The Global Language of Business

GS1 Digital Signatures Technical Implementation Guideline

Using digital signatures to support brand protection and product authenticity

Release 1.1.0, Ratified, Jan 2026

Document Summary

Document Item	Current Value
Document Name	GS1 Digital Signatures Technical Implementation Guideline
Document Date	Jan 2026
Document Version	1.1
Document Issue	0
Document Status	Ratified
Document Description	Using digital signatures to support brand protection and product authenticity

Contributors

Name	Organisation
Pete Alvarez	GS1 Global Office
Daniel Anders	WIPOTEC-OCS GmbH
Anette Andersson	ICA Sverige AB
Albert Arbones	GS1 Spain
Koji Asano	GS1 Japan
Pascal Aulagnet	Pfizer
Andrea Ausili	GS1 Italy
Adam Bartlett	Excellis Health Solutions
Robert Beideman	GS1 Global Office
Salima Bekraoui	GS1 Italy
Chuck Biss	GS1 Global Office
Marc Blanchet	Viagenie
Elizabeth Board	GS1 Global Office

Name	Organisation
Miklos Bolyky	GS1 Global Office
Josh Bonczkowski	FoodLogiQ
Scott Brown	1WorldSync, Inc.
Jose Manuel Cantera Fonseca	IOTA Foundation
Jiraporn Chalermjirarat	GS1 Thailand
Luiz Costa	GS1 Brasil
Jay Crowley	US Data Management, LLC (USDM)
Oscar Cruz	GS1 Mexico
Chase Cunningham	Wal-Mart Stores, Inc.
Kevin Dean	Dolphin Data Development Ltd.
Elsa Dietrich	Cosmetics Europe
Peta Ding (editor)	**GS1 Global Office**
Volker Ditscher	WIPOTEC-OCS GmbH
Roland Dominici	COBUILDER FRANCE
Jeanne Duckett	Avery Dennison RFID
Valérie Dumez	GS1 France
Fabris Ekeu	GS1 Cameroon
Paul Elizondo	MEPS Real-Time, Inc.
René Elspaß	WIPOTEC-OCS GmbH
Xinxin Fan	IoTeX
Eleanor Gayle	GS1 Global Office
Steyn Geldenhuys	Sieveo
Tarik Gemei	Amgen Inc.
Nicole Golestani	GS1 Canada
Juan Pablo Gomez Sepulveda	GS1 Mexico

Name	Organisation
Nadi (Scott) Gray	GS1 Global Office
Mark Harrison (lead editor)	**Milecastle Media Limited**
Philip Heggelund	DuckScape Inc
Bernie Hogan	Independent Consultant - Bernie Hogan
Nuran Idris	GS1 Global Office
Michael Isabell	eAgile Inc.
Yohan Jeon	GS1 Korea
Steven Keddie	GS1 Global Office
Kimmo Keravuori	GS1 Finland
Mads Kibsgaard	GS1 Denmark
Anna Klapper	GS1 Germany
Brent Koeppel	Cardinal Health
Cihan Korucu	GS1 Türkiye
Alexey Krotkov	GS1 Russia
Chris Lai	GS1 Hong Kong, China
Stephen Lam	GS1 Hong Kong, China
Piergiorgio Licciardello	GS1 Global Office
Joseph Lipari	Systech International
Geraldine Lissalde-Bonnet	GS1 Global Office
Wayne Luk	GS1 Hong Kong, China
Ilka Machemer	GS1 Germany
Ned Mears	GS1 US
Jan Merckx	GS1 Netherlands
Edward C Merrill	GS1 Global Office
Andreas Micke	GS1 Germany

Name	Organisation
Nuno Miranda	GS1 Portugal
Andrew Morehead	GS1 US
Gena Morgan	GS1 US
Elif Muftuoglu	GS1 Türkiye
Alice Mukaru	GS1 Sweden
Shekhar Nambi	JOHNSON & JOHNSON PTE LTD
Giada Necci	GS1 Italy
Alexander Pakhomov	LLC "Center for the Development of Advanced Technologies"
Luis Paniagua	GS1 Costa Rica
John Pearce	Axicon
Christophe Pereira	la poste
Peter Phaneuf	eAgile Inc.
Justin Picard	ScanTrust
Neil Piper	GS1 Global Office
Francesca Poggiali	GS1 Global Office
Maryam Poorsafari Balasi	GS1 Canada
Albertus Pretorius (co-chair, lead editor)	**Tonnjes ISI Patent Holding GmbH**
Aruna Ravikumar	GS1 Australia
Tayyab Rehman	GS1 Canada
Craig Alan Repec (editor)	**GS1 Global Office**
Eduardo Retana	GS1 Costa Rica
Octavio Rodriguez	Systech International
John Ryu (facilitator)	**GS1 Global Office**
Zbigniew Sagan	Advanced Track and Trace
Alexander Sanchez	GS1 Mexico

Name	Organisation
Laura Sanchez Cabrera	EM Microelectronic
Yuki Sato	GS1 Japan
Sebastian Schmittner	European EPC Competence Center GmbH (EECC)
Eugen Sehorz	GS1 Austria
Steven Simske	Colorado State University
Dana Smith	Baxter Healthcare
Tania Snioch	GS1 Global Office
Olga Soboleva	GS1 Russia
Jim Springer	EM Microelectronic
Xiaoyun Sun	GS1 China
Claude Tetelin	GS1 Global Office
Ann Tindale	GS1 Australia
Elena Tomanovich	GS1 Global Office
Laurent Tonnelier	mobiLead
Ralph Troeger	GS1 Germany
Alec Tubridy	GS1 Ireland
Vivian Underwood	GS1 US
Stephanie Van Rossum	GS1 Global Office
Linda Vezzani	GS1 Italy
Gwen Volpe	Fresenius Kabi AG
Amber Walls	GS1 US
Lei Wang	GS1 China
Chunguang Wang	GS1 China
Wenyu Wang	GS1 China
Wilfried Weigelt	REA Elektronik GmbH

Name	Organisation
Roman Winter	GS1 Germany
Kevin Wu	Creative Sensor Inc.
Huang Xin	GS1 China
Ruoyun Yan	GS1 China
Chao Zhang	Zhongguancun Industry & Information Research Institute of Two-dimensional Code Technology

Log of Changes

Release	Date of Change	Changed By	Summary of Change
1.0	June 2024	P. Archer F. Chiovenda, P. Ding, K. Dean, M. Harrison, F. Poggiali, A. Pretorius & C. Repec	Initial publication developed under WR 23-172, with additional changes per WR 24-213 "RAIN RFID" to "EPC/RFID" and references section updated to align with GS1 Style Guide.
1.1	May 2025	P. Ding & M. Harrison	WR 24-234 New note for section 1.1 clarifying independent verification requires software or Web page not linked from the 2D barcode and Figure 3-1 revised to indicate starting point

Disclaimer

Table of Contents

1 Introduction

1.1 Executive summary

For centuries, humans have devised methods to detect counterfeits and assign accountability to ensure, among other things, fair trade. One of these methods is the concept of a signature. Digital signatures facilitate trust and accountability for trade within the digital world, in the same way that handwritten signatures are used for global trade in the physical world. Since the 1990s, digital signatures have become a fundamental enabler of electronic commerce and the integration of trade, financial transactions, secure communications and trust on the Internet, as experienced every day by our increasingly digitised society across the globe.

Data carriers such as barcodes and RFID tags have become the primary method to apply identification and optional attribute information to an entity such as a trade item, logistics unit, asset or document, for example. This encoded data is used to obtain additional information about the entity, with the additional attributes used, often immediately, for localised actions or processes.

When data is digitally signed, it supports detection of fraudulent tampering of the data, the checking of its integrity and authenticity, as well as providing accountability for the data (i.e., non-repudiation by the party that digitally signs the data). For this reason, digital signatures can support regulatory requirements such as the European Union (EU) Regulation 2023/1542 concerning batteries and waste batteries [EU2023/1542], which notes that “the technical design should ensure that the battery passport carries data in a secure way which respects privacy rules” and that “data authentication, reliability and integrity shall be ensured” and that “the battery passport shall be such that a high level of security and privacy is ensured and fraud is avoided”. Another example includes the proposed EU legislation on the safety of toys [EU2023/0298COD] and on detergents and surfactants [EU2023/0124COD] which both note that there is “the need to allow for the verification of the authenticity of the product passport”.

Digitally signing data is a mechanism to support verification of the authenticity of data. The technical approaches detailed in the GS1 Digital Signatures Technical Implementation Guide are based on existing established open standards for digital signatures that enable anyone to independently verify the authenticity and integrity of the digitally signed data.

In summary, the main business requirements for using digital signatures include:

- Ensuring that digitally signed data corresponds to a specific individual physical entity that is identified or described in that data. This can include additional authentication factors, such as embedding unique codes within the digitally signed data that are securely marked on the individual physical entity using technologies such as watermarks, holograms, microprinting or ink visible under ultraviolet light.
- The ability to verify that the identifier (and where required by industry, specific data elements) encoded within a data carrier displayed or carried on an entity, is authentic i.e., untampered and from a known and trusted source.
- To provide a verifiable link between the entity who is accountable for the data and the data encoded in the data carrier.
- Open, free, and interoperable verification i.e., digital signatures from different signers must be independently verifiable.

✓ To minimise the impact on systems to ensure efficient interoperable deployment without disruption, whether or not they can process digital signatures. Notably, as illustrated in many other ecosystems, unrestricted verification using more than one independent "channel" (i.e. a second opinion) is an underlying requirement for general and open trust.

✓ **Note:** It is easy for counterfeiters to copy or create fraudulent linear and 2D barcodes that could result in the user being redirected to false information or dangerous online resources such as malware, computer viruses or inappropriate videos or images that cause distress or harm. Therefore, users should not rely solely on any verification software or verification Web page that is linked directly from the 2D barcode or recommended in a Web page linked from the 2D barcode. Open standards for security allow users to independently choose and use any credible verification software that correctly supports such open standard approaches, whether based on ISO/IEC 20248 [DigSig], JSON Web Signatures [JWS], XML Digital Signatures [XML-DS], or Verifiable Credentials [VC]. This is an important aspect of what is meant by 'independent verification' and helps to protect a user from being deceived with verification software provided by the counterfeiter, which gives a false assurance that a counterfeit product is genuine or authentic without doing robust verification checks that correctly conform to the open standard approaches. The gs1:jws link type mentioned in this technical implementation guideline only directs to digitally signed data in JSON Web Signature format [JWS], and the GS1 Application Identifier (8030) defined by the GS1 General Specifications [GenSpecs] provides only a DigSig data construct as defined by ISO/IEC 20248 [DigSig]. Neither of these approaches redirects to a preferred verification app – both approaches support independent verification, with further detail provided within this Technical Implementation Guideline

1.2 Purpose

This technical implementation guideline intends to provide the GS1 community, particularly the technical community (e.g. solution provider community), with an authoritative GS1 resource to explain how the GS1 Digital Signatures application standard utilises existing open standard technologies and how it can be used as a tool to detect counterfeit or non-authentic products and raise barriers for counterfeiters.

The GS1 Digital Signatures application standard, which was ratified in 2023 for publication in the GS1 General Specifications [GenSpecs], is an open standard solution that supports independent verification of digitally signed data by leveraging ISO/IEC 20248 [DigSig], JSON Web Signatures [JWS] or XML Digital Signatures [XML-DS]. All these are ITU X.509 [X.509] application standards.

For everyone (including consumers):

Section 2 provides a general introduction to digital signatures and how they work, with example workflows for how a consumer might use a smartphone app to verify a digital signature. Further background information about cryptography can be found in Appendix F.

For brand owners and solution providers:

The remainder of this document is primarily intended for a brand owner's or solution provider's technical team. It provides an overview about how open standards approaches to digital signatures can be used for brand protection purposes and product anti-counterfeiting, in a way that enables any customer to independently verify the digital signature to check product authenticity.

For public policy teams:

This entire document may be used in discussions with regulators and shared with their technical teams, for awareness of open standards approaches to checking authenticity of an entity such as a product or document, as a better alternative to proprietary approaches that do not support independent verification by members of the public.

Note: ISO/IEC 20248 is maintained by the ISO/IEC Joint Technical Committee 1 and Standards Committee 31 for AIDC technologies, known as ISO/IEC JTC1/SC 31 [SC31]. This committee, in which GS1 are active participants, specifies barcode and RFID standards for a number of industries such as the manufacturing and logistics industries. JSON Web Signatures [JWS] is specified by RFC 7515 and uses JSON (JavaScript Object Notation), a text syntax that facilitates structured data interchange between all programming languages, and is defined by RFC 8259 [JSON]. Both standards are maintained by the Internet Engineering Task Force from the IETF (Internet Engineering Task Force). XML Digital Signatures [XML-DS] are specified and maintained by the World Wide Web Consortium (W3C). X.509 [X.509] is developed and maintained by the International Telecommunication Union - Telecommunication Standardization Sector of ITU (ITU-T) and is equivalent to ISO/IEC 9594-8.

Note: For simplicity, many of the examples and details described throughout this document reference Global Trade Item Numbers (GTIN) and products only. However, this guidance is applicable to other types of entities, as the GS1 Digital Signatures application standard is approved for use with several GS1 identification keys that identify entities such as logistics units, documents, assets etc.

Note: Currently the smallest digital signatures are 256-bits long. It is unlikely to become shorter. As such, its use is limited to high-capacity data carriers. Appendix E discusses the transition to high-capacity data carriers for the use of serialisation and digital signatures.

1.2.1 Scope

The scope of this document, the GS1 Digital Signatures Technical Implementation Guideline, is to:

- Explain what digital signatures are, and why they are used e.g., detect tampering, non-repudiation.
- Document the two approaches, carrier centric and web-centric, supported by GS1 standards.
- Provide worked examples to demonstrate the two approaches supported by GS1 standards using ISO/IEC 20248 [DigSig] or JSON Web Signature [JWS].
- Note other security considerations regarding management of ITU-T X.509 public key infrastructure (PKI) keys [X.509] e.g., how long should a public key certificate remain valid relative to the lifespan of an entity, how to manage rollover or replacement of keys, as well as revocation when needed.

Note: An ISO/IEC 20248 [DigSig] data construct is commonly known as a “DigSig”, a named thing with a specific meaning, while “digital signature” in lower case refers to the general and common digital signature.

1.2.2 Out of Scope

In line with the GS1 Digital Signatures application standard, the following are out of scope for this Technical Implementation Guideline:

- Closed or proprietary solutions.
- Solutions which do not support independent verification.
- Approaches not based on ISO/IEC 20248 [DigSig], JSON Web Signatures [JWS] or ITU-T X.509 [X.509].
- Commercial complexities related to third parties assigned by brand owners to carry out contracted operations such as packaging, labelling and generation and applying of digital signatures by such parties.

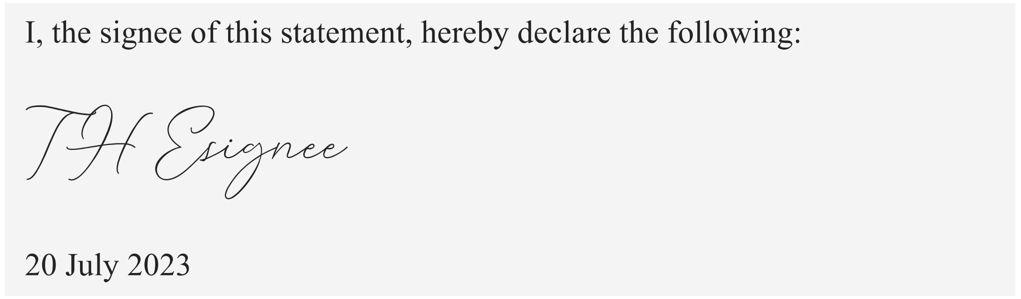
Important: Future work on Verifiable Credentials [VC] may become an alternative or complement to ITU-T X.509 [X.509]. GS1 is following these developments and assessing their potential impacts as they evolve.

1.3 Handwritten signatures

Handwritten signatures are commonly used to sign charters, contracts and financial documents to provide legitimacy to such documents. However, handwritten signatures can be forged or copied to another document to impersonate the original author of the handwritten signature and to fraudulently indicate that the author agreed to the contents of the document to which the copied signature is applied. Forensic methods are successful in detecting forged signatures and handwriting. However this takes time, which is not feasible for any scale of operations. It is important to note that forensic methods deal with the wiggles a hand makes when writing. The wiggles are compared to a known and trusted specimen of the signee/writer of the words. In this context, the following two examples have a different trust outcome:

Example A

In example A shown in Figure 1-1, only the signature is written. Therefore, only the paper provides a link between the signature and the statement.


I, the signee of this statement, hereby declare the following:

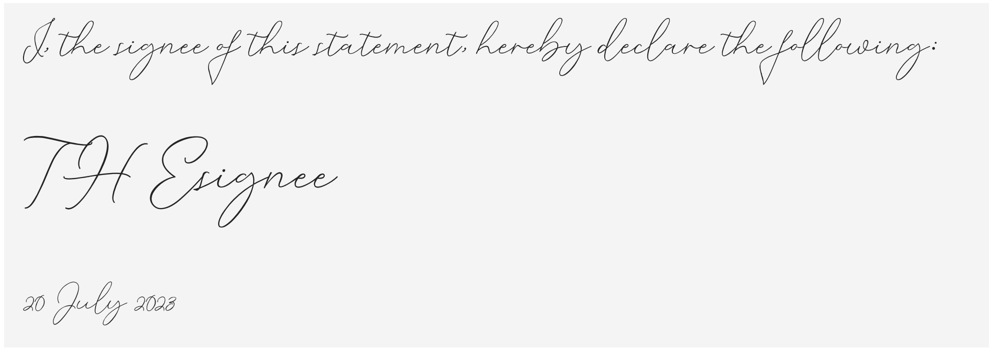
TH Esignee

20 July 2023

Figure 1-1 Example A: handwritten signature only

Example B

In example B shown in Figure 1-2, both the statement and the signature are written. Therefore, tampering of the statement can be detected, even if the signature is not verifiably linked to an entity, by comparing the signature handwriting to the statement handwriting.



I, the signee of this statement, hereby declare the following:

TH Esignee

20 July 2023

Figure 1-2 Example B: handwritten statement and signature

The following concepts illustrated by written signatures are applicable to digital signatures:

- Digital signatures use the method demonstrated by example B, to ensure tamper detection of the message, and that the message is part and parcel of the signature.
- Digital signatures do not provide secrecy.
- To counter the copying of signatures, some inherent natural characteristic should be included in the signature e.g., watermarks and "invisible" inks in paper certificates.
- The signee is a legal person who can be held accountable.
- A trusted third party needs to vouch for the identity of the party responsible for making the digital signature. This is practiced in the real world as well as the digital world, as specified in ITU-T X.509 [X.509].
- Signatures are timestamped, even when the date is not written due to the ability to age the application of the ink. Notably, a timestamp is in most cases an inherent requirement for the signature to be legally binding.

1.4 Digital signatures

Digital signatures are much more secure than handwritten signatures because they are:

- Cryptographically generated using an asymmetric public/private key pair.

- Highly sensitive to any changes in the data content, so that a digital signature for one document or dataset will not be valid for another document or dataset unless the data content is identical.

Important: Digital signatures do not provide or intend to provide secrecy. They are typically used, like in HTTPS and SSL, to authenticate the keys which are used to provide secrecy.

As digital signatures are extremely sensitive to changes in data content, they provide a mechanism to detect any tampering of the data; even changing one minor bit of information will result in a completely different digital signature and the verification of the signature is rejected.

The electronic computational nature of digital signatures ensures that the speed of their generation and verification is fast and scalable, to match modern data systems and operations. However, any given generated signature has a limited lifespan because of the ever-increasing computational power of computers. This cryptographic issue is well managed by established best practices.

Digital signatures based on certified public/private key pairs inherently support non-repudiation. This means that when someone digitally signs a document or dataset, even if they deny that they did so, that denial would be considered invalid in many legal jurisdictions. This is because there is an extremely low mathematical probability that the signature could have been generated without knowledge of, or access to, the private key used for signing. The corresponding public key is also certified as belonging to the signing party, following appropriate due diligence checks that were performed at the time the public key certificate was issued by a credible public key certificate authority.

Digital signatures are **generated** in a two-step process:

1. The data content is processed using a hashing algorithm such as SHA-256 [SHS], to produce a compact hash value.
2. Then the hash value is encrypted using a private key, to construct the bare digital signature value. Details of the hashing algorithm, encryption algorithm and a reference to the corresponding public key certificate should be provided in addition to the bare digital signature value, in order to support independent verification of the digital signature.

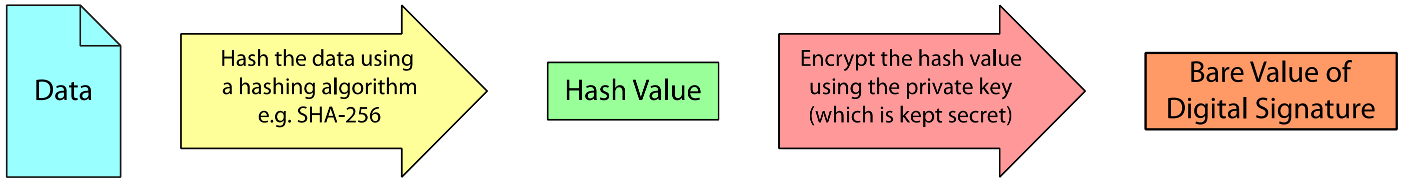

Data
Hash the data using a hashing algorithm e.g. SHA-256
Hash Value
Encrypt the hash value using the private key (which is kept secret)
Bare Value of Digital Signature

Figure 1-3 Two-step process to *generate* a digital signature

Digital signatures are **verified** in a two-step process:

1. The data content is processed using a hashing algorithm such as SHA-256 [SHS], to produce a compact hash value.
2. Then the digital signature value is decrypted* using the referenced public key, to produce a result.

Note: *the process described is a simplification of the actual cryptographic process

If that result is equal to the hash value from the first step and if the public key certificate is valid and has a valid chain of trust back to a credible root certificate authority, the digital signature is considered to be valid. There can be a high level of confidence that the data content has not been altered since it was signed and that the data content or document was digitally signed by the person or organisation specified in the public key certificate for the public key that successfully verified the signature.

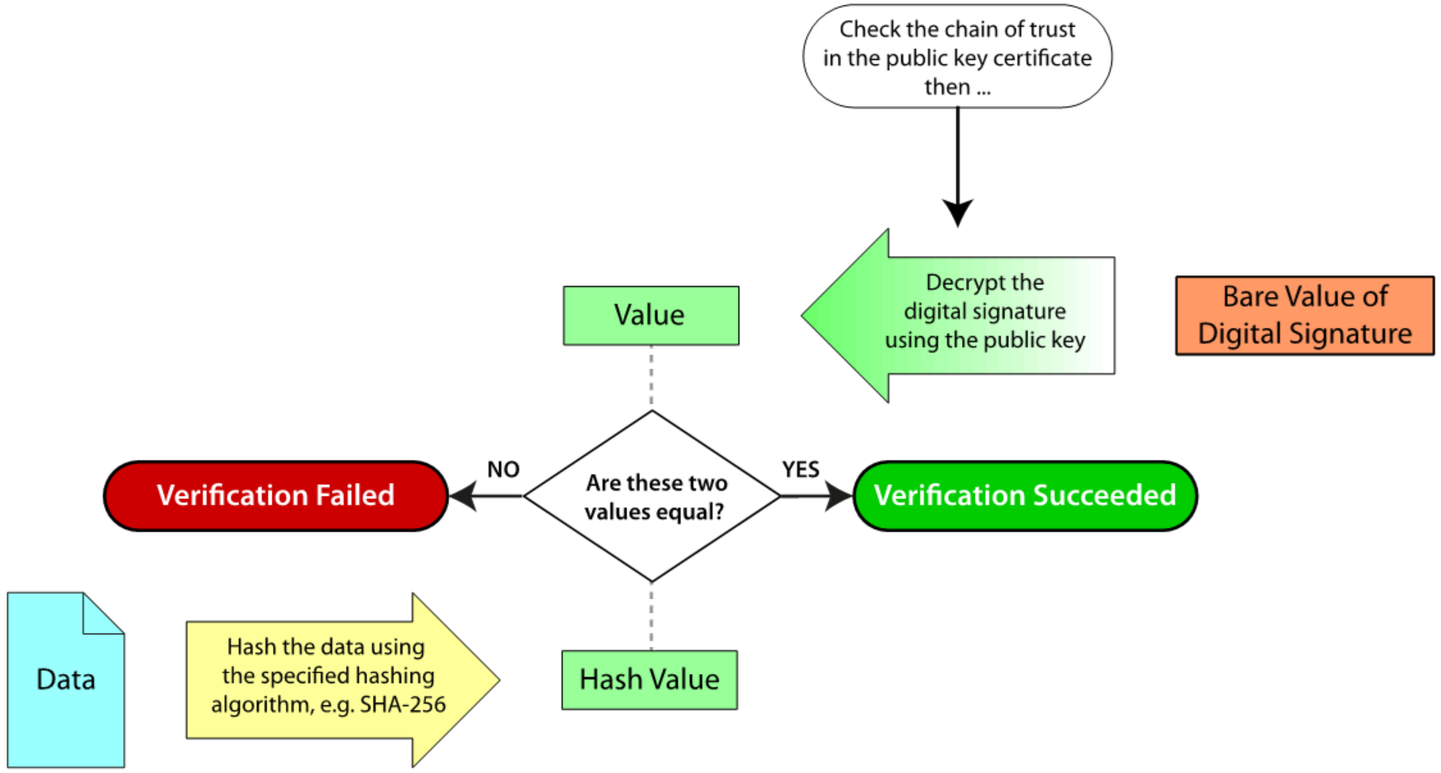

Check the chain of trust in the public key certificate then ...
Value
Decrypt the digital signature using the public key
Bare Value of Digital Signature
Verification Failed
NO
Are these two values equal?
YES
Verification Succeeded
Data
Hash the data using the specified hashing algorithm, e.g. SHA-256
Hash Value

Figure 1-4 Two-step process to *verify* a digital signature

Digital signatures are highly sensitive to change, which is desired. For example, the addition of a comma in a sentence of a contract may alter the meaning of the contract, or the removal or move of the dot/dash/comma in a value.

Consider two image files where only one pixel has been changed - in this case, a pixel in the centre of the eye of the sitting puffin is changed to white in the modified image shown in Figure 1-6.

Figure 1-5 Original puffins image

Original image: https://ref.gs1.org/guidelines/digital-signatures/1.0.0/Original_puffins_image_hash_demo.jpg

Figure 1-6 Modified puffins image

Modified image: https://ref.gs1.org/guidelines/digital-signatures/1.0.0/Modified_puffins_image_hash_demo.jpg

SHA-256 hash value of original image shown in Figure 1-5 is:

- 5E2D12927BF44B0CB2A03C5081A786824562D48A505928FE422669DE9A7ACE7B

SHA-256 hash value of modified image shown in Figure 1-6 is:

- `3F795869DDD4AE5B358FF817248820E84CFD6C6DEC17956E525EC6B2ABEE9182`

SHA-256 hash values can be calculated using free online tools or from a POSIX conformant operating system command line, using the command `shasum -a 256`

e.g., `shasum -a 256 original_puffins_image_hash_demo.jpg`

Or working with the online files remotely:

```
curl -s -H 'Cache-Control: no-cache' https://ref.gs1.org/guidelines/digital-
signatures/1.0.0/assets/Fig_original_puffins_image_hash_demo.jpg | shasum -a 256


curl -s -H 'Cache-Control: no-cache' https://ref.gs1.org/guidelines/digital-
signatures/1.0.0/assets/Fig_modified_puffins_image_hash_demo.jpg | shasum -a 256
```

Even by only changing the colour of one pixel out of 449,600 pixels, the resulting hash values are completely different and show no obvious similarity, other than in length. As a result, any digital signatures constructed using those hash values will also be completely different. This simple demonstration shows how hash values and digital signatures are extremely sensitive to even the tiniest changes in the data content.

The next stage in constructing a digital signature is to encrypt the hash value using a private key. A public/private key pair is generated using a cryptographic algorithm. The public key can be calculated from the private key, but the private key cannot be easily calculated from a known public key. For this reason, the use of public/private keys is known as asymmetric cryptography.

A public key can be openly communicated to anyone, but the private key must be kept secret by its owner and must not be shared or communicated to anyone else.

