PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In the **PATENT APPLICATION** of:	
Swoop IP Holdings LLC	
Application No.: 19/024,922	
Confirmation No.: 5317	Our File: SWO-PT014.6
Filed: January 16, 2025	
For: VENDOR TOKEN GENERATOR	
Group: 3699	
Examiner: DAVIDA LEE KING	

DECLARATION UNDER 37 C.F.R. §1.132

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

I, John P. Killoran, Jr., submit this declaration in connection with U.S. Patent Application No. 19/024,922, entitled "Vendor Token Generator."

Contents

I. Introduction

1. I understand that the pending claims in this application have been rejected under 35 U.S.C. § 101. I further understand that this declaration is being submitted under 37 C.F.R. § 1.132 as evidence relevant to subject matter eligibility. This declaration is directed solely to subject matter eligibility under 35 U.S.C. § 101.

2. I am not submitting this declaration to address novelty under 35 U.S.C. § 102, obviousness under 35 U.S.C. § 103, written description or enablement under 35 U.S.C. § 112(a), definiteness under 35 U.S.C. § 112(b), or any other ground of patentability. I understand that those issues are separate from the subject matter eligibility issue addressed in this declaration.

3. The purpose of this declaration is to explain, from the perspective of a person having ordinary skill in the art at the relevant time, how the specification and pending claims of U.S. Patent Application No. 19/024,922 describe and claim a specific technological improvement to secure SMTP-based e-commerce transaction processing. In particular, this declaration addresses how a person of ordinary skill would understand the claimed architecture to improve the way an e-commerce system processes SMTP-based transaction messages that contain tokenized transaction instructions.

4. As explained below, the specification and claims do not merely describe using email to conduct a purchase. Rather, in my opinion, a person of ordinary skill in the art would understand the specification and claims to describe an ordered security architecture in which an e-commerce system receives an SMTP message containing a transaction token, decodes the token, validates token freshness and one-time-use status, records token use, authenticates the sender against decoded token information, verifies that the associated vendor is authorized to generate the

token, and performs the transaction only when the required security checks are successful.

5. My opinions are based on my personal knowledge, my experience with email-based e-commerce systems and tokenized transaction workflows, and my review of materials relating to this application. Those materials include the specification and drawings of U.S. Patent Application No. 19/024,922, the pending claims, the Office Action rejecting the pending claims, and the references cited by the Examiner, including U.S. Patent No. 8,775,263 to Killoran et al., on which I am a named inventor. My opinions are also based on my understanding of how a person having ordinary skill in the art would have understood the technology described in the specification and recited in the claims at the relevant time.

II. My Background

6. I am John P. Killoran, Jr. I am currently the Chief Executive Officer of Swoop In Technologies LLC. I have worked for many years in technology development, including in the fields of electronic commerce, secure identity systems, passwordless authentication, payment systems, email-based transaction systems, and cybersecurity.

7. Throughout my career, I have focused on developing practical technical solutions to problems involving trust, identity, authentication, and transaction processing. My work has included technologies involving cryptography, email, QR technology, payments, identity, artificial intelligence, passwordless systems, and cybersecurity. I have been involved in the development of systems intended to improve how users, vendors, and transaction-processing systems interact across electronic networks, including systems designed to reduce user friction while preserving security and trust.

8. I have also been involved in the development of technologies relating to secure customer/vendor transaction workflows. In such systems, a customer may interact with a vendor-facing interface, while a separate transaction-processing or payment system determines whether a transaction should be accepted and processed. Based on my experience, the technical design of such systems requires attention to how transaction instructions are generated, transmitted, authenticated, validated, and processed by networked computer systems.

9. I am familiar with email-based e-commerce systems and with the technical issues that arise when email messages are used to initiate, confirm, modify, or complete electronic transactions. My experience includes systems in which email messages, mailto links, tokens, vendor systems, payment systems, and customer devices interact as part of an electronic transaction workflow. I am also familiar with the use of tokenized information in transaction processing, including the use of tokens to carry machine-readable information used to identify a customer, identify a vendor, identify transaction data, authenticate a transaction request, or determine whether a transaction request should be processed.

10. I am a named inventor on U.S. Patent No. 8,775,263 to Killoran et al., titled "System and Method for Email-Based E-Commerce." I understand that this patent has been cited by the Examiner in the present Office Action. That patent reflects my prior work involving email-based e-commerce, including the use of email messages and mailto hyperlinks to facilitate e-commerce transactions.

11. The cited Killoran patent describes, among other things, advertisement email messages that may include mailto hyperlinks. When selected by a user, those mailto hyperlinks may generate order email messages containing transaction-related parameters, such as customer identifiers, product identifiers, quantities, campaign identifiers, or other information associated with a transaction. That prior work

concerned the use of email as part of an e-commerce workflow and reflects my experience with systems in which email messages can carry transaction-related information.

12. Based on that work and my other experience, I am familiar with technical problems encountered in email-based e-commerce systems. Those problems include determining whether an email message corresponds to a valid transaction instruction, whether a transaction message has been generated by or on behalf of an authorized vendor, whether the sender of the email message corresponds to the intended customer, and whether a transaction instruction has already been used or may be replayed. I am also familiar with risks associated with unauthorized transaction messages, stale links, duplicate or replayed messages, spoofed sender information, and vendor-generated transaction requests.

13. In my experience, those risks are not merely business concerns. They are technical issues that arise from using email messages, including SMTP-based messages, as carriers for machine-readable transaction instructions. Email messages may be generated at one time and processed later. They may remain in a mailbox, be forwarded, be resent, be copied, or be delayed in transmission. If such messages contain transaction instructions or tokens, the receiving transaction-processing system must determine whether the message and token should be trusted before executing a transaction.

14. I am also familiar with the state of the art at the relevant time concerning email-based checkout systems, tokenized payment systems, vendor-side systems, e-commerce systems, and payment-processing systems. At the relevant time, persons of ordinary skill in the art understood that email links and generated email messages could be used to facilitate e-commerce transactions. They also understood that tokenized information could be used to carry machine-readable transaction or

authentication information. In addition, persons of ordinary skill understood that payment or e-commerce servers could be used to validate transaction information and determine whether a transaction should be processed.

15. My background gives me personal knowledge of the technical context in which the present application was filed. I understand the difference between a system that merely uses email or a mailto link to initiate a purchase and a system that uses an ordered token-security architecture to determine whether an SMTP-based transaction message should be trusted and processed. My opinions in this declaration are based on that experience, my knowledge of the field, and my review of the present application, pending claims, Office Action, and cited references.

III. Level of Ordinary Skill in the Art

16. For purposes of this declaration, I understand the relevant field of art to include secure electronic commerce systems, email-based transaction systems, tokenized transaction authorization, payment-processing systems, and computer-network security. This field includes systems in which customer devices, vendor systems, e-commerce systems, payment servers, and financial-processing systems communicate over electronic networks to initiate, authenticate, authorize, and process transactions.

17. In my opinion, a person of ordinary skill in the art at the relevant time would have had a bachelor's degree in computer science, computer engineering, electrical engineering, or a related technical field, and at least two years of practical experience with electronic commerce systems, email-based communication protocols, payment-processing systems, tokenized transaction systems, identity/authentication systems, or computer-network security. Additional practical experience with secure transaction-processing systems could substitute for formal

education, and advanced education in a relevant technical field could substitute for some practical experience.

18. A person of ordinary skill in the art would have been familiar with conventional e-commerce checkout workflows, including web-based checkout, payment-gateway interactions, stored customer payment credentials, vendor checkout interfaces, and transaction confirmation workflows. Such a person also would have understood conventional email-based checkout workflows, including systems in which a user selects a link or button and an email client generates an email message containing transaction-related information.

19. A person of ordinary skill also would have understood the technical characteristics of email communication, including that SMTP-based email messages are asynchronous, may be stored, forwarded, copied, resent, delayed, or delivered after the circumstances surrounding the original message have changed. Such a person would have recognized that these characteristics create technical security issues when email messages are used as transaction-initiation or transaction-confirmation vehicles.

20. A person of ordinary skill would also have been familiar with tokenized transaction authorization. Such a person would have understood that tokens may be used to carry machine-readable transaction data, customer information, vendor information, expiration information, cryptographic authentication information, or other information used by a receiving system to determine whether a request should be accepted. Such a person also would have understood that tokens used in transaction-processing systems may require server-side validation before a transaction is executed.

21. In addition, a person of ordinary skill would have understood common security concerns associated with networked transaction systems, including stale

requests, replayed messages, duplicate submissions, unauthorized token generation, sender spoofing, mismatches between message sender information and token-encoded identity information, and compromised cryptographic credentials. Such a person would have recognized that these concerns are particularly important when machine-readable transaction instructions are transmitted through email rather than through a closed or synchronous checkout session.

22. A person of ordinary skill would therefore have understood that merely receiving an email message containing product, price, quantity, or customer information is not necessarily sufficient to determine whether the message should be processed as an authorized transaction instruction. Such a person would have understood the need for technical controls that allow the receiving e-commerce system to determine whether the message is current, whether the token has already been used, whether the sender corresponds to the customer identified in the token, and whether the token was generated by an authorized vendor.

23. In my opinion, the level of ordinary skill described above is consistent with the subject matter of the present application. The specification describes a vendor token generator, tokenized email and web checkout, vendor-side token generation, e-commerce-system token decoding, cryptographic key usage, sender email verification, token expiration, token-use tracking, and payment processing. A person having the level of skill described above would have been able to understand the specification and would have recognized the technical significance of using an ordered security architecture to process SMTP-based transaction messages.

IV. Relationship to My Prior Work

24. I am a named inventor on U.S. Patent No. 8,775,263 to Killoran et al., which is cited in the Office Action in this application. My prior work reflected in that patent involved email-based e-commerce systems, including the use of email

messages and mailto links to facilitate the purchase of products or the making of donations. In general, that work addressed ways in which an email message could be used as part of an e-commerce workflow, including by allowing a user to select a mailto link that generated an email message containing transaction-related information.

25. Based on my experience with that technology, I understand that using a mailto link to generate a purchase email is different from the token-security architecture described and claimed in the present application. A mailto link may cause an email client to generate an email message containing product, quantity, customer, campaign, or order information. That type of functionality can be useful because it allows a customer to initiate or confirm a transaction through an email client. But the mere generation of an email message from a mailto link does not necessarily provide a complete security protocol for determining whether a later-received SMTP message should be trusted and processed by an e-commerce system.

26. In my opinion, a person of ordinary skill in the art would recognize that the cited Killoran patent reflects one stage of email-based e-commerce technology. It shows that email messages and mailto links can be used to carry transaction-related information and facilitate customer interaction with an e-commerce system. However, a person of ordinary skill would also recognize that the presently claimed invention is directed to a more specific transaction-security architecture for SMTP-based transaction processing.

27. The presently claimed architecture is not limited to the concept of generating a purchase email. Rather, the claimed architecture requires the e-commerce system to evaluate a received SMTP transaction message through multiple technical checks before transaction execution. Those checks include token decoding, expiration validation, nonce or one-time-use validation, token-use

datastore lookup, recording of token use, sender authentication based on decoded token information, and vendor-authorization verification using a token-authenticator bound to the token.

28. In my opinion, the distinction is important. A system that merely uses a mailto link to generate a purchase email may receive an email message containing order information. But that alone does not necessarily determine whether the email message is stale, whether the same token has already been used, whether the message sender corresponds to the customer identified by the token, or whether the token was generated by a vendor authorized to generate transaction tokens. Those are technical questions that must be answered by the e-commerce system before processing the transaction if the system is to provide the claimed type of transaction security.

29. The present application addresses those technical questions by describing a vendor-side token generator and an e-commerce system that cooperate to generate, transmit, decode, authenticate, and process tokens. The specification explains that a token may be a sequence of byte data, string, or file used to authenticate a transaction, and that a token may include encrypted strings, files, passwords, ciphers, or other data containing information used to perform or authenticate a transaction when sent to payment servers. Present Application ¶ [0017]. The specification also explains that tokens may contain information used to perform or authenticate a transaction. Present Application ¶ [0027]. A person of ordinary skill would therefore understand the token in the present application as a machine-readable transaction-authentication object, not merely as a customer-visible order label.

30. The present application also describes token-encoded validity and transaction information that goes beyond ordinary mailto-generated order content. The token parameters may include, among other things, a partner ID, token type,

card token, email address, amount, user data, expiration date, user-agent information, and IP address. Present Application ¶¶ [0028]-[0043]. The specification also describes encrypted token contents including token type, card token, email address, amount, partner ID, signup URL, expiration date, group information, user data, and computer/browser identifiers. Present Application ¶¶ [0057]-[0067]. In my opinion, a person of ordinary skill would understand that these fields are used by the e-commerce system to evaluate whether a received transaction message corresponds to an authorized transaction context.

31. The present application further describes expiration and one-time-use controls that address technical risks associated with email-based transaction messages. The specification explains that a token may include an expiration date, after which the token is no longer considered valid. Present Application ¶ [0064]. The specification also explains that the e-commerce system may store a checksum of a token after it has been used and prevent the same token from being used more than once. Present Application ¶ [0070]. In addition, the specification describes storing a group reference number for a used token and preventing that group reference number from being used more than a predetermined number of times. Id. A person of ordinary skill would understand these features as replay-prevention and token-use-state mechanisms, not merely as ordinary order-entry features.

