing to trust an object until malicious intent is demonstrated, trust can be interrupted when integrity can no longer be demonstrated.

What if trust could travel with the file?

I began this investigation by wondering what existing cybersecurity technologies might not do. I now think that is the less interesting question.

Microsoft, Palo Alto, CrowdStrike and others are continually improving their ability to detect sophisticated attacks, correlate activity and understand behaviour across systems.

The question isn't whether those technologies are good enough. It is whether we are asking them to provide an assurance that belongs somewhere else. Threat detection tells us something about the object we have.

Provenance can tell us whether it is the object we were supposed to have.

Connected Trust asks whether that assurance can persist as the object moves through relationships we don't completely control.

I don't yet know whether this is already adequately solved, whether the pieces exist but remain disconnected, or whether there is a genuine architectural gap.

That is what I want to investigate. Because the consequence of getting the answer wrong may be considerably greater than a corrupted document. Perhaps the question we should ask before **“Is this dangerous?”** is much simpler:

“Is this still what was sent?”

If it isn't:

Who changed it? Where did trust break? And why would we allow it to continue its journey until we know?

What if trust could travel with the file?