A public key certificate can be issued by a certificate authority to provide a high level of assurance about who owns a public key. This typically involves some interactive checks of other identity documents as well as one or more random challenges to check that whoever is applying for the public key certificate actually controls the corresponding private key. This is done without the applicant revealing the private key to the certificate authority; instead, the certificate authority sends one or more challenges for the applicant to encrypt some specified data (which could be random or at least not known in advance to the applicant) using their private key. If the certificate authority can decrypt the data using the corresponding public key, then the certificate authority can be sure that the applicant does control the corresponding private key.

However, to have trust in the digitally signed data it is also necessary to check the integrity of the public key certificate chain and ensure that there is an unbroken chain of trust (via digitally signed assertions from a higher-level certificate authority to a lower-level certificate authority) that connects one or more global well-trusted root certificate authorities to the certificate authority that issued the public key certificate.

2 Motivation for using digital signatures

Digital signatures can be a valuable tool in improving consumer confidence in the authenticity of a physical product instance, as a digital signature cannot be trivially copied from one dataset or document to another if the digitally signed data contains a reference to a specific serialised product identifier. Especially if the digitally signed data also contains one or more references to other complex security codes that appear within security features on the physical product, such as watermarks, holograms, microprinting or fluorescent ink that is only visible under ultraviolet light. In turn, this raises the barrier to counterfeiters, to make it economically unattractive to target a specific product instance that is accompanied by digitally signed data from the corresponding brand owner.

In practice, digital signatures are an important tool amongst many tools that can be used in the fight against counterfeiters and illicit products. Other helpful techniques include mass serialisation where each product instance has a globally unique identifier and the use of mass-serialised traceability data to detect implausible distribution routes or the presence of 'cloned' individual product instances, in which two or more objects with the same individual product instance identifier appear in multiple locations at the same time or within an overlapping time interval; a real-world macroscopic object cannot simultaneously be in more than one place at one time, so the detection of duplicate or conflicting observations at the same time may indicate suspicious activity. Instance level identification of each entity with a globally unique identifier, is the most practical method of ensuring an entity is not counterfeited, as it is the only effective and pragmatic way to detect if counterfeits are present.

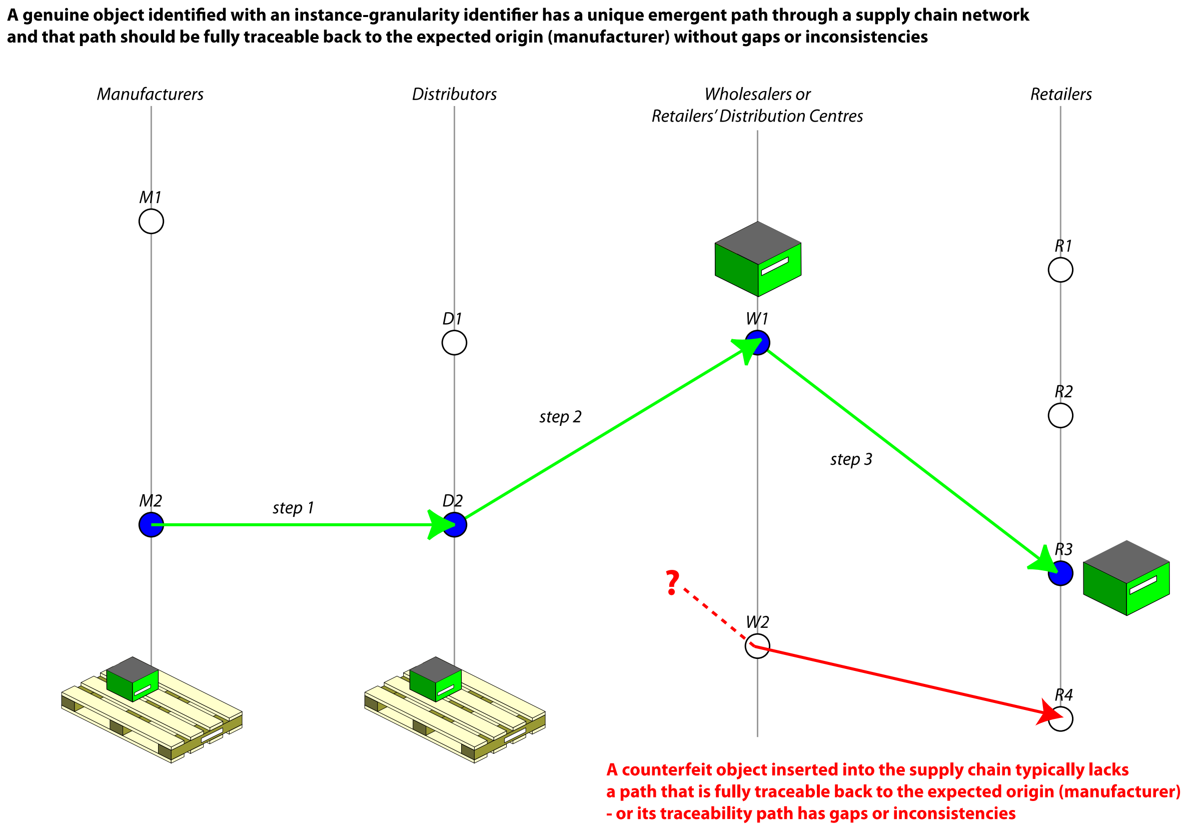

A genuine object identified with an instance-granularity identifier has a unique emergent path through a supply chain network and that path should be fully traceable back to the expected origin (manufacturer) without gaps or inconsistencies
Manufacturers
Distributors
Wholesalers or Retailers' Distribution Centres
Retailers
M1
M2
D1
D2
W1
W2
R1
R2
R3
R4
step 1
step 2
step 3
?
A counterfeit object inserted into the supply chain typically lacks a path that is fully traceable back to the expected origin (manufacturer) - or its traceability path has gaps or inconsistencies

Figure 2-1 Example of supply chain network path for a genuine and counterfeit product instance

A combination of these tools is shown to be more effective than any on their own, e.g. digital signatures with serialisation of items and transactions is a well-practiced method in many verticals. See Appendix E for more information on serialisation.

The GS1 Digital Signatures mission-specific work group (MSWG) was chartered to demonstrate to brand owners, manufacturers, logistics operators, regulators and legislators, that such technology can be helpful in the fight against counterfeit goods. This included the validation that digital signature approaches based on existing open standards are de facto in use and are widely considered to be sufficient. This validation aims to avoid closed, proprietary or emerging solutions. An open standards, non-proprietary approach enables the brand owner or their appointed agent e.g., contract manufacturer or contract packer, to digitally sign data for each individual product instance, while also enabling any individual (including end-consumers who purchase an individual product) to independently verify the integrity of the digitally signed data as well as the correspondence of that digitally signed data to a specific physical product instance, all without any reliance on proprietary approaches whatsoever.

The work group investigated and specified two approaches that are supported by GS1 standards:

1. **Carrier centric**

In this method, the Digital Signature (DigSig) is encoded in a data carrier as an attribute of the instance level GS1 identification key e.g., serialised GTIN, at the time of entity identification e.g., time of production for a trade item instance, in a way that does not disrupt non-signature aware systems. This ISO/IEC 20248 [DigSig] digital signature data construct is called a "DigSig" and has been assigned the GS1 Application Identifier (AI) (8030) to support encoding in high capacity 2D barcodes and EPC/RFID tags, as defined by the GS1 General Specifications [GenSpecs].

   a. Verification can be performed offline or in environments where communication and data access is restricted. This requires the availability of the digital signature certificate at the point of verification. ISO/IEC 20248 [DigSig] specifies the method for obtaining such certificates, though other trusted open methods of certificate distribution may also be used.

   b. Special care must be taken to ensure the validity period of the digital signature and its verification certificates are longer than the expected lifespan of the product.

   c. Certificate revocation is handled by the ITU-T X.509 [X.509] open standard methods, which are widely practised and supported on the Internet. ISO/IEC 20248 [DigSig] also specifies a method to revoke individual DigSigs.

   d. The product may be linked to the digital signature by adding data fields to the DigSig representing some intrinsic features or security marks of the product. See a description of how this works in Section 3. ISO/IEC 20248 [DigSig] specifies the data definition methods for the DigSig data fields to ensure interoperability between verification applications. This data definition includes a method to instruct a verifier on the use of such a field, which can also be provided in multiple languages.

2. **Web centric**

In this method, the Digital Signature (DigSig) is obtained from the Web by resolving the instance level GS1 identification key e.g., serialised GTIN. A JSON Web Signature [JWS] is used in the Web-centric examples shown in this document. The Web centric method has the potential for the data to be more up to date.

   a. Verification is online and may be a hurdle for some use cases. However, it facilitates two important benefits:

      i. The status of the product instance and digital signature provider can and should be current. The digitally signed product information may nevertheless be outdated and the verifier must take this into account.

      ii. The lifespan of the product instance's digital signature starts at the time of the request. As such, it may be short lived e.g., less than 18 months, typically a few days, to satisfy the transaction requiring the verification. This validity period falls neatly within current best practices for validity periods.

   b. The product instance may be linked to the digital signature by using an "incomplete" token, that requires the "missing part", often a secret code value, to be added by the verifier. The "missing part" constitutes some intrinsic features or security marking on the physical product instance. See the description of how it works in Section 4.

The following table provides a summary of the benefits and constraints of the carrier-centric approach and Web-centric approach, as well as features and capabilities that are supported by both approaches. It should be noted that the two approaches are not mutually exclusive and can be used in combination. The Web-centric approach is less constrained and more suitable for large amounts of static data attributes. Whereas data capacity is limited with the carrier-centric approach, therefore consideration should be given to prioritising the minimum data set required to support local actions, such as expiration date for example, to enable inventory or business processes to avoid the sale of an out-of-date product.

Table 2-1 Summary of benefits, constraints and shared capabilities of the two approaches supported by GS1 standards

Digital Signature approaches supported by GS1 standards	Benefits	Constraints	Common to both approaches
Carrier-centric approach	▪ Verification of the signature can be done offline or in environments with restricted (time and access) data access. ▪ Data required for immediate decision/actions (e.g., expiry date) can be stored in the data carrier and be authenticated at the time of reading to facilitate the immediate action.	▪ Memory capacity on data carriers is limited. ▪ Adding a digital signature will typically increase the size of a 2D data carrier or may increase the cost of an EPC/RFID tag, to ensure sufficient capacity. ▪ Data encoded in a data carrier might become out-of-date, if more recent updates to the data about a product instance or its status have occurred since the data carrier was encoded. ▪ The validity period for the signature and its associated verification certificates is required to be longer than the expected lifespan of the product.	▪ Both approaches use open standards that support independent verification of the digital signature and digitally signed data by anyone; ISO/IEC 20248 provides such an open standard for the carrier-centric approach, while JSON Web Signatures, XML Digital Signatures and Verifiable Credentials provide open standards for the Web-centric approach. These approaches have no need for intermediate trust agents but nevertheless ensure trust and interoperability across independent autonomous actors within supply chains and value networks. ▪ This is in some ways analogous to the use of different Web browsers to retrieve online information such as online banking, without any lock-in and without sharing of secrets with intermediary agents. ▪ Both ISO/IEC 20248 and JSON Web Signatures can support multi-lingual instructions to support the user through each step of the verification process. ▪ Both ISO/IEC 20248 and any of the open standard Web approaches can support strong binding of the digitally signed data to a specific unique individual product instance, through the inclusion of data specific to that individual product instance, which may include unique codes present in security marking such as holograms, watermarks, microprinting or ink visible under ultraviolet light. ▪ Both approaches are considered a two-factor or multi-factor authentication process, as they support a more challenging workflow in which the user verifying the digitally signed data is required to inspect the individual product instance, find such unique code values and input these as a required part of the verification process. If the signature is only checked for data about the physical entity's identifier, it is still not known whether that identifier really corresponds to that specific instance of the physical entity. However, if the digitally signed data includes other details that can only be found by inspection of a specific physical entity, then it's possible to provide a higher level of confidence that the digitally signed data is intended to describe that specific physical entity. ▪ Each EPC/RFID tag has a globally unique read-only Tag ID that is factory-programmed by the manufacturer of the tag's chip. With either the carrier-centric approach or Web-centric approach, such a Tag ID can be included within the digitally signed data, as an additional authentication factor; if Tag ID is present in the digitally signed data, any mismatch between the Tag ID read from an EPC/RFID tag and the Tag ID recorded in the digitally signed data indicates that the tagged object is not genuine.

Digital Signature approaches supported by GS1 standards	Benefits	Constraints	Common to both approaches
Web-centric approach	▪ Effectively no limit on how much data can be stored online, digitally signed and retrieved. ▪ Digitally signed data that is retrieved online could potentially be more up-to-date than digitally signed data encoded in the data carrier and provides an opportunity to check the current status of each product instance and the provider of the digital signature. ▪ Potential to digitally sign the data on-demand, at the time of the request – and to use a shorter validity period for the signature.	▪ No access to digitally signed data in environments with no Internet connectivity ▪ Potential to act on online data believing that it is up-to-date, even though it could still be stale.	

3 Accessing digital signatures from data carriers

3.1 General overview

This section explains the method where the Digital Signature (DigSig) is encoded in a 2D barcode or EPC/RFID tag, with consideration for minimising the size of the data encoded in the data carrier, to ensure small barcode footprints and efficient RFID reads.

Section 3.2 through to 3.6 shows how this method works, with details explaining the preparation, signing and verification of a Digital Signature (DigSig) encoded in a data carrier, and technical examples provided in Appendix A.

Figure 3-1 Illustrative overview of a carrier-centric digital signature process using ISO/IEC 20248

3.2 Method

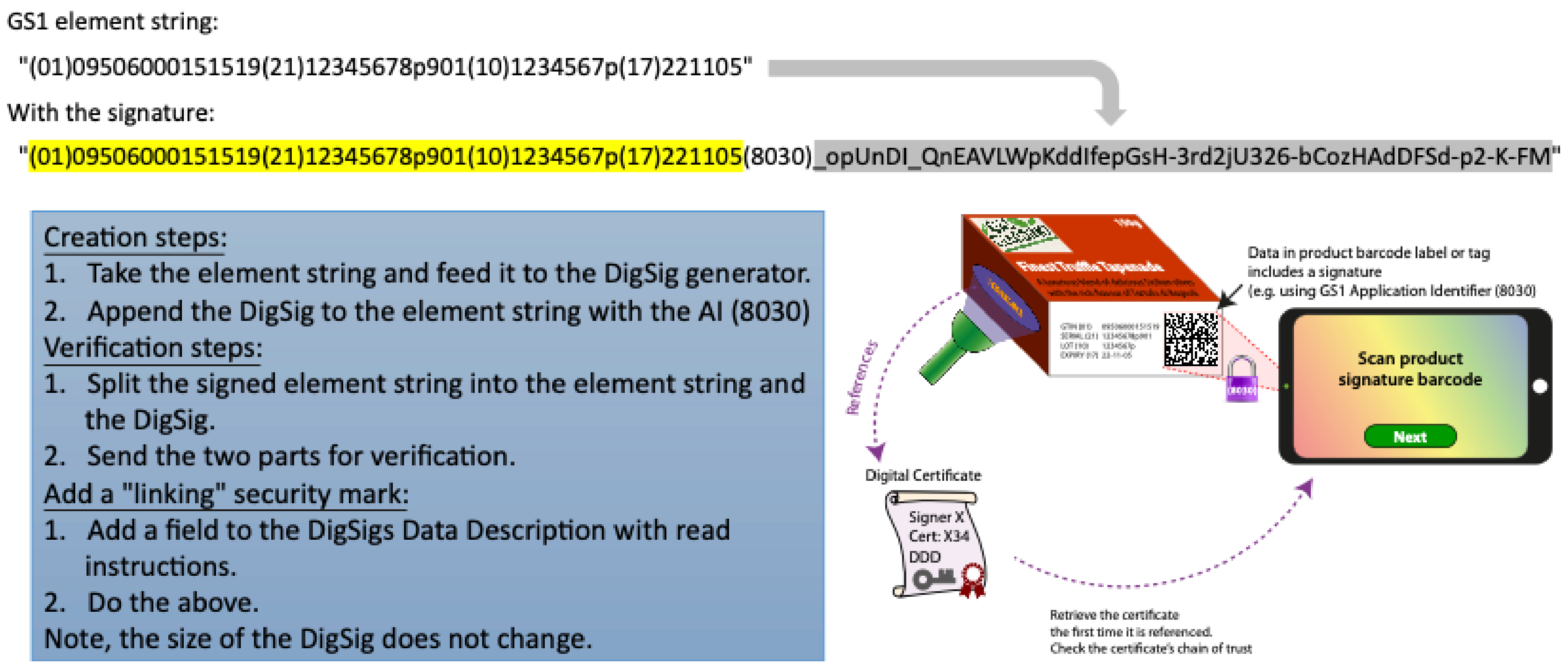
With this method, the digital signature as defined by ISO/IEC 20248 [DigSig] is encoded as a data attribute of an instance level identifier e.g., serialised GTIN, in a 2D barcode or EPC/RFID tag. This method emphasises the ability to verify offline, specifically where connectivity and data access is constrained, or where immediate actions are required.

A DigSig can be added to an instance level GS1 identification key as a supporting data attribute with the AI (8030). Figure 3-2 shows the process for a GS1 element string, with an optional UV ink packaging mark also shown. See Appendix A for more detail on the processes described in this section. The use of DigSigs are demonstrated with the fictitious product Dal Giardino Finest Truffle Tapenade and demonstration Global Trade Item Number (GTIN) from the GS1 Prefix ‘950’. See Appendix C for information about the Dal Giardino brand and Appendix D for information on example GTINs based on the GS1 Prefix ‘950’.

Additional features of ISO/IEC 20248 [DigSig] include:

- The ability to carry additional data fields within the DigSig or DigSig certificate. The recommended use of this feature is to add attributes, for which AIs do not exist, to link the encoded data with a physical entity such as trade item (or its tamper evident packaging), e.g., an UV ink mark, or weight with tolerances. ISO/IEC 20248 [DigSig] was designed with this in mind and provides methods to describe to an operator or a machine, the actions to be taken to obtain such linking-data. This linking data should not be included in the tag data, although it must be signed with the other data, and as such, forms part of the signature.
- Multi-lingual data descriptions to assist with the previous point.
- The ability to direct the reading of other barcodes and EPC/RFID tags, to include the information of those in the DigSig. For example, a DigSig encoded barcode can be added as a supplementary or secondary symbol to an entity e.g., trade item, which has an existing barcode e.g., EAN/UPC barcode, to allow systems which are not DigSig aware to operate without impact.

Note: ISO/IEC 20248 [DigSig] defines an open standard mechanism for encoding an Automated Identification and Data Capture (AIDC) dedicated digital signature construct known as a ‘DigSig’, in a data carrier such as a EPC/RFID tag or a high-capacity 2D barcode. The GS1 Application Identifier (AI) (8030) is assigned for the purpose of expressing such a digital signature construct as defined in ISO/IEC 20248 [DigSig] and plays an equivalent role to alternative data identifiers provided by non-GS1 issuing agencies for example, the MH10 data identifier ‘6R’.



GS1 element string:

"(01)09506000151519(21)12345678p901(10)1234567p(17)221105"

With the signature:

"(01)09506000151519(21)12345678p901(10)1234567p(17)221105(8030)_opUnDI_QnEAVLWpKddIfepGsH-3rd2jU326-bCozHAdDFSd-p2-K-FM"

Creation steps:
1. Take the element string and feed it to the DigSig generator.
2. Append the DigSig to the element string with the AI (8030)

Verification steps:
1. Split the signed element string into the element string and the DigSig.
2. Send the two parts for verification.

Add a "linking" security mark:
1. Add a field to the DigSigs Data Description with read instructions.
2. Do the above.

Note, the size of the DigSig does not change.

Data in product barcode label or tag includes a signature (e.g. using GS1 Application Identifier (8030)
Scan product signature barcode
Next
References
Digital Certificate
Signer X
Cert: X34
DDD
Retrieve the certificate the first time it is referenced.
Check the certificate's chain of trust

Figure 3-2 Example of DigSig AI (8030) encoded as GS1 element string in a 2D barcode

ISO/IEC 20248 [DigSig] specifies that the DigSig Data Description (DDD) is certified with the brand owner's public key, resulting in the DigSig certificate. The DigSig certificate is required for the verification of a DigSig. It contains the brand owner credentials, the public key for verification and the DDD for decoding. A digital certificate is a verifiable document. As such, a brand owner may include, using the DDD format, information which is applicable to all the DigSigs, as a group, generated with the certified DDD. For example, usage information and certification marks. The DDD of this example can be used for all UV ink marked products of the brand owner, see Appendix A for further details.

Note: ISO/IEC 20248 [DigSig] uses open standards such as RFC 8259 for JSON [JSON], the predominant method for data messages on the Internet.

The returned verification data in the machine-to-machine format, for the DigSig shown in Figure 3-2, is:

```
{"20248":
  {"ResponseCode": {"Code":0, "Desc": "Success"},
   "DDDdataTagged":
    {"specificationversion": "ISO/IEC 20248:2022",
     "daid": "GS1 9506000151540",
     "cid": 10001,
     "dauri": "https://dalgiardino.com",
     "GS1_element_string": "(01)09506000151519(21)12345678p901(10)1234567p(17)221105",
     "UVink": "UV389JKG4M2",
     "signature": ":706B:3BA5:D2D6:A7B3:AA70:CCC8:A71A:1057:2807:5189:3D06:8EDB:C5F2:9DA1:5D89:D69C",
     "timestamp": "2023-10-20"},
     "Languages": ["en","fr", "de"]}}
```

The response message provides the following information:

- The verification was successful, i.e. neither the data nor the certificate was tampered; both are authentic.
- The issuer of the DigSig is identified with the GS1 Party GLN 9506000151540.
- The GS1 element string is: `(01)09506000151519(21)12345678p901(10)1234567p(17)221105`.
- The packaging has a UV ink mark: UV389JKG4M2.
- The languages in which operator instructions are available.

The UV ink code using the DDD can be made to display or to request the code. The description in the DDD to instruct a user to read and enter the UV ink code may look like this:

```
{"fieldid": "UVink",
 "fieldname":
  {"en": "UV ink mark",
   "fr": "Marque d'encre UV",
   "de": "Markierung mit UV-Tinte"},
 "description":
  {"en": "Shine a UV light on the security mark. Enter the text which become visible.",
   "fr": "Faites briller une lumière UV sur la marque de sécurité. Saisissez le textequi devient visible.",
   "de":"Beleuchten Sie das Sicherheitszeichen mit UV-Licht. Geben Sie den Text ein, der sichtbar wird."},
  "type": "string",
  "pragma": {"entertext": null}}
```

When the UV ink value is not supplied, the verifier will respond with the following message:

```
{…
 "MissingDataSnips": ["UVink"],
 "MissingDataSnipPragma":
  [ {"fieldid": "UVink",
        "pragma": {"entertext": null},
        "description": "Shine a UV light on the security mark. Enter the text which become visible."}],
  …}
```

The description will change to French when French is selected from the response. Appendix A provides details about the DDD.

GS1 Digital Link:

"https://dalgiardino.com/01/09506000151519/10/1234567p/21/12345678p901?17=221105"

With the signature:

"https://dalgiardino.com/01/09506000151519/10/1234567p/21/12345678p901?17=221105 &8030=_opUnDI_QnEAVLWpKddIfepGsH-3rd2jU326-bCozHAdDFSd-p2-K-FM"

The URI is included in the signature, which assist in detecting spoofing attacks.
This is achieved by first verifying on the local device before resolving the GS1 Digital Link.

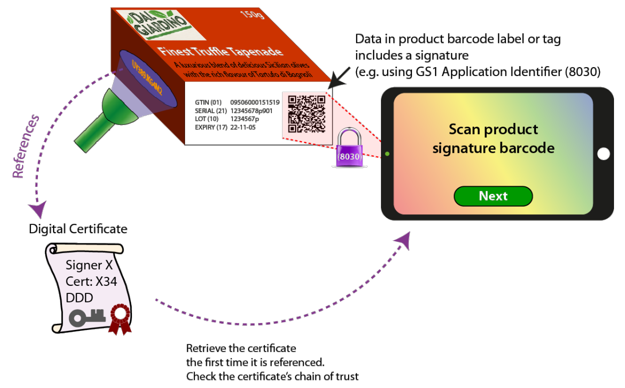

DAL GIARDINO
150g
Finest Truffle Tapenade
A luxurious blend of delicious Sicilian olives with the rich flavour of Tortufo di Bagnoli
UY389JKG4M2
GTIN (01) 09506000151519
SERIAL (21) 12345678p901
LOT (10) 1234567p
EXPIRY (17) 22-11-05
(8030)
Data in product barcode label or tag includes a signature (e.g. using GS1 Application Identifier (8030)
Scan product signature barcode
Next
References
Digital Certificate
Signer X
Cert: X34
DDD
Retrieve the certificate the first time it is referenced. Check the certificate's chain of trust

Figure 3-3 Example of DigSig AI (8030) encoded as GS1 Digital Link URI in a 2D barcode

EPC/RFID tags encode the same data in an interoperable way, as defined by the GS1 EPC Tag Data Standard [TDS]. EPC/RFID tags, unlike barcodes, store data in memory banks, each to be read independently. Bank 01 is always read first. Afterwards, the system controlling the read point decides if it needs to read the unique chip identifier (TID) and user memory. It is important to understand that the inclusion of the TID in the signature binds all the tag data to that specific chip, which means copies can be detected electronically. The steps are:

1. Convert the GS1 element string to an EPC, which is binary.

2. Read the TID (whilst optional, it is recommended).
3. Generate the DigSig, in this example with EPC, TID and UV ink code, which is also binary.
4. Encode the EPC to memory bank 01 and the DigSig, prefixed with the DSFID 17, to memory bank 11.

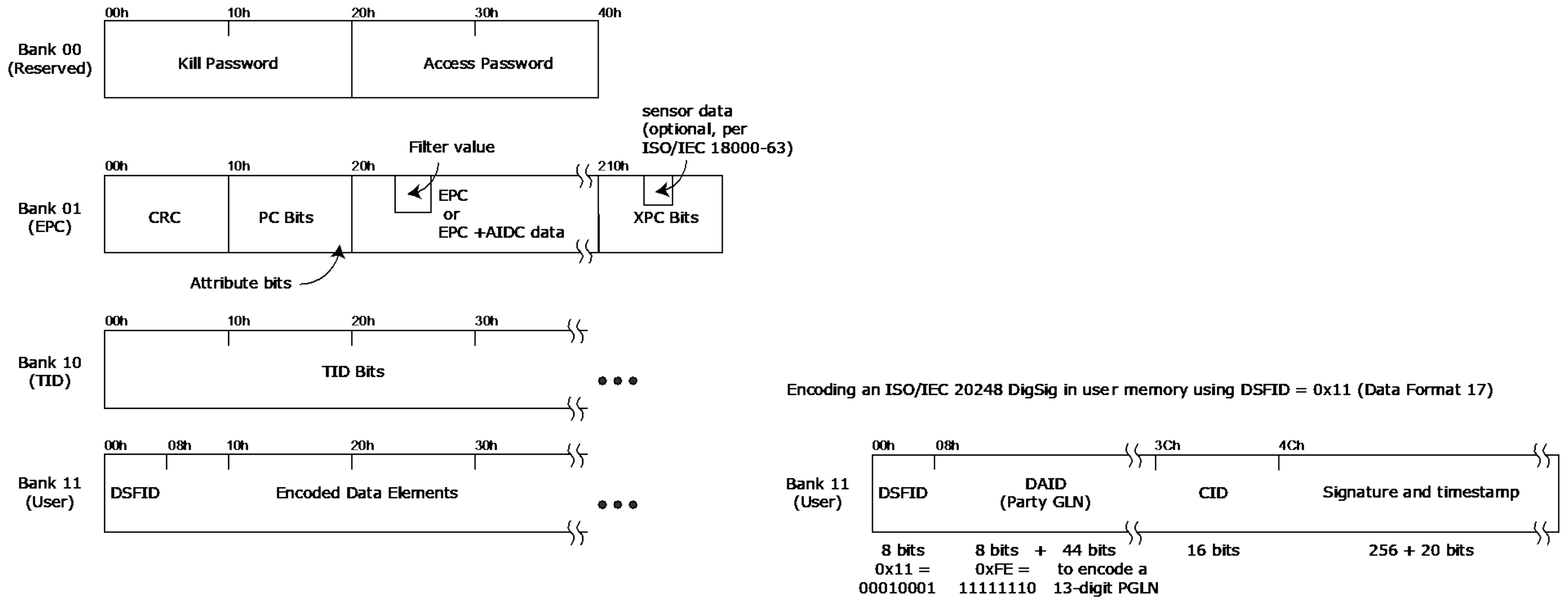


Figure 3-4 Example of DigSig encoded as an EPC on an EPC/RFID tag

3.3 Preparation

The brand owner, a legal entity, is typically the "signer", as the owner of and accountable party of the item data as contained in the barcode or tag. The following steps need to be completed before the encoded product data can be signed, in the context of the examples shown in section 3.1 and 3.2:

1. A PGLN must be registered and active, with a valid licence from a GS1 Member Organisation.

2. Decide what data is to be encoded and what is stored in the cloud. In this example, the encoded data includes the GTIN, the serial number, expiration date and batch/lot number with and without an UV ink code.
3. Decide on how to apply/add the DigSig. In all the examples shown here, it is added to the existing barcode or tag.
4. Design the DigSig Data Description (DDD). In this case the same DDD is used by the brand owner for several product ranges. See Appendix A.