32. The present application also describes sender-token matching. The specification explains that, when a transaction is attempted, the payment server decodes the token, authenticates the sender of the email, and may process the transaction. Present Application ¶ [0048]. The specification further explains that the e-commerce system may verify an email address associated with the sender of an email, thereby permitting the payment server to verify that the payload was intended for use by the recipient because the email is included in the decrypted payload

message. Present Application ¶ [0072]. A person of ordinary skill would understand that this binds the SMTP message sender to customer information contained in the decrypted token payload.

33. The present application also describes vendor authorization through cryptographic or equivalent authentication mechanisms. The specification explains that token generation may involve public-key cryptography, including encryption using an XSalsa20 stream cipher, authentication using Poly1305 MAC, and public keys using Curve25519 high-speed elliptic-curve cryptography. Present Application ¶ [0054]. The specification further explains that the vendor server may receive a unique private and public key associated with the vendor and may generate a token using its unique key. Present Application ¶ [0056]. The e-commerce system then receives the encoded token, decrypts token information, confirms key validity, ensures that the vendor server is authorized to generate tokens, and processes payment. Present Application ¶ [0068]. A person of ordinary skill would understand this as a vendor-authorization and token-authentication architecture.

34. Based on my experience with email-based e-commerce systems, I would not characterize the presently claimed invention as merely the use of email to conduct a transaction. The use of email provides the communication channel, but the claimed improvement concerns how the e-commerce system determines whether a tokenized SMTP message should be trusted. The e-commerce system does not process the transaction simply because an email message was received. Instead, the system decodes the token, validates the token's freshness and prior-use status, records token use, authenticates the sender against decoded token information, verifies vendor authorization, and performs the transaction only when the required checks succeed.

35. In my opinion, a person of ordinary skill would therefore understand the cited Killoran patent and the presently claimed invention as addressing different aspects of email-based e-commerce technology. Killoran reflects the use of email and mailto links to facilitate customer interaction and transaction initiation. The presently claimed invention addresses technical security limitations that arise when such email-based transaction messages are processed by an e-commerce system. Those limitations include stale-message risk, replay risk, sender mismatch risk, and unauthorized vendor-token-generation risk.

36. A person of ordinary skill would understand that these are technical limitations of email-based transaction systems, not merely commercial considerations. SMTP messages may persist, be forwarded, be resent, be delayed, or be received outside the original transaction context. A tokenized transaction instruction carried in such a message therefore needs to be evaluated by the receiving e-commerce system before transaction execution. The presently claimed architecture provides that evaluation through expiration validation, nonce or one-time-use replay prevention, token-use datastore recording, sender-token matching, and vendor-authenticator verification.

37. Accordingly, in my opinion, the presently claimed invention is not merely a restatement of email-based checkout or the use of a mailto link. It is directed to a specific security architecture for processing SMTP-based transaction messages. A person of ordinary skill would understand the claims as addressing technical security limitations of email-based transaction systems by changing how the e-commerce system evaluates, authenticates, records, and conditionally processes tokenized transaction instructions.

V. Overview of the Technological Problem

38. Email-based checkout systems permit an e-commerce transaction to be initiated, confirmed, modified, or canceled through an email message. In such systems, a customer may select a link, button, or other control in an offer message, webpage, or checkout interface, and that selection may cause an email client to generate an email message containing transaction-related information. This is consistent with the technical operation of the mailto URI scheme. RFC 6068 explains that a mailto URI may specify not only a destination email address, but also mail header fields and a message body. M. Duerst et al., RFC 6068, The 'mailto' URI Scheme §§ 1-3, 6.1 (Internet Eng'g Task Force Oct. 2010). RFC 6068 further explains that resolving a mailto URI does not cause an immediate interaction with a server; instead, the client creates a message to the designated address with default header fields, and the user may edit the message, send the message unedited, or choose not to send the message. Id. § 3.

39. Patent publications in this field also recognized that email-based transaction systems could use mailto links or buttons to generate transaction-related email messages. For example, U.S. Patent No. 8,775,263 to Killoran et al. describes advertising email messages containing mailto hyperlinks that generate order reply email messages including customer, product, and quantity information. U.S. Pat. No. 8,775,263 col. 4 ll. 4-31, col. 6 l. 62-col. 7 l. 27. Similarly, U.S. Patent No. 9,704,190 to Custer describes an email checkout system in which a customer may receive an email message containing a mailto hyperlink or embedded button, and selecting the button may generate a reply email message that includes a token. U.S. Pat. No. 9,704,190 col. 2 ll. 36-54, col. 10 ll. 38-50. These references confirm that, at the relevant time, persons working in this field understood that email-based checkout

could use links, buttons, and generated email messages as part of an electronic transaction workflow.

40. In my experience, this type of architecture can reduce customer friction because the customer can initiate or confirm a transaction through familiar email functionality rather than manually re-entering payment information through a conventional checkout page. However, the use of email messages to initiate or confirm transactions creates technical security issues that are specific to the use of SMTP and Internet mail infrastructure in an electronic transaction-processing architecture. Internet mail is an asynchronous store-and-forward system. RFC 5598 describes basic message transfer from author to recipients as using an "asynchronous store-and-forward communication infrastructure" through a sequence of independent transmissions through mail transfer agents. D. Crocker, RFC 5598, Internet Mail Architecture § 5 (Internet Eng'g Task Force July 2009). RFC 5598 further explains that relays perform transfer-service routing and store-and-forward by transmitting or retransmitting messages to recipients. Id. § 2.2.2. RFC 5321 similarly provides that when delivery cannot be completed immediately, the SMTP client retains responsibility for delivery and may requeue the message for a subsequent attempt. J. Klensin, RFC 5321, Simple Mail Transfer Protocol §§ 3, 4.5.4.1 (Internet Eng'g Task Force Oct. 2008).

41. Because of that architecture, an SMTP message may be generated at one time and received or processed at a later time. It may remain in a sender's outbox, sent-items folder, or mailbox; it may be queued and retried; and it may traverse multiple relays or intermediaries before reaching the transaction-processing system. See Crocker, RFC 5598 § 5; Klensin, RFC 5321 § 4.5.4.1. In addition, Internet mail messages can be resent or forwarded. RFC 5322 describes "Resent" header fields for messages reintroduced into the transport system and recognizes

forwarding a message as a common message operation. P. Resnick, RFC 5322, Internet Message Format §§ 3.6.6, 3.6.7 (Internet Eng'g Task Force Oct. 2008). RFC 5598 likewise describes forwarding through a mediator and explains that a forwarded message may be treated as a new message. Crocker, RFC 5598 § 5.

42. SMTP sender information also presents technical authentication limitations. RFC 5321 states that SMTP mail is "inherently insecure" because users can create messages that may trick a recipient into believing that the messages came from somewhere else. Klensin, RFC 5321 § 7.1. RFC 5321 further states that SMTP mail inherently cannot be authenticated, and integrity checks cannot be provided, at the transport level. Id. RFC 5321 also states that there is no inherent relationship between either the reverse-path or forward-path addresses in the SMTP envelope and the addresses in the message header section. Id. § 7.2. Therefore, in an email-based transaction system, the fact that an SMTP message appears to come from a particular address does not, by itself, establish that the sender corresponds to the customer identified in a transaction token.

43. Conventional email checkout, by itself, does not necessarily resolve these issues. A conventional email-based checkout message may include product information, price information, quantity information, customer-identifying information, or other transaction parameters. But the mere presence of such information in an email message does not necessarily establish that the message corresponds to a current transaction authorization. For example, an email message generated from a link may be transmitted or re-transmitted after the underlying offer has expired, after the price or inventory has changed, or after the customer's authorization context has changed. In token-based security systems generally, expiration is a recognized technical control. RFC 6819 explains that short token expiration times may protect against replay, token leakage, and online guessing. T.

Lodderstedt et al., RFC 6819, OAuth 2.0 Threat Model and Security Considerations § 5.1.5.3 (Internet Eng'g Task Force Jan. 2013). Industry guidance likewise recognizes expiration as a security property of sensitive tokens; for example, the OWASP Forgot Password Cheat Sheet recommends that password reset tokens be randomly generated, sufficiently long, stored securely, single use, and expire after an appropriate period. OWASP Found., Forgot Password Cheat Sheet, OWASP Cheat Sheet Series (2024).

44. Likewise, a conventional email checkout message does not necessarily establish that the token or transaction instruction contained in the message has not already been used. Because email messages can persist after transmission and can be resent, forwarded, copied, or delivered more than once, the same transaction token or transaction instruction may be presented to the e-commerce system more than once. Replay risk is a recognized computer-security problem. RFC 4949 defines a replay attack as an attack in which a valid data transmission is maliciously or fraudulently repeated, either by the originator or by a third party who intercepts and retransmits the data. R. Shirey, RFC 4949, Internet Security Glossary, Version 2, at 245 (Internet Eng'g Task Force Aug. 2007). RFC 4949 also explains that "liveness" provides assurance that data is freshly transmitted and not replayed from a previous transmission. Id. at 163.

45. The use of freshness data, nonces, timestamps, or unique message identifiers to prevent replay was also well recognized in the academic literature. Needham and Schroeder's work on authentication protocols used nonces in cryptographic exchanges to establish freshness and authenticate communication between networked computers. Roger M. Needham & Michael D. Schroeder, Using Encryption for Authentication in Large Networks of Computers, 21 Commc'ns ACM 993, 994-96 (1978). Denning and Sacco later explained that replay of old

authentication messages can compromise a protocol and proposed timestamps as a way to address stale authentication data. Dorothy E. Denning & Giovanni Maria Sacco, Timestamps in Key Distribution Protocols, 24 Commc'ns ACM 533, 533-36 (1981). More recent literature likewise recognizes that message authentication alone may not be sufficient to prevent replay. Boudhayan Gupta explains that HMACs are effective for keyed message authentication, but if used alone may still be susceptible to replay attacks, and proposes combining time-dependent authentication with a unique message identifier so that recipients can detect and ignore replayed messages. Boudhayan Gupta, A Replay-Attack Resistant Message Authentication Scheme Using Time-Based Keying Hash Functions and Unique Message Identifiers, arXiv:1602.02148, at 1-2 (2016).

46. A nonce or one-time-use value is a recognized technical mechanism for addressing replay risk. RFC 4949 defines a nonce as a random or non-repeating value included in protocol data, usually to guarantee liveness and detect and protect against replay attacks. Shirey, RFC 4949, at 199. RFC 6819 similarly recognizes limiting token usage or enforcing one-time token usage as a countermeasure against replay of tokens. Lodderstedt et al., RFC 6819 § 5.1.5.4. RFC 6819 further explains that, if an authorization server observes more than one attempt to redeem an authorization code, the authorization server may reject the current request. Id. Industry guidance concerning request tokens is consistent with the same principle. The OWASP Cross-Site Request Forgery Prevention Cheat Sheet explains that synchronizer tokens should be secret, unique, and unpredictable and should be verified by the server to determine whether a received request is authorized. OWASP Found., Cross-Site Request Forgery Prevention Cheat Sheet, OWASP Cheat Sheet Series (2024). If an e-commerce system does not maintain and check token-use state,

the system may be unable to determine whether a received token represents a first legitimate use or a later replay of a previously used transaction instruction.

47. Patent publications in adjacent transaction-security systems also recognized replay concerns. For example, U.S. Patent Application Publication No. 2014/0061302 to Hammad describes validating a physical verification token and checking token serial numbers and/or IP addresses against a database to prevent replay attacks by fraudsters. U.S. Patent Appl. Pub. No. 2014/0061302 ¶ 50. Although Hammad addresses a different physical card-verification-token architecture, it reflects the general technical understanding that replayed token-related messages can create transaction-security risks and that server-side state or database checks may be used to reduce those risks. Id. ¶¶ 50, 97.

48. Conventional email checkout also does not necessarily ensure that a token was generated by an authorized vendor. In a vendor-token architecture, the e-commerce system may receive an email message containing a token that purports to identify a vendor and a transaction. Unless the e-commerce system verifies that the token is associated with a vendor that is authorized to generate such tokens, the system may be unable to distinguish an authorized vendor-generated token from an unauthorized or fabricated token. This is a technical origin-authentication and token-integrity problem. In analogous token systems, RFC 6819 describes binding tokens to a particular resource server or client identifier and signing or encrypting token content as countermeasures that reduce replay impact and token abuse. Lodderstedt et al., RFC 6819 §§ 5.1.5.5, 5.1.5.8-5.1.5.10. Custer similarly describes a vendor receiving public and private keys, a vendor using a private key to generate a token, and a payment server decoding and authenticating the token before processing the transaction. U.S. Pat. No. 9,704,190 col. 2 ll. 15-54, col. 3 ll. 46-54, col. 5 ll. 5-65, Fig. 2. Custer also describes token parameters including amount, user data, and

expiration, confirming that persons working in this field understood tokens as machine-readable authentication objects that could carry transaction and validity information. Id. col. 5 ll. 5-65.

49. A further problem is that the sender address of an SMTP message does not, by itself, necessarily establish that the sender corresponds to the customer identified in the token. In an email-based transaction system, the e-commerce system may need to determine whether the email address from which the message was received matches the customer identity encoded or otherwise identified in the token. Without such a comparison, the system may process a message even though the communication-layer sender information and the token-encoded customer information are inconsistent. The technical need for binding a request to an authenticated state or identity is reflected in RFC 6819, which describes using a state parameter to link client requests to a user-agent authenticated state and to verify request validity by matching a binding value to that authenticated state. Lodderstedt et al., RFC 6819 § 5.3.5. In the email-checkout context, Custer similarly describes authenticating a second email using both the customer email address and the token. U.S. Pat. No. 9,704,190 col. 2 ll. 15-35.