The brand owner, in preparation to sign the products, must be assigned a PGLN to identify themselves as a legal entity and which represents the signer or DigSig generator. The brand owner may assign a third party to generate the DigSigs, with the third party's PGLN indicating the accountable party for the signed data. The third party should have a PGLN assigned by their own organisation, however if this is not possible, the brand owner may allocate a PGLN to the third party, in support of their own business operations. Please refer to the GS1 General Specifications for further details on Allocating Global Location Numbers. Please note that third party contract scenarios e.g., with contract manufacturers or contract packers, for digital signatures is a complex topic and is beyond the scope of this implementation guideline.

The brand owner designs the DigSig data structure to include the reference to the UV ink mark code and the description for how to read it, which is part of the DDD, and publishes it in a digital certificate known as a DigSig Cert. The DigSig does not contain the DigSig Cert since it is far too large. Instead, it contains a unique reference to the DigSig Cert, using the appropriate PGLN and DigSig Cert identifier, as specified by ISO/IEC 20248 [DigSig] clause 7 with the DigSig Cert repository specified in Annex N. This reference contains cross validation to enable detection of DigSig Cert spoofing and DDD tampering. It is important to note that the brand owner is required to specify a set of authoritative certificate repositories from where the DigSig Cert can be obtained, and where an online verification may be done. These authoritative URIs are also contained in the DDD to be cross-checked. A verifier must have this DigSig Cert available to verify. The recommended method is for the verifier to use a trusted repository as described in ISO/IEC 20248 [DigSig] Annex N. A brand owner may host its own trusted repository, although they should also use an external well-known public repository to prevent denial-of-service (DOS) and spoofing attacks.

The DigSig Cert also contains the credentials of the brand owner, including their PGLN and the public key which is to be used for the verification. The brand owner keeps the private key secret, as this is used for the signing. Typically, a brand owner will have one DDD with its DigSig Cert for a range of products.

DigSig Certs should be periodically reissued with new public/private key pairs, typically annually, in order to maintain cryptographic integrity over time. DigSig Certs must have a validity period within the verification lifespan of the products it verifies, to be considered as valid. Note, the validity period must be longer than the sum of the signing period and the product verification lifespan.

3.4 Signing

The product is signed by creating the DigSig with the appropriate information, known as the DDD data. In this example, the information includes:

- The PGLN
- The DigSig Cert ID
- The element string to be encoded on the product’s barcode: `(01)09506000151519(21)12345678p901(10)1234567p(17)221105`
- The UV ink mark code from the product.

This information is passed to a service, as specified in ISO/IEC 20248 [DigSig] Annex D. This service should be secured and is typically sited at the same physical location where the products are quality-assured and where the barcode is applied to it. It returns the DigSig as a string using URI-safe Base64 characters, as specified in section 5 of RFC 4648 [RFC4648], which is appended to the element string following the AI (8030). Note, the DigSig has been shortened for the example shown in Figure 3-5. The element string is then encoded in a barcode or EPC/RFID tag in conformance with GS1 standards and applied to the product: `(01)09506000151519(21)12345678p901(10)1234567p(17)221105(8030)wJgJlkA…A`

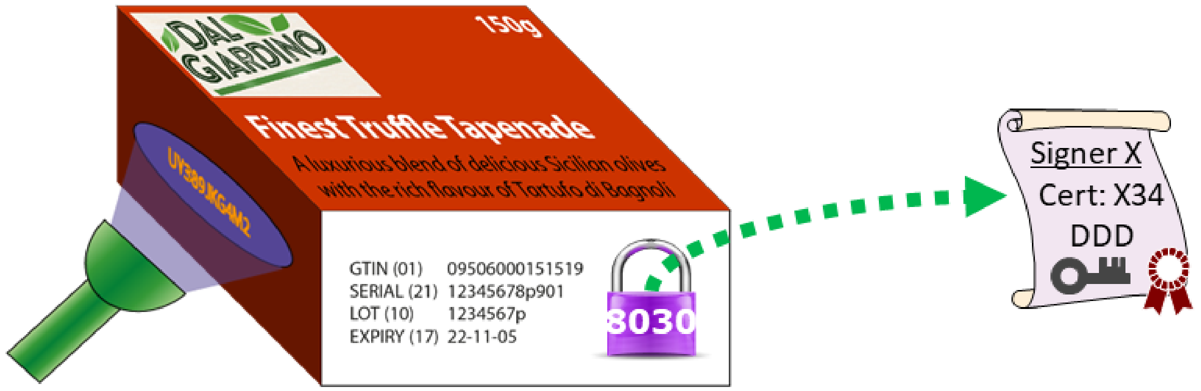


Figure 3-5 Example of referencing the DigSig cert

Note, the size of the encoded data matters! It influences the size of the barcode and the time to read the data from the EPC/RFID tag. It is useful to know that the DigSigs in all these cases are a fixed length of 352 bits or 59 URI-safe Base64 characters. Please refer to the EPC Tag Data Standard (TDS) [TDS] for full details on how to encode a DigSig in an EPC/RFID tag.

3.5 Verification in context

The following series of figures illustrates how a human operator can perform the verification process, although the methods work in a similar way for automated verification processes performed by machines, provided that the data is machine interpretable.

In the first step shown in Figure 3-6, the barcode is scanned. The application software detects that a DigSig is present in the data carrier and requires the corresponding DigSig certificate. If the application software does not already have the DigSig certificate, it would retrieve it before proceeding to the next step of verification.

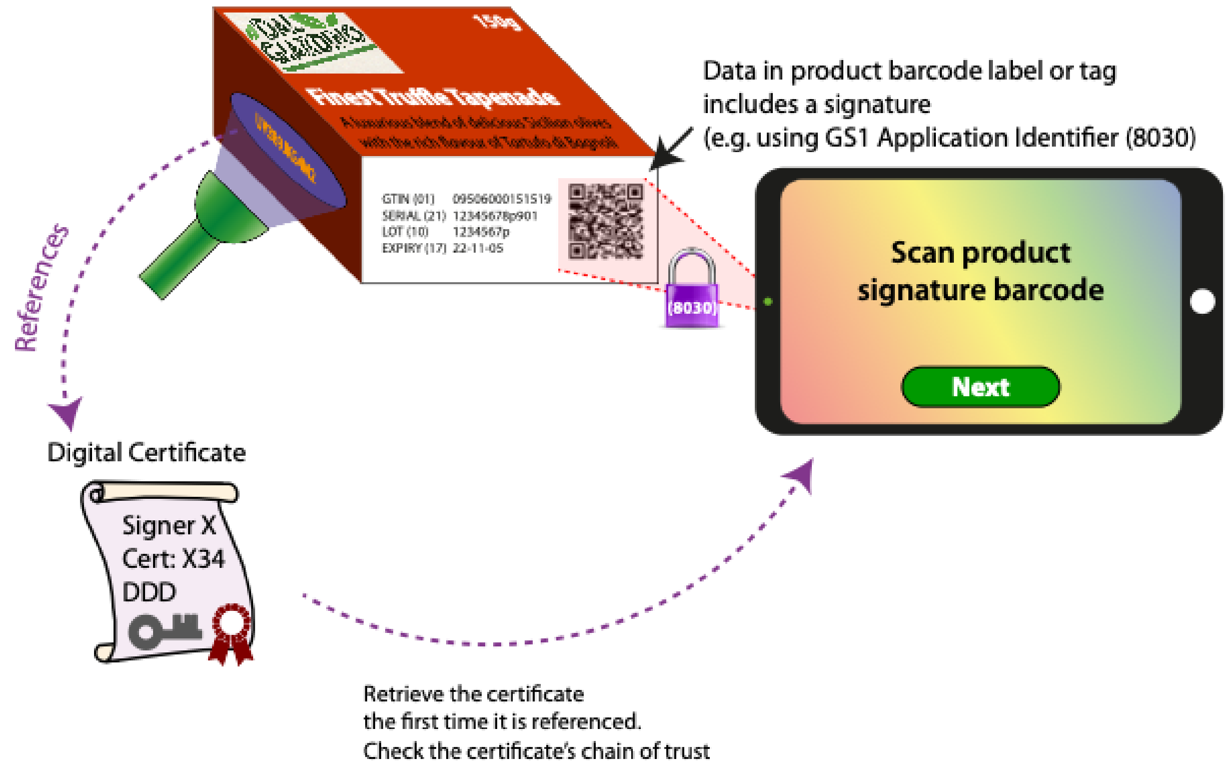

Data in product barcode label or tag includes a signature (e.g. using GS1 Application Identifier (8030)
Finest Truffle Tapenade
150g
GTIN (01) 09506000151519
SERIAL (21) 12345678p901
LOT (10) 1234567p
EXPIRY (17) 22-11-05
(8030)
Scan product signature barcode
Next
References
Digital Certificate
Signer X
Cert: X34
DDD
Retrieve the certificate the first time it is referenced. Check the certificate's chain of trust

Figure 3-6 The DigSig verification process

The DigSig certificate indicates that a security marking needs to be checked, as shown in Figure 3-7. The human operator follows the instructions provided, checks the security marking and then inputs the secret code read from the security marking, to allow the remainder of the verification to be completed automatically.

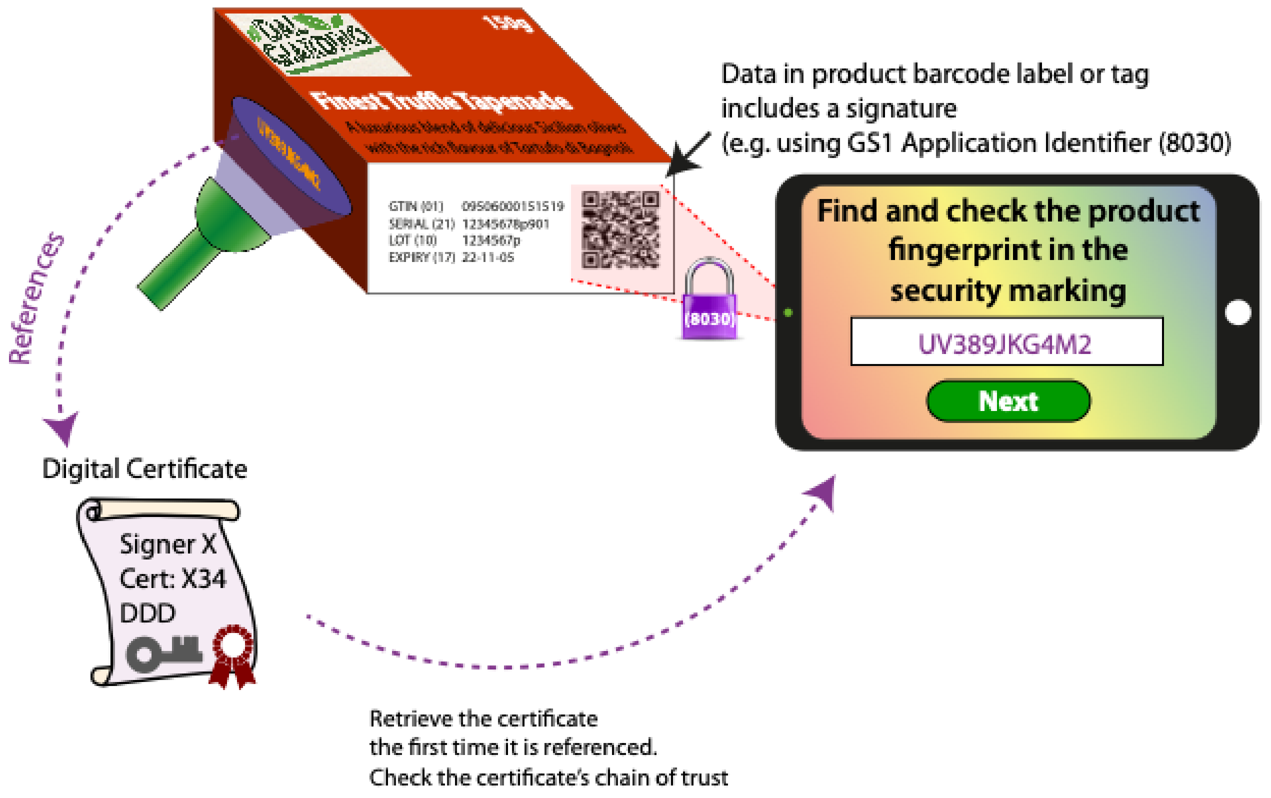


Figure 3-7 Instructions to locate a security marking for verification

The local application and human operator now have trusted product data, as shown in Figure 3-8, to allow them to perform their local actions, such as putting away into inventory or selling a product instance, with confidence.

The DigSig Certificate is a public document, that is verifiable and is not intended to be a secret. It can therefore be cached locally to support further verifications that use that same DigSig Certificate in environments where reliable connectivity to the Internet is highly constrained. Note that revocation checks should be done online.

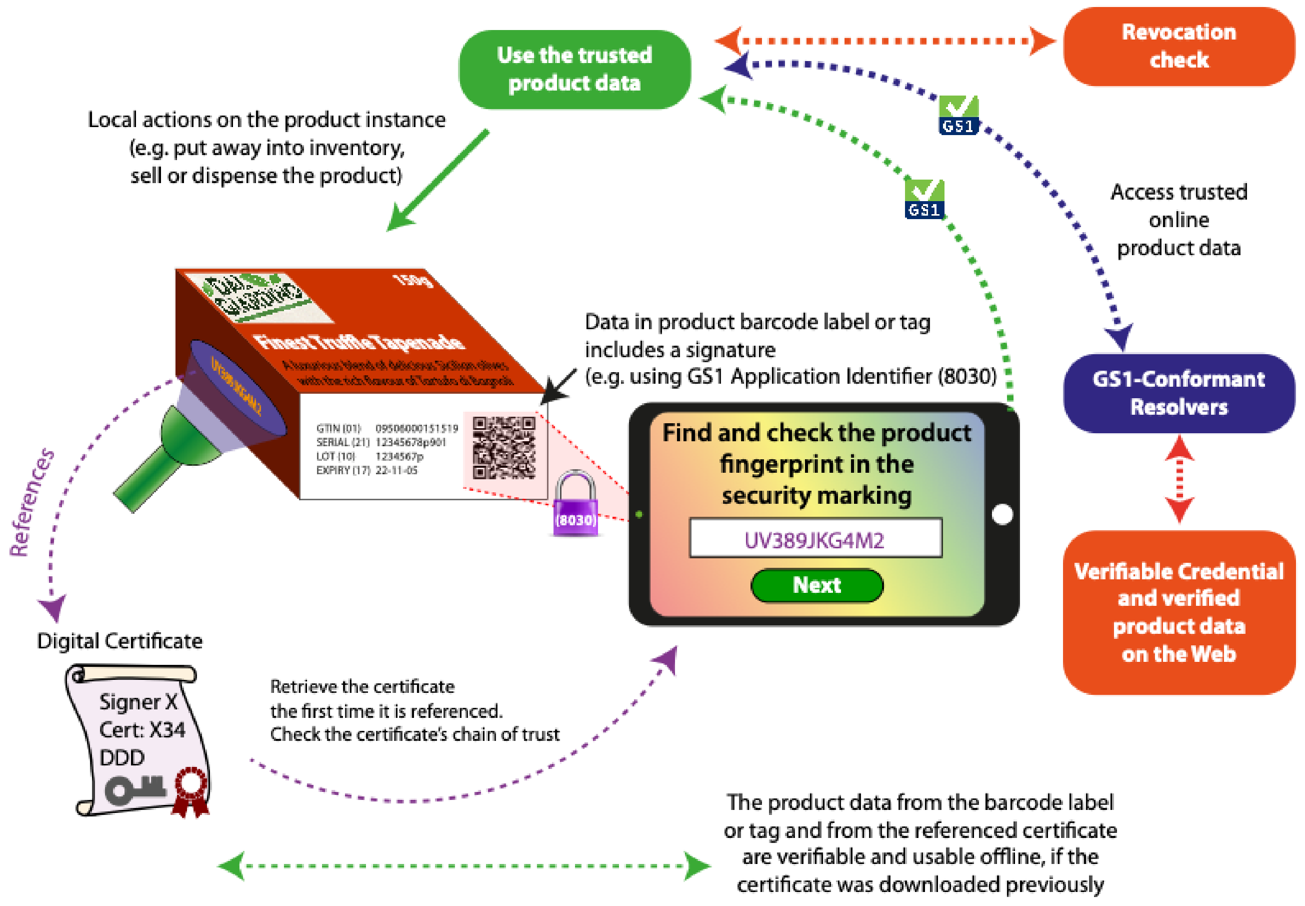

Use the trusted product data
Revocation check
Local actions on the product instance (e.g. put away into inventory, sell or dispense the product)
Access trusted online product data
Data in product barcode label or tag includes a signature (e.g. using GS1 Application Identifier (8030)
GS1-Conformant Resolvers
Find and check the product fingerprint in the security marking
UV389JKG4M2
Next
Verifiable Credential and verified product data on the Web
References
Digital Certificate
Signer X
Cert: X34
DDD
Retrieve the certificate the first time it is referenced. Check the certificate's chain of trust
The product data from the barcode label or tag and from the referenced certificate are verifiable and usable offline, if the certificate was downloaded previously

Figure 3-8 Using trusted product data after verification

When online connectivity is available, it is also possible to obtain other trusted information about the product. If the online information available about the product agrees with the offline information encoded in the data carrier on the product, this increases the level of trust in such information.

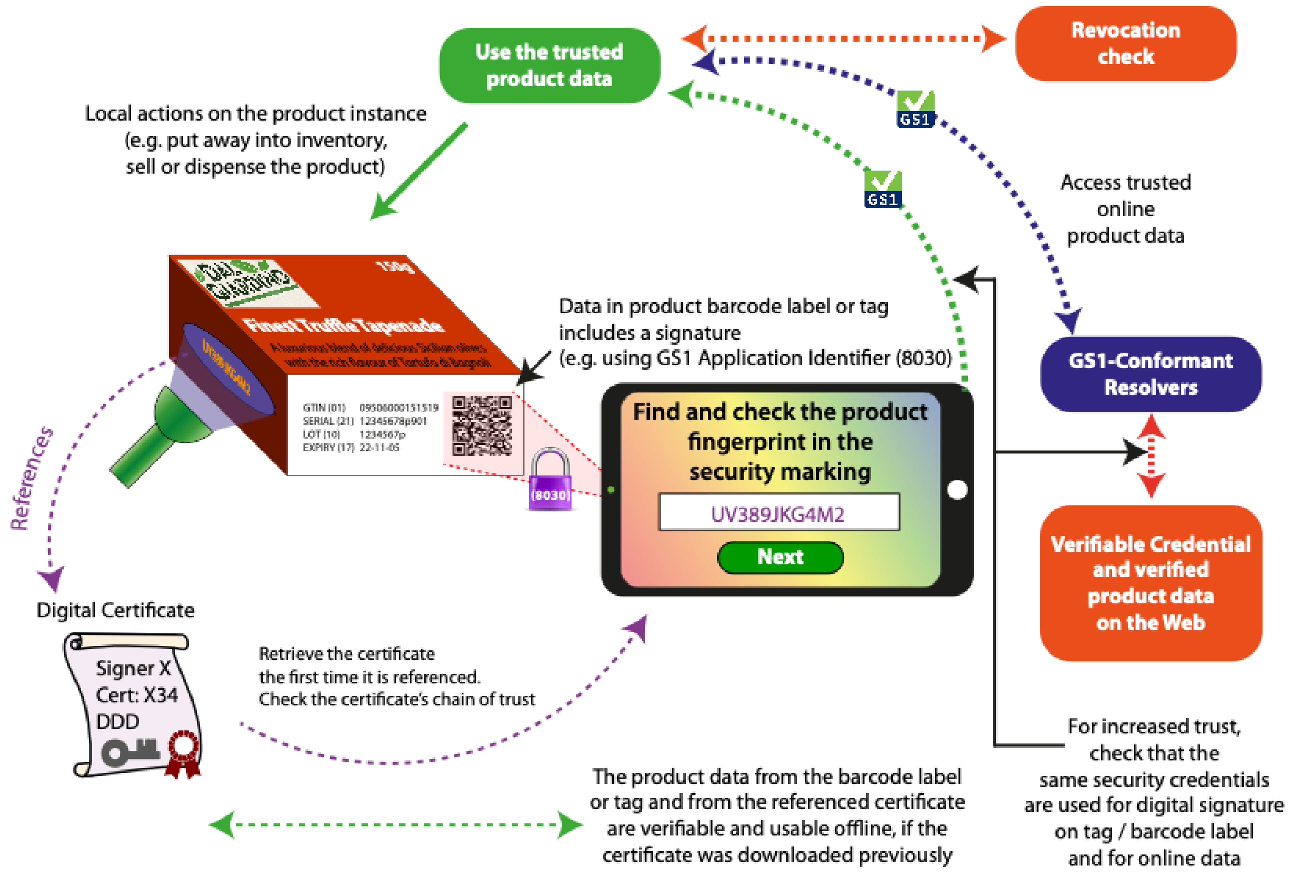

Use the trusted product data
Revocation check
Local actions on the product instance (e.g. put away into inventory, sell or dispense the product)
Access trusted online product data
GS1
Data in product barcode label or tag includes a signature (e.g. using GS1 Application Identifier (8030)
GS1-Conformant Resolvers
Finest Truffle Tapenade
150g
GTIN (01) 09506000151519
SERIAL (21) 12345678p901
LOT (10) 1234567p
EXPIRY (17) 22-11-05
(8030)
Find and check the product fingerprint in the security marking
UV389JKG4M2
Next
Verifiable Credential and verified product data on the Web
References
Digital Certificate
Signer X
Cert: X34
DDD
Retrieve the certificate the first time it is referenced. Check the certificate's chain of trust
The product data from the barcode label or tag and from the referenced certificate are verifiable and usable offline, if the certificate was downloaded previously
For increased trust, check that the same security credentials are used for digital signature on tag / barcode label and for online data

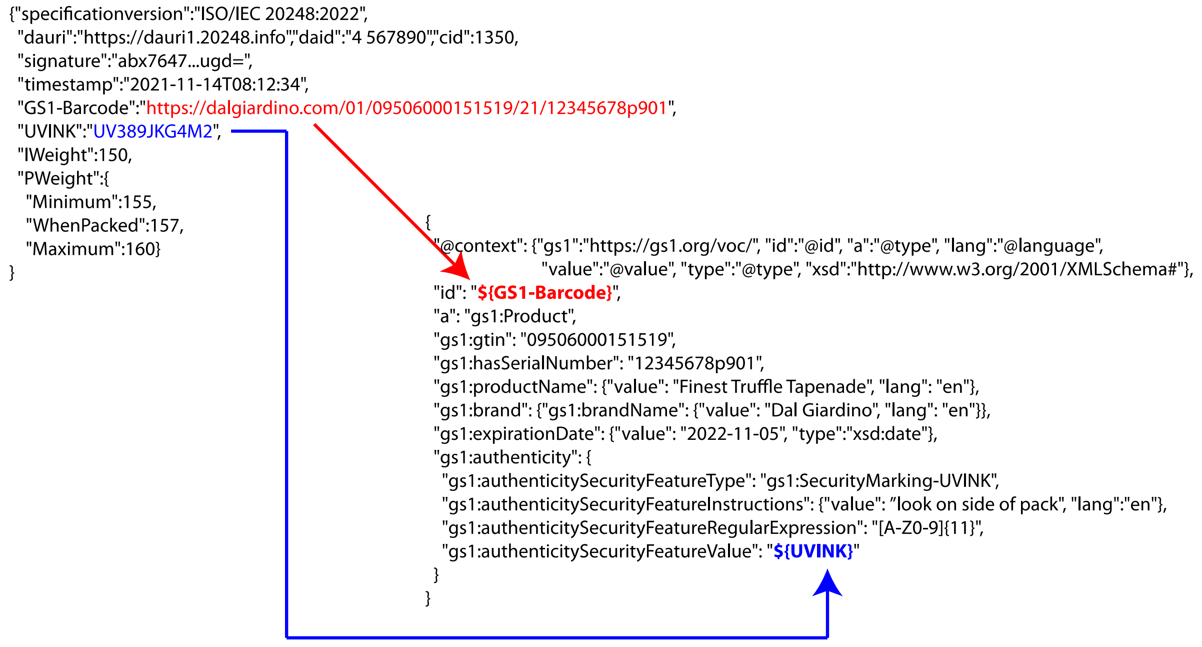
Figure 3-9 Matching online and offline product data to further increase trust

It should be obvious that the level of verification effort is directly related to the risk of the transaction enabled by the product read scenario and identification. As in the physical world, a driver's licence is mostly taken on face-value, since there is an intrinsic trust that it is verified in detail at some other point where counterfeits will be prosecuted. It stands to reason that many fast-moving consumer goods (FMCG) will have a low frequency of "full verification" due to the high volume of consumption and low profit margins. The pure fact that somebody may perform the "full verification" which can expose a counterfeit operation at any time and place, becomes a hurdle for counterfeiters. This illustrates the importance for an easy and robust method to add a digital signature to existing product identifiers and systems. The discussed examples illustrate that the only real impact on the data stack needs to be an extra operation at the time of generating the barcode label or EPC/RFID tag. Verification can be manual and effective.

3.6 Verification output

The DigSig decoder verifier outputs the decoded carrier data and the verification result: accepted, rejected (indicates data tampering), not attempted, or error. It always attempts to decode the data and return what can be decoded, regardless of the verification result. Data decoding may be incomplete for various reasons like, a damaged data carrier, reference data not available, or no support for the specified crypto method.

The decoded data is provided, using JSON [JSON], in a machine-to-machine format (tagged data - `DDDdataTagged`) and/or in a display format (`DDDdataDisplay`) to be used in a user interface. `DDDdataDisplay` uses language, number and time formats as specified in the DigSig Data Description (DDD). `DDDdataTagged` can be mapped to a specified "structure document" as illustrated in Figure 3-10 . It shows the `DDDdataTagged` as JSON-LD [JSON-LD] which is a common format used on the Internet, since it is both machine-readable and human-readable.



```
{"specificationversion":"ISO/IEC 20248:2022",
  "dauri":"https://dauri1.20248.info","daid":"4 567890","cid":1350,
  "signature":"abx7647...ugd=",
  "timestamp":"2021-11-14T08:12:34",
  "GS1-Barcode":"https://dalgiardino.com/01/09506000151519/21/12345678p901",
  "UVINK":"UV389JKG4M2",
  "lWeight":150,
  "PWeight":{
    "Minimum":155,
    "WhenPacked":157,
    "Maximum":160}
}
```

```
{
 "@context": {"gs1":"https://gs1.org/voc/", "id":"@id", "a":"@type", "lang":"@language",
              "value":"@value", "type":"@type", "xsd":"http://www.w3.org/2001/XMLSchema#"},
 "id": "${GS1-Barcode}",
 "a": "gs1:Product",
 "gs1:gtin": "09506000151519",
 "gs1:hasSerialNumber": "12345678p901",
 "gs1:productName": {"value": "Finest Truffle Tapenade", "lang": "en"},
 "gs1:brand": {"gs1:brandName": {"value": "Dal Giardino", "lang": "en"}},
 "gs1:expirationDate": {"value": "2022-11-05", "type":"xsd:date"},
 "gs1:authenticity": {
  "gs1:authenticitySecurityFeatureType": "gs1:SecurityMarking-UVINK",
  "gs1:authenticitySecurityFeatureInstructions": {"value": "look on side of pack", "lang":"en"},
  "gs1:authenticitySecurityFeatureRegularExpression": "[A-Z0-9]{11}",
  "gs1:authenticitySecurityFeatureValue": "${UVINK}"
 }
}
```

Figure 3-10 JSON-LD message populated with the verified DigSig data

4 Accessing digital signatures via the Web

4.1 General overview

As an alternative to encoding a digital signature construct in a data carrier, digitally signed data can be retrieved via a Web request, such as a lookup of a GS1 Digital Link URI [DL-URI] for the physical entity. GS1 Digital Link URIs [DL-URI] and resolver infrastructure [DL-Resolution] are a modern way to connect barcodes from physical entities such as trade items, assets or logistics units, to information on the Web. GS1 and other organisations such as individual brand owners, or solution providers, can operate resolver infrastructure for GS1 Digital Link [DL-Resolver] or serve content directly from those URIs.

Existing open standard technical approaches for digitally signing data include JSON Web Signatures (JWS) [JWS], XML Signatures [XML-DS] or Verifiable Credentials [VC]. In all of these approaches, the data payload can link the instance level identifier of the physical entity e.g., GTIN + Serial, SSCC etc. to structured master data about the physical entity.

A dedicated 'link type' property gs1:jws (https://gs1.org/voc/jws) is already defined within the GS1 Web Vocabulary [WebVoc] to enable a Web request to specify that it would like to receive a JSON Web Signature (JWS) [JWS] for the physical entity identified by the specified GS1 Digital Link URI. Brand owners can configure resolver records so that such requests are redirected to a target Web resource that returns the corresponding JSON Web Signature [JWS] data.

The structure of JSON Web Signatures [JWS] is defined by RFC 7515. Each JSON Web Signature [JWS] consists of three parts:

- A header that contains information about the algorithms used and typically a link to a public key certificate
- A body section containing the data payload
- A footer containing the actual digital signature

The data payload may also include details about authentication codes marked within security markings on the physical entity, such as holograms, microprinting, watermarks or UV ink printing that becomes visible under ultra-violet light.

In this section, we describe three alternative workflows, with increasing levels of sophistication, as shown in Figure 4-1.

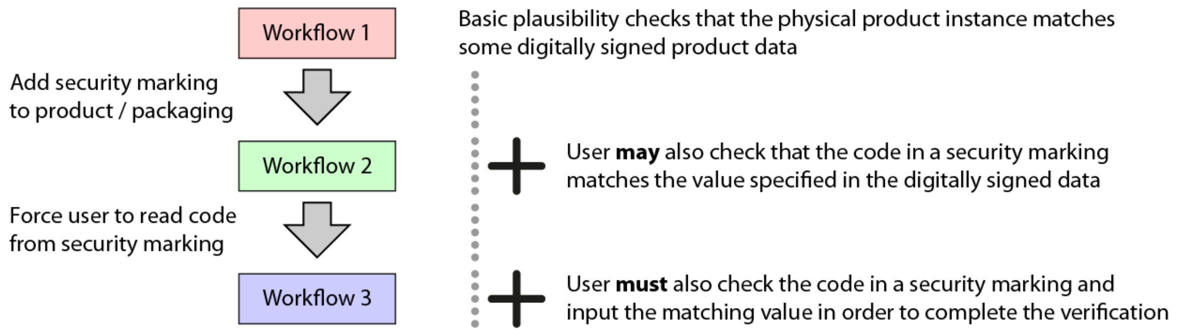

Workflow 1
Basic plausibility checks that the physical product instance matches some digitally signed product data
Add security marking to product / packaging
Workflow 2
User **may** also check that the code in a security marking matches the value specified in the digitally signed data
Force user to read code from security marking
Workflow 3
User **must** also check the code in a security marking and input the matching value in order to complete the verification

Figure 4-1 Three alternative workflows for accessing Digital Signatures via the Web

1. In the first workflow, the digitally signed data includes details that are characteristic of the physical entity or marked on its packaging to enable a basic plausibility check.
2. The second workflow can provide the same features as the first workflow, but can additionally reveal the presence and value of an authentication code marked within a physical security marking on the entity or its packaging. This enables an additional check to confirm that the digitally signed data really does correspond to that specific instance of the entity.
3. The third workflow takes the second workflow a step further and instead of immediately revealing the actual value of the authentication code, the user is presented with instructions about what kind of security marking to look for and where to look. The user must then perform a physical inspection of the entity and read the actual value of the authentication code from that instance of the entity, before verification of the digital signature can be attempted.

4.2 Workflow 1: Enabling basic plausibility checks

Mobile applications might make use of the "Verified by GS1" service [VBG] to enable a basic plausibility check about the product and could display the product image along with a small number of basic attributes for the product identified by that GTIN.

Alternatively, a mobile application might make use of a GS1 Digital Link URI syntax [DL-URI] and resolver infrastructure [DL-Resolution] to find various kinds of data, information resources and services on the World Wide Web that are relevant for the product identified by that GTIN. Note that GS1 Digital Link URIs and GS1 conformant resolvers [DL-Resolution] also support redirection for more granular identification such as GTIN + Serial Number for instance-level information, or GTIN + batch/lot number for specific batch/lot information about that product GTIN, so details can also be checked against an expiration date printed on the packaging of a specific product instance. The global GS1 resolver service [DL-Resolver] for GS1 Digital Link URIs will redirect only to Web URLs specified by the respective licensee of the GS1 Company Prefix (GCP) or individual GTIN license, typically the brand owner of the product.

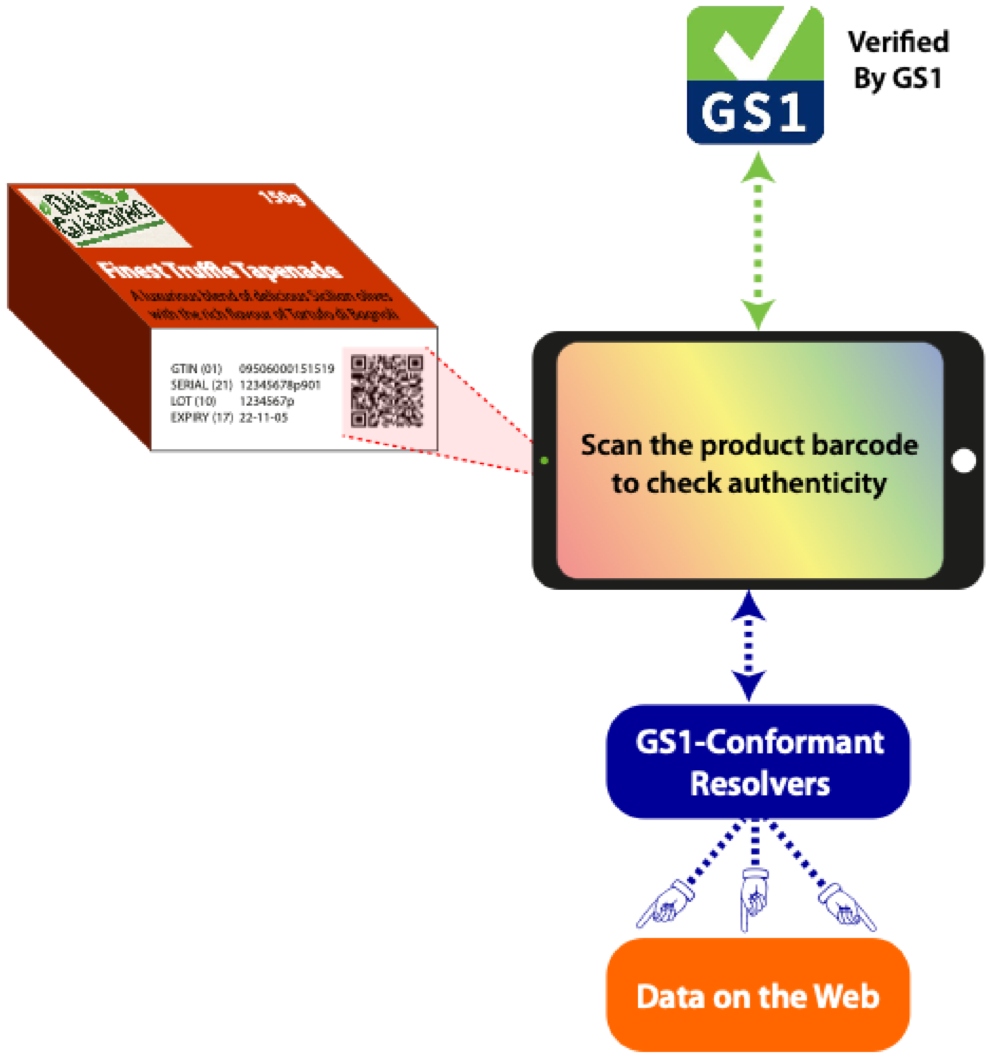


Figure 4-2 Using Verified by GS1 and a resolver infrastructure to enable basic plausibility checks

Whether basic information is retrieved via the “Verified by GS1” service [VBG] or by using GS1 Digital Link URIs [DL-URI] and a resolver infrastructure [DL-Resolution] as shown in Figure 4-2, it should be possible to display some information to enable a basic plausibility check. This could include an image of the product, or basic product attribute information such as its product name, weight or net content and potentially data encoded in the barcode using GS1 Application Identifiers (AI), such as the GTIN and a specific batch/lot identifier, serial number or expiration date, as shown in Figure 4-3. The GS1 Web Vocabulary [WebVoc] and schema.org [Schema] can be used to express such details as structured Linked Data in JSON-LD format [JSON-LD], which a mobile app can then display or further process. For example, the digitally signed data could include information about the presence of allergens or additives, suitability for specific diet types or certifications and accreditations about the product. A dedicated mobile app might then alert a user if the details in the digitally signed data conflict with user configurations in the app settings regarding their dietary preferences and intolerances or ethical, environmental or sustainability preferences. As this type of data can be digitally signed by the brand owner, the consumer can have a high level of trust that the data is as specified by the brand owner and should be correct. Other data such as net weight, product dimensions or a product image can also be included in the digitally signed data to be checked against the physical product instance to ensure that they match; otherwise, this should raise suspicions.

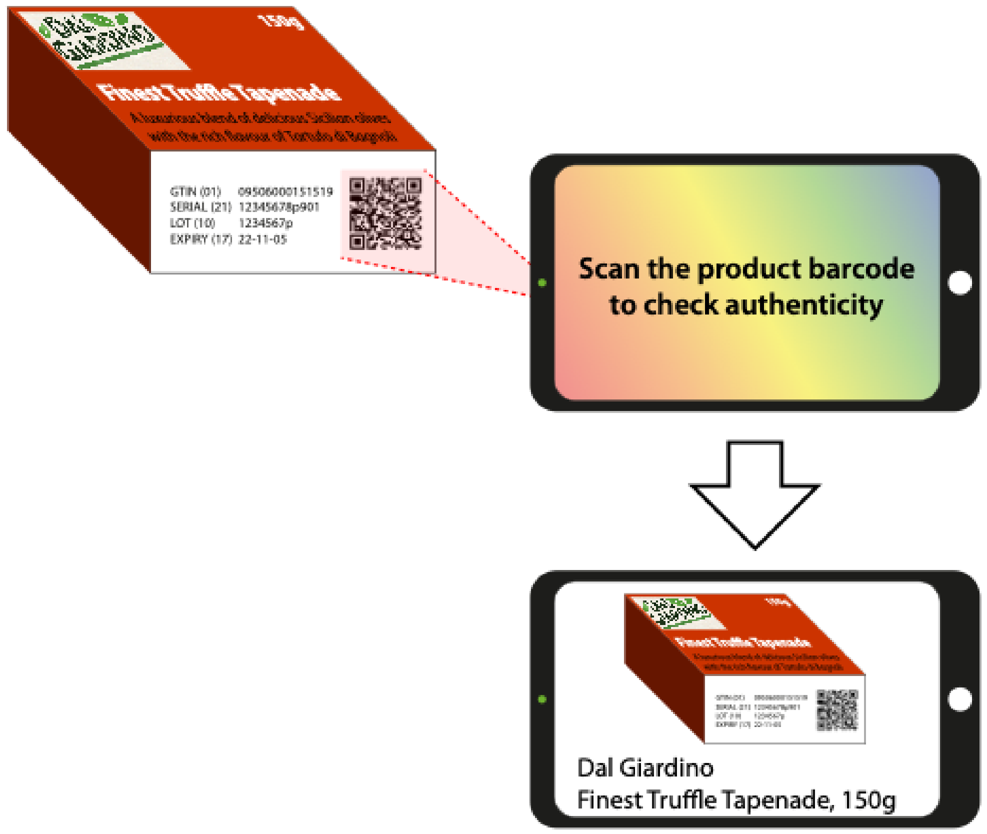


Figure 4-3 Using product information and image to enable basic plausibility checks

4.3 Workflow 2: Displaying an authentication code from a physical security marking

The second and third workflows involve verifying a JSON Web Signature [JWS] and checking an authentication code that appears within a physical security marking (e.g. hologram, ultra-violet ink markings, watermarks, microprinting) appearing on the product instance.

While a product barcode (even at serial level) could simply be copied and reprinted on counterfeit or illicit products, using an additional physical security marking in combination with digitally signed data, can provide a much higher level of assurance that the digitally signed data does actually correspond to that specific physical entity and matches the additional authentication codes marked within security markings on the physical object. Although this adds some complexity and cost, it also raises the economic barriers to counterfeiters, to deter them from targeting products that are protected with physical security markings combined with digitally signed data.

In a simple approach, the digitally signed data payload might simply indicate what the expected authentication code should be and where to look for it on the product, as shown in Figure 4-4. In this scenario, the user scans the product barcode with a mobile application and in addition to the basic plausibility checks from workflow 1 shown in Section 4.2, the mobile application can advise the user to inspect for one or more specific types of security marking in a particular location(s) and to check that each contains a specific code.

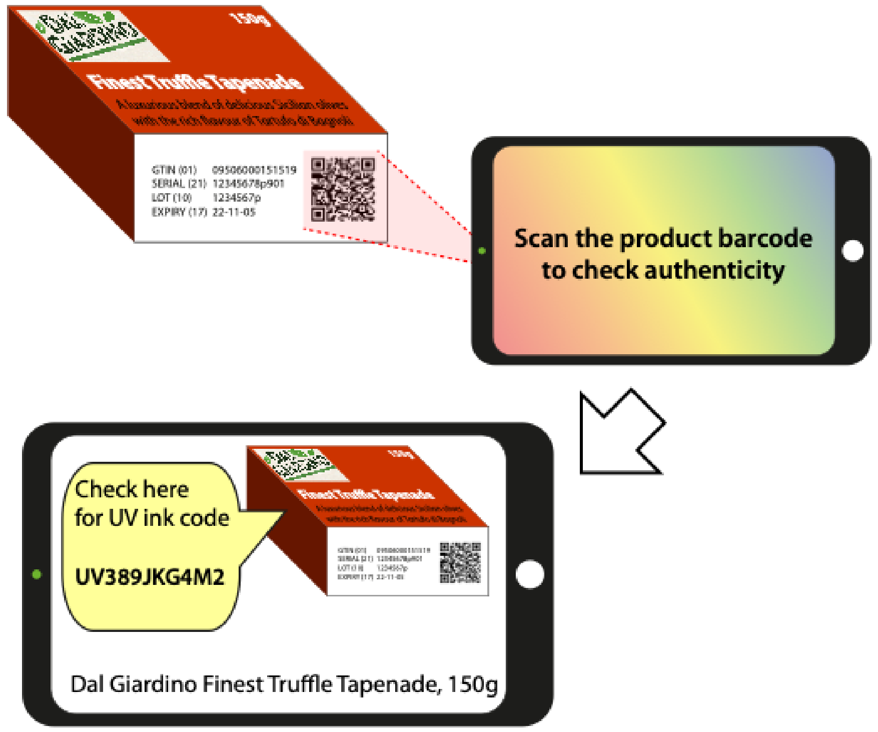


Figure 4-4 Matching authentication codes to verify the digital signature

4.4 Workflow 3: Input of an authentication code from a physical security marking to verify the digital signature

Alternatively, in a slightly more challenging version of workflow 2, the expected value of the authentication code might not be immediately revealed to the user – instead the user is instructed to perform a physical inspection of the product instance and read the code from the security marking, then input this value before attempting to verify the digital signature, as illustrated in Figure 4-5. All three workflows follow a similar approach, with some incremental modifications in workflows 2 and 3.

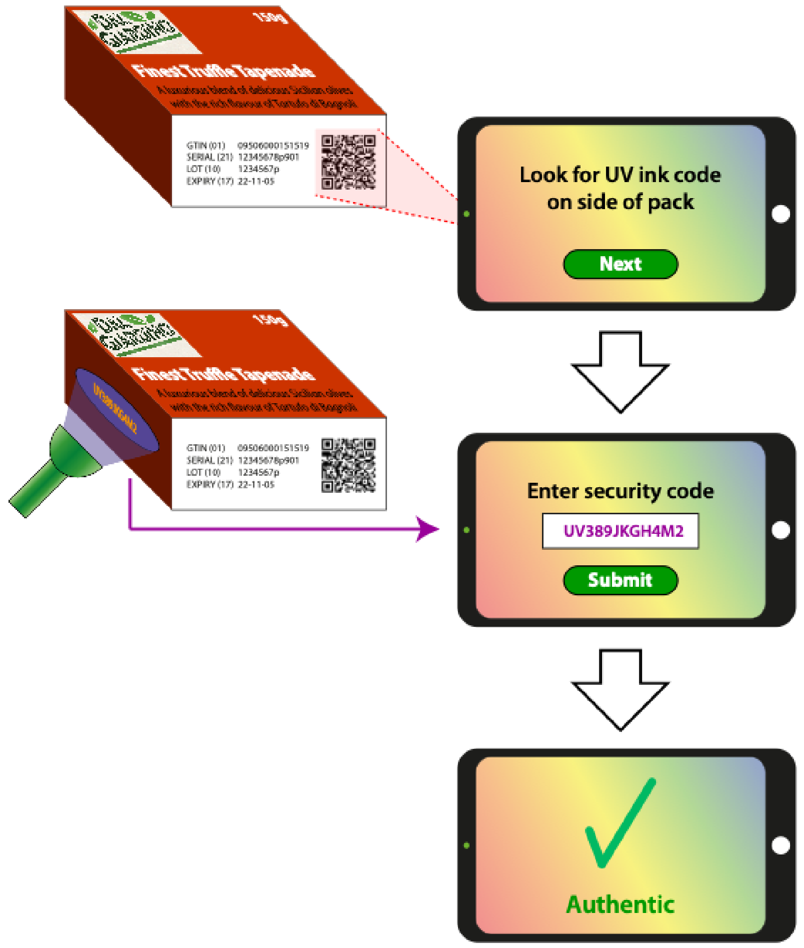


Figure 4-5 Input of an authentication code to verify the digital signature

In all workflows, the process begins when the user's mobile phone scans the barcode found on the physical entity, which in these examples are a product instance. That barcode could be a QR Code that natively encodes a GS1 Digital Link URI [DL-URI] which can be easily read by a mobile phone's camera. Alternatively, it could be a GS1 barcode such as a GS1 DataMatrix that encodes a GS1 element string, for which a corresponding GS1 Digital Link URI can be constructed. A GS1 Digital Link URI behaves like any other URL, for making a Web request and retrieving online information. Often this will default to whatever default link type was configured by the brand owner, often a human-readable Web page with information about the product or perhaps a specific promotion connected with that product. However, GS1 Digital Link URIs also enable multi-functional linking to different kinds of information and services on the Web. A mobile app can express which kind of information it would like to receive by specifying a link type within the query string of the URI when making a Web request. For example, to request usage instructions or an instruction manual, an app could specify linkType=gs1:instructions within the URI query string. A full list of currently defined link types is available at https://gs1.org/voc/?show=linktypes. Within these, a dedicated link type gs1:jws can be used to request redirection to an online JSON Web Signature [JWS], if the licensee of the GTIN (typically the brand owner) has provided that for the product instance.

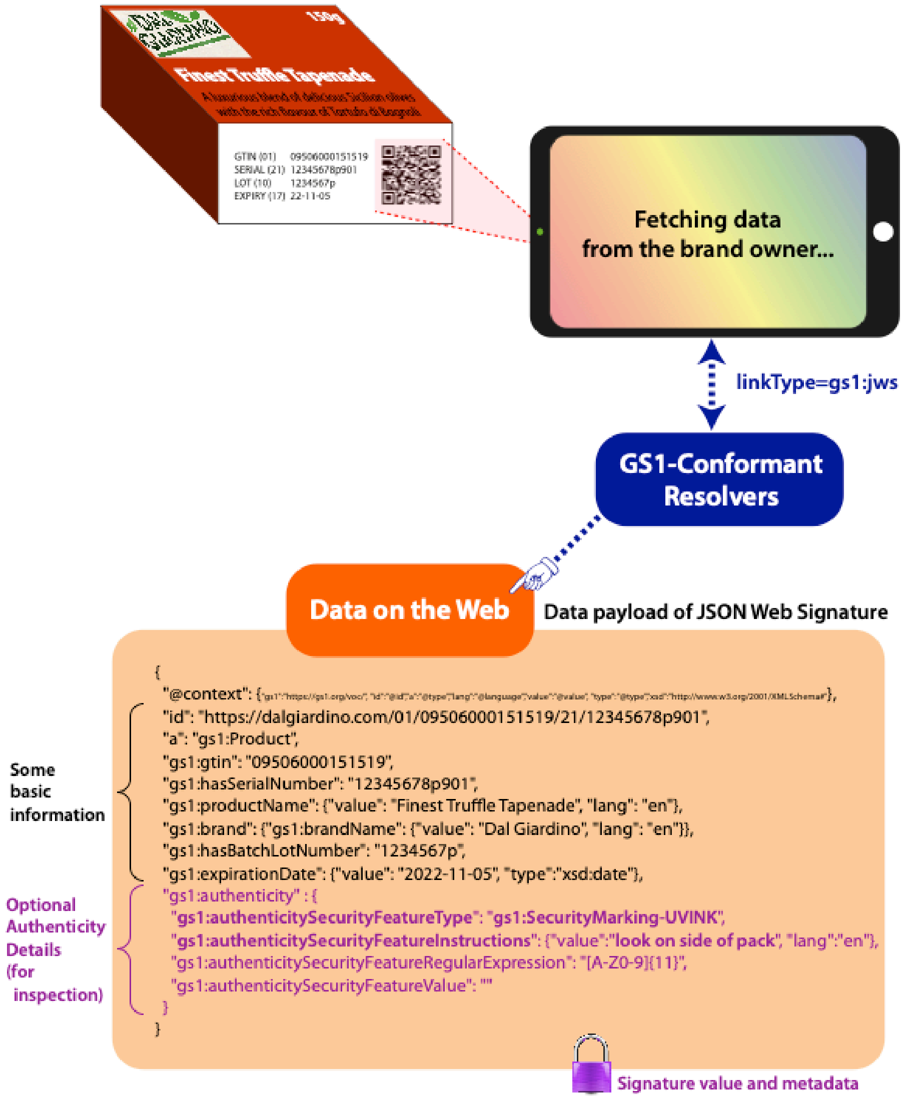
Referring to RFC 7515, which defines JSON Web Signatures [JWS] or by using one of the toolkits or libraries to support JSON Web Signatures, the JSON Web Signature can be split into its header data, body data payload and the footer containing the signature. The body data payload will typically be in JSON format [JSON], but these three workflows actually use JSON-LD [JSON-LD] (which is still a valid JSON format but with some extra features for Linked Data and multiple namespaces).

In workflow 1, the body data payload typically expresses some basic information about the product or product instance but nothing about security markings. The GS1 Digital Link URI [DL-URI] can be used as the identifier for various Linked Data factual claims, i.e., as the RDF Subject. It is also possible to use GS1 Digital Link URIs at different levels of granularity to express facts that are defined for the entire product class identified by that GTIN, versus other facts that are defined for a specific batch/lot or even for a specific serial number.

Workflows 2 and 3 involve an authentication code present in a physical security marking on the product or its packaging. The digitally signed data can now also include information about what kind of physical security features are present and should be inspected, as well as details about where to find these, as illustrated in Figure 4-6, which shows some additional data (shown in purple) as values of a Linked Data property gs1:authenticity, defined within the GS1 Web Vocabulary [WebVoc]. Its value is the class gs1:AuthenticityDetails that can express various properties to indicate which kind of security markings appear on the product and instructions about where to find them. These additional details for workflows 2 and 3 are shown in purple in Figure 4-6 and Figure 4-7.

In workflow 2, the code value within the security marking is still present within the data payload, as the value of the property gs1:authenticitySecurityFeatureValue.

In workflow 3, the code value within the security marking was present within the data payload, as the value of the property gs1:authenticitySecurityFeatureValue at the time when the digital signature is generated, typically by the brand owner directly or by an authorised agent such as a contract manufacturer or contract packer. However after the signature has been generated, the body data payload is altered to remove the actual value of that code, e.g. by setting the value of the property gs1:authenticitySecurityFeatureValue back to an empty string, in order to force the user to inspect the security marking(s) on the product, read the code value(s) and input the code value(s) into the mobile app, which inserts the inspected code values back into the data before attempting to verify the digital signature.



```
{
 "@context": {"gs1":"https://gs1.org/voc/","id":"@id","a":"@type","lang":"@language","value":"@value", "type":"@type","xsd":"http://www.w3.org/2001/XMLSchema#"},
 "id": "https://dalgiardino.com/01/09506000151519/21/12345678p901",
 "a": "gs1:Product",
 "gs1:gtin": "09506000151519",
 "gs1:hasSerialNumber": "12345678p901",
 "gs1:productName": {"value": "Finest Truffle Tapenade", "lang": "en"},
 "gs1:brand": {"gs1:brandName": {"value": "Dal Giardino", "lang": "en"}},
 "gs1:hasBatchLotNumber": "1234567p",
 "gs1:expirationDate": {"value": "2022-11-05", "type":"xsd:date"},
 "gs1:authenticity" : {
  "gs1:authenticitySecurityFeatureType": "gs1:SecurityMarking-UVINK",
  "gs1:authenticitySecurityFeatureInstructions": {"value":"look on side of pack", "lang":"en"},
  "gs1:authenticitySecurityFeatureRegularExpression": "[A-Z0-9]{11}",
  "gs1:authenticitySecurityFeatureValue": ""
 }
}
```

Figure 4-6 Digitally signed data payload

When the digital signature is generated, the actual value of the authentication code should be present within the data payload – but in workflow 3, this value can later be removed from the data payload as illustrated in Figure 4-7. Note that the value of gs1:authenticitySecurityFeatureValue has been reset to an empty string, in order to prompt a human user to perform a physical inspection to read the actual code value from the product instance and re-insert this missing value within the data payload before attempting to verify the digital signature. Until the missing value of the authentication code is found and re-inserted within the data payload as the value of gs1:authenticitySecurityFeatureValue, there is no possibility that the digital signature can be verified successfully. This is because essential data, that was present when the signature was generated, is initially missing until the user enters the correct code value read from the security marking. When the code from the security marking has been entered, the mobile app inserts that code value into the data payload and can then begin to verify the digital signature.

Figure 4-6 shows a simple example of the data payload of a JSON Web Signature [JWS] in which a block of JSON-LD [JSON-LD] data provides some basic information about the product instance, as well as some optional additional details about authentication codes present in security markings on the product or its packaging. The example data is shown in Figure 4-7 and illustrates the use of properties defined by the GS1 Web Vocabulary [WebVoc], specifically properties defined within the gs1:AuthenticityDetails class. However, in this data, the actual value of the authentication code in the security marking is still present as the value of the property gs1:authenticitySecurityFeatureValue. In workflow 3, the data payload can be modified after calculating the digital signature, by setting this value to an empty string as shown in yellow highlight in Figure 4-7, in order to force a physical inspection of the product and its security markings, as described above.

```
{
  "@context": {
     "gs1": "https://gs1.org/voc/",
     "id": "@id",
     "a": "@type",
     "lang": "@language",
     "value": "@value",
     "type": "@type",
     "xsd": "http://www.w3.org/2001/XMLSchema#"
  },
  "id": "https://dalgiardino.com/01/09506000151519/21/12345678p901",
  "a": "gs1:Product",
  "gs1:gtin" : "09506000151519",
  "gs1:hasSerialNumber" : "12345678p901",
  "gs1:productName" : {"value": "Finest Truffle Tapenade", "lang": "en"},
  "gs1:brand" : {"gs1:brandName": {"value": "Dal Giardino", "lang": "en"}},
  "gs1:hasBatchLotNumber" : "1234567p",
  "gs1:expirationDate" : {"value": "2022-11-05", "type": "xsd:date"},
  "gs1:authenticity" : {
    "gs1:authenticitySecurityFeatureType" : "gs1:SecurityMarking-UVINK",
    "gs1:authenticitySecurityFeatureInstructions" : {"value": "look on side of pack", "lang": "en"},
    "gs1:authenticitySecurityFeatureRegularExpression" : "[A-Z0-9]{11}",
    "gs1:authenticitySecurityFeatureValue" : "UV389JKG4M2"
  }
}
```

Figure 4-7 Extract of JSON-LD data from JSON Web Signature data payload

Appendix B provides further detail about how a JSON Web Signature [JWS] can make use of recent extensions to the GS1 Web Vocabulary [WebVoc] to include details of unique codes in physical security markings within the digitally signed data, to ensure that the unique identifier of the physical product instance cannot be trivially copied to an instance of an unauthentic or counterfeit product.