50. In addition, conventional email checkout does not necessarily prevent the same transaction token from being replayed to cause duplicate or unauthorized transaction processing. This risk is particularly important in email-based transaction systems because a transaction email may be resent intentionally or inadvertently, forwarded to another user, retrieved from a prior message thread, queued and retried by SMTP infrastructure, or delivered more than once due to email-client or mail-server behavior. See Crocker, RFC 5598 §§ 2.2.2, 5; Klensin, RFC 5321 § 4.5.4.1; Shirey, RFC 4949, at 199, 245; Denning & Sacco, supra, at 533-36. If the e-commerce system lacks a mechanism for determining that a nonce, one-time-use

value, checksum, or other token-use indicator has already been recorded, the system may process the same transaction instruction multiple times.

51. In my opinion, these issues are not merely business problems associated with deciding whether to accept or reject a purchase. They are computer-network and transaction-security problems that arise from using SMTP messages and tokenized transaction instructions as part of an electronic payment-processing architecture. The relevant technical problem is how an e-commerce system should process an asynchronous email message containing machine-readable transaction data in a manner that verifies token freshness, prevents token replay, authenticates the customer indicated by the message and token, and confirms that the vendor associated with the token was authorized to generate the token before transaction execution occurs.

VI. Overview of the Claimed Technological Solution

52. The pending claims do not merely use email to conduct a purchase. In my opinion, a person of ordinary skill in the art would understand the claims to recite a specific ordered security architecture for processing SMTP-based e-commerce transaction messages. The claimed architecture uses the email message as a communication vehicle, but the security of the transaction does not arise merely from the use of email. Rather, the claimed architecture requires the e-commerce system to decode, validate, authenticate, record, and verify specific token and sender information before allowing transaction execution. The specification describes this type of architecture by explaining that the vendor server may include a token generator, that tokens may contain information used to perform or authenticate a transaction, and that the payment server may decode the token, authenticate the sender of the email, and process the transaction. Present Application ¶¶ [0017], [0023], [0027], [0048].

53. The claimed approach is directed to a technical problem that arises when SMTP email messages are used as transaction instructions. Because an SMTP message may be generated from a mailto link, delayed, resent, forwarded, copied, or received after the surrounding transaction context has changed, the e-commerce system must determine whether the received message is eligible for processing. The specification describes an email-based transaction workflow in which emails contain hyperlinks corresponding to vendor offers, selection of a mailto hyperlink causes a new email to contain a token holding identifiers of the associated offer and purchaser, and the email may then be sent to the email payment gateway for decoding and processing. Present Application ¶ [0007]. The claims address the security risks of that workflow by requiring the e-commerce system to perform multiple coordinated checks on the email message, the token contained in the email message, the sender of the email message, and the third-party vendor associated with the token.

54. As I understand the pending claims, the claimed architecture begins with an email message that is generated in response to activation of a mailto link. This is significant because the claimed system is not merely receiving arbitrary email correspondence. The received email message is part of a structured transaction workflow in which the mailto link causes an email message to be generated for communicating tokenized transaction information to the e-commerce system. The specification similarly describes generating an email message including a mailto hyperlink including the token, where the mailto hyperlink generates an email response message addressed to the payment server and including the token. Present Application ¶ [0008].

55. The claims further require that the token is generated for the mailto link. In my opinion, this ties the token to the specific link-driven transaction workflow.

The token is not merely a generic order number or human-readable purchase instruction. The token is a structured data object that the e-commerce system later decodes and validates to determine whether the SMTP message should be accepted for transaction processing. The specification explains that a token may be a sequence of byte data, string, or file used to authenticate a transaction, and may include encrypted strings, files, passwords, ciphers, or other data containing information used to perform or authenticate a transaction when sent to payment servers. Present Application ¶ [0017]. The specification further explains that the token generator may create a token to provide a purchase opportunity and that the generated token may be transmitted as part of an email message. Present Application ¶¶ [0053], [0074].

56. The token includes multiple pieces of technical and transaction-specific information. In particular, the token includes a vendor identifier, transaction data identifying at least one of a product identifier, a price, or a quantity, an expiration time, and a nonce or one-time-use value. These fields serve different technical purposes. The vendor identifier permits the e-commerce system to determine the vendor associated with the token. The transaction data permits the e-commerce system to determine the transaction context associated with the token. The expiration time permits the e-commerce system to determine whether the token is stale. The nonce or one-time-use value permits the e-commerce system to determine whether the token has already been used or replayed. The specification supports these types of token fields by describing token parameters including a partner ID, token type, card token, email address, amount, user data such as an item reference number or SKU, expiration date, and other identifying information. Present Application ¶¶ [0027]-[0043]. The specification further describes encrypted token contents including token type, card token, email address, transaction amount, unique partner

ID, signup URL, expiration date, group information, user data, and computer/browser identifiers. Present Application ¶¶ [0057]-[0067].

57. The claims then require the e-commerce system to decode the token to form a decoded token that identifies the third-party vendor and a customer email address. In my opinion, this decoding step is important because it causes the e-commerce system to obtain trusted transaction-context information from the token itself, rather than relying only on visible email-body text or SMTP header information. A person of ordinary skill in the art would understand that this decoding operation permits the e-commerce system to compare information derived from the token against other information associated with the received email message before the transaction is performed. The specification describes an e-commerce system including a token decoding unit configured to decode tokens and a message processing unit configured to process received email messages. Present Application ¶¶ [0050]-[0051]. The specification also explains that the e-commerce system's message processing unit receives an email containing an encoded token, decrypts the token into token information, confirms key validity, ensures that the vendor server is authorized to generate tokens, and processes a payment for the user with the given email address. Present Application ¶ [0068].

58. The claims further require validating the token by determining that the expiration time has not elapsed. This addresses the technical risk that an email message generated from a mailto link may be delayed, resent, or received after the token or underlying offer should no longer be honored. By requiring expiration validation, the claimed architecture causes the e-commerce system to determine whether the token is still temporally valid before transaction execution. The specification describes an expiration-date token parameter and explains that, after the expiration date, the token is no longer considered valid. Present Application ¶

[0064]. The specification also describes that tokens may be configured to expire by default after two weeks, after another predetermined time, or never, and that the vendor server may extend or shorten the expiration time of an offer associated with a token. Present Application ¶ [0044].

59. The claims also require determining, using a token-use datastore, that the nonce or one-time-use value has not previously been used. This is a replay-prevention mechanism. In my opinion, a person of ordinary skill in the art would understand that this step gives the e-commerce system stateful knowledge about prior token use. Without such a check, the e-commerce system could receive the same email message, or another message containing the same token, and be unable to determine whether the token represents a first valid use or a later replay. The specification describes this type of token-use state by explaining that the e-commerce system may be configured to store a checksum of a token after it has been used and to prevent the same token from being used more than once. Present Application ¶ [0070]. The specification also describes storing a group reference number for a used token and preventing the group reference number from being used more than a predetermined number of times. Id. In addition, the specification describes token groups in which members may be invalidated when one token is processed, thereby preventing additional processing of related tokens. Present Application ¶¶ [0044], [0065].

60. The claims further require recording use of the nonce or one-time-use value in the token-use datastore. This recording step is part of the same replay-prevention architecture. It updates the e-commerce system's stored token-use state so that later attempts to use the same nonce or one-time-use value may be identified and rejected. In my opinion, the datastore is not merely a recordkeeping component for business information. It is part of the security architecture because it enables the

e-commerce system to enforce one-time use of transaction tokens received through an asynchronous email channel. This is consistent with the specification's description that the e-commerce system stores a checksum after token use to prevent the same token from being used more than once and stores group-use information to limit use of related token groups. Present Application ¶ [0070].

61. The claims also require authenticating the sender of the email message as the customer by comparing the sender email address to the customer email address identified by the decoded token. This comparison binds information from two different sources: the communication-layer sender information associated with the received SMTP message and the customer identity information identified by the decoded token. In my opinion, this is a technical authentication step because it prevents the e-commerce system from relying solely on the apparent sender of the email message or solely on token content. Instead, the system determines whether those two sources of information are consistent before permitting the transaction to proceed. The specification supports this architecture by describing that the payment server decodes the token, authenticates the sender of the email, and may process the transaction. Present Application ¶ [0048]. The specification further explains that the e-commerce system may comprise an email processing system configured to verify an email address associated with the sender of an email, which permits the payment server to verify that the payload was intended for use by the recipient because the email is included in the decrypted payload message. Present Application ¶ [0072].

62. The claims further require determining that the third-party vendor is authorized to generate the token by verifying a token-authenticator bound to the token. The token-authenticator may include, for example, a cryptographic signature, a message authentication code, or an encrypted token portion that is verifiable using vendor authorization data stored by the e-commerce system. In my opinion, this step

addresses a separate technical security problem: whether the token was generated by, or on behalf of, an authorized vendor. By verifying a token-authenticator that is bound to the token, the e-commerce system can determine whether the token should be trusted as vendor-authorized transaction data. The specification describes that token generation may involve public-key cryptography, including encryption, authentication using Poly1305 MAC, and public keys using Curve25519, and further describes that the vendor server may receive a unique private and public key associated with the vendor and may generate a token using its unique key. Present Application ¶¶ [0054], [0056]. The specification also explains that the e-commerce system decrypts the encoded token, confirms key validity, and ensures that the vendor server is authorized to generate tokens before processing a payment. Present Application ¶ [0068]. The specification further describes rejecting encrypted transaction tokens after revoking signing keys when a vendor key is compromised. Present Application ¶ [0069].

63. The claimed architecture therefore uses multiple different checks that address multiple different technical risks. The expiration time addresses stale-token risk. The nonce or one-time-use value and token-use datastore address replay risk. The sender-email comparison addresses customer-authentication risk. The token-authenticator verification addresses unauthorized-vendor and token-integrity risk. These checks operate together before the transaction is performed. The specification describes the same type of coordinated architecture by disclosing token generation by the vendor server, token transmission by email or other electronic messaging, token decoding by the e-commerce system, sender email verification, vendor-authorization verification, checksum-based prevention of token reuse, and transaction processing only after the required validation steps are satisfied. Present Application ¶¶ [0050]-[0056], [0068]-[0072], [0074], [0078].

64. The claims also require that the e-commerce transaction is performed only when sender authentication is successful, token validation is successful, and vendor-authorization verification is successful. This conditional transaction execution is important. In my opinion, a person of ordinary skill in the art would understand that the claimed e-commerce system is not merely processing a purchase request. It is enforcing an ordered security protocol in which transaction execution is gated by multiple technical validations directed to the received SMTP message and token. The specification describes that, when a transaction is attempted, the payment server decodes the token, authenticates the sender of the email, and may process the transaction. Present Application ¶ [0048]. It also describes that the e-commerce system receives an email containing an encoded token, decrypts the token, confirms key validity, ensures vendor authorization, and processes a payment for the user with the given email address. Present Application ¶ [0068].

65. The ordered nature of the claimed architecture is also significant. The system receives an email message generated from a mailto link, obtains a token generated for that link, decodes the token, validates token freshness and prior-use state, records token use, authenticates the sender against decoded token information, verifies that the vendor is authorized to generate the token, and only then performs the e-commerce transaction. In my opinion, this ordered combination changes how the e-commerce system processes SMTP-based transaction messages. The system does not treat an email message as sufficient simply because it contains purchase information. Instead, the system treats the email message as a candidate transaction instruction that must satisfy a defined set of token-security and identity-verification checks before execution. The specification supports this understanding by describing that the system receives an email containing an encoded token, decodes and validates token contents, verifies sender and vendor-related authorization information,

processes the transaction, and sends notice of a successful transaction. Present Application ¶¶ [0007]-[0008], [0048], [0068]-[0072], [0074].

66. In my opinion, this ordered combination provides a specific transaction-security protocol for SMTP-based e-commerce messages. The claimed protocol addresses risks that arise from using asynchronous email messages as transaction instructions, including stale messages, replayed tokens, sender mismatch, and unauthorized vendor-generated tokens. A person of ordinary skill in the art would understand the claimed architecture as an improvement to secure electronic transaction processing because it adds technical controls for determining whether an SMTP-based transaction message should be trusted and processed by the e-commerce system. The specification describes that the disclosed system uses a secure and backwards-compatible process, provides the vendor server with improved security by embedding more functionality in the vendor server, and reduces bandwidth by streamlining communications with the e-commerce system. Present Application ¶ [0052].

VII. How a Person of Ordinary Skill Would Understand the Specification

67. A person of ordinary skill in the art would understand the specification to describe more than an ordinary commercial checkout process. The specification does not merely describe a customer deciding to purchase an item through email. Rather, the specification describes a technical architecture for generating, transmitting, decoding, authenticating, and processing machine-readable transaction tokens in connection with email-based and web-based e-commerce transactions. In my opinion, a person of ordinary skill would understand the specification as addressing how an e-commerce system should technically evaluate a tokenized transaction instruction before the system permits a payment transaction to be executed.