4.5 Completing the verification of the digital signature

A mobile app examines the JSON Web Signature [JWS] metadata in the header to extract the information it needs to retrieve the public key certificate or Verifiable Credentials [VC]. It then checks that the chain of trust is unbroken and credible within the public key certificate or Verifiable Credential [VC] before verifying the signature using the specified algorithm and public key. If the digital signature can be verified successfully, the user is informed that the product appears to be authentic.

4.6 Further exploratory work within GS1

GS1 is also investigating and prototyping Verifiable Credentials [VC] to provide a robust chain of trust from GS1 Global Office via the relevant GS1 Member Organisation (MO) to the licensee of a GS1 Company Prefix (GCP) or single-issue GTIN, which can link to the public key of the signing party.

This means that GS1 can act as a trust anchor for the connection between the public key of the signing party and the GTIN of the product, without relying on a certificate authority for this relationship.

This can help to determine which organisation has authority to make factual assertions about the product instance, typically the brand owner.

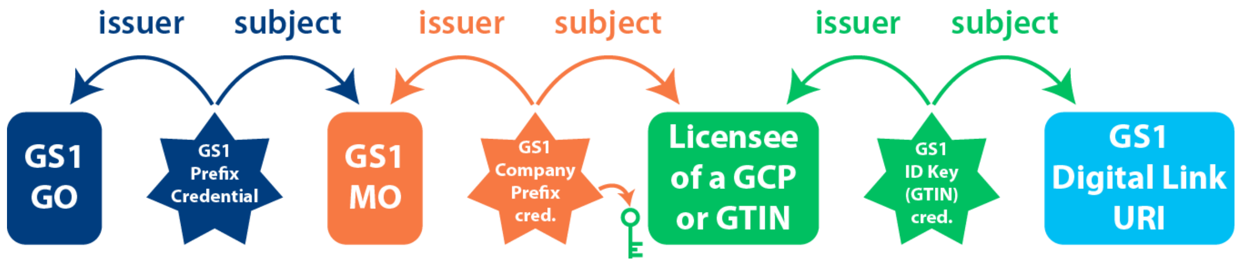


Figure 4-8 Using Verifiable Credentials to provide a chain of trust

In the example shown in Figure 4-8, the star shapes represent verifiable credentials that specify an issuer and a subject. The issuer points to the organisation that issued the credential, whereas the subject points to the organisation or entity for which various claims are made.

The first verifiable credential, shown in Figure 4-8 as dark blue, would be issued by GS1 Global Office and asserts that a specific GS1 Member Organisation (e.g. GS1 Netherlands) has GS1 Prefixes in a specific range (e.g. 870-879 for GS1 Netherlands).

The second verifiable credential, shown in Figure 4-8 as orange, is asserted by the GS1 Member Organisation about an individual member company and asserts that the individual member company has licensed either a specific GS1 Company Prefix (GCP) from that GS1 Member Organisation or has licensed a single-issue GS1 identification key (e.g. single-issue GTIN or single-issue GLN) from that GS1 Member Organisation. The credential could also include a reference to the public key of the individual member company.

Lastly, the third verifiable credential, shown in Figure 4-8 as green, is asserted by an individual company that licences either a GS1 Company Prefix (GCP) or single-issue GS1 identification key from a GS1 Member Organisation. This credential then specifies a subject which may be a GS1 Digital Link URI [DL-URI], shown in Figure 4-8 as light blue, based on the single issue GS1 identification key or the GS1 Company Prefix that was licensed from the GS1 Member Organisation.

This collection of verifiable credentials therefore represents a chain of trust connecting GS1 Global Office to every GS1 Digital Link URI that has been constructed using a GS1 Company Prefix or individual single-issue GS1 identification key that has been licensed via a GS1 Member Organisation. Note however that the current ontology for Verifiable Credentials [VC] does not provide much practical guidance about how to navigate through such chains of credentials, even though this would appear to be a typical use case for any identification system, such as international telephone numbers for example, that involve a hierarchical system of delegation.

The green rectangle shown in Figure 4-8 indicates an individual company that has a current and valid licence issued by a GS1 Member Organisation. It should be noted, that an individual organisation should be identified by their Party Global Location Number (PGLN), corresponding to GS1 Application Identifier (417), as defined by the GS1 General Specifications [GenSpecs] - and that a GS1 Company Prefix does not uniquely identify a company. A company may licence multiple GS1 Company Prefixes or none, in the case of a single-issue identification key licences. There can also be situations where a GS1 Member Organisation retains control over a GS1 Company Prefix and uses it to issue single-issue identification keys to separate individual companies who have no relationship in common, other than all of them being licensees of single-issue identification keys from that GS1 Member Organisation.

5 References

Abbreviation	Title	Referenced link (if known)
[DigSig]	ISO/IEC 20248: Information technology – Automatic identification and data capture techniques – Digital signature data structure schema	ISO/IEC (International Organization for Standardization / International Electrotechnical Commission) https://www.iso.org/standard/81314.html
[DL-Resolution]	GS1 conformant resolver standard	GS1 https://ref.gs1.org/standards/resolver/
[DL-Resolver]	GS1 Resolver service	GS1 https://www.gs1.org/standards/resolver
[DL-URI]	GS1 Digital Link Standard: URI Syntax	GS1 https://ref.gs1.org/standards/digital-link/uri-syntax/
[EPCIS]	EPC Information Services (EPCIS) Standard	GS1 https://ref.gs1.org/standards/epcis/
[EU2023/0124COD]	Proposal for a REGULATION OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL on detergents and surfactants, amending Regulation (EU) 2019/1020 and repealing Regulation (EC) No 648/2004	The European Parliament and the Council of the European Union https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A52023PC0217
[EU2023/0298COD]	Proposal for a REGULATION OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL on the safety of toys and repealing Directive 2009/48/EC	The European Parliament and the Council of the European Union https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A52023PC0462&qid=1714461913721
[EU2023/1542]	REGULATION (EU) 2023/1542 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 12 July 2023 concerning batteries and waste batteries, amending Directive 2008/98/EC and Regulation (EU) 2019/1020 and repealing Directive 2006/66/EC	The European Parliament and the Council of the European Union https://eur-lex.europa.eu/eli/reg/2023/1542/oj
[GenSpecs]	The GS1 General Specifications	GS1 https://ref.gs1.org/standards/genspecs/
[JSON-LD]	JSON-LD: A JSON-based Serialization for Linked Data	The World Wide Web Consortium (W3C) https://www.w3.org/TR/json-ld11
[JSON]	RFC 8259 - The JavaScript Object Notation (JSON) Data Interchange Format	Internet Engineering Task Force (IETF) https://www.rfc-editor.org/info/rfc8259

Abbreviation	Title	Referenced link (if known)
[JWS]	RFC 7515 - JSON Web Signature (JWS)	Internet Engineering Task Force (IETF) https://www.rfc-editor.org/info/rfc7515
[RFC4648]	The Base16, Base32, and Base64 Data Encodings	Internet Engineering Task Force (IETF) https://www.rfc-editor.org/info/rfc4648
[SC31]	ISO/IEC JTC1/SC31: Automatic identification and data capture techniques	ISO/IEC (International Organization for Standardization / International Electrotechnical Commission) https://www.iso.org/committee/45332.html
[Schema]	Schema.org vocabulary	Schema.org https://schema.org/
[SHS]	Secure Hash Standard (SHS)	The Federal Information Processing Standards Publication Series of the National Institute of Standards and Technology (NIST) https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf
[TDS]	EPC Tag Data Standard (TDS)	GS1 https://ref.gs1.org/standards/tds/
[UHFG2]	EPC® Radio-Frequency Identity Generation-2 UHF RFID Standard	GS1 https://www.gs1.org/standards/rfid/uhf-air-interface-protocol
[VBG]	Verified by GS1 service	GS1 https://www.gs1.org/services/verified-by-gs1
[VC]	GS1 Verifiable Credentials – White Paper on Data Model and Validations	GS1 https://ref.gs1.org/gs1/vc/data-model/
[WebVoc]	The GS1 Web Vocabulary	GS1 https://gs1.org/voc/
[X.509]	ITU-T X.509 PKI: Information technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks	International Telecommunication Union – Telecommunication Standardization Sector of ITU (ITU-T) https://www.itu.int/rec/T-REC-X.509
[XML-DS]	XML Signature Syntax and Processing	The World Wide Web Consortium (W3C) https://www.w3.org/TR/xmldsig-core1/

A Appendix A - DigSig examples

A.1 Signing of a GS1 element string

This example shows how a GS1 element string is signed and verified.

- The signing entity is expressed as a Domain Authority ID “daid” which uses the example Party GLN from ISO/IEC 20248: 9506000151540.
- The example element string is used: `(01)09506000151519(21)12345678p901(10)1234567p(17)221105`

DigSig Data Description (DDD)

```
{"digsiginfo":
  {"specificationversion": "ISO/IEC 20248:2022",
   "daid": "GS1 9506000151540",
   "cid":10000,
   "dauri": "https://dalgiardino.com",
   "verificationuri": ["https://ds.idetrust.io","https://dalgiardino.com"],
   "revocationuri": ["https://ds.idetrust.io","https://dalgiardino.com"],
   "preverify": {"en": "Demonstration Validation Label - NOT FOR USE",
                 "fr": "Étiquette de validation de démonstration - NE PAS UTILISER",
                 "de": "Validierungslabel zur Demonstration - NICHT ZUR VERWENDUNG"},
   "acceptverify": {"en": "This Dal Giardino item is genuine.",
                    "fr": "Cet article Dal Giardino est authentique.",
                    "de": "Dieser Dal Giardino-Artikel ist echt."},
   "rejectverify": {"en": "Please contact info@dalgiardino.com.",
                    "fr": "Veuillez contacter info@dalgiardino.com.",
                    "de": "Bitte kontaktieren Sie info@dalgiardino.com."},
   "postverify": {"en": "Enjoy the product - Please care for the environment.",
                  "fr": "Profitez du produit – Veuillez prendre soin de l'environnement.",
                  "de": "Genießen Sie das Produkt – Bitte achten Sie auf die Umwelt."}},
 "datafields":
  [ {"fieldid": "specificationversion"},
    {"fieldid": "dauri"},
    {"fieldid": "daid"},
    {"fieldid": "cid"},
    {"fieldid": "signature"},
    {"fieldid": "timestamp", "range": "[2023-01-01..+5]"},
    {"fieldid": "GS1_element_string",
     "type": "string","range":"[[:print:]]",
     "fieldname": {"en": "GS1 element string",
                   "fr": "Chaîne d'éléments GS1",
                   "de": "GS1 Datenelement"},
     "description": {"en": "Please provide the GS1 element string.",
                     "fr": "Veuillez fournir la chaîne de l'élément GS1.",
                     "de": "Bitte geben Sie die GS1 Datenelement an."},
     "pragma": {"readmethod": {"methodname": ["ISO/IEC 15417"],
                               "methodsnipencoding": "TEXT"}}}]}
```

Input data to generate a DigSig

```
{"daid": "GS1 9506000151540","cid":10000,
 "GS1_element_string": "(01)09506000151519(21)12345678p901(10)1234567p(17)221105"}
```

DigSig generation result

```
{"CIDSnip": "_opUnDI_QnEAVLWpKddIfepGsH-3rd2jU326-bCozHAdDFSd-p2-K-FM",
 "DataSnips": {"GS1_element_string": "(01)09506000151519(21)12345678p901(10)1234567p(17)221105"}}
```

The signing app now takes the CIDSnip, which is the DigSig, and adds it to the element string:

`(01)09506000151519(21)12345678p901(10)1234567p(17)221105`**`(8030)_opUnDI_QnEAVLWpKddIfepGsH-3rd2jU326-bCozHAdDFSd-p2-K-FM`**

Verification

The verification app reads the GS1 element string, as above. It removes the DigSig AI (8030) from the element string:

`(01)09506000151519(21)12345678p901(10)1234567p(17)221105`

`(8030)_opUnDI_QnEAVLWpKddIfepGsH-3rd2jU326-bCozHAdDFSd-p2-K-FM`

It then creates the verification request:

```
{"20248": {"CIDSnip": "_opUnDI_QnEAVLWpKddIfepGsH-3rd2jU326-bCozHAdDFSd-p2-K-FM",
           "DataSnips": ["GS1_element_string": "(01)09506000151519(21)12345678p901(10)1234567p(17)221105"}]}
```

The result is:

```
{"20248": {"ResponseCode": {"Code": 0, "Desc": "DigSig verification accepted"},
 "DDDdataTagged":
  {"specificationversion": "ISO/IEC 20248:2022",
   "daid":"GS1-9506000151540","cid":10000,"dauri":"https://dalgiardino.com",
   "signature":":054B:5A92:9D74:87DE:A46B:07FB:7ADD:DA35:37DB:AF9B:0A8C:C701:D0C5:49DF:A9DB:E2BE",
   "timestamp":"2023-10-12",
   "GS1_element_string":"(01)09506000151519(21)12345678p901(10)1234567p(17)221105"},
   "Languages":["en","fr","de"]}}
```

The language indicator informs the verification app of the languages available in the DigSig Data Description (DDD) of the DigSig. The languages are applicable for `DDDdataDisplay`. The default verification return is `DDDdataTagged`, which is used in machine-to-machine applications.

The `DDDdataDisplay` for the above example, in French is:

```
{"DDDdataDisplay":
  {"digsiginfo":
    {"specificationversion": "ISO/IEC 20248:2022",
     "dauri": "https://dalgiardino.com",
     "daid": "GS1 9506000151540",
     "cid": 10000,
     "verificationuri": ["https://ds.idetrust.io","https://dalgiardino.com"],
     "revocationuri": ["https://ds.idetrust.io","https://dalgiardino.com"],
     "preverify": "Étiquette de validation de démonstration - NE PAS UTILISER",
     "acceptverify": "Cet article Dal Giardino est authentique.",
     "rejectverify": "Veuillez contacter info@dalgiardino.com.",
     "postverify": "Profitez du produit - Veuillez prendre soin de l'environnement."},
   "datafields":
    [ {"fieldid": "specificationversion",
       "displayvalue": "ISO/IEC 20248:2022"},
      {"fieldid": "dauri",
       "displayvalue": "https://dalgiardino.com"},
      {"fieldid": "daid",
       "displayvalue": "GS1 9506000151540"},
      {"fieldid": "cid",
       "displayvalue": "10000"},
      {"fieldid": "signature",
       "fieldname": "signature",
       "displayvalue": ":054b5a929d7487dea46b07fb7addda3537dbaf9b0a8cc701d0c549dfa9dbe2be"},
      {"fieldid": "timestamp",
       "fieldname": "timestamp",
       "displayvalue": "2023-10-12"},
      {"fieldid": "GS1_element_string",
       "fieldname": "Chaîne d'éléments GS1",
       "displayvalue": "(01)09506000151519(21)12345678p901(10)1234567p(17)221105",
       "description": "Veuillez fournir la chaîne de l'élément GS1."}]}}
```

A.2 Signing of a GS1 element string including the UV ink mark

The UV ink mark or any other physical security mark is added to the DigSig by adding a field for it to the DigSig Data Description (DDD), as shown below in blue.

DigSig Data Description

```
{"digsiginfo":
  {"specificationversion": "ISO/IEC 20248:2022",
   "dauri": "https://dalgiardino.com",
   "daid": "GS1 9506000151540",
   "cid": 10001,
   "verificationuri": ["https://ds.idetrust.io","https://dalgiardino.com"],
   "revocationuri": ["https://ds.idetrust.io","https://dalgiardino.com"],
   "preverify": {"en": "Demonstration Validation Label - NOT FOR USE",
                 "fr": "Étiquette de validation de démonstration - NE PAS UTILISER",
                 "de": "Validierungslabel zur Demonstration - NICHT ZUR VERWENDUNG"},
   "acceptverify": {"en": "This Dal Giardino item is genuine.",
                    "fr": "Cet article Dal Giardino est authentique.",
                    "de": "Dieser Dal Giardino-Artikel ist echt."},
   "rejectverify": {"en": "Please contact info@dalgiardino.com.",
                    "fr": "Veuillez contacter info@dalgiardino.com.",
                    "de": "Bitte kontaktieren Sie info@dalgiardino.com."},
   "postverify": {"en": "Enjoy the product - Please care for the environment.",
                  "fr": "Profitez du produit – Veuillez prendre soin de l'environnement.",
                  "de": "Genießen Sie das Produkt – Bitte achten Sie auf die Umwelt."}},
 "datafields":
  [ {"fieldid": "specificationversion"},
    {"fieldid": "dauri"},
    {"fieldid": "daid"},
    {"fieldid": "cid"},
    {"fieldid": "signature"},
    {"fieldid": "timestamp", "range": "[2023-01-01..+5]"},
    {"fieldid": "GS1_element_string",
     "type": "string", "range": "[[:print:]]",
     "fieldname": {"en": "GS1 element string",
                   "fr": "Chaîne d'éléments GS1",
                   "de": "GS1 Datenelement"},
     "description": {"en": "Please provide the GS1 element string.",
                     "fr": "Veuillez fournir la chaîne de l'élément GS1.",
                     "de": "Bitte geben Sie die GS1 Datenelement an."},
     "pragma": {"readmethod": {"methodname": ["ISO/IEC 15417"],
                               "methodsnipencoding": "TEXT"}}},
    {"fieldid": "UVink",
     "fieldname":
      {"en": "UV ink mark",
       "fr": "Marque d'encre UV",
       "de": "Markierung mit UV-Tinte"},
     "description":
      {"en": "Shine a UV light on the security mark. Enter the text which becomes visible.",
       "fr": "Faites briller une lumière UV sur la marque de sécurité. Saisissez le texte qui devient visible.",
       "de": "Beleuchten Sie das Sicherheitszeichen mit UV-Licht. Geben Sie den Text ein, der sichtbar wird."},
     "type": "string",
     "pragma": {"entertext": null}}]}
```

The `entertext` pragma is used in the UV ink case, since the operator must read the security mark and enter it. The pragma is an instruction for the verification app.

Input data to generate a DigSig

```
{"daid": "GS1 9506000151540", "cid":10001,
 "GS1_element_string": "(01)09506000151519(21)12345678p901(10)1234567p(17)221105",
 "UVink": "UV389JKG4M2"}
```

The UV ink code must be read, likely in an automated way, during or before the time of signing.

DigSig generation result

```
{"CIDSnip": "dshvklhakdhvuiew89lksajd8",
 "DataSnips": {"GS1_element_string": "(01)09506000151519(21)12345678p901(10)1234567p(17)221105",
               "UVink":"UV389JKG4M2"}}
```

The signing app now takes the CIDSnip, which is the DigSig, and adds it to the element string. The UV ink code is included in the signature, so that is not added to the element string.

`(01)09506000151519(21)12345678p901(10)1234567p(17)221105`**`(8030)dshvklhakdhvuiew89lksajd8`**

Verification

The verification app reads the GS1 element string, as above. It removes the DigSig AI (8030) from the element string:

`(01)09506000151519(21)12345678p901(10)1234567p(17)221105`**`(8030)dshvklhakdhvuiew89lksajd8`**

It then creates the verification request:

```
{"20248": {"CIDSnip": "dshvklhakdhvuiew89lksajd8",
           "DataSnips": {"GS1_element_string": "(01)09506000151519(21)12345678p901(10)1234567p(17)221105"}}}
```

The result is:

```
{"20248": {"ResponseCode": {"Code": 19, "Desc": "Snips incomplete"},
 "DDDdataTagged":
  {"specificationversion": "ISO/IEC 20248:2022",
   "dauri": "https://dalgiardino.com",
   "daid": "GS1-9506000151540",
   "cid": 10001,
   "signature": "kdlsjfoirguogre",
   "timestamp": "2024-04-23T17:12:25",
   "GS1_element_string": "(01)09506000151519(21)12345678p901(10)1234567p(17)221105"},
   "MissingDataSnips": ["UVink"],
   "MissingDataSnipPragma":
    [ {"fieldid": "UVink",
       "pragma": {"entertext": null},
       "description": "Shine a UV light on the security mark. Enter the text which become visible."}],
   "Languages":["en","fr","de"]}}
```

The verification now has the information to provide to an operator to obtain the UV ink code. The completed verification request is then submitted:

```
{"20248": {"CIDSnip": "dshvklhakdhvuiew89lksajd8",
           "DataSnips":{"GS1_element_string": "(01)09506000151519(21)12345678p901(10)1234567p(17)221105",
                        "UVink": "UV389JKG4M2"}}}
```

A.3 Signing of an EPC binary string

This example shows an SGTIN-198 EPC encoding on an EPC/RFID tag as illustrated in Figure 3-4 Note the specification for the various memory banks.

DigSig Data Description

```
{"digsiginfo":
  {"cid":  10002,
   "specificationversion": "ISO/IEC 20248:2022",
   "daid": "GS1 9506000151540",
   "dauri": "https://dalgiardino.com",
   "verificationuri": ["https://dalgiardino.com"],
   "revocationuri": ["https://dalgiardino.com"],
   "preverify": {"en": "Demonstration Validation Label - NOT FOR USE",
                 "fr": "Étiquette de validation de démonstration - NE PAS UTILISER",
                 "de": "Validierungslabel zur Demonstration - NICHT ZUR VERWENDUNG"},
   "acceptverify": {"en": "This Dal Giardino item is genuine.",
                    "fr": "Cet article Dal Giardino est authentique.",
                    "de": "Dieser Dal Giardino-Artikel ist orginal."},
   "rejectverify": {"en": "Please contact info@dalgiardino.com.",
                    "fr": "Veuillez contacter info@dalgiardino.com.",
                    "de": "Bitte kontaktieren Sie info@dalgiardino.com."},
   "postverify": {"en": "Enjoy the product - Please care for the environment.",
                  "fr": "Profitez du produit – Veuillez prendre soin de l'environnement.",
                  "de": "Genießen Sie das Produkt – Bitte achten Sie auf die Umwelt."}},
"datafields":
  [ {"fieldid": "specificationversion"},
    {"fieldid": "dauri"},
    {"fieldid": "daid"},
{"fieldid": "cid"},
{"fieldid": "signature"},
{"fieldid": "timestamp","range":"[2023-01-01..+5]"},
{"fieldid": "GS1_EPC_SGTIN-198",
 "type": "bstring", "binaryformat": "{198}",
 "fieldname": {"en": "GS1 EPC SGTIN-198 Code",
              "fr": "Code GS1 SGTIN-198 EPC",
              "de": "GS1 EPC-SGTIN-198-Code"},
 "description": {"en": "Please provide the EPC.",
                "fr": "Veuillez fournir le EPC.",
                "de": "Bitte geben Sie den EPC an."},
 "pragma": {"readmethod": {"methodname":["GS1 EPC Gen2", "ISO/IEC 18000-63", "INVENTORY"],
                          "methodmemory":[1],
                          "methodoffset":0,
                          "methodlength":198}}},
{"fieldid": "UVink",
 "fieldname":
  {"en": "UV ink mark",
   "fr": "Marque d'encre UV",
   "de": "Markierung mit UV-Tinte"},
 "description":
  {"en": "Shine a UV light on the security mark. Enter the text which become visible.",
   "fr": "Faites briller une lumière UV sur la marque de sécurité. Saisissez le texte qui devient visible.",
   "de": "Beleuchten Sie das Sicherheitszeichen mit UV-Licht. Geben Sie den Text ein, der sichtbar wird."},
 "type": "string",
 "pragma": {"entertext": null}},
{"fieldid": "TID",
 "type": "bstring", "binaryformat": "{96}",
 "fieldname": {"en": "EPC/RFID chip identifier (TID)",
              "fr": "Identifiant de la puce EPC/RFID (TID)",
              "de": "EPC/RFID-Chip-Kennung (TID)"},
 "description": {"en": "Please provide the TID.",
                "fr": "Veuillez fournir le TID.",
                "de": "Bitte geben Sie den TID an."},
 "pragma": {"readmethod": {"methodname":["GS1 EPC Gen2", "ISO/IEC 18000-63", "ACCESS"],
                          "methodmemory":[2],
                          "methodoffset":0,
                          "methodlength":96}}}]}
```

Input data to generate a DigSig

The TID must first be read from the EPC/RFID tag, and the UV ink code from the packaging.

```
{"daid":"GS1 9506000151540","cid":10002,
 "GS1_EPC_SGTIN-198": ":3606:C440:905D:6058:B266:D1AB:66EE:3839:60C4:0000:0000:0000:0000",
 "UVink": "UV389JKG4M2",
 "TID": ":E2C0:6892:2000:B502:1E25:4DF0"}
```

Note that, in the example above, the `"GS1 EPC SGTIN-198"` *field, encoded in hex with a colon between each 16-bit "word", as* `":3606:C440:905D:6058:B266:D1AB:66EE:3839:60C4:0000:0000:0000:0000",` *corresponds to an EPC Hex encoding (from bit 20h of Memory Bank 01) of* `3606C440905D6058B266D1AB66EE383960C40000000000000000.` *The corresponding EPC Tag URI is* `urn:epc:tag:sgtin-198:0.95060001515.01.12345678p901.` *Similarly, the* `"TID"` *field, encoded in hex with a colon between each 16-bit "word", as* `":E2C0:6892:2000:B502:1E25:4DF0",` *corresponds to a TID Hex encoding (from bit 00h of TID Memory) of* `E2C068922000B5021E254DF0.`

DigSig generation result

The Snips are generated to be shown in hex since it need to be programmed into an EPC/RFID tag. The DigSig specification uses the leading ":" to distinguish between URI-safe Base64 and hex.

```
{"CIDSnip": ":FE8A:549C:323F:4271:264E:E9D0:53F4:CCBC:72C0:D973:3406:89A1:8FC8:AF9A:179C:293D:8033:7F95:A0D4:F1F4:B2F4",
 "DataSnips": {"GS1_EPC_SGTIN-198": ":3606:C440:905D:6058:B266:D1AB:66EE:3839:60C4:0000:0000:0000:00",
               "TID": ":E2C0:6892:2000:B502:1E25:4DF0",
               "UVink": "UV389JKG4M2"}}
```

The CIDSnip must be prepended with the DSFID 0x11 and then padded to a 32-bit word boundary. It is then written to MB11 (User Memory). If the tag is conformant to the EPC® Radio-Frequency Identity Generation-2 UHF RFID Standard [UHFG2], and the RFID chip is capable of the RUM/UWC (Read User Memory/User Memory Word Count) feature, then set the UWC to the length of the prepended and padded CIDSnip. In this way the reader, when the RUM bit is set, can read the CIDSnip without knowing the length of it.

The same applies for the EPC, but in this case the Len bits are set in the PC-word (Protocol Control).

Reading the EPC with a DigSig from an EPC/RFID tag

In the case of a [UHFG2] conformant tag:

1. The EPC/RFID tag is INVENTORIED and the application stores the EPC.
2. The reader notes that the RUM bit is set and then reads the User Memory. The DSFID indicates this is a DigSig. The application then stores this DigSig, omitting the DSFID, as the CIDSnip.
3. The reader should be configured so that when it sees a DigSig in User Memory, it then also reads the TID.
4. The CIDSnip with the DataSnips EPC and TID are then passed to the DigSig decoder verifier. The DigSig decoder verifier will notify the calling application that the UVink must be supplied, see the entertext pragma of the UVink field.
5. The UVink code can be directly displayed by removing the entertext pragma. In this case the operator must check that the code is correct. DigSig Verifier Apps have OCR (Optical Character Recognition) capability, which can be used to reduce the operator error. The UVink part can be automated, for example on a production line, by adding a camera to the RFID read-point to read the UVink code.
6. The application then requests the operator to enter the UV ink code. All the DigSig data elements (CIDSnip and DataSnips) are now complete.
7. The DigSig is resubmitted for verification, of which the result is display by the application to the operator.