68. The specification describes an architecture in which functionality is divided between a vendor-side system and an e-commerce or payment system. The vendor server includes, among other components, a token generator, a vendor email server, and a vendor website. Present Application ¶¶ [0023], [0050]. The e-commerce system includes, among other components, a token decoding unit, an interface unit, a database module, a message processing unit, an order execution unit, and an account management unit. Present Application ¶¶ [0050]-[0051]. A person of ordinary skill would understand from this architecture that token generation, token delivery, token decoding, message processing, and transaction execution are separate technical functions distributed across cooperating networked systems.

69. The specification further explains that the architecture enables vendors to generate two-click tokens within their own vendor server without requiring communication with the e-commerce system at the time the tokens are created. Present Application ¶ [0050]. In my opinion, that disclosure would have conveyed to a person of ordinary skill that the vendor-side token generator is not merely a user-interface convenience. It changes the architecture of the email-based payment system by moving token-generation functionality to the vendor server while preserving later e-commerce-system validation and transaction processing. The specification expressly describes this as a secure and backwards-compatible process that provides improved security by embedding more functionality in the vendor server and reduces bandwidth by streamlining communications with the e-commerce system. Present Application ¶ [0052].

70. A person of ordinary skill would also understand the disclosed token to function as a machine-readable transaction-authentication object, not merely as a label, order number, or business identifier. The specification defines a token as a sequence of byte data, string, or file used to authenticate a transaction. It further

explains that a token may include one or more encrypted strings, files, passwords, ciphers, or other data containing information used to perform or authenticate a transaction when sent to payment servers. Present Application ¶ [0017]. The specification similarly states that tokens may be encrypted strings containing information to perform a transaction when sent to payment servers and may contain information used to perform or authenticate a transaction. Present Application ¶ [0027]. Those disclosures would have informed a person of ordinary skill that the token itself is part of the transaction-security mechanism.

71. The specification describes token contents that bind transaction information to vendor, customer, and security-related information. For example, the token parameters may include a private key, public key, partner ID, environment, token type, card token, email address, signup URL, amount, user data, expiration information, user-agent information, accept-language information, accept-charset information, and IP-address information. Present Application ¶¶ [0028]-[0043]. The specification also describes encrypted token contents including token type, card token, email address for the recipient, transaction amount, unique partner ID, signup URL, expiration date, group information, user data, and computer/browser identifiers. Present Application ¶¶ [0057]-[0067]. In my opinion, a person of ordinary skill would understand these disclosures to describe a token that carries machine-readable information used by the receiving e-commerce system to determine whether the transaction instruction is valid and should be processed.

72. The specification's discussion of email-based checkout further confirms that the token is used as a transaction-authentication object. The specification explains that the vendor server sends emails with tokens to customers, that the emails contain hyperlinks corresponding to vendor offers, and that when the mailto hyperlink is selected, the new email contains a token holding identifiers of

the associated offer and purchaser. Present Application ¶ [0007]. The email may then be sent to the email payment gateway, which receives the email, decodes the contents, and processes the transaction. Id. A person of ordinary skill would understand this disclosure to describe an email message carrying a tokenized transaction instruction that the e-commerce system must decode and evaluate before transaction execution.

73. A person of ordinary skill would also understand that the token binds transaction information to vendor and customer information so that the e-commerce system can later determine whether a received SMTP message is eligible for transaction processing. The token may include an amount, user data, a card token, a recipient email address, a partner ID, and an expiration date. Present Application ¶¶ [0034]-[0039], [0060]-[0064]. The e-commerce system's message processing unit receives an email containing an encoded token, decrypts the token into token information, confirms the key's validity, ensures that the vendor server is authorized to generate tokens, and processes a payment for the user with the given email address. Present Application ¶ [0068]. In my opinion, that disclosure would inform a person of ordinary skill that the e-commerce system uses decoded token information to evaluate whether the received message corresponds to a valid vendor-authorized transaction for a particular customer and transaction context.

74. The specification also describes technical controls for determining whether a token remains valid. For example, the token may include an expiration date, after which the token is no longer considered valid. Present Application ¶ [0064]. The specification further explains that a token may be configured to expire after a default period, after another predetermined time, or never, and that the vendor server may extend or shorten the expiration time of a particular offer associated with a token without resending an email or generating a new token. Present Application

¶ [0044]. A person of ordinary skill would understand these disclosures as addressing the technical problem of stale transaction tokens in email-based transaction systems, where an email message may remain available to the customer after the underlying authorization context changes.

75. A person of ordinary skill would further understand the claimed token-authenticator to provide cryptographic or equivalent technical assurance that the token is associated with a vendor authorized to generate transaction tokens. The specification explains that token generation may involve public-key cryptography, including encryption using an XSalsa20 stream cipher, authentication using Poly1305 MAC, and public keys using Curve25519 high-speed elliptic-curve cryptography. Present Application ¶ [0054]. The specification also explains that the vendor server may receive a unique private and public key associated with the vendor, and a public key that the e-commerce system uses for multiple partners. Present Application ¶ [0056]. The vendor server may then generate a token using its unique key and provide the e-commerce system's public key as the intended target for messages. Id.

76. In my opinion, a person of ordinary skill would understand those disclosures to mean that the token is not trusted merely because it appears in an email message. Instead, the e-commerce system evaluates cryptographic or equivalent authentication information associated with the token to determine whether the token should be trusted. This understanding is confirmed by the specification's disclosure that the e-commerce system decrypts the token information, confirms the key's validity, ensures that the vendor server is authorized to generate tokens, and then processes payment. Present Application ¶ [0068]. The specification also describes key-revocation functionality. If a vendor key is compromised, the e-commerce system may reject encrypted transaction tokens after revoking the signing keys.

Present Application ¶ [0069]. A person of ordinary skill would understand this disclosure as further confirming that vendor authorization and token authenticity are technical prerequisites to trusted transaction processing.

77. A person of ordinary skill would also understand the claimed nonce or one-time-use value and token-use datastore to provide a replay-prevention mechanism specifically adapted for asynchronous SMTP transaction messages. The specification does not need to use the word "nonce" for a person of ordinary skill to understand the disclosed one-time-use functionality. The specification explains that the e-commerce system may be configured to store a checksum of a token after it has been used and to prevent the same token from being used more than once. Present Application ¶ [0070]. The specification further explains that the e-commerce system may store a group reference number for a used token and prevent the group reference number from being used more than a predetermined number of times. Id. A person of ordinary skill would understand these features as providing token-use state that permits the e-commerce system to detect and prevent repeated use of the same token or related token group.

78. That replay-prevention functionality is particularly significant in the SMTP context. The specification describes email-based checkout in which a generated token may be transmitted as part of an email message and a customer may send a response email with the token embedded. Present Application ¶¶ [0053], [0074]. A person of ordinary skill would understand that such email messages may be retained, resent, or otherwise presented to the e-commerce system more than once. In view of that context, the disclosed checksum storage, group reference number storage, and prevention of multiple use would be understood as technical controls that protect the e-commerce system against repeated processing of the same tokenized transaction instruction.

79. The specification also describes sender verification as part of the same technical architecture. The payment server decodes the token, authenticates the sender of the email, and may process the transaction. Present Application ¶ [0048]. The e-commerce system may include an email processing system configured to verify an email address associated with the sender of an email, which permits the payment server to verify that the payload was intended for use by the recipient because the email is included in the decrypted payload message. Present Application ¶ [0072]. A person of ordinary skill would understand this disclosure as binding the SMTP sender information to customer information carried in the decrypted token payload.

80. In view of these disclosures, a person of ordinary skill would understand the specification to describe a technical transaction-processing architecture. The vendor-side token generator creates machine-readable encrypted token payloads containing transaction, customer, vendor, expiration, and security-related information. The token is transmitted through email or another electronic messaging channel. The e-commerce system receives the message, decodes the token, verifies token and key validity, confirms that the vendor is authorized to generate tokens, verifies sender/customer information, checks token-use state to prevent reuse, and then processes the transaction if the required checks are satisfied. Present Application ¶¶ [0007]-[0008], [0017], [0048], [0050]-[0056], [0068]-[0072], [0074].

81. Accordingly, in my opinion, a person of ordinary skill would not read the specification as describing only a commercial checkout idea. A person of ordinary skill would understand the specification to describe an improvement to secure electronic transaction processing in which the e-commerce system uses tokenized, cryptographically verifiable, statefully tracked transaction data to

determine whether an asynchronous SMTP-based transaction message should be trusted and processed.

VIII. Technological Improvement: Replay Prevention in SMTP-Based Transaction Processing

82. Replay attacks are a recognized technical risk in networked computer systems that process machine-readable instructions. In a replay attack, a valid data transmission or valid message is repeated so that a receiving system treats the repeated transmission as a new or valid instruction. As explained in RFC 4949, a replay attack includes an attack in which a valid data transmission is maliciously or fraudulently repeated, either by the originator or by a third party who intercepts and retransmits the data. R. Shirey, RFC 4949, Internet Security Glossary, Version 2, at 245 (Internet Eng'g Task Force Aug. 2007). RFC 4949 also explains that "liveness" provides assurance that data is freshly transmitted and not replayed from a previous transmission. Id. at 163. A person of ordinary skill in the art would have understood that a transaction-processing system receiving machine-readable transaction data must be able to distinguish a current, authorized instruction from a previously used instruction that is being presented again.

83. SMTP-based transaction systems are particularly susceptible to stale-message and duplicate-message risks because SMTP email is not a synchronous request-response protocol in the same way as a live checkout session. Internet mail is an asynchronous store-and-forward communication system. RFC 5598 describes Internet mail transfer as using an "asynchronous store-and-forward communication infrastructure." D. Crocker, RFC 5598, Internet Mail Architecture § 5 (Internet Eng'g Task Force July 2009). RFC 5321 likewise explains that, when delivery cannot be completed immediately, the SMTP client retains responsibility for delivery and may requeue the message for a later delivery attempt. J. Klensin, RFC

5321, Simple Mail Transfer Protocol §§ 3, 4.5.4.1 (Internet Eng'g Task Force Oct. 2008). A person of ordinary skill would therefore understand that an email message used as a transaction instruction may be generated at one time, delivered at a later time, retained in a mailbox, resent, forwarded, copied, or otherwise presented to the e-commerce system more than once.

84. That persistence and retransmission risk matters when the email message contains a tokenized transaction instruction. If the e-commerce system simply processes a token because the token appears in a received SMTP message, the system may be unable to determine whether the token represents a first authorized use or a repeated use of an earlier authorization. In an email-based checkout environment, that can result in a duplicate transaction, an unauthorized repeated charge, or processing of a stale transaction context. The risk is not limited to intentional fraud. A duplicate or stale message may also result from normal email-client behavior, user forwarding, user resending, mailbox retention, or delayed message delivery.

85. The claimed nonce or one-time-use value addresses this technical problem by making each token use traceable at the e-commerce system. In my opinion, a person of ordinary skill would understand a nonce or one-time-use value in this context as a data value that permits the receiving system to determine whether a token has already been used. RFC 4949 defines a nonce as a random or non-repeating value included in protocol data, usually to guarantee liveness and detect and protect against replay attacks. Shirey, RFC 4949, at 199. Similarly, RFC 6819 recognizes limiting token usage or enforcing one-time token usage as a countermeasure against replay of tokens. T. Lodderstedt et al., RFC 6819, OAuth 2.0 Threat Model and Security Considerations § 5.1.5.4 (Internet Eng'g Task Force Jan. 2013).

86. The specification describes the same replay-prevention concept in the context of the disclosed e-commerce system. The specification states that the e-commerce system may be configured to store a checksum of a token after it has been used and to prevent the same token from being used more than once. Present Application ¶ [0070]. A person of ordinary skill would understand that this is not merely storing historical business information. Rather, the stored checksum provides the e-commerce system with persistent token-use state that can be checked when a later message containing the same token, or a token corresponding to the same stored checksum, is received.

87. The specification further describes group-based token-use state. In particular, the specification explains that the e-commerce system may store a group reference number for a used token and prevent the group reference number from being used more than a predetermined number of times. Present Application ¶ [0070]. The specification also explains that the vendor server may generate groups of token values that automatically invalidate members of the group when one token is processed. Present Application ¶ [0044]. The specification further identifies "group" as encrypted token information, explaining that tokens with a group equal to a given value may not be considered valid after execution of a single member. Present Application ¶ [0065]. A person of ordinary skill would understand these disclosures as state-based mechanisms for controlling whether the same token or a related group of tokens remains eligible for later transaction processing.

88. The token-use datastore therefore provides the e-commerce system with persistent state for determining whether the token, nonce, one-time-use value, checksum, or related group-use indicator has already been used. This is important because SMTP messages themselves do not inherently provide the e-commerce system with reliable one-time-use state. The same email message or tokenized

payload may appear again after it was previously processed. By maintaining a token-use datastore, the e-commerce system can determine whether a later-received transaction message corresponds to a previously used authorization.

89. In my opinion, a person of ordinary skill would understand the claimed token-use datastore as part of the e-commerce system's security architecture. The datastore changes how the e-commerce system processes incoming SMTP-based transaction messages. The system does not simply decode a token and process the transaction if the token contains product or payment information. Instead, the system checks stored token-use state to determine whether the token, nonce, or one-time-use value is still eligible for use. That stored state permits the e-commerce system to reject a replayed or duplicate tokenized transaction instruction.

90. Recording use of the nonce or one-time-use value after successful validation and before transaction execution further improves the security of the transaction-processing architecture. In that ordering, the system first determines that the token is facially valid and has not previously been used. The system then records use of the nonce or one-time-use value in the token-use datastore before the transaction is performed. This prevents a second message containing the same token or same one-time-use value from being treated as a new transaction instruction. In my opinion, that ordering is significant because it closes the window in which the same token could be accepted more than once.

91. This recording step also addresses the particular characteristics of email-based transaction processing. Because SMTP messages may be delayed, duplicated, resent, or forwarded, an e-commerce system may receive the same tokenized transaction instruction more than once. A persistent record of token use gives the system a memory of prior processing that is independent of the contents of the current email message. Once the system records the token-use value, a later-

received message containing the same token-use value may be identified as a replay or duplicate transaction attempt.

92. In my opinion, this improves the security of the e-commerce system itself. The improvement is not merely that a human or business decides that a customer should be charged only once. The improvement is that the e-commerce system is configured to maintain and apply token-use state when processing machine-readable transaction instructions received through SMTP. The system changes from a stateless or message-content-only processing model to a stateful validation model in which prior token use affects whether a later SMTP message is eligible for transaction execution.

93. This is not merely a business rule. A business rule might state, in general terms, that a customer should not be charged twice. The claimed replay-prevention architecture provides a technical control for enforcing that result in a networked e-commerce system. The e-commerce system uses token-encoded information, a nonce or one-time-use value, and a token-use datastore to make an automated determination regarding whether a received SMTP message contains a token that has already been used. That determination changes the information flow through the e-commerce system: the incoming email message is not routed directly to transaction execution, but instead is routed through token validation, token-use lookup, token-use recording, and only then conditional transaction execution.

94. A person of ordinary skill would also understand that the replay-prevention mechanism cooperates with other claimed security features. Expiration validation addresses stale-token risk. Sender authentication addresses customer-identity risk. Vendor-authenticator verification addresses unauthorized-token-generation risk. The token-use datastore addresses replay and duplicate-processing risk. Together, these features provide an ordered security architecture for SMTP-

based transaction processing. The replay-prevention component is a technological improvement because it gives the e-commerce system persistent state and one-time-use enforcement for tokenized transaction instructions received through an asynchronous email channel.

95. Accordingly, in my opinion, the claimed nonce or one-time-use value and token-use datastore provide a technical improvement to SMTP-based transaction processing. They allow the e-commerce system to determine whether a tokenized transaction instruction is fresh and unused before transaction execution. They also prevent the same token from being reused to trigger a duplicate or unauthorized transaction. A person of ordinary skill would understand this as an improvement to the security and reliability of the e-commerce system's transaction-processing architecture, not merely as an implementation of a commercial policy.

IX. Technological Improvement: Vendor-Authorization Verification

96. In a vendor-token architecture, a technical problem arises if the e-commerce system cannot distinguish tokens generated by authorized vendors from tokens generated by unauthorized systems. In such an architecture, the e-commerce system receives a machine-readable token that purports to represent a transaction involving a particular vendor, customer, and transaction context. If the e-commerce system accepts the token merely because it appears in an email message or because the token contains transaction information, then an unauthorized system could potentially generate or modify tokenized transaction data and cause the e-commerce system to process an unauthorized transaction.

97. A person of ordinary skill in the art would have understood this as a token-origin and token-integrity problem. The problem is not simply whether a vendor and customer have a commercial relationship. Rather, the technical problem is whether the receiving e-commerce system can determine that the token was

generated by a vendor system that the e-commerce system recognizes and has authorized to generate transaction tokens. In the absence of such a verification mechanism, the e-commerce system would lack a reliable technical basis for distinguishing authorized vendor-generated tokens from fabricated, altered, or unauthorized tokens.

98. The claimed token-authenticator addresses this problem by binding an authenticator to the token and associating that authenticator with the third-party vendor. In my opinion, a person of ordinary skill would understand this to mean that the token is not simply a passive container for business information. Instead, the token includes, or is associated with, technical authentication information that the e-commerce system can verify before deciding whether to trust the token. That authenticator may take the form of a cryptographic signature, a message authentication code, or an encrypted token portion that is verifiable using vendor authorization data stored by the e-commerce system.

99. The specification supports this understanding. It explains that token generation by the vendor server may involve public-key cryptography, including encryption using an XSalsa20 stream cipher, authentication using Poly1305 MAC, and public keys using Curve25519 high-speed elliptic-curve cryptography. Present Application ¶ [0054]. A person of ordinary skill would understand those disclosures to describe cryptographic techniques that can be used to protect token confidentiality, token integrity, and token authenticity. In particular, a message authentication code or cryptographic signature provides technical evidence that the token was generated using appropriate cryptographic material and has not been altered in a way that escapes detection by the e-commerce system.

100. The specification further explains that the vendor server may be configured with keys provided by the e-commerce system. The vendor server

receives a unique private and public key associated with the vendor, and a public key that the e-commerce system uses for multiple partners. The vendor server may then generate a token with its unique key and provide the e-commerce system's public key as the intended target for messages. Present Application ¶ [0056]. In my opinion, a person of ordinary skill would understand this keying architecture as establishing vendor-specific authorization data that can later be used by the e-commerce system to determine whether a received token is associated with an authorized vendor.

101. The e-commerce system's verification function is also described in the specification. The specification states that the e-commerce system's message processing unit receives an email containing an encoded token, decrypts the key into token information, confirms the key's validity, ensures that the vendor server is authorized to generate tokens, and processes a payment for the user with the given email address. Present Application ¶ [0068]. A person of ordinary skill would understand that sequence as a technical verification process, not merely a business approval step. The e-commerce system obtains token information, evaluates cryptographic validity or key validity, determines whether the vendor server is authorized to generate tokens, and only then processes the payment.

102. The claimed vendor-authorization verification therefore improves security by changing how the e-commerce system handles tokenized transaction instructions. Instead of treating the token as valid solely because it identifies a vendor or transaction, the e-commerce system verifies a token-authenticator using vendor authorization data stored by the e-commerce system. This permits the e-commerce system to determine whether the token is associated with a vendor that the system recognizes as authorized to generate transaction tokens.

103. A cryptographic signature, message authentication code, or encrypted token portion provides technical evidence that the token originated from, or was authorized by, a recognized vendor. A signature or MAC can be verified against stored vendor-specific authorization data. An encrypted token portion can be decrypted or otherwise verified using key material associated with the vendor and the e-commerce system. In either case, the e-commerce system is not relying solely on visible token fields, visible email-body text, or SMTP sender information. The e-commerce system is instead using cryptographic or equivalent technical verification to determine whether the token should be trusted.

104. The specification's key-revocation discussion further demonstrates that the vendor-authorization mechanism is a technical security feature of the e-commerce system. The specification explains that, if a keyset is compromised, the e-commerce system may disable the compromised keyset and generate another. If a vendor key is compromised, the e-commerce system may reject encrypted transaction tokens after revoking the signing keys. Present Application ¶ [0069]. A person of ordinary skill would understand that this capability is part of a technical security architecture. The e-commerce system can change its stored authorization state so that tokens associated with compromised or revoked vendor signing keys are no longer accepted.

105. This key-revocation functionality is important because it confirms that vendor authorization is not merely a static business designation. It is a technical trust relationship enforced by the e-commerce system through stored authorization data, key validity checks, and rejection of tokens associated with compromised or revoked keys. A person of ordinary skill would understand that this architecture permits the e-commerce system to respond to security events by modifying which vendor-generated tokens are accepted for transaction processing.

106. In my opinion, the claimed vendor-authorization verification improves the security of the e-commerce system by preventing unauthorized token generation from resulting in transaction execution. The system does not simply ask whether the token identifies a known vendor. The system verifies whether the token includes or is bound to an authenticator associated with that vendor and whether that authenticator is verifiable using vendor authorization data stored by the e-commerce system. If the authenticator cannot be verified, the token is not trusted and the transaction is not performed.

107. This verification also works together with the other claimed security controls. Sender authentication addresses whether the SMTP sender corresponds to the customer identified by the decoded token. Token validation addresses whether the token is fresh and unused. Vendor-authorization verification addresses whether the token itself was generated by an authorized vendor. These checks address different security risks and operate together before the e-commerce transaction is performed.

108. A person of ordinary skill would therefore understand the claimed vendor-authorization verification as improving the transaction-processing architecture itself. The improvement lies in how the e-commerce system evaluates a received SMTP-based transaction message before execution. The system uses vendor authorization data, cryptographic or equivalent token-authenticator verification, and key-validity determinations to decide whether the tokenized instruction should be accepted. This changes the information flow through the system: a received email message containing a token is not routed directly to payment processing; it is first routed through vendor-authenticator verification and other security checks.

109. Accordingly, in my opinion, the claims do not merely improve or define the commercial relationship between a vendor and a customer. They improve the technical architecture by which an e-commerce system receives, evaluates, and conditionally processes tokenized transaction instructions. The vendor-authorization verification mechanism provides the e-commerce system with a technical control for rejecting unauthorized or untrusted vendor tokens before transaction execution. A person of ordinary skill would understand this as an improvement to secure electronic transaction processing.

X. Technological Improvement: Sender Authentication Based on Decoded Token Information

110. SMTP sender information alone may not be sufficient to determine whether a transaction email should be processed. In an email-based transaction architecture, an e-commerce system may receive an email message that appears to originate from a particular email address. However, the apparent sender address, standing alone, does not necessarily establish that the sender is the customer for whom the tokenized transaction instruction was generated. SMTP messages may be generated by different email clients, forwarded, resent, modified, or otherwise transmitted in ways that may cause the apparent sender information and the intended customer authorization information to diverge.

111. This is a technical problem in SMTP-based transaction processing. The issue is not merely whether a customer wishes to purchase a product. The issue is whether the e-commerce system can reliably determine, before executing a transaction, that the received email message corresponds to the customer identity encoded in the transaction token. A person of ordinary skill in the art would understand that a system that processes transaction instructions received by email should not rely solely on the fact that an email message was received from some

address. The system should determine whether the sender information associated with the received SMTP message is consistent with the customer information contained in the tokenized transaction payload.

112. The claimed system addresses this problem by requiring the e-commerce system to decode the token and identify a customer email address from the decoded token. The e-commerce system then authenticates the sender of the email message as the customer by determining that the email address of the sender matches the customer email address identified by the decoded token. In my opinion, this is an important technical step because the e-commerce system uses decoded token content as an authentication reference for evaluating the communication-layer sender information.

113. The specification supports this understanding. It explains that a customer wishing to use the payment server may register his or her email address and payment information with the payment server. Present Application ¶ [0048]. The specification further explains that, when a transaction is attempted, the payment server decodes the token, authenticates the sender of the email, and may process the transaction. Id. A person of ordinary skill would understand this disclosure as describing a technical process in which sender authentication is performed by the payment server before the transaction is processed.

114. The specification also explains that the e-commerce system receives an email containing an encoded token, decrypts the token into token information, confirms the key's validity, ensures that the vendor server is authorized to generate tokens, and processes a payment for the user with the given email address. Present Application ¶ [0068]. In my opinion, this disclosure would inform a person of ordinary skill that the e-commerce system uses information recovered from the token

payload, including user-identifying information, as part of determining whether the transaction should be processed.

115. The specification further explains that the e-commerce system may comprise an email processing system configured to verify an email address associated with the sender of an email. Present Application ¶ [0072]. The specification states that this may permit the payment server to verify that the payload was indicated for use by the recipient because the email is included in the decrypted payload message. Id. A person of ordinary skill would understand this disclosure as describing a comparison between the sender email information associated with the received email message and the email information included in the decrypted token payload.

116. The claimed sender-authentication step therefore binds two different layers of information. The first layer is the communication-layer information associated with the received SMTP message, including the email address from which the message was received. The second layer is the token-encoded transaction-authorization information obtained by decoding or decrypting the token, including the customer email address identified by the decoded token. By comparing those two pieces of information, the e-commerce system determines whether the received SMTP message corresponds to the intended customer for the tokenized transaction instruction.

117. In my opinion, this comparison is a technical authentication step. The e-commerce system is not merely reading a customer name or recording a purchase preference. The system is evaluating whether the message sender information and the token-encoded customer information are consistent before transaction execution. If they are not consistent, the system can determine that the email message should

not be treated as an authorized transaction instruction for the customer identified in the token.

118. This improves the security of SMTP-based transaction processing because it prevents the e-commerce system from treating any received email containing a token as sufficient for transaction execution. Instead, the system must determine that the sender email address corresponds to the customer email address identified by the decoded token. In an email-based transaction system, that comparison helps address risks associated with forwarded messages, copied messages, resent messages, or messages sent from an email address other than the intended customer's address.

119. This sender-token comparison also works together with the other claimed security checks. Token validation determines whether the token is fresh and unused. Vendor-authorization verification determines whether the token was generated by an authorized vendor. Sender authentication determines whether the email message was received from the customer identified by the token. The transaction is performed only when those checks are successful. In my opinion, a person of ordinary skill would understand that the sender-authentication step is part of the ordered security architecture that determines whether an SMTP-based transaction message should be trusted.

120. The claimed system therefore does not rely merely on receiving an email from any address. Nor does it rely merely on token content in isolation. Instead, the e-commerce system decodes the token to identify customer information and compares that token-derived customer information with the sender email address associated with the received message. This comparison changes how the e-commerce system processes SMTP-based transaction messages by requiring sender-token consistency before transaction execution.