A.4 DigSigs, an X.509 application

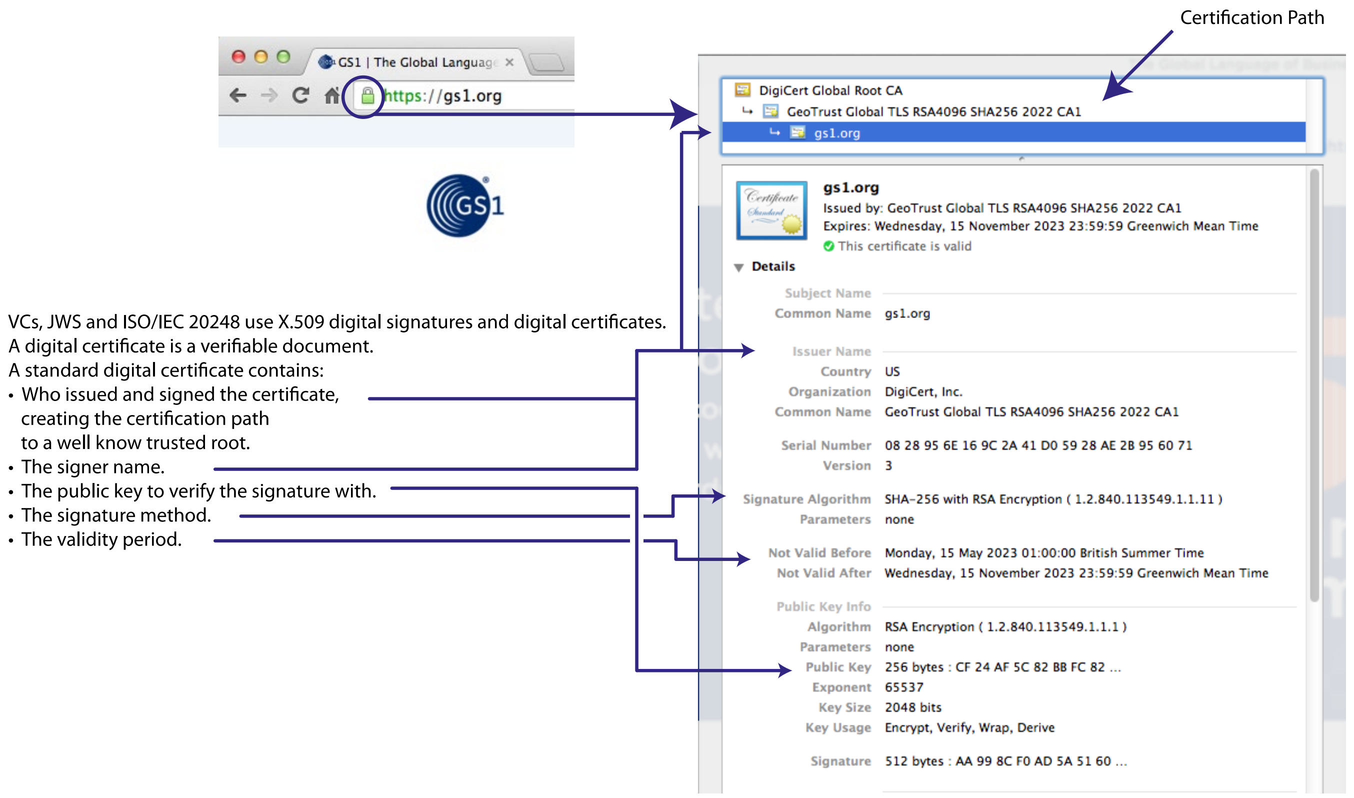
ITU-T X.509 digital certificates [X.509] play an important role in digital signatures. An ITU-T X.509 digital certificate [X.509] is the verifiable document that carries all verification and data structure information of a specific group of digital signatures. Figure A-1 provides a closer look of the different components of a standard webserver certificate.

When a Web browser accesses a secure URL using “https://” instead of “http://” it displays a small padlock icon within or near the URL bar. By clicking on the padlock icon, details of the website’s verified digital certificate, such as the certificate information and certificate path can be accessed. Figure A-1 shows what may be seen, with an example of the web server certificate for the public GS1 website as it was, during development of this guideline in 2023.

By clicking on any of the certificates in the path, the corresponding certification information can be shown for certificates higher up the certificate trust path or certificate hierarchy. The top-level certificate should be a well-known public trusted certificate and applies to ISO/IEC 20248 DigSigs [DigSig], JSON Web Signatures [JWS] and Verifiable Credentials [VC].

ISO/IEC 20248 [DigSig] also known as the DigSig method, is a X.509 application. A DigSig Certificate is an ITU-T X.509 standard [X509] digital certificate. ISO/IEC 20248 [DigSig] specifies how a DigSig certificate is used, and how a data structure is specified, which is collectively known as the DigSig Data Description (DDD).

To decode and verify a DigSig, only the associated DigSig certificate is needed, as it includes the DDD. ISO/IEC 20248 [DigSig] specifies the interface for trusted repositories to host and provide the DigSig Certificate, information on the status of the DigSig Certificate, and provide revocation information on individual DigSigs. It is important to note that DigSig Certificates should be hosted at more than one trusted repository, including on the Domain Authority website. This counters impersonation and denial-of-service (DOS) attacks. GS1 may host such a repository. X.509 digital certificates are public (not secret) by design and are independently verifiable. As such, it supports open and free verification.



VCs, JWS and ISO/IEC 20248 use X.509 digital signatures and digital certificates.
A digital certificate is a verifiable document.
A standard digital certificate contains:
- Who issued and signed the certificate, creating the certification path to a well know trusted root.
- The signer name.
- The public key to verify the signature with.
- The signature method.
- The validity period.

```
DigiCert Global Root CA
 ↳ GeoTrust Global TLS RSA4096 SHA256 2022 CA1
    ↳ gs1.org

gs1.org
Issued by: GeoTrust Global TLS RSA4096 SHA256 2022 CA1
Expires: Wednesday, 15 November 2023 23:59:59 Greenwich Mean Time
This certificate is valid

Details
       Subject Name
        Common Name  gs1.org

        Issuer Name
            Country  US
       Organization  DigiCert, Inc.
        Common Name  GeoTrust Global TLS RSA4096 SHA256 2022 CA1

      Serial Number  08 28 95 6E 16 9C 2A 41 D0 59 28 AE 2B 95 60 71
            Version  3

Signature Algorithm  SHA-256 with RSA Encryption ( 1.2.840.113549.1.1.11 )
         Parameters  none

   Not Valid Before  Monday, 15 May 2023 01:00:00 British Summer Time
    Not Valid After  Wednesday, 15 November 2023 23:59:59 Greenwich Mean Time

    Public Key Info
          Algorithm  RSA Encryption ( 1.2.840.113549.1.1.1 )
         Parameters  none
         Public Key  256 bytes : CF 24 AF 5C 82 BB FC 82 ...
           Exponent  65537
           Key Size  2048 bits
          Key Usage  Encrypt, Verify, Wrap, Derive

          Signature  512 bytes : AA 99 8C F0 AD 5A 51 60 ...
```

Figure A-1 X.509 server certificate content

A DigSig certificate can consist of several hundreds of characters, which means that it cannot be contained in the associated DigSig that is encoded in a data carrier. Instead, it uses a reference model, which may be summarised as:

- The signer of a DigSig is called the Domain Authority (with "domain" in the context of a domain name service or DNS) and is identified using the Domain Authority Identifier (DAID).
  - As a Domain Authority must be a legal entity, this party can be identified with Party Global Location Number (PGLN), which forms part of a DAID.
- A Domain Authority issues a certificate identifier (CID) for each DigSig certificate it publishes.
- The DAID and CID are the first two data elements of a DigSig.

- The combination of a DAID, CID and DAURI references the DigSig Certificate.
  - The Domain Authority URI (DAURI) is contained in the DigSig Cert and forms part of the DigSig signature. This URI provides the verifier, which can be machine or human, with a means to access the official or authoritative website of the Domain Authority, to determine the level of trust they are prepared to assign to the DigSig. Typically, an official website must exist, and have information indicating that it is current to be considered authoritative. It should also contain legal locations which should not be in conflict with the details of the PGLN.

The DigSig certificate information and the DDD is specified in a manner to detect tampering or misrepresentation of any part of the certificate, the DDD and the DigSigs.

An example of the DigSig Data Description (DDD) for the fictitious Dal Giardino Truffle Tapenade is shown, with the different components numbered to correspond to the list below which explains each of the components contained in the DDD example:

1. Current version information of the DigSig method, which can be updated when the referenced ISO/IEC 20248 [DigSig] specification version changes. Note the 2022 version is expected to remain stable for some time.
2. DigSig signer information, which includes the DAID, CID and DAURI.
3. Verification and next steps information for the operator, provided in multiple languages.
4. Digital signature and timestamp (note the timestamp range is the signing period in years, which must be shorter than the certificate validity period).
5. Instructions for the operator to explain how to locate and read the GS1 barcode.
6. Instructions for the operator to explain how to locate and read an authentication code within a security marking, which in this example is a UV ink mark with instructions to illuminate the security marking with a UV light to find the code and type it in to complete verification of the DigSig.

The DDD has two parts: the "`digsiginfo`" part which describes the credentials and use of the DDD, and the "`datafields`" part which provides the field definitions. The compulsory fields "`specificationversion`", "`dauri`", "`daid`", "`cid`", "`signature`" and "`timestamp`", are cross referenced between the DigSig Cert, DDD and DigSig to detect tampering.

```
{"digsiginfo":
1  {"specificationversion": "ISO/IEC 20248:2022",
2   "daid": "GS1 9506000151540", "cid": 10001, "dauri": "https://dalgiardino.com",
3   "verificationuri": [https://ds.idetrust.io/https://ds.idetrust.io, "https://dalgiardino.com"],
3   "revocationuri": [https://ds.idetrust.io/https://ds.idetrust.io, "https://dalgiardino.com"],
3   "preverify": {"en": "Demonstration Validation Label - NOT FOR USE",
3                 "fr": "Étiquette de validation de démonstration - NE PAS UTILISER",
3                 "de": "Validierungslabel zur Demonstration - NICHT ZUR VERWENDUNG"},
3   "acceptverify": {"en": "This Dal Giardino item is genuine.",
3                    "fr": "Cet article Dal Giardino est authentique.",
3                    "de": "Dieser Dal Giardino-Artikel ist echt."},
3   "rejectverify": {"en": "Please contact info@dalgiardino.com.",
3                   "fr": "Veuillez contacter info@dalgiardino.com.",
3                   "de": "Bitte kontaktieren Sie info@dalgiardino.com."},
3   "postverify": {"en": "Enjoy the product - Please care for the environment.",
3                 "fr": "Profitez du produit – Veuillez prendre soin de l'environnement.",
3                 "de": "Genießen Sie das Produkt – Bitte achten Sie auf die Umwelt."}},
 "datafields":
1  [ {"fieldid": "specificationversion"},
2    {"fieldid": "dauri"},
2    {"fieldid": "daid"},
2    {"fieldid": "cid"},
4    {"fieldid": "signature"},
4    {"fieldid": "timestamp", "range": "[2023-01-01..+5]"},
5    {"fieldid": "GS1_element_string",
5     "type":"string","range": "[[:print:]]",
5     "fieldname": {"en": "GS1 element string",
5                   "fr": "Chaîne d'éléments GS1",
5                   "de": "GS1 Datenelement"},
5     "description": {"en": "Please provide the GS1 element string.",
5                     "fr": "Veuillez fournir la chaîne de l'élément GS1.",
5                     "de": "Bitte geben Sie die GS1 Datenelement an."},
5     "pragma": {"readmethod": {"methodname": ["ISO/IEC 15417"], "methodsnipencoding": "TEXT"}}},
6    {"fieldid": "UVink",
6     "fieldname": {"en": "UV ink mark",
6                   "fr": "Marque d'encre UV",
6                   "de": "Markierung mit UV-Tinte"},
6     "description": {"en": "Shine a UV light on the security mark. Enter the text which become visible.",

6                     "fr": "Faites briller une lumière UV sur la marque de sécurité. Saisissez le
6                            texte qui devient visible.",
6                     "de": "Beleuchten Sie das Sicherheitszeichen mit UV-Licht. Geben Sie den Text
6                            ein, der sichtbar wird."},
6     "type": "string",
6     "pragma": {"entertext": null}}]}
```

B Appendix B - JSON Web Signature example

B.1 GS1 Web Vocabulary properties for GS1 Digital Signatures

A JSON Web Signature, as defined by RFC 7515 [JWS], includes a data payload in JavaScript Object Notation (JSON) format [JSON], which can also be JSON-LD format (JSON for Linked Data) [JSON-LD]. This means that the data payload can be a block of master data about an individual entity instance, such as a serialised trade item or a logistics unit, expressed using terms from the GS1 Web Vocabulary [WebVoc] and formatted in JSON-LD.

Schema.org [Schema] is a widely used Web vocabulary for describing all kinds of entities such as places, people, organisations, events and products, in a machine-interpretable way, as Linked Data. The GS1 Web Vocabulary [WebVoc] enables all entities identified by GS1 identification keys and associated attribute data, to be described in greater detail, serving as an extension to schema.org [Schema].

As a result of the work from the GS1 Digital Signatures MSWG, GS1 has expanded the GS1 Web Vocabulary [WebVoc] to add support for describing such authentication markings, using properties defined within the class https://gs1.org/voc/AuthenticityDetails shown below in Table B-1. This means that the GS1 Web Vocabulary [WebVoc] is able to support equivalent functionality for what is natively provided in ISO/IEC 20248 [DigSig] regarding the binding to unique codes in physical security markings.

Table B-1 GS1 Web Vocabulary properties from the gs1:AuthenticityDetails class

Property Compact URI	Expected value	Description
gs1:authenticitySecurityFeatureInstructions	rdf:langString (*)	Provides human-readable instructions about how to locate a physical security marking and read a value from it.
gs1:authenticitySecurityFeatureInstructionsURL	xsd:anyURI	Links to online instructions about how to locate a physical security marking and read a value from it.
gs1:authenticitySecurityFeatureRegularExpression	xsd:string	Links to a regular expression to be used to perform syntax validation (plausibility checking) of a string value read from a physical security marking.
gs1:authenticitySecurityFeatureType	gs1:SecurityMarking (**)	Links to a URI code value indicating a particular type of physical security marking.
gs1:authenticitySecurityFeatureValue	xsd:string	Links to a string value read from a physical security marking.
NOTES: (*) a language-tagged string (**) a URI value from a defined code list - see Table B-2		

Within the GS1 Web vocabulary [WebVoc], the gs1:SecurityMarking code list defines the code values shown in Table B-2 to reference the different types of security markings that can appear on an entity or its packaging.

Table B-2 GS1 Web Vocabulary code list for gs1:SecurityMarking

Code value	Code value description
gs1:SecurityMarking-HOLOGRAM	A holographic image formed using light diffraction and interference of light waves, capable of representing a three-dimensional object or multiple images, depending on the angle of observation.
gs1:SecurityMarking-MICROPRINTING	A hidden pattern that is too small to read easily without means of optical magnification such as a microscope.
gs1:SecurityMarking-UVINK	A hidden pattern or marking that is invisible when observed under normal visible light but which fluoresces when observed using ultra-violet light.
gs1:SecurityMarking-WATERMARK	A conventional watermark in which a pattern within a translucent sheet becomes visible when observed via transmitted light (typically due to reduced opacity of the pattern), whereas the pattern remains hidden when observed via light reflected off the surface.

By making these updates to the GS1 Web Vocabulary [WebVoc], it is now also possible to express details about unique authentication codes in security features that are unique to each product or entity instance, to strongly bind the digitally signed data to a specific individual physical entity in the real world.

This code list can be extended in future to support other kinds of security markings, with new standardised properties added to the gs1:AuthenticityDetails class, if needed through the submission of a new Work Request, as part the Global Standards Management Process (GSMP)

B.2 Constructing a JSON Web Signature

Every JSON Web Signature [JWS] has a data payload formatted using JSON (JavaScript Object Notation) [JSON]. The data payload is what is digitally signed; the digital signature ensures that any tampering of the data payload can be detected and the party who digitally signed the data is accountable for it and cannot deny signing it.

JSON is a simple data format that can express arrays as a comma-separated list of elements enclosed in square brackets, whereas associative arrays or data objects are expressed as a comma-separated set of key:value pairs enclosed within curly brackets.

A simple data payload might contain some basic facts about an individual product instance, for example. This could include details such as the product GTIN, serial number, batch/lot number, product name and description, expiration date (if perishable) and any other details considered to be important or useful for distinguishing between a genuine product instance and a counterfeit.

A GS1 Digital Link URI [DL-URI] can express a product GTIN and a serial number, to uniquely identify an individual product instance. Here is an example: https://dalgiardino.com/01/09506000151519/21/12345678p901

Factual claims about this product instance can be written using JSON-LD, which is JSON for Linked Data, a Linked Data format standardised by W3C [JSON-LD]. The following JSON-LD data provides an example of some factual claims:

```
{
  "@context": {
        "gs1": "https://gs1.org/voc/",
        "id": "@id",
        "a": "@type",
        "lang": "@language",
        "value": "@value",
        "type": "@type",
        "xsd": "http://www.w3.org/2001/XMLSchema#"
      },
  "id": "https://dalgiardino.com/01/09506000151519/21/12345678p901",
  "a": "gs1:Product",
  "gs1:gtin": "09506000151519",
  "gs1:hasSerialNumber": "12345678p901",
  "gs1:productName": {"value": "Finest Truffle Tapenade", "lang": "en"},
  "gs1:brand": {"gs1:brandName": {"value": "Dal Giardino", "lang": "en"}},
  "gs1:hasBatchLotNumber": "1234567p",
  "gs1:expirationDate": {"value": "2022-11-05", "type": "xsd:date"}
}
```

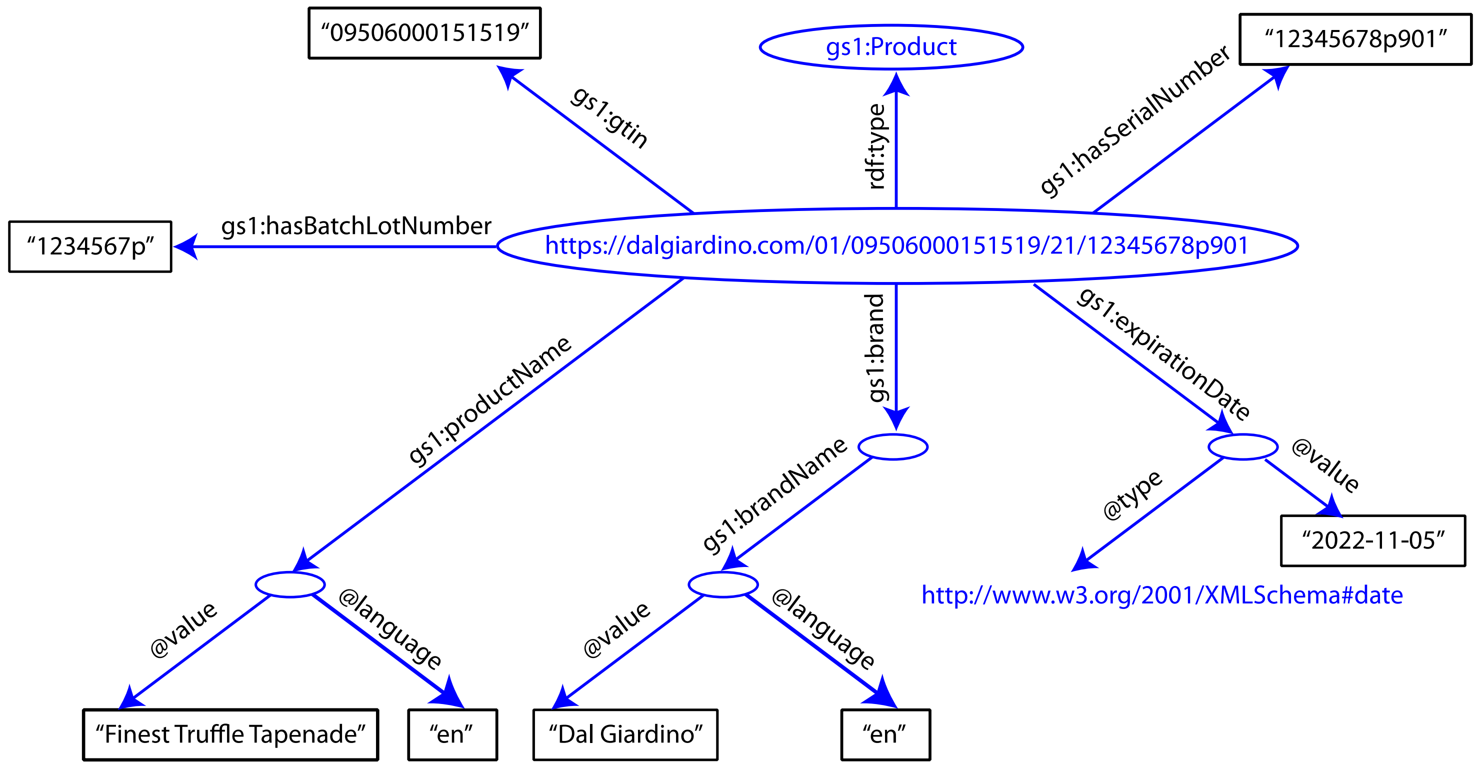
Linked Data uses Web URIs to identify properties, classes, code lists and code values, and to express relationships between entities.

As Web URIs can be quite long, this example uses compact URI expressions (CURIEs) such as gs1:gtin, as a compact way of writing a Web URI such as https://gs1.org/voc/gtin (the property defined in the GS1 Web Vocabulary that indicates the GTIN assigned to a product). CURIEs are defined in the W3C standard at https://www.w3.org/TR/curie/.

The "`@context`" resource explains how to expand such CURIEs - we see that "`gs1`" is a CURIE prefix corresponding to https://gs1.org/voc/, whereas "`xsd`" is a CURIE prefix corresponding to http://www.w3.org/2001/XMLSchema# (which is used for date/time datatypes such as xsd:date).

The "`@context`" resource also provides some mappings to special JSON-LD keywords. In this example, "`id`" is mapped to the special JSON-LD keyword "`@id`" which indicates that the value is considered to be the RDF Subject of various facts contained within that data object (within the same set of curly brackets). In this example, the value of "`id`" / "`@id`" is the GS1 Digital Link URI https://dalgiardino.com/01/09506000151519/21/12345678p901, which is the RDF Subject of various relationships that express values for its product name, brand name, batch/lot number and expiration date, as well as the product GTIN.

The JSON-LD [JSON-LD] example is simply a way of expressing the relationships shown in Table B-1 as structured text.


"09506000151519"
gs1:Product
"12345678p901"
gs1:gtin
rdf:type
gs1:hasSerialNumber
"1234567p"
gs1:hasBatchLotNumber
https://dalgiardino.com/01/09506000151519/21/12345678p901
gs1:productName
gs1:brand
gs1:expirationDate
gs1:brandName
@value
@type
"2022-11-05"
http://www.w3.org/2001/XMLSchema#date
@value
@language
@value
@language
"Finest Truffle Tapenade"
"en"
"Dal Giardino"
"en"

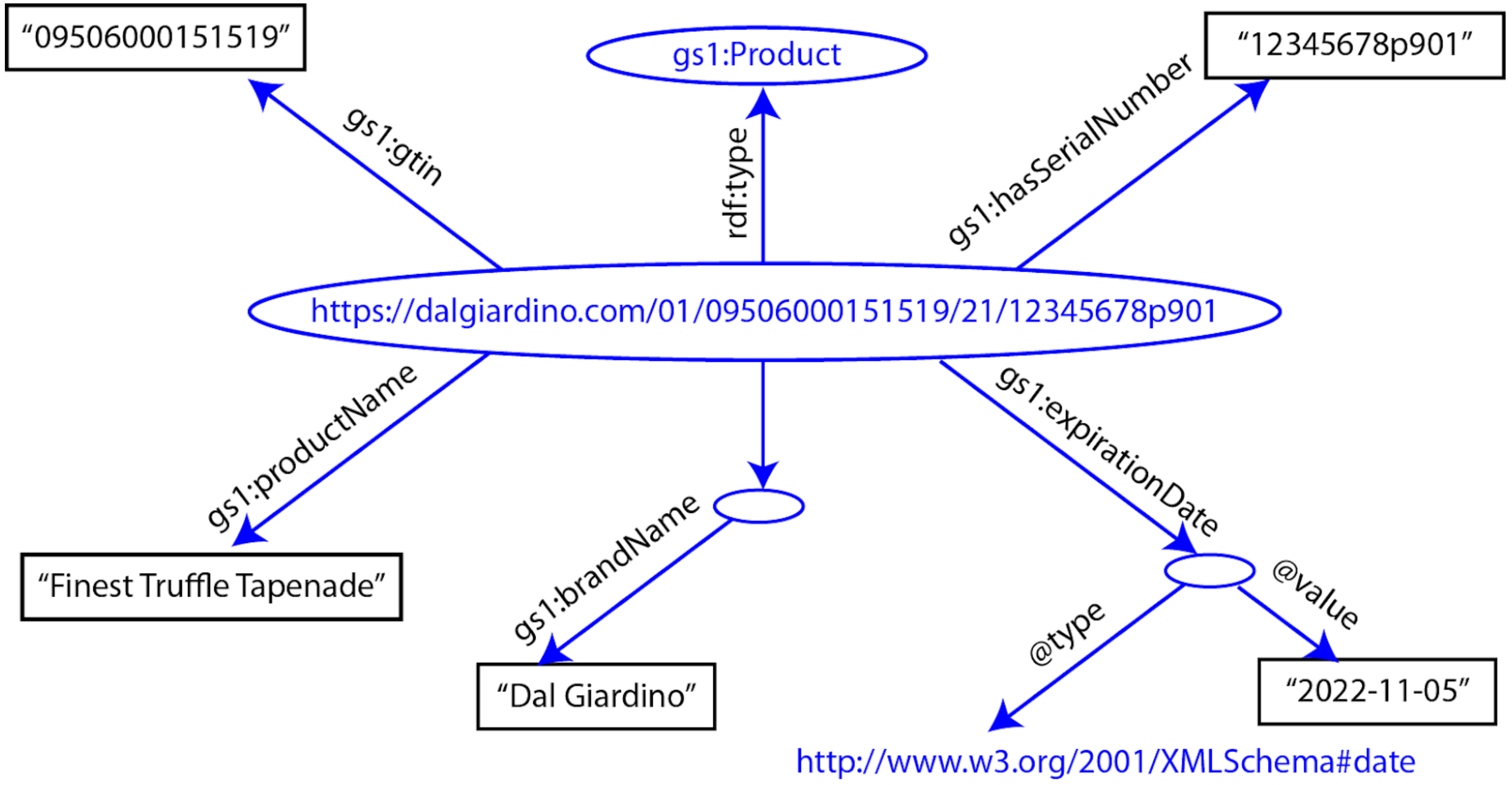


Figure B-1 Expressing relationships between entities, identifiers and other properties about the entities

In our example data, the JSON fields named "`type`" and "`a`" are both mapped to the special JSON-LD keyword "`@type`", which corresponds to the property rdf:type, which is used to indicate the class of things to which the subject belongs. In this example, "`a`": "`gs1:Product`" means that the GS1

Digital Link URI https://dalgiardino.com/01/09506000151519/21/12345678p901 identifies a product and is a member of the class gs1:Product. In our example data, the JSON field named "value" is mapped to a special JSON-LD keyword "@value" which is used within language-tagged strings and also within numeric values of specific data types, such as xsd:date formats. In our JSON-LD example data, the JSON field named "lang" is mapped to a special JSON-LD keyword "@language" which indicates the ISO 639Set 1 two-letter language codes for language-tagged strings, such as "en" for English, "de" for German, etc.

The GS1 Digital Link URI [DL-URI] looks like a URL, typically behaves like a URL and can be encoded in a QR Code and scanned with a smartphone. However, a GS1 Digital Link URI is a structured URI and a multifunctional link that can point to more than one type of online information or service. If an app wants to retrieve a JSON Web Signature [JWS] for that product instance, instead of making a Web request for the GS1 Digital Link URI to retrieve the default web page, the app specifies a link type, by expressing linkType=gs1:jws in the URI query string, so it makes a Web request for https://dalgiardino.com/01/09506000151519/21/12345678p901?linkType=gs1:jws instead of just https://dalgiardino.com/01/09506000151519/21/12345678p901. Resolver infrastructure for GS1 Digital Link, as defined by the GS1 conformant resolver standard [DL-Resolution], understands link types. So if the brand owner has configured a redirection for the gs1:jws link type, the brand owner can configure a resolver to redirect to the online URL of the JSON Web Signature [JWS] for that individual product instance.