121. Accordingly, in my opinion, a person of ordinary skill would understand the claimed sender authentication based on decoded token information as a technological improvement to secure electronic transaction processing. It provides a technical control for determining whether a received SMTP message corresponds to the intended customer authorization encoded in the token, and it prevents transaction execution unless that authentication requirement is satisfied.

XI. Ordered Combination and Practical Application

122. In my opinion, the claimed security features should not be viewed as isolated or unrelated steps. A person of ordinary skill in the art would understand that the claimed validation steps operate in coordination with one another to determine whether an SMTP-based transaction message should be trusted and processed by the e-commerce system. Each step addresses a different technical risk that arises when a machine-readable transaction token is transported through an asynchronous email channel.

123. The specification describes a distributed architecture in which token generation may occur at the vendor server, while token decoding, message processing, database operations, order execution, and account management may occur at the e-commerce system. Present Application ¶¶ [0050]-[0051]. The specification also explains that this architecture permits vendors to generate two-click tokens within their own vendor server without requiring communication with the e-commerce system at the time the tokens are created. Present Application ¶ [0050]. A person of ordinary skill would understand that this architecture requires later validation by the e-commerce system because the token may be generated outside the e-commerce system and later received through email or another electronic messaging channel.

124. The first aspect of the ordered architecture is token generation and transmission. The specification explains that the vendor server may generate an email offer, that the token generator may create a token to provide a purchase opportunity, and that the generated token may be transmitted as part of an email message. Present Application ¶¶ [0053], [0074]. The specification further explains that emails contain hyperlinks corresponding to offers from the vendor, and when the mailto hyperlink is selected, the new email contains a token that holds identifiers of the associated offer and purchaser. Present Application ¶ [0007]. Thus, a person of ordinary skill would understand that the email message is not simply ordinary correspondence. It carries a machine-readable tokenized transaction instruction.

125. Once the e-commerce system receives the tokenized message, the ordered validation architecture begins. The e-commerce system receives an email containing an encoded token, decrypts the token information, confirms key validity, ensures that the vendor server is authorized to generate tokens, and processes a payment for the user with the given email address. Present Application ¶ [0068]. In my opinion, a person of ordinary skill would understand this disclosure as describing a coordinated security sequence in which transaction execution follows token decoding, key validation, vendor authorization, and user identification.

126. Expiration validation addresses stale-token risk. An email message generated from a mailto link may remain available to the user after the underlying offer or authorization context has changed. The specification addresses this issue by identifying an expiration-date token parameter and explaining that, after the expiration date, the token is no longer considered valid. Present Application ¶ [0064]. The specification also explains that the token may be configured to expire after a default time, another predetermined time, or never, and that the vendor server may extend or shorten the expiration time of a particular offer associated with a

token. Present Application ¶ [0044]. A person of ordinary skill would understand expiration validation as a technical control that determines whether the received token is still temporally eligible for processing.

127. Nonce or one-time-use validation addresses replayed-token risk. In this context, a replayed token is a tokenized transaction instruction that is presented to the e-commerce system after it has already been used or after a related one-time-use authorization has already been consumed. The specification supports this functionality by explaining that the e-commerce system may store a checksum of a token after it has been used and prevent the same token from being used more than once. Present Application ¶ [0070]. A person of ordinary skill would understand this as a mechanism for determining whether the received token, or a value corresponding to that token, has already been used.

128. Token-use recording creates persistent state for future replay prevention. Without persistent token-use state, the e-commerce system would be limited to the contents of the current email message and token. The system would not have a reliable technical basis for determining whether the same tokenized transaction instruction had previously been received and processed. The specification addresses that problem by disclosing that the e-commerce system may store a checksum of a token after use and may store a group reference number for a used token to prevent that group reference number from being used more than a predetermined number of times. Present Application ¶ [0070]. The specification also explains that token groups may be configured so that other members of the group are invalidated when one token is processed. Present Application ¶¶ [0044], [0065]. A person of ordinary skill would understand these disclosures to describe stateful replay prevention rather than merely a business record of prior transactions.

129. Sender authentication binds the SMTP message to the customer identity encoded in the token. The specification explains that, when a transaction is attempted, the payment server decodes the token, authenticates the sender of the email, and may process the transaction. Present Application ¶ [0048]. The specification further explains that the e-commerce system may include an email processing system configured to verify an email address associated with the sender of an email, which permits the payment server to verify that the payload was intended for use by the recipient because the email is included in the decrypted payload message. Present Application ¶ [0072]. A person of ordinary skill would understand that this comparison binds communication-layer sender information to token-encoded customer information before transaction execution.

130. Vendor-authenticator verification binds the token to an authorized vendor. The specification explains that token generation may involve public-key cryptography, including encryption, authentication using Poly1305 MAC, and public keys using Curve25519 high-speed elliptic-curve cryptography. Present Application ¶ [0054]. The specification further explains that the vendor server may receive a unique private and public key associated with the vendor and may generate a token using its unique key. Present Application ¶ [0056]. The e-commerce system then confirms key validity and ensures that the vendor server is authorized to generate tokens before processing payment. Present Application ¶ [0068]. A person of ordinary skill would understand this as a technical trust mechanism that prevents the e-commerce system from accepting tokenized transaction instructions unless the token is associated with an authorized vendor.

131. These validation steps operate together. Expiration validation determines whether the token is stale. One-time-use validation determines whether the token has already been used. Token-use recording updates persistent state so that

future duplicate or replayed attempts may be detected. Sender authentication determines whether the email sender corresponds to the customer identity encoded in the token. Vendor-authenticator verification determines whether the token was generated by a vendor authorized by the e-commerce system. In my opinion, a person of ordinary skill would understand that each of these checks addresses a different failure mode in SMTP-based transaction processing.

132. The claims also require that transaction execution occurs only when the required checks succeed. That ordering is important. The e-commerce system does not perform the transaction merely because an email message was received or because the message contains transaction data. Instead, the e-commerce system treats the email as a candidate transaction instruction and routes it through a series of token-security, sender-authentication, and vendor-authorization checks. Only after those checks are satisfied does the system perform the e-commerce transaction.

133. This ordered combination is consistent with the specification's disclosed transaction flows. In the email transaction flow, the vendor posts or sends an offer, the token generator generates a token, the customer sends a response email with the token embedded or checks out from a site, the e-commerce system decodes the token and processes the transaction, and a notification of successful transaction is provided. Present Application ¶ [0074]. In the web transaction flow, the vendor generates a token with associated signatures, the e-commerce system decrypts the payload, compares identifying values to values independently signed by the vendor system, determines that the request is authorized by the vendor server, performs the transaction, and notifies the vendor server. Present Application ¶ [0078]. A person of ordinary skill would understand these flows as reflecting an architecture in which receipt of a token is not enough; the token and associated request information must be validated before the transaction is executed.

134. In my opinion, this ordered combination integrates any commercial transaction concept into a practical computer-security application. The claims may operate in the context of e-commerce, but the claimed improvement is not merely the idea of buying a product or authorizing a purchase. The claimed improvement is the particular way in which the e-commerce system processes an SMTP-based transaction message: decoding a token, validating token freshness and one-time-use state, recording token use, authenticating the sender against token-encoded customer information, verifying vendor authorization using a token-authenticator, and conditionally performing the transaction only after the security checks succeed.

135. This architecture changes how SMTP-based transaction requests are processed by the e-commerce system. A conventional system might treat an email message containing product and customer information as an instruction to process a transaction. The claimed architecture instead interposes a security-processing layer between message receipt and transaction execution. That layer uses token contents, persistent token-use state, sender-token comparison, and vendor-authenticator verification to determine whether the message should be trusted.

136. A person of ordinary skill would understand this as a change in information flow and state management within the e-commerce system. The information flow is changed because the incoming SMTP message is not routed directly to order execution. It is routed through token decoding, expiration validation, one-time-use validation, token-use recording, sender authentication, and vendor-authenticator verification. State management is changed because the e-commerce system maintains token-use data, such as checksum or group-use data, that affects whether later messages containing the same or related token information are accepted or rejected. Present Application ¶ [0070].

137. In my opinion, this is a practical application of computer-security techniques to a specific electronic transaction-processing environment. The ordered combination addresses technical risks created by using asynchronous SMTP messages to carry transaction instructions. It provides the e-commerce system with mechanisms for determining whether the token is fresh, unused, tied to the correct customer, and generated by an authorized vendor. The claims therefore improve the way the e-commerce system evaluates and processes SMTP-based transaction requests, rather than merely implementing a commercial checkout concept on generic computer components.

138. Accordingly, a person of ordinary skill would understand the ordered combination as an improvement to secure electronic transaction processing. The claimed architecture coordinates expiration validation, one-time-use validation, token-use recording, sender authentication, vendor-authenticator verification, and conditional transaction execution to determine whether an SMTP-based transaction message should be trusted. This ordered combination provides the e-commerce system with a practical computer-security protocol for processing tokenized transaction messages received through an asynchronous email channel.

XII. Distinction From Conventional Email Checkout

139. Conventional email checkout systems may use a mailto link, button, or similar user-interface control to generate an email message containing transaction-related information. For example, the generated email may include information identifying a product, quantity, price, customer, campaign, or other transaction context. The technical baseline for this type of workflow is reflected in the mailto URI standard. RFC 6068 explains that a mailto URI can specify a destination email address and may also specify mail header fields and body text. M. Duerst et al., RFC 6068, The 'mailto' URI Scheme §§ 1, 3 (Internet Eng'g Task Force Oct. 2010). RFC

6068 further explains that resolving a mailto URI does not cause an immediate server interaction; instead, the client creates a message draft that the user may edit, send, or abandon. Id. § 3. Developer documentation likewise describes ordinary email links as opening a new outgoing email message and permitting subject, cc, bcc, and body fields to be encoded in the link. MDN Contributors, Creating Links, MDN Web Docs, "Email Links" (Nov. 23, 2025).

140. Patent publications in the email-checkout field reflect the same conventional baseline. U.S. Patent No. 8,775,263 to Killoran et al. describes advertising email messages containing mailto hyperlinks that generate order-reply email messages including customer, product, and quantity information. U.S. Pat. No. 8,775,263 col. 4 ll. 4-31, col. 6 l. 62-col. 7 l. 27. Similarly, U.S. Patent No. 9,704,190 to Custer describes an email checkout system in which a customer may receive an email message containing a mailto hyperlink or embedded button, and selection of the button may generate a reply email message that includes a token. U.S. Pat. No. 9,704,190 col. 2 ll. 36-54, col. 10 ll. 38-50. These references confirm that one ordinary form of email checkout is the use of a link or button to create an email message containing transaction-related content.

141. In my opinion, however, a person of ordinary skill in the art would understand that merely using a mailto link to generate an email containing transaction information is different from the ordered security architecture recited in the pending claims. A conventional email checkout system may permit a customer to generate and send an email message to place, confirm, modify, or cancel an order. But the mere generation of an email message does not necessarily provide the e-commerce system with technical assurance that the message is current, that the token has not been used before, that the sender corresponds to the customer identified in the token, or that the token was generated by a vendor authorized to generate tokens.

142. That distinction is particularly important because email is persistent, asynchronous, and user-controllable. Internet mail is an asynchronous store-and-forward communication infrastructure. D. Crocker, RFC 5598, Internet Mail Architecture § 5 (Internet Eng'g Task Force July 2009). SMTP also contemplates queueing and later retrying of messages when delivery cannot be completed immediately. J. Klensin, RFC 5321, Simple Mail Transfer Protocol §§ 3, 4.5.4.1 (Internet Eng'g Task Force Oct. 2008). Internet message standards also recognize that messages may be resent or reintroduced into the transport system. Pete Resnick, RFC 5322, Internet Message Format § 3.6.6 (Internet Eng'g Task Force Oct. 2008). Later academic work concerning email forwarding also confirms that forwarding behavior can alter email-security assumptions and can create security consequences that are not addressed merely by treating the email as a normal message. Enze Liu et al., Forward Pass: On the Security Implications of Email Forwarding Mechanism and Policy, arXiv:2302.07287 (Apr. 19, 2023).

143. SMTP sender information alone also does not provide a complete transaction-authentication mechanism. RFC 5321 states that SMTP mail is inherently insecure and that users can create messages that may cause a recipient to believe that the messages came from somewhere else. Klensin, RFC 5321 § 7.1. RFC 5321 further states that SMTP mail cannot inherently be authenticated or integrity-checked at the transport level and that there is no inherent relationship between SMTP envelope addresses and message-header addresses. Id. §§ 7.1-7.2. Thus, ordinary email checkout does not necessarily establish that the apparent sender of the email is the customer authorized by the tokenized transaction data.

144. The pending claims address these problems by requiring more than an email containing purchase information. The claims require a token generated for the mailto link. In the claimed architecture, the token is not merely a business label,

order number, or visible product identifier. The token is a machine-readable transaction-authentication object that the e-commerce system decodes and evaluates. This is consistent with the specification's definition of a token as a sequence of byte data, string, or file used to authenticate a transaction, and as one or more encrypted strings, files, passwords, ciphers, or other data containing information used to perform or authenticate a transaction when sent to payment servers. Present Application ¶ [0017]. The specification similarly explains that tokens may be encrypted strings containing information to perform or authenticate a transaction. Present Application ¶ [0027].