However, at this stage, even if that JSON-LD data is digitally signed and formatted as a JSON Web Signature [JWS], there is nothing to prevent any counterfeiter from simply replicating the QR Code containing that GS1 Digital Link URI [DL-URI] and printing that on a label of a counterfeit product. So something else is needed in the data, that includes details about a secret code located on the genuine product, that is unique to that individual product instance and which is not practical or cost-effective for counterfeiters to reproduce at scale on counterfeit products in a way that the codes match the codes generated by the brand owner for each individual product instance.

To include such a secret code in the set of facts about that individual product instance, we can use properties defined within the gs1:AuthenticityDetails class, to indicate which kind of security marking to look for, how to find it on the product, as well as some basic validation checks (using regular expressions) e.g., to check that the correct number of characters has been entered.

When such details about a secret code are included in the JSON-LD data payload, it might now look something like this:

```
{
  "@context": {
    "gs1": "https://gs1.org/voc/",
    "id": "@id",
    "a": "@type",
    "lang": "@language",
    "value": "@value",
    "type": "@type",
    "xsd": "http://www.w3.org/2001/XMLSchema#"
  },
  "id": "https://dalgiardino.com/01/09506000151519/21/12345678p901",
  "a": "gs1:Product",
  "gs1:gtin": "09506000151519",
  "gs1:hasSerialNumber" : "12345678p901",
  "gs1:productName": {"value": "Finest Truffle Tapenade", "lang": "en"},
  "gs1:brand": {"gs1:brandName":{"value": "Dal Giardino", "lang": "en"}},
  "gs1:hasBatchLotNumber": "1234567p",
  "gs1:expirationDate": {"value": "2022-11-05", "type": "xsd:date"},
  "gs1:authenticity": {
    "gs1:authenticitySecurityFeatureType": "gs1:SecurityMarking-UVINK",
    "gs1:authenticitySecurityFeatureInstructions": {"value": "look on side of pack", "lang":"en"},
    "gs1:authenticitySecurityFeatureRegularExpression": "[A-Z0-9]{11}",
    "gs1:authenticitySecurityFeatureValue": "UV389JKGH4M2"
  }
}
```

The property gs1:authenticitySecurityFeatureType indicates what type of security marking contains the secret code, gs1:authenticitySecurityFeatureInstructions provides text instructions about how to find the security marking, gs1:authenticitySecurityFeatureInstructionsURL can link to a web page, online image or video with more detailed instructions (or even machine-readable instructions for use by an automated system for checking authenticity).

The value of the property gs1:authenticitySecurityFeatureValue must be a non-empty string before attempting to verify the signature. If the data payload shows an empty string for this property, it means that a human being or an automated system must first inspect that individual product instance, find the security marking, read the secret code, then insert the value of the secret code back into the data payload instead of the empty string, before attempting to verify the signature.

Although the data payload in our example is JSON-LD, all JSON-LD is valid JSON data, so the procedure for constructing and verifying the JSON Web Signature (JWS) is exactly as detailed in RFC 7515 [JWS].

However, the value of gs1:authenticitySecurityFeatureValue (which matches the secret code in the security marking) must be included in the data payload at the time of signing the data, although the brand owner can afterwards set the value to an empty string within the data payload, in order to force an inspection of the individual product instance, to find the security marking and read the value of the secret code that is unique to that individual product instance.

B.3 Verifying a JSON Web Signature that includes gs1:authenticitySecurityFeatureValue in its payload

At the time of attempting to verify a JSON Web Signature (JWS) [JWS], if the data payload includes the property gs1:authenticitySecurityFeatureValue, this may be a non-empty string. If so, the value should be compared with the value of the secret code appearing in the specified security marking. If the value of the property gs1:authenticitySecurityFeatureValue is an empty string, the specified security marking must first be inspected, the value of its secret code read and set as the value of the corresponding gs1:authenticitySecurityFeatureValue property, before attempting to perform the verification of the digital signature

C Appendix C – About the fictitious ‘Dal Giardino’ brand

Dal Giardino is an entirely fictitious brand developed originally by SGK/Schawk to support the GS1 Mobile Ready Hero Image Guideline. Since then, the ‘Dal Giardino’ brand has been used in a variety of demonstrations related to GS1 Digital Link. The https://dalgiardino.com website is the destination for a number of links associated with example GTINs.

D Appendix D – Example GTINs based on the GS1 Prefix ‘950’

GS1 uses the GS1 Prefix ‘950’ to construct a number of managed GTINs used for examples in its standards documentation to enable functional demonstrations. Each 950-based GTIN is assigned to a specific fictitious product and is managed by GS1 Global Office, in conformance with GS1 GTIN allocation and management rules. In contrast, the GS1 Prefix ‘952’ may be used for demonstrations and examples of the GS1 system but as an unmanaged sandbox, on the understanding that there is no central coordination or management about what such 952-based GTINs correspond to, when used in examples, so the same 952-based GTIN might correspond to different products in different examples. More information on GS1 prefix ‘950’ and ‘952’ for demonstration and example purposes, can be found in the GS1 Style Guide.

E Appendix E – Transitioning to authenticated product data and mass serialisation

Assigning each product instance a combination of a product GTIN and Serial Number enables each product instance to have a globally unique identifier which enables the individual traceability history of each unique product instance, shown as a well-defined path through the supply chain network in Figure E-1. Serialisation also enables this traceability history to be captured as a set of EPCIS event data [EPCIS] that is unique to that product instance.

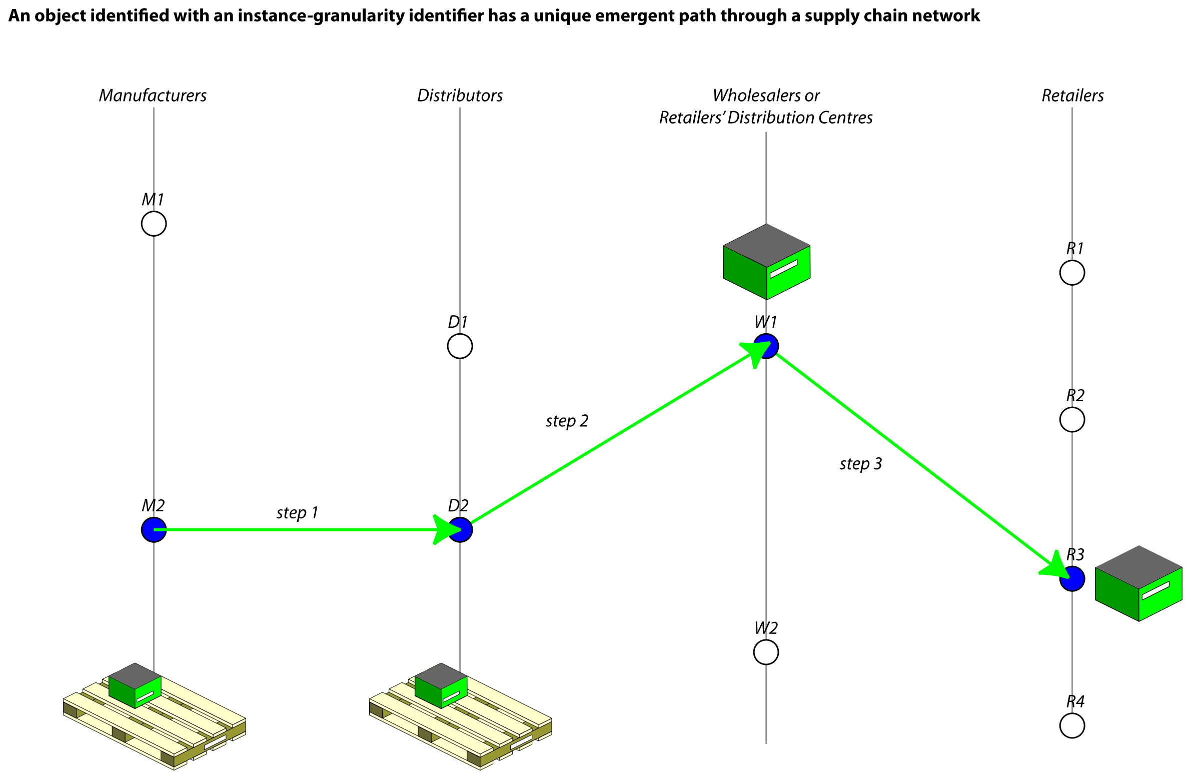

An object identified with an instance-granularity identifier has a unique emergent path through a supply chain network
Manufacturers
Distributors
Wholesalers or Retailers' Distribution Centres
Retailers
M1
M2
D1
D2
W1
W2
R1
R2
R3
R4
step 1
step 2
step 3

Figure E-1 Example of supply chain network path for a genuine serialised product instance

In contrast, when identification is only via a combination of product GTIN and Batch/Lot number, there are multiple indistinguishable objects that all share the same identifier and it is perfectly reasonable for these to appear in multiple locations at the same time. There is no well-defined path through the supply chain network as a result, as shown in Figure E-2.

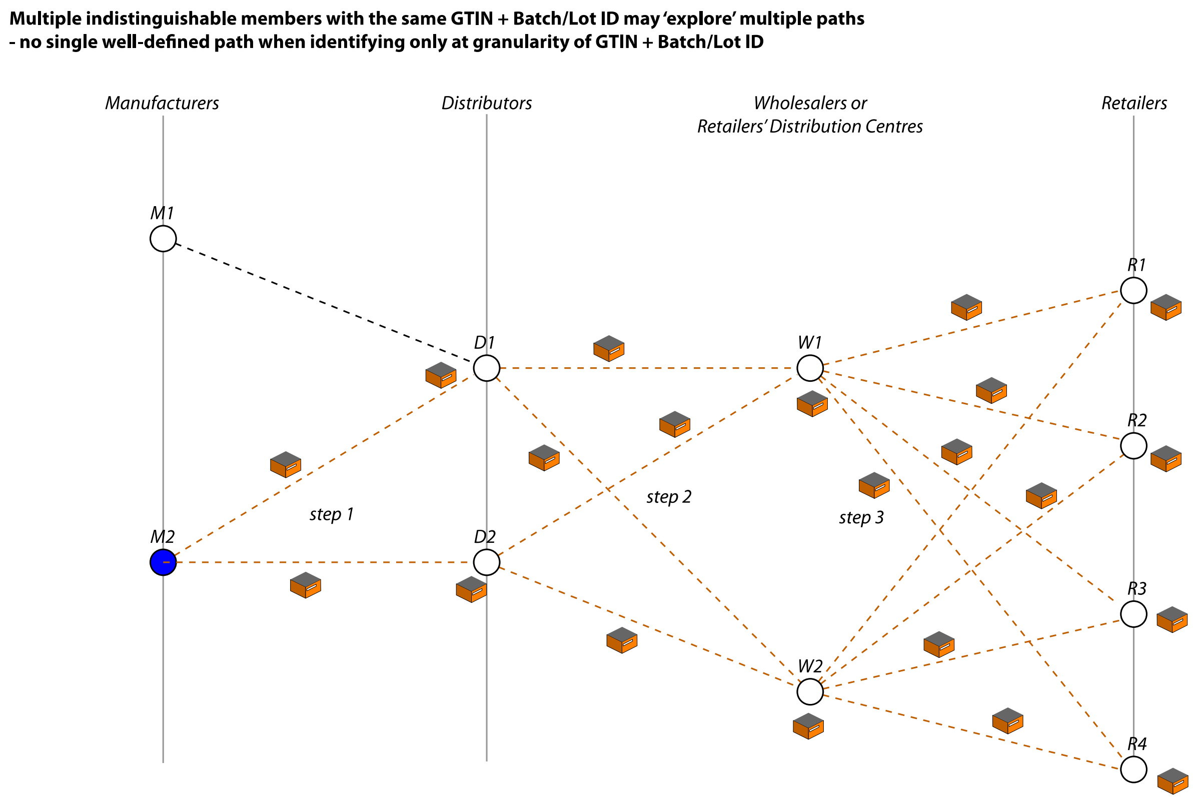
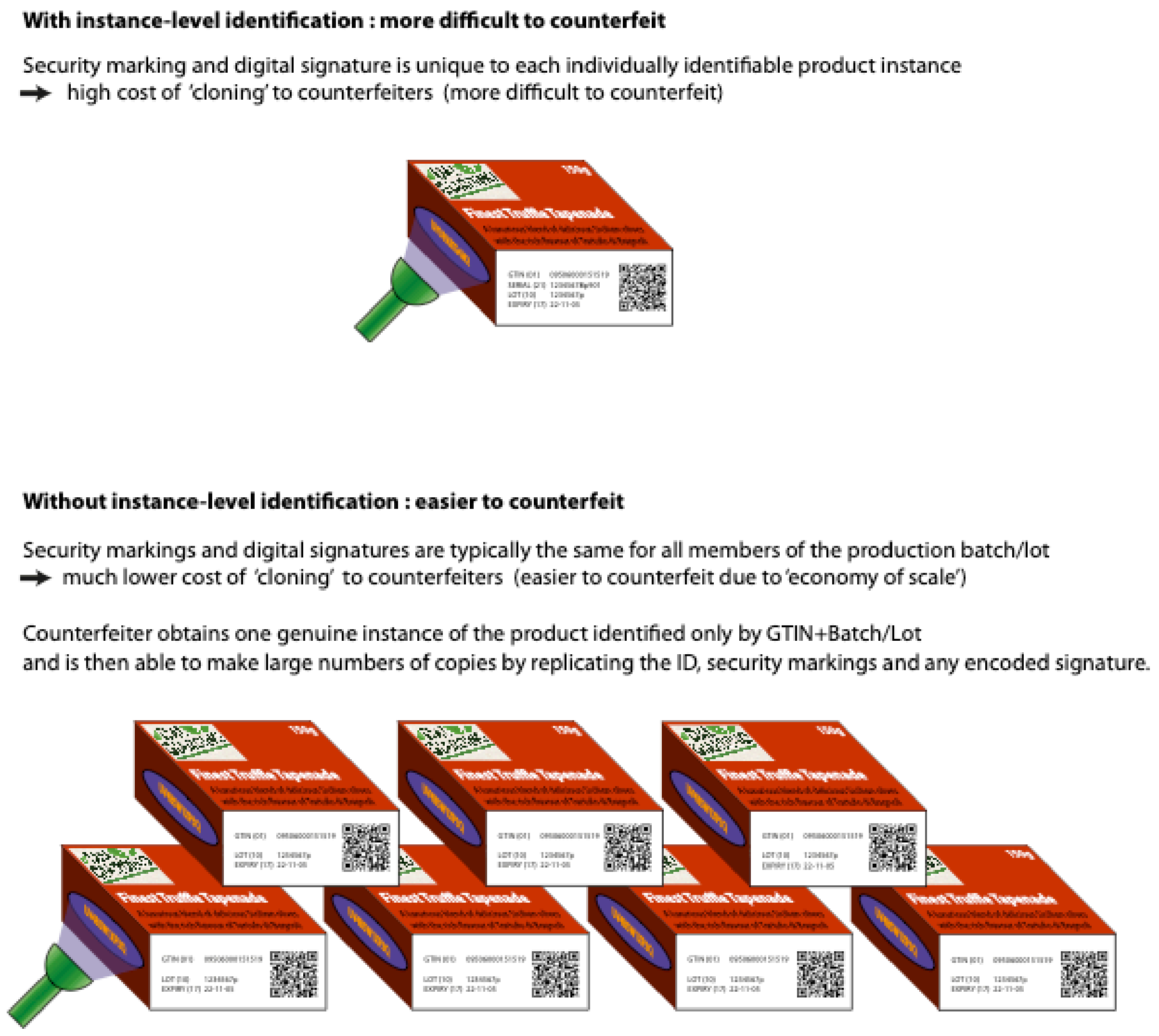

Multiple indistinguishable members with the same GTIN + Batch/Lot ID may 'explore' multiple paths
- no single well-defined path when identifying only at granularity of GTIN + Batch/Lot ID
Manufacturers
Distributors
Wholesalers or
Retailers' Distribution Centres
Retailers
M1
M2
D1
D2
W1
W2
R1
R2
R3
R4
step 1
step 2
step 3

Figure E-2 Example of supply chain network path for a genuine non-serialised product instance

For anti-counterfeit and brand protection applications, having a globally unique identifier for each product instance also means that any associated authentication codes appearing on that product instance in a physical security marking (e.g., inside holograms, watermarks, UV ink marks or using microprinting etc.) can also be unique to that product instance. This provides the highest barrier to counterfeiters since the corresponding secret code for each unique product instance is effectively unpredictable without having actual physical custody of each product instance. So there is no 'economy of scale' for counterfeiters to exploit by having physical custody of one product instance; they will not be able to predict what the corresponding authentication codes should be for other instances of the same product GTIN.



With instance-level identification : more difficult to counterfeit

Security marking and digital signature is unique to each individually identifiable product instance
→ high cost of 'cloning' to counterfeiters (more difficult to counterfeit)

Without instance-level identification : easier to counterfeit

Security markings and digital signatures are typically the same for all members of the production batch/lot
→ much lower cost of 'cloning' to counterfeiters (easier to counterfeit due to 'economy of scale')

Counterfeiter obtains one genuine instance of the product identified only by GTIN+Batch/Lot
and is then able to make large numbers of copies by replicating the ID, security markings and any encoded signature.

Figure E-3 How serialisation versus batch/lot level identification can deter counterfeiters

It should be remembered that not all industry sectors are ready to adopt mass serialisation and globally unique identification for each product instance. For many applications, identification of a specific product GTIN and its batch/lot identifier might be considered sufficient. However, in such situations, where several product instances share the same product GTIN and batch/lot identifier without having unique serial numbers, then if authentication codes are used in physical security markings, they will most likely be valid for each product instance within that specific batch or lot.

This presents a somewhat lower barrier to counterfeiters because if a counterfeiter obtains one instance of a targeted product and reads the security marking that is valid for that batch or lot, as well as the corresponding digital signature encoded in the data carrier, then it is quite easy for the counterfeiter to produce large numbers of counterfeit products having the same product GTIN, batch/lot identifier, digital signature and corresponding authentication code appearing within a physical security marking. Thus making the cloned replica counterfeit product instances effectively indistinguishable from the genuine product instances belonging to that batch/lot and product GTIN, as shown in Figure E-3.

The main difference between using the combination of product GTIN and Serial Number instead of a product GTIN and batch/lot identifier is that the specific combination of product GTIN and Serial Number should only appear once globally, on one individual product instance. Whereas the combination of product GTIN and batch/lot identifier could appear on many hundreds or thousands of product instances that are otherwise indistinguishable from each other. Relying only on identification using product GTIN and batch/lot identifier provides a potential 'economy of scale' that a counterfeiter could exploit, to introduce cloned counterfeit products that appear indistinguishable and appear to pass verification checks, even when using digital signatures (at least when the digital signature is online or encoded in a barcode).

Identifiers and data encoded in most GS1-conformant barcodes can be trivially copied. In contrast with barcodes, EPC/RFID tags include a non-programmable read-only unique Tag ID that is set by the manufacturer of the tag, such that each EPC/RFID tag should have a unique Tag ID. NFC tags have a similar feature, referred to as UID (Unique ID).

Although such a Tag ID is not a GS1-conformant way of constructing a unique instance-level identifier that works in an interoperable manner across barcodes and EPC/RFID data carriers, a Tag ID can nevertheless be included within digitally signed data, as an additional authentication factor as specified by ISO/IEC 20248 (see the example in section A.3); the digitally signed data for a uniquely identified product instance expresses what the Tag ID should be, as recorded by the brand owner or manufacturer, so if a different Tag ID is read for a product instance identified by that unique instance-level identifier, this indicates that the product instance is likely to be a counterfeit, rather than the genuine authentic product instance from the brand owner.

Note that a Tag ID included in digitally signed data should begin with the Allocation Class Identifier at address 00h of MB 10, and end with the final bit of the TID Serial Number Segment. Additional TID segments, beginning with the Optional Command Support Segment, should not be included in the digitally signed data, even if these segments are present in TID memory. The structure of TID memory is normatively specified in section 16 of GS1's EPC Tag Data Standard [TDS].

If the individual product instance has an EPC/RFID tag, it is possible to include the corresponding Tag ID in the digitally signed data, whether the digital signature is encoded within the data carrier or is retrieved online via a Web-centric approach.

For high value products or products where the risks of counterfeiting are considered too high, a higher barrier to counterfeiters can be achieved through the use of a unique product instance identifier (product GTIN + Serial Number) and the inclusion within the digitally signed data of a unique Tag ID from the specific tag in which the product GTIN + Serial Number are encoded. In contrast, identification using product GTIN + batch/lot ID provides a much lower barrier to counterfeiters.

F Appendix F - Overview of cryptography

F.1 Cast of characters

To better understand the concepts in this appendix, the fictitious characters introduced below are used to illustrate examples of each concept. Any resemblance to actual people or events, is purely coincidental and unintentional.

Alice and Bob are individuals that want to communicate with each other. The data they want to share, referred to generically as a document, is sensitive, and needs to be secured; exactly how depends on the level of trust between Alice and Bob and the environment in which they operate. Different security protocols will apply to different levels of trust and different environments.

Darth is a hacker. He may be interested in reading the documents Alice and Bob share, using the business intelligence in them to buy and sell shares in the companies that they represent. Or he may be interested in tampering with the documents to change the bank account number on an invoice to one he controls.

F.2 Concepts

F.2.1 Security vs. Integrity

Security is about ensuring that the document can be read only by those permitted to do so. Integrity is about ensuring that the document hasn’t been tampered with. Whether you apply neither, one, or both depends on your requirements. If you’re communicating with your bank over the Internet, you want the connection to be secure so that no one can intercept any of the information going back and forth. If you have been asked to share proof of your bank balance to a lender, your bank will supply some additional data, embedded in your monthly statement, to demonstrate the statement’s integrity, i.e., that you haven’t tampered with it before providing it to the lender to get more favourable terms or to borrow a higher amount.

F.2.2 Data at Rest vs. Data in Transit

The terms “data at rest” and “data in transit” refer to two different states of data. Data at rest refers to data that is stored and not actively moving from device to device or network to network. Data in transit is data that is actively moving from one location to another, whether over the Internet or through a private network.

The security and integrity applied to data at rest say nothing about how it is handled in transit. Similarly, the security and integrity applied to data in transit say nothing about how it is handled at rest. The two are entirely independent of each other, and different decisions need to be made about safeguarding stored data versus securing it as it moves across networks.

F.2.3 Verification vs. Validation

Although the two terms are often used interchangeably, for the purposes of this document, and in line with their usage in the W3C Verifiable Credentials Data Model, verification is about the structure (including any proofs of integrity required) and validation is about the content.

For example, the verification of a passport is about the physical passport itself:

- Does the document pass the overt security checks, e.g., holograms, coloured inks, security threads, raised lettering, corrugated cover?
- Does the document pass covert security checks, e.g., ultraviolet images, hidden text or images?
- Does the document pass forensic security checks, e.g., nano images, embedded DNA, materials analysis?

The level of verification required depends on the environment; forensic analysis is typically reserved for situations where there is strong reason to believe that the passport is fraudulent, it passes overt and covert security checks, and there is a high risk associated with accepting a fraudulent document.

Once the passport has been verified, it must now be validated:

- Is the picture that of the person presenting it?
- Does the name in the passport match the name on the airline ticket?
- Was the passport issued by a country recognized by the country the person is trying to enter?
- Does the country the person is trying to enter require a visa and does the passport have a valid visa in it?

Verification is a prerequisite for validation; if a document can’t be verified, it’s not valid. A document that can’t be validated, though, doesn’t imply that its verification process is flawed in any way.

F.3 Encryption

Encryption is a process used to secure and protect data by encoding it into an unreadable format using a reversible algorithm and an encryption key. The encoded format, known as ciphertext, can only be decoded (decrypted) back into its original readable form, called plaintext, by someone who knows how it was encoded and who possesses the correct decryption key. The purpose of encryption is to ensure confidentiality and prevent unauthorized access to data, either at rest or in transit.

F.3.1 Symmetric Encryption

In symmetric encryption, the same key is used for both encrypting and decrypting the data. This means that both the sender and the receiver must have access to the same key (referred to as a shared key) and keep it secure. Symmetric encryption is fast and efficient, making it suitable for encrypting large amounts of data.

Alice wants to send a document to Bob, but she’s concerned about it being intercepted. Alice’s concerns are justified because, unbeknownst to either her or Bob, Darth has hacked into Bob’s email, so he can see everything that Bob sends and receives. Alice prepares the document, encrypts it, and emails it.

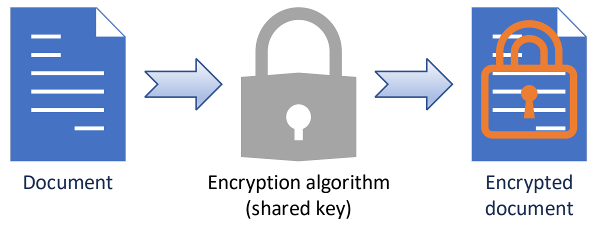


Figure F-1 Encrypting a document with a shared key

The challenge now is how to get the shared key to Bob. Alice is mindful of good security practices and doesn't send the password by email, because she knows that if Bob's email account has been hacked then both the document and the shared key will be accessible to the hacker. Instead, Alice sends the shared key as a text message.

Bob gets the document by email and the shared key by text message. With the two in hand, he can decrypt the document.

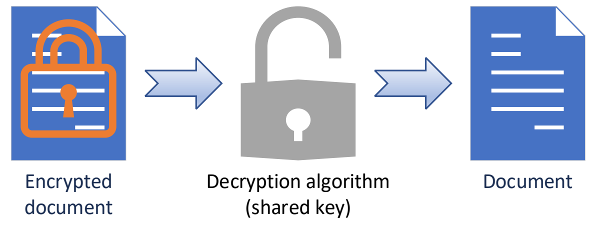


Figure F-2 Decrypting a document with a shared key

Although Darth has access to Bob's email, he can't decrypt the document without the shared key. There are three ways for him to get it: he can hack Alice's systems, he can hack Bob's systems, or he can hack the phone network. Bob's email has already been hacked, so the rest of his systems are likely the easiest target.

If Darth gets access to the shared key, the level at which he has compromised Alice and Bob depends on how well they adhere to best practices. If Alice and Bob use the same shared key time and again, then every document Alice sends to Bob is compromised. If Alice uses the same shared key with multiple parties, then every document she sends is at risk, and if Darth hacks other recipients, he will be able to read the documents Alice shares with them as well. The longer the shared key is in use and the more people it is shared with, the less secure it becomes.

F.3.2 Asymmetric Encryption

Also known as public-key encryption, asymmetric encryption uses two different keys (hence "asymmetric"): a private key, which is kept secret, and a public key, which can be shared with anyone. The keys are mathematically bound: if you know the private key, you can generate the public key very easily. The reverse, deriving the private key from the public key, is intentionally designed to be computationally impractical with current technology. The security is based on mathematical problems that are easy to perform in one direction (generating a public key from a private key) but extremely difficult to reverse (deriving the private key from the public key).

Which key is used and when depends on the direction of communication and what needs to be secured. Rather than discuss any particular scenario, this section shows what each option provides.

F.3.3 Encrypting With a Private Key

Alice encrypts a document with her private key and emails it to Bob.

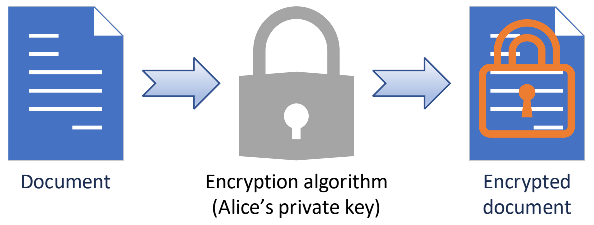


Figure F-3 Encrypting a document with Alice's private key

Alice shares her public key widely, perhaps by publishing it on her personal website or including it in her email signature. Bob receives the email and decrypts the document with Alice's public key.

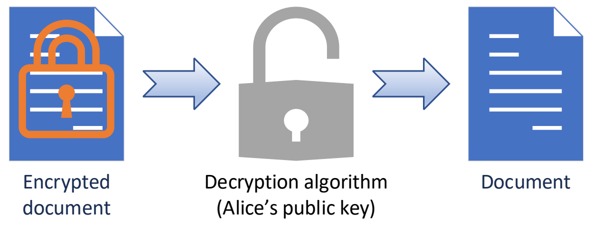


Figure F-4 Decrypting a document with Alice's public key

The fact that Bob was successful in decrypting the document tells him that it originated with Alice. Assuming her private key is secure, Alice is the only one who could have encrypted the document.