145. The pending claims also require token-encoded expiration information. Expiration information allows the e-commerce system to determine whether the token remains temporally valid before processing the transaction. The specification supports this feature by identifying an expiration-date token parameter and explaining that, after the expiration date, the token is no longer considered valid. Present Application ¶ [0064]. The specification also explains that a token may be configured to expire after a default time, another predetermined time, or never, and that the vendor server may extend or shorten the expiration time of an offer associated with a token. Present Application ¶ [0044]. Token expiration is a recognized technical control in tokenized systems. RFC 6819 explains that short token expiration times may protect against replay, token leakage, and online guessing. T. Lodderstedt et al., RFC 6819, OAuth 2.0 Threat Model and Security Considerations § 5.1.5.3 (Internet Eng'g Task Force Jan. 2013). RFC 7519 similarly defines the exp claim as identifying the time after which a JSON Web Token must not be accepted for processing. Michael Jones et al., RFC 7519, JSON Web Token (JWT) § 4.1.4 (Internet Eng'g Task Force May 2015). OWASP guidance for password-reset tokens likewise recommends that security tokens be single use and

expire after an appropriate period. OWASP Cheat Sheet Series, Forgot Password Cheat Sheet, "Introduction" and "General Security Practices" (last visited May 30, 2026).

146. The pending claims further require a nonce or one-time-use value. In the claimed architecture, the nonce or one-time-use value provides the e-commerce system with a way to determine whether a tokenized transaction instruction has already been used. Conventional email checkout does not necessarily provide such a one-time-use value, and the mere presence of product, quantity, or customer information in an email message does not inherently prevent the message or token from being resent or replayed. RFC 4949 defines a replay attack as an attack in which a valid data transmission is maliciously or fraudulently repeated. R. Shirey, RFC 4949, Internet Security Glossary, Version 2, at 245 (Internet Eng'g Task Force Aug. 2007). RFC 4949 defines a nonce as a random or non-repeating value included in protocol data, usually to guarantee liveness and detect and protect against replay attacks. Id. at 199. RFC 7519 similarly defines the jti claim as a unique identifier for a token. Jones et al., RFC 7519 § 4.1.7. The specification supports this one-time-use concept by explaining that the e-commerce system may store a checksum of a token after it has been used and prevent the same token from being used more than once. Present Application ¶ [0070].

147. The claimed token-use datastore lookup is another feature that distinguishes the claimed architecture from ordinary email checkout. In a conventional email checkout system, an incoming email may be parsed to determine what product, quantity, or customer information it contains. But the system does not necessarily perform a lookup against persistent token-use state to determine whether the token or one-time-use value has already been used. The claimed token-use datastore provides persistent state that exists outside the current email message. That

persistent state permits the e-commerce system to determine whether a later-received SMTP message contains a token, nonce, one-time-use value, checksum, or related group-use indicator that has already been consumed. NIST guidance treats replay resistance as a verifier-side property and, in the OTP context, states that a verifier shall accept a given OTP only once while valid. Nat'l Inst. of Standards & Tech., Digital Identity Guidelines: Authentication and Authenticator Management, NIST SP 800-63B §§ 3.1.4, 3.2.7. OWASP CSRF guidance likewise describes server-side token verification, including verifying the existence and validity of the token and rejecting the request if validation fails. OWASP Cheat Sheet Series, Cross-Site Request Forgery Prevention Cheat Sheet, "Synchronizer Token Pattern" (last visited May 30, 2026). These technical controls are consistent with the claimed use of a datastore to determine whether a token or one-time-use value has already been used.

148. The claims also require recording successful use of the nonce or one-time-use value. This is significant because replay prevention depends not only on checking whether a token has been used, but also on updating the system's stored state after use so that future attempts can be detected. The specification describes this state-update concept by explaining that the e-commerce system may store a checksum of a token after it has been used and may prevent the same token from being used more than once. Present Application ¶ [0070]. The specification also explains that token groups may be generated so that members of the group are automatically invalidated when one token is processed, and that encrypted token information may include a group value such that tokens with the same group value may no longer be considered valid after execution of a single member. Present Application ¶¶ [0044], [0065]. OWASP JWT guidance similarly recommends a server-side denylist storing a digest of revoked tokens and explains that placing a token on such a denylist causes immediate invalidation for further use. OWASP

Cheat Sheet Series, JSON Web Token for Java Cheat Sheet, "No Built-In Token Revocation by the User" (last visited May 30, 2026). Academic literature likewise recognizes the combination of finite validity and unique message identifiers as a replay-resistance technique. Boudhayan Gupta, A Replay-Attack Resistant Message Authentication Scheme Using Time-Based Keying Hash Functions and Unique Message Identifiers, arXiv:1602.02148, at 1-2 (Feb. 5, 2016).

149. The pending claims further require sender authentication against a decoded customer email address. Conventional email checkout may receive an email from an address and may include customer-identifying information in the body of the email. But the claimed architecture does not rely merely on receiving an email from any address. The e-commerce system decodes the token to identify a customer email address and authenticates the sender by determining whether the sender email address matches the customer email address identified by the decoded token. The specification supports this architecture by explaining that, when a transaction is attempted, the payment server decodes the token, authenticates the sender of the email, and may process the transaction. Present Application ¶ [0048]. The specification further explains that the e-commerce system may include an email processing system configured to verify an email address associated with the sender of an email, thereby permitting the payment server to verify that the payload was intended for use by the recipient because the email is included in the decrypted payload message. Present Application ¶ [0072]. RFC 6819 provides an analogous token-security concept by describing binding a request to an authenticated state using a state value. Lodderstedt et al., RFC 6819 § 5.3.5. OWASP JWT guidance likewise describes binding a token to additional user context, such as a fingerprint value, before accepting the token. OWASP Cheat Sheet Series, JSON Web Token for Java Cheat Sheet, "Token Sidejacking" (last visited May 30, 2026).

150. The pending claims also require vendor-authorization verification using a token-authenticator bound to the token. This is distinct from conventional email checkout because ordinary email checkout does not necessarily determine whether a token was generated by a vendor authorized by the e-commerce system to generate transaction tokens. The specification describes a cryptographic vendor-token architecture. Token generation may involve public-key cryptography, including encryption using an XSalsa20 stream cipher, authentication using Poly1305 MAC, and public keys using Curve25519 high-speed elliptic-curve cryptography. Present Application ¶ [0054]. The vendor server may receive a unique private and public key associated with the vendor and may generate a token with its unique key. Present Application ¶ [0056]. The e-commerce system receives an email containing an encoded token, decrypts the token information, confirms key validity, ensures that the vendor server is authorized to generate tokens, and processes the payment for the user with the given email address. Present Application ¶ [0068].

151. This vendor-authenticator verification is consistent with established technical approaches to signed or MACed tokens and messages. RFC 7515 defines a JSON Web Signature as a digitally signed or MACed message, and describes algorithm and key-identification information used for verification. Michael Jones et al., RFC 7515, JSON Web Signature (JWS) §§ 2, 4.1.1, 4.1.4-4.1.9 (Internet Eng'g Task Force May 2015). DKIM provides an email-specific example of binding domain responsibility to a message through a cryptographic signature that can be validated by a verifier. D. Crocker et al., RFC 6376, DomainKeys Identified Mail (DKIM) Signatures §§ 1, 3, 6 (Internet Eng'g Task Force Sept. 2011). RFC 5280 provides a public-key infrastructure example of using certificate and revocation information so that verifiers can determine whether key material remains valid. D. Cooper et al., RFC 5280, Internet X.509 Public Key Infrastructure Certificate and

Certificate Revocation List (CRL) Profile § 3.3 (Internet Eng'g Task Force May 2008). These sources support the technical character of verifying whether a token is associated with an authorized signer or key holder.

152. The specification's discussion of key revocation further distinguishes the disclosed security architecture from conventional email checkout. If a vendor key is compromised, the e-commerce system may reject encrypted transaction tokens after revoking signing keys. Present Application ¶ [0069]. A conventional email checkout system that merely processes email messages containing product or customer information does not necessarily include such a vendor-key authorization and revocation architecture. The disclosed key-revocation functionality confirms that token acceptance depends on technical authorization data maintained by the e-commerce system.

153. The claims also condition transaction execution on successful completion of the required checks. In other words, the e-commerce system does not perform the transaction merely because an email was received or because the email includes transaction information. The transaction is performed only after the token is decoded, token validity is confirmed, token-use state is checked and recorded, the sender is authenticated against token-encoded customer information, and the vendor is verified as authorized to generate the token. This conditional execution is consistent with the specification's disclosure that the e-commerce system decrypts the token, confirms key validity, ensures vendor authorization, and processes the payment for the user with the given email address. Present Application ¶ [0068]. It is also consistent with security guidance that sensitive actions should proceed only after validating the relevant token. See OWASP Forgot Password Cheat Sheet; OWASP Cross-Site Request Forgery Prevention Cheat Sheet.

154. In my opinion, a person of ordinary skill would understand this ordered combination as materially different from ordinary email checkout. Ordinary email checkout may provide convenience by letting a customer initiate or confirm a purchase through email. The claimed architecture, by contrast, adds a transaction-security protocol that determines whether an SMTP transaction message should be trusted and processed. That protocol includes token generation for a mailto link, token-encoded expiration, token-encoded one-time-use information, token-use datastore lookup, recording of token use, sender authentication against decoded token information, vendor-authorization verification using a token-authenticator, and conditional transaction execution.

155. The specification's examples further illustrate this distinction. In the email transaction flow, the vendor may post or send an offer, the token generator generates a token, the customer sends a response email with the token embedded or checks out from a site, the e-commerce system decodes the token and processes the transaction, and a success notification is sent. Present Application ¶ [0074]. In the web transaction flow, the vendor generates a token with associated signatures, the e-commerce system decrypts the payload, compares identifying values to values independently signed by the vendor system, determines that the request is authorized by the vendor server, performs the transaction, and notifies the vendor server. Present Application ¶ [0078]. These examples show that transaction processing depends on token decoding, authorization, and validation, not merely on the existence of an email or web request.

156. Accordingly, in my opinion, the claims improve upon ordinary email checkout by adding a specific security architecture for determining whether an SMTP transaction message should be processed. The improvement lies in how the e-commerce system evaluates incoming tokenized transaction messages before

execution. The system uses token-encoded validity information, token-use state, sender-token matching, and vendor-authenticator verification to decide whether to trust the message. A person of ordinary skill would understand that this is a technical improvement to secure electronic transaction processing, rather than merely a description of a commercial transaction implemented using email.

XIII. Why the Claimed Improvement Would Have Been Recognized at the Relevant Time

157. At the relevant time, persons working in the field of electronic commerce understood that email could be used as part of an online transaction workflow. Email messages, links, and buttons could be used to initiate or confirm transactions, and payment servers could be used to process transactions based on information received through electronic communications. The specification reflects this state of the art by explaining that vendors wishing to adopt an email-based payment method may need to integrate their own systems with a third-party system to use an email payment process. Present Application ¶ [0003]. The specification further explains that an email payment system may require close integration between the vendor and e-commerce system, which may be time consuming and cumbersome for the vendor. Id.

158. At the same time, persons of ordinary skill in the art understood that e-commerce systems needed to reduce transaction friction while preserving transaction security. The specification expressly describes both aspects of that problem. On the friction and scalability side, the specification explains that communication between the gateway and the vendor may be considerable depending on the size of the customer base and the array of offers, and that larger numbers of users may require more bandwidth. Present Application ¶ [0004]. The specification also explains that many online businesses depend on the ability to scale their

customer base and that a system using less bandwidth would be advantageous. Id. On the security side, the specification states that online security is a concern for any vendor and that limiting communication between systems communicating encrypted data over public networks reduces potential interference from outside parties. Present Application ¶ [0005].

159. In my opinion, a person of ordinary skill would have understood that the design challenge was not simply to make purchasing easier. The design challenge was to reduce checkout friction and communication overhead without weakening the security of the transaction-processing system. The specification addresses that challenge by describing a token generator integrated directly in the vendor's system, thereby allowing the vendor to generate tokens for email-based payments and two-click web checkout while reducing communication with the email payment gateway. Present Application ¶ [0007]. The specification also explains that the token generator allows vendors to include secure tokens in advertising emails, and that when a mailto hyperlink is selected, a new email contains a token holding identifiers of the associated offer and purchaser. Id.

160. The state of the art also included the use of tokens to carry machine-readable transaction or authorization information. The specification defines a token as a sequence of byte data, string, or file used to authenticate a transaction, and explains that a token may include encrypted strings, files, passwords, ciphers, or other data containing information used to perform or authenticate a transaction when sent to payment servers. Present Application ¶ [0017]. The specification also describes token parameters including private key, public key, partner ID, environment, token type, card token, email address, amount, user data, expiration date, user-agent information, and IP address. Present Application ¶¶ [0027]-[0043]. A person of ordinary skill would have understood from these disclosures that the

token was not merely a visual label or order number. It was a machine-readable data object used by the payment or e-commerce system to authenticate and process the transaction.

161. At the relevant time, persons of ordinary skill also would have recognized that reducing friction by allowing mailto-based confirmation created technical risks that needed to be addressed by server-side validation. A mailto-based transaction message is not the same as a live, synchronous checkout session. The technical standards governing email confirm that Internet mail operates through asynchronous, store-and-forward infrastructure. See D. Crocker, RFC 5598, Internet Mail Architecture § 5 (Internet Eng'g Task Force July 2009). SMTP also provides for queuing and later retrying of messages when delivery cannot be completed immediately. See J. Klensin, RFC 5321, Simple Mail Transfer Protocol §§ 3, 4.5.4.1 (Internet Eng'g Task Force Oct. 2008). A person of ordinary skill would therefore have recognized that a transaction email generated from a mailto link might be delayed, resent, forwarded, copied, or received after the original transaction context had changed.