The problem, though, is that Darth also knows Alice's public key, and he can read the document as easily as Bob.

F.3.4 Encrypting With a Public Key

Alice encrypts a document with Bob’s public key and emails it to Bob. Assuming Bob’s private key is secure, Alice knows that he is the only one who can decrypt the document.

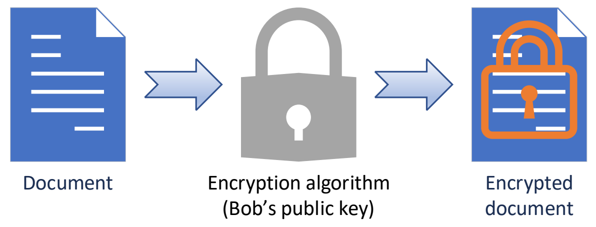


Figure F-5 Encrypting a document with Bob’s public key

Bob receives the email and decrypts the document with his private key.

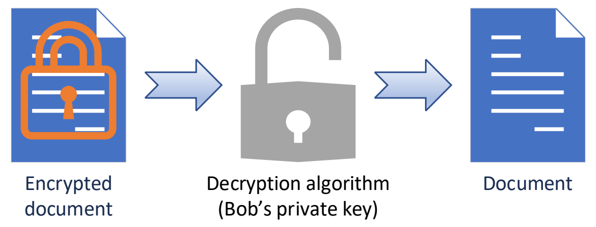


Figure F-6 Decrypting a document with Bob’s private key

The fact that Bob was successful in decrypting the document tells him that it was intended for him.

Darth has no access to Bob’s private key, so he can’t read the document. However, nothing stops him from sending a fake email to Bob, purportedly from Alice, with an encrypted document (encrypted using Bob’s public key) that directs Bob to take some action that would be to Darth’s benefit.

F.3.5 Double Encryption

Alice encrypts a document with her private key and again with Bob’s public key and emails it to Bob. Assuming Bob’s private key is secure, Alice knows that Bob is the only one who can decrypt the document.

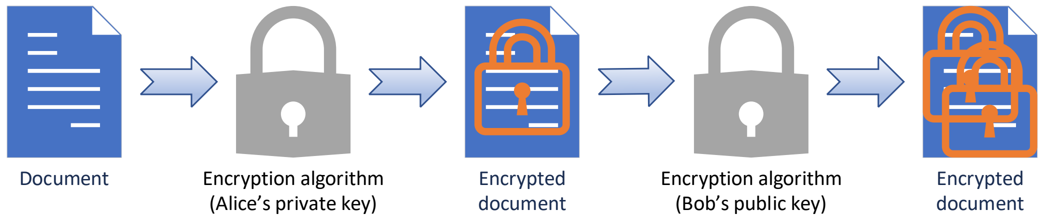

Document
Encryption algorithm (Alice's private key)
Encrypted document
Encryption algorithm (Bob's public key)
Encrypted document

Figure F-7 Encrypting a document with Alice's private key and Bob's public key

Bob receives the email and decrypts the document with his private key and Alice's public key.

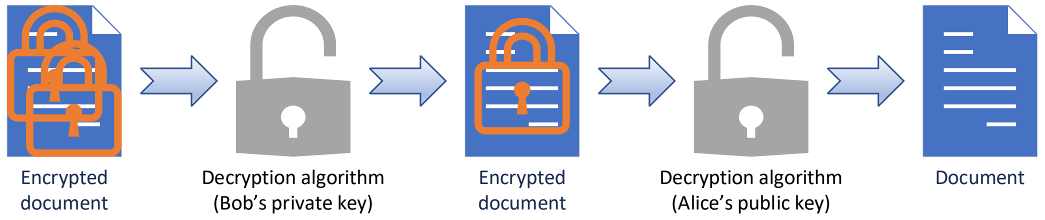

Encrypted document
Decryption algorithm (Bob's private key)
Encrypted document
Decryption algorithm (Alice's public key)
Document

Figure F-8 Decrypting a document with Bob's private key and Alice's public key

The fact that Bob was successful in decrypting the document tells him that it originated with Alice. Assuming her private key is secure, Alice is the only one who could have encrypted the document.

Darth has no access to Bob's private key, so he can't read the document. Furthermore, he can't send a fake email to Bob, purportedly from Alice, because he has no access to Alice's private key either. Naturally, if Darth can compromise a single party's private key, he can impersonate them as a sender and read everything they receive. However, only that party is compromised; no one else's correspondence is accessible to Darth.

F.3.6 Triple Encryption

Because of its complexity, asymmetric encryption is considerably slower than symmetric encryption, which can prove challenging for encrypting large amounts of data. A practical alternative, one that drives how secure connections on the web work, is to use both asymmetric and symmetric encryption:

1. Alice uses a random data generator to generate a one-time use shared key.
2. Alice applies double asymmetric encryption to the shared key, using her private key and Bob's public key.
3. Alice sends the encrypted shared key to Bob.
4. Alice applies symmetric encryption to the document, using the unencrypted shared key she generated.

5. Alice sends the encrypted document to Bob.
6. Bob applies double asymmetric decryption to the encrypted shared key, using his private key and Alice's public key.
7. Bob applies symmetric decryption to the document, using the decrypted shared key he received from Alice.

Web servers and browsers do this to establish a secure (HTTPS) connection. There are additional steps involved to prove that the web server the browser is connected to is the one that is expected, which is discussed later.

F.3.7 Summary

Based on the above, the following statements are true:

- A document encrypted with Alice's private key can be read by anyone with her public key and the reader is assured that the document originated with Alice.
    - The document has integrity but no security.
- A document encrypted with Bob's public key can be read only by Bob.
    - The document has security but no integrity.
- A document encrypted with both Alice's private key and Bob's public key can be read only by Bob and Bob is assured that the document originated with Alice.
    - The document has both security and integrity.
- A randomly generated, one-time use shared key may be delivered using double asymmetric encryption and the document itself delivered using symmetric encryption.
    - The key and by extension the document have both security and integrity.

F.4 Hashes and Digital Signatures

Very often, a document being shared is public in nature, or it's private in nature but shouldn't have to be decrypted every time it is read. Either way, it may be necessary to verify the integrity of the document to ensure that it hasn't been tampered with since it was generated. If you can verify some aspect of the document, you can have confidence that its original content hasn't been tampered with; the level of confidence depends on the security of the verification process.

Suppose, for example, that you have been told that the document is exactly 3,195,254,046 bytes in size. If the document size doesn't match, you can be certain that it's the wrong document or that it has been tampered with. It's easy to modify a document while preserving the size, so even if the document size matches, the confidence you have in its integrity is not going to be very high. What's needed is a way to generate something related to the document that is:

- Deterministic: The same input will always produce the same output.
- Fast: The process can quickly compute the output for any given input.
- Pre-image resistant: Given the output, it should be computationally infeasible to reverse it and find the original input.
- Sensitive: Any change to the input, no matter how small, should produce a significantly different output.
- Collision resistant: It should be extremely difficult to create two different inputs that produce the same output.

The class of algorithms that meet the above criteria are collectively called hash algorithms. A hash algorithm converts an input into a fixed-size string of bytes, called a hash or a digest, that appears random.

Suppose that Alice wants to send a document to Bob. She wants Bob to be able to read it, but also to trust that it hasn't been tampered with. Alice calculates the hash for the document and emails both the document and the hash to Bob.

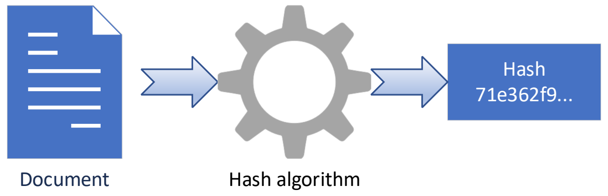


Figure F-9 Hash generation

Bob saves the document and, some time later, wants to read it. In between receiving and reading it, Darth has gained access to Bob's systems and modified the document. Before reading it, Bob verifies the document against the hash provided by Alice.

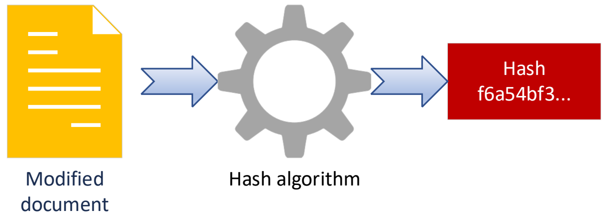


Figure F-10 Hash verification

- The hash that Bob generates doesn't match the one provided by Alice, so Bob knows that the document has been modified. He doesn't know where the modifications are, but he knows that the document can't be trusted.

- The obvious risk here is that, if Darth has access to Bob's systems and is able to modify the document, he may also be able to modify the hash that Bob recorded when he received the document. When Bob verifies the document, he verifies it against the hash generated by Darth, not the one originally provided by Alice, and ends up acting on its (fraudulent) contents.
- To protect against this risk, Alice doesn't provide the hash directly. Instead, she encrypts it with her private key and provides the encrypted result instead.

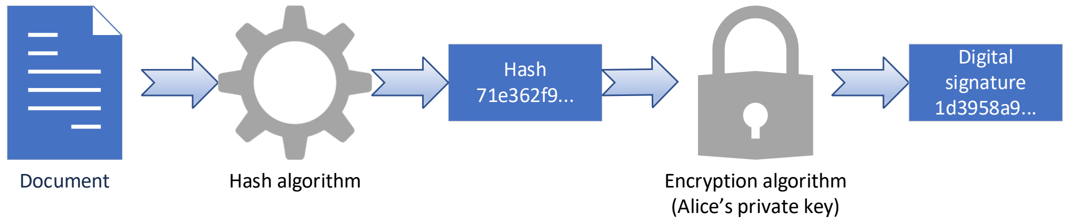


Figure F-11 Digital signature generation

In the same way that a physical signature is Alice's attestation to the contents of a physical document (cheque, contract, birthday card, etc.), an encrypted hash is Alice's attestation to the contents of an electronic document. The result is called a digital signature.

Now, Darth's access to Bob's systems doesn't matter as much. He can still modify the document, and he can still determine the original hash (using Alice's public key), but he can't generate a new digital signature because he doesn't have access to Alice's private key. Regardless, hoping that Bob doesn't do the verification, Darth modifies the document.

When Bob returns to the document, he verifies it against the digital signature provided by Alice.

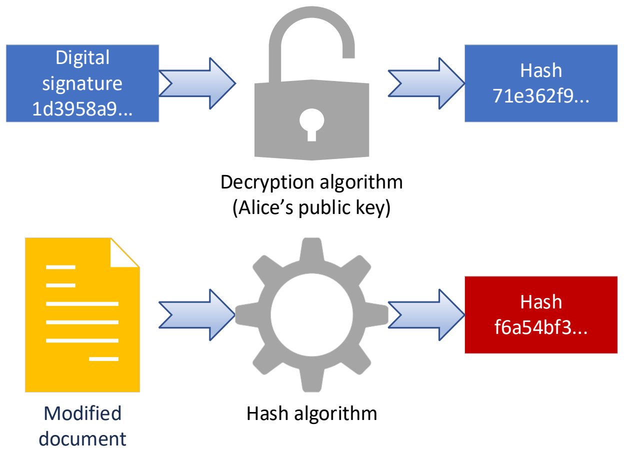

Digital signature 1d3958a9...
Decryption algorithm (Alice's public key)
Hash 71e362f9...
Modified document
Hash algorithm
Hash f6a54bf3...

Figure F-12 Digital signature verification

The hash that Bob generates doesn't match the one decrypted from the digital signature provided by Alice, so Bob knows that the document has been modified. He doesn't know where the modifications are, but he knows that the document can't be trusted.

F.5 Digital Signatures in Practice

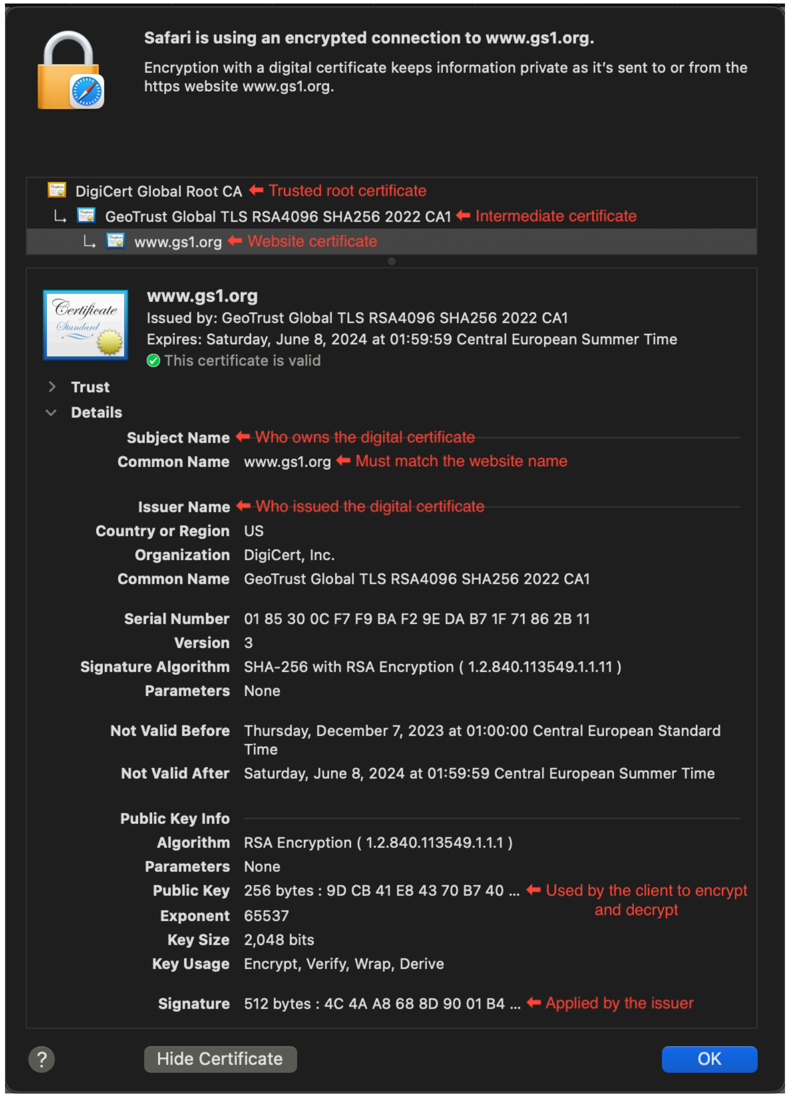
Digital signatures are very common. Banks and other high-security institutions will often sign documents they produce (e.g., monthly bank statements) so that they can't be tampered with. If you apply for a car loan, for example, the lender may want a copy of your latest pay stub (digitally signed by your employer or their payroll service provider) and a copy of your latest bank statement (digitally signed by your bank). Tampering with these documents to inflate either your salary or your net worth will be obvious to the lender the moment they try to verify the digital signature.

By far the most common use case for digital signatures is in securing your Internet experience: encrypting the data in transit between your browser and the server using triple encryption (security) and ensuring that your browser is connected to the server it's expecting (integrity).

Signing documents and encrypting data in transit both rely on a digital certificate, which is itself just another digitally signed document, but in a standardized format known as X.509, developed by the International Telecommunication Union (ITU). In general, a digital certificate contains the name of the entity to which it applies, other identifying information as required, the public key belonging to the entity, and a digital signature applied by another, trusted entity.

The digital certificate, whether used for signing documents or encrypting data, needs to be signed because otherwise, anyone with the right tools could impersonate any organisation or website they want. The remainder of this section will focus on securing a connection to a website, but the same principles apply to signing documents.

There are approximately 200 million active websites in the world, and it’s impossible to maintain a list of all of them and their public keys. Instead, trust is delegated through a digital certificate chain. Let’s look at the digital certificate for the GS1 website.



Safari is using an encrypted connection to www.gs1.org.

Encryption with a digital certificate keeps information private as it's sent to or from the https website www.gs1.org.

DigiCert Global Root CA ⬅ Trusted root certificate
└ GeoTrust Global TLS RSA4096 SHA256 2022 CA1 ⬅ Intermediate certificate
  └ www.gs1.org ⬅ Website certificate

www.gs1.org
Issued by: GeoTrust Global TLS RSA4096 SHA256 2022 CA1
Expires: Saturday, June 8, 2024 at 01:59:59 Central European Summer Time
✓ This certificate is valid

> Trust

⌄ Details

Subject Name	⬅ Who owns the digital certificate
Common Name	www.gs1.org ⬅ Must match the website name
Issuer Name	⬅ Who issued the digital certificate
Country or Region	US
Organization	DigiCert, Inc.
Common Name	GeoTrust Global TLS RSA4096 SHA256 2022 CA1
Serial Number	01 85 30 0C F7 F9 BA F2 9E DA B7 1F 71 86 2B 11
Version	3
Signature Algorithm	SHA-256 with RSA Encryption (1.2.840.113549.1.1.11)
Parameters	None
Not Valid Before	Thursday, December 7, 2023 at 01:00:00 Central European Standard Time
Not Valid After	Saturday, June 8, 2024 at 01:59:59 Central European Summer Time
Public Key Info	
Algorithm	RSA Encryption (1.2.840.113549.1.1.1)
Parameters	None
Public Key	256 bytes : 9D CB 41 E8 43 70 B7 40 ... ⬅ Used by the client to encrypt and decrypt
Exponent	65537
Key Size	2,048 bits
Key Usage	Encrypt, Verify, Wrap, Derive
Signature	512 bytes : 4C 4A A8 68 8D 90 01 B4 ... ⬅ Applied by the issuer

Figure F-13 The digital certificate for the GS1 website

At the top of the digital signature dialog is the digital certificate chain, starting with a trusted root certificate and continuing down to the website certificate, with zero or more intermediate certificates along the way. Verification starts at the bottom and works up to the trusted root certificate. If everything is verified, the digital certificate is valid, and the connection to the website can be trusted.

But what makes a certificate a trusted root certificate? How does your browser know to trust DigiCert Global Root CA when connecting to the GS1 website? It knows this because there is, installed with and secured by your operating system, a database of trusted root certificates. The database is maintained by the Certification Authority Browser Forum (CA/Browser Forum), a voluntary gathering of Certificate Issuers and suppliers of Internet browser software and other applications that use certificates. As long as you trust that your operating system hasn't been compromised, you can trust the trusted root certificates installed by it, and you can therefore trust the connections to the websites you are browsing.

The website verification algorithm looks something like this:

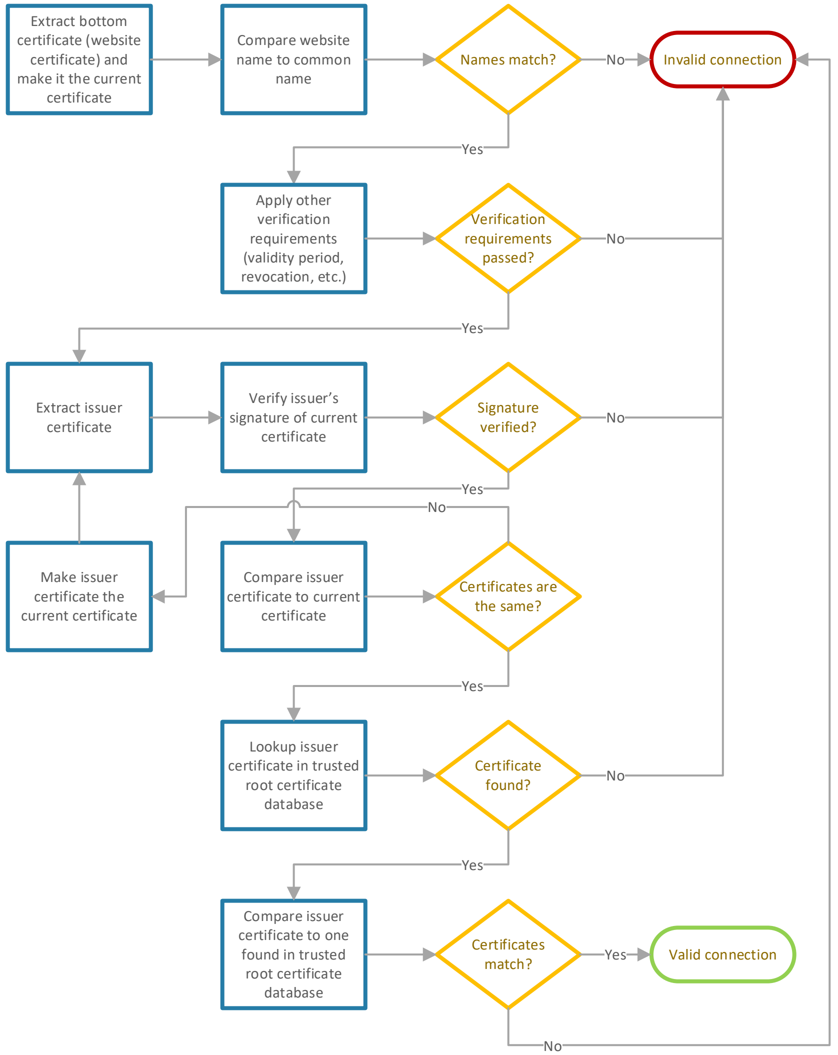

Extract bottom certificate (website certificate) and make it the current certificate
Compare website name to common name
Names match?
No
Invalid connection
Yes
Apply other verification requirements (validity period, revocation, etc.)
Verification requirements passed?
No
Yes
Extract issuer certificate
Verify issuer's signature of current certificate
Signature verified?
No
Yes
No
Make issuer certificate the current certificate
Compare issuer certificate to current certificate
Certificates are the same?
Yes
Lookup issuer certificate in trusted root certificate database
Certificate found?
No
Yes
Compare issuer certificate to one found in trusted root certificate database
Certificates match?
Yes
Valid connection
No

Figure F-14 Website verification

Note that trusting the connection doesn't imply that you can trust the website itself; the digital certificate authenticates and secures the connection but says nothing about the content. It's as easy for a malicious website to get a digital certificate as it is for your bank, so caution is still required when acting on the content of an unknown website.

The number of trusted root certificates is relatively small (less than 200) and the verification processes that the companies behind them require websites to go through can vary considerably, from automated installation when setting up a new business website with a credit card, to full business registration verification for larger websites. In practice, though, there is no way for the user to tell which ones have had stricter verification applied. More importantly, it's difficult to add an entry to the CA/Browser Forum database, especially for non-web-related purposes, which limits the scalability of the digital certificate infrastructure.

There are other digital certificate authorities for other purposes, such as the International Civil Aviation Organization (ICAO), which manages the ICAO Master List containing the Country Signing Certificate Authority (CSCA) public key certificates for validating e-passports.

Digital certificates can be used without relying on any centralized authority. For example, your company might have its own self-signed root certificate for signing internal documents, but it would have little use outside the company, as others could not be assured of its validity. The Pretty Good Privacy (PGP) standard supports root certificates outside the CA/Browser Forum database, but it requires that anyone verifying a PGP digital signature make an independent decision about whether to trust the root certificate.

F.6 Verifiable Credentials

Like a digital certificate, a verifiable credential is just a digitally signed document in a standardized format. What makes it different from a digital certificate is that its content is highly extensible, which means that it has no predefined purpose.

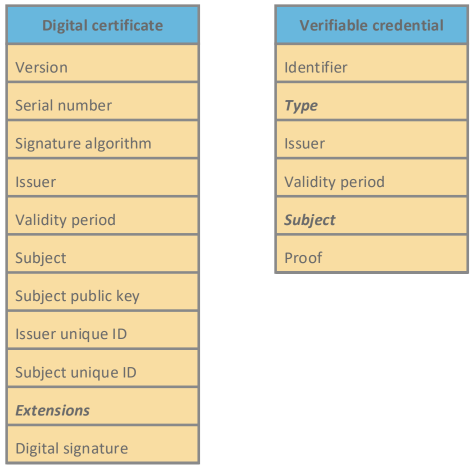


Figure F-15 Digital certificate and verifiable credential structures with extension points in boldface

A digital certificate has a singular purpose: the association of a public key with a subject. Everything else is subordinate to that purpose. It is extensible, but only in a limited fashion (individual fields only, not structured data), and only in ways that support its purpose. By comparison, the purpose of a verifiable credential is declared via its type, with structured content (aligned with the type) provided in the subject.

Digital certificates don’t require a centralized authority. In practice, the CA/Browser Forum is the de facto centralized authority for most digital certificates, with others, such as ICAO, providing a similar service for specialized applications.

Although verifiable credentials are digitally signed, they don’t rely on X.509 digital certificates for verification. Verification constraints are much looser by design: if you trust the issuer and can verify the credential using the issuer’s public key (along with other structural verification requirements), the credential is verified. Every issuer has a unique identifier, which must be a URI that, if dereferenced, the must result in a document containing machine-readable information about the identifier (including the issuer’s public key).

This approach doesn’t require a central database of issuer documents (such as the CA/Browser Forum database), as each issuer provides its own means for retrieving its public key. If the dereferencing mechanism can be trusted, the public key can be retrieved with confidence.

Verifiable credentials can be chained together, just like X.509 digital certificates, but in contrast, they don’t have to have the same purpose. For example, a physician could have a verifiable credential, issued by their local regulatory authority, confirming their licence to practice medicine. The physician could then issue a verifiable credential to a patient to prove vaccination status for travel. The vaccination credential would contain either a link to or copy of the doctor’s licence to practice. Verifying the vaccination status requires that the vaccination credential be verified as having been issued by a doctor licensed by a known regulatory authority (for which there might be a centrally managed database). The mechanism is similar to that for website verification, but is much more flexible:

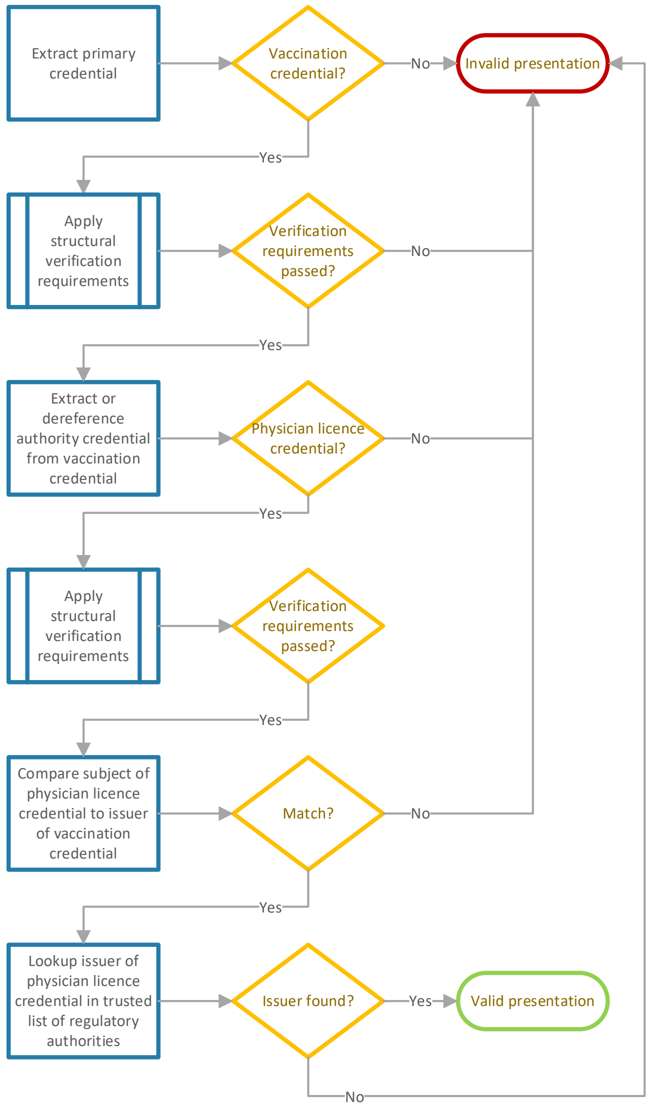

Extract primary credential
Vaccination credential?
No
Invalid presentation
Yes
Apply structural verification requirements
Verification requirements passed?
No
Yes
Extract or dereference authority credential from vaccination credential
Physician licence credential?
No
Yes
Apply structural verification requirements
Verification requirements passed?
Yes
Compare subject of physician licence credential to issuer of vaccination credential
Match?
No
Yes
Lookup issuer of physician licence credential in trusted list of regulatory authorities
Issuer found?
Yes
Valid presentation
No

Figure F-16 Vaccination credential verification

Once the vaccination credential has been verified, it can be validated (e.g., by confirming that the vaccination date was long enough before the date of travel for the vaccine to be effective and that the vaccinations received are appropriate for the destination).