162. Those email-specific characteristics would have made server-side validation important. A person of ordinary skill would have understood that an e-commerce system receiving an SMTP transaction message could not safely rely only on the fact that an email message was received or that the message contained product, price, quantity, or customer information. The receiving system needed to determine whether the token was still valid, whether the token had already been used, whether the email sender corresponded to the intended customer, and whether the token was generated by an authorized vendor.

163. The specification describes the claimed improvement in a way that would have made these technical advantages apparent to a person of ordinary skill.

The specification describes an architecture that enables vendors to generate two-click tokens within their own vendor server without requiring communication with the e-commerce system at the time the tokens are created. Present Application ¶ [0050]. It also describes an e-commerce system that includes a token decoding unit, database module, order execution unit, message processing unit, and account management unit. Present Application ¶¶ [0050]-[0051]. In my opinion, a person of ordinary skill would have recognized that this division of functionality required the e-commerce system to perform later validation of tokens that were generated by the vendor-side system and transmitted through email or another electronic messaging channel.

164. The specification further states that the system is configured to use a secure and backwards-compatible process, provides improved security by embedding more functionality in the vendor server, and reduces bandwidth by streamlining communications with the e-commerce system. Present Application ¶ [0052]. A person of ordinary skill would have understood those benefits as technical improvements to the architecture of the payment system. The architecture reduces the need for repeated gateway communication at token-generation time while preserving e-commerce-system validation at transaction-processing time.

165. A person of ordinary skill would also have recognized the improvement provided by expiration validation. The specification identifies an expiration-date token parameter and explains that, after that time, the token is no longer considered valid. Present Application ¶ [0064]. The specification also explains that tokens may be configured to expire after a default period, after another predetermined time, or never, and that the vendor server may extend or shorten the expiration time of a particular offer associated with a token. Present Application ¶ [0044]. This would have informed a person of ordinary skill that the system addresses stale-token risk

by causing the e-commerce system to determine whether the token remains valid before processing the transaction.

166. A person of ordinary skill would also have recognized the improvement provided by checking and recording one-time-use token state. The specification explains that the e-commerce system may store a checksum of a token after it has been used and prevent the same token from being used more than once. Present Application ¶ [0070]. It further explains that the e-commerce system may store a group reference number for a used token and prevent the group reference number from being used more than a predetermined number of times. Id. In addition, the specification describes token groups in which members may be automatically invalidated when one token is processed. Present Application ¶¶ [0044], [0065]. A person of ordinary skill would have recognized these features as addressing replay and duplicate-processing risks created by persistent, asynchronous email messages.

167. This recognition would also have been consistent with known computer-security principles. Replay attacks were a recognized technical risk, and nonces or non-repeating values were known mechanisms for guaranteeing freshness and protecting against replay. See R. Shirey, RFC 4949, Internet Security Glossary, Version 2, at 199, 245 (Internet Eng'g Task Force Aug. 2007). Token-expiration and one-time-use restrictions were likewise recognized token-security controls. See T. Lodderstedt et al., RFC 6819, OAuth 2.0 Threat Model and Security Considerations §§ 5.1.5.3-.4 (Internet Eng'g Task Force Jan. 2013). In view of those principles, a person of ordinary skill would have recognized that a server-side token-use datastore improves SMTP-based transaction processing by giving the e-commerce system persistent state for determining whether a tokenized transaction instruction has already been used.

168. A person of ordinary skill would also have recognized the improvement provided by sender authentication based on decoded token information. The specification explains that, when a transaction is attempted, the payment server decodes the token, authenticates the sender of the email, and may process the transaction. Present Application ¶ [0048]. The specification also explains that the e-commerce system may include an email processing system configured to verify an email address associated with the sender of an email, thereby permitting the payment server to verify that the payload was intended for use by the recipient because the email is included in the decrypted payload message. Present Application ¶ [0072]. In my opinion, a person of ordinary skill would have understood that this improves security by binding the SMTP sender information to customer identity information recovered from the decrypted token payload.

169. A person of ordinary skill would also have recognized the improvement provided by vendor-authorization verification. The specification describes token generation using public-key cryptography, including encryption, authentication using Poly1305 MAC, and public keys using Curve25519. Present Application ¶ [0054]. It further explains that the vendor server receives a unique private and public key associated with the vendor, and a public key that the e-commerce system uses for multiple partners. Present Application ¶ [0056]. The e-commerce system receives an email containing an encoded token, decrypts token information, confirms key validity, ensures that the vendor server is authorized to generate tokens, and processes payment. Present Application ¶ [0068]. A person of ordinary skill would have understood this as a technical mechanism for determining whether the token originated from, or was authorized by, a vendor recognized by the e-commerce system.

170. The specification's key-revocation disclosure would have reinforced that understanding. The specification explains that, if a keyset is compromised, the e-commerce system may disable the compromised keyset and generate another, and that if a vendor key is compromised, the e-commerce system may reject encrypted transaction tokens after revoking signing keys. Present Application ¶ [0069]. A person of ordinary skill would have recognized this as a security control for maintaining trust in vendor-generated tokens and preventing compromised or unauthorized token generation from resulting in transaction processing.

171. In my opinion, the claimed improvement would have been apparent from the specification to a person of ordinary skill because the specification expressly describes the relevant components and their cooperative operation. The vendor-side token generator creates tokens. Present Application ¶¶ [0023], [0053]-[0056]. The token carries transaction, customer, vendor, expiration, group, and browser or system-identifying information. Present Application ¶¶ [0027]-[0043], [0057]-[0067]. The e-commerce system decodes the token, confirms key validity, verifies vendor authorization, verifies sender email information, stores token-use state, prevents token reuse, and processes the transaction only after validation. Present Application ¶¶ [0048], [0068]-[0072]. The specification further describes the transaction flows in which a customer sends a response email with the token embedded, the e-commerce system decodes the token and processes the transaction, and a successful-transaction notification is provided. Present Application ¶ [0074].

172. A person of ordinary skill would therefore have recognized that the claimed architecture improves the security and reliability of SMTP-based transaction processing. It does so by adding server-side checks that are specifically suited to tokenized transaction instructions carried through email: expiration validation for stale tokens, one-time-use validation and recording for replay

prevention, sender-token matching for customer authentication, and vendor-authenticator verification for token-origin assurance. These features cooperate to determine whether an SMTP transaction message should be trusted before transaction execution occurs.

173. Accordingly, in my opinion, the claimed improvement would have been recognizable at the relevant time from both the specification and the surrounding technical context. The specification describes an architecture that reduces checkout friction and communication overhead while preserving security through server-side validation of tokenized transaction messages. A person of ordinary skill would understand that this architecture improves the e-commerce system's processing of SMTP-based transaction requests by changing the system from a simple email-order processor into a stateful, token-validating, sender-authenticating, vendor-verifying transaction-processing system.

XIV. Claimed Features Corresponding to the Technological Improvement

174. The technological improvement described above is reflected in the pending claims. In my opinion, a person of ordinary skill in the art would understand claim 1 to recite the core security architecture for SMTP-based e-commerce transaction processing. Claim 1 does not merely recite the receipt of an email message or the completion of a purchase. Instead, claim 1 recites a coordinated sequence of token generation, token decoding, token validation, token-use recording, sender authentication, vendor-authorization verification, conditional transaction execution, and transaction notification.

175. In particular, claim 1 recites receiving, by an e-commerce system, an email message that contains a token, where the email message is generated in response to activating a mailto link and where the token is generated for the mailto link. A person of ordinary skill would understand that this limitation ties the token

to a structured SMTP-based transaction workflow, rather than to arbitrary email correspondence. The email message is the communication vehicle, but the token is the machine-readable transaction-authentication object that the e-commerce system must evaluate before transaction execution.

176. Claim 1 further recites that the token includes a vendor identifier, transaction data identifying at least one of a product identifier, a price, or a quantity, an expiration time, and a nonce or one-time-use value generated for the token. These limitations correspond to the security architecture described above. The vendor identifier identifies the vendor associated with the transaction token. The transaction data identifies the relevant transaction context. The expiration time provides a basis for determining whether the token is stale. The nonce or one-time-use value provides a basis for determining whether the token has already been used or replayed.

177. Claim 1 also recites decoding the token to form a decoded token, where the decoded token identifies a third-party vendor and a customer email address. In my opinion, this decoding step reflects the claimed system's use of token-derived information as a basis for later validation and authentication. The e-commerce system does not rely only on visible email text or the apparent sender address. It decodes the token and uses information obtained from the token to determine whether the SMTP message is eligible for transaction processing.

178. Claim 1 then recites validating the token by determining that the expiration time has not elapsed and by determining, using a token-use datastore, that the nonce or one-time-use value has not previously been used. These limitations correspond to two different technical controls. The expiration check addresses stale-token risk. The token-use datastore lookup addresses replay risk. In my opinion, a person of ordinary skill would understand these limitations as requiring the e-

commerce system to perform server-side validation before the transaction is executed.

179. Claim 1 further recites, after validating the token is successful and before performing the e-commerce transaction, recording, by the e-commerce system, in the token-use datastore, use of the nonce or one-time-use value. This limitation corresponds to the persistent-state aspect of the replay-prevention architecture. The e-commerce system does not merely check information in the current email message. It updates stored token-use state so that future attempts to use the same nonce or one-time-use value may be detected and rejected.

180. Claim 1 also recites authenticating a sender of the email message as a customer by determining that an email address of the sender matches the customer email address identified by the decoded token. This limitation corresponds to the sender-token matching described above. It binds the communication-layer sender information associated with the SMTP message to the customer identity information obtained from the decoded token. A person of ordinary skill would understand this as a technical authentication step performed by the e-commerce system before transaction execution.

181. Claim 1 further recites determining that the third-party vendor is authorized to generate the token by verifying a token-authenticator that is bound to the token and associated with the third-party vendor. The token-authenticator may include a cryptographic signature, a message authentication code, or an encrypted token portion that is verifiable using vendor authorization data stored by the e-commerce system. This limitation corresponds to the vendor-authorization verification described above. A person of ordinary skill would understand this as requiring the e-commerce system to verify that the token is associated with a vendor authorized to generate transaction tokens before the token is trusted.

182. Claim 1 then recites performing the e-commerce transaction when sender authentication is successful, token validation is successful, and determining that the third-party vendor is authorized to generate the token is successful. This conditional-execution limitation is important. It reflects that transaction execution is gated by the security checks. The e-commerce system does not execute the transaction merely because an email was received or because a token contains transaction information. It executes the transaction only when the required sender-authentication, token-validation, and vendor-authorization checks succeed.

183. Finally, claim 1 recites notifying the third-party vendor of successful processing of the e-commerce transaction. In the claimed architecture, notification occurs after the e-commerce system has processed the transaction through the required security controls. A person of ordinary skill would understand this notification as part of the transaction-processing workflow that follows successful validation and execution.

184. The system claim recites a corresponding implementation of the same security architecture. Claim 7 recites a communications interface, one or more processors, and memory storing instructions that cause the system to perform corresponding operations: receive the email message containing the token generated in response to activating a mailto link, decode the token, validate expiration and prior-use state, record use of the nonce or one-time-use value, authenticate the sender by comparing sender email information to decoded token information, verify vendor authorization using a token-authenticator, perform the transaction only when the required checks succeed, and notify the third-party vendor. In my opinion, a person of ordinary skill would understand claim 7 as reciting a system implementation of the same SMTP transaction-security protocol recited in claim 1.

185. The computer-readable-medium claim also recites a corresponding implementation of the same architecture. Claim 13 recites a non-transitory computer-readable medium storing instructions that, when executed by one or more processors of an e-commerce system, cause the processors to perform operations corresponding to the method of claim 1. A person of ordinary skill would understand claim 13 as reciting software instructions for implementing the same token-decoding, token-validation, token-use-recording, sender-authentication, vendor-authorization, and conditional-transaction-execution architecture.

186. The dependent claims further reflect practical implementations of that technological improvement. Claims 2, 8, and 14 recite that the mailto link is included in an offer message provided by the third-party vendor. These claims correspond to an implementation in which the vendor provides an offer message containing a mailto link that initiates the tokenized transaction workflow. A person of ordinary skill would understand these limitations as applying the claimed security architecture to vendor-provided offer messages.

187. Claims 3, 9, and 15 recite that the offer message includes a plurality of mailto links corresponding to a plurality of tokens. These limitations reflect a practical implementation in which a single offer message may present multiple selectable transaction options. Each mailto link corresponds to a token, and the e-commerce system may later decode and validate the token associated with the selected link. In my opinion, these claims show that the security architecture can be applied to multiple tokenized options within an offer-message environment.

188. Claims 4, 10, and 21 recite that at least two of the plurality of tokens are associated with different transaction data identifying different products, different prices, different quantities, or a combination thereof. These limitations further illustrate how the claimed architecture applies to practical e-commerce offers.

Different selectable tokens may correspond to different product, price, or quantity options, while the e-commerce system still applies the same token-validation, sender-authentication, and vendor-authorization checks before transaction execution.

189. Claims 5, 11, and 22 recite that the offer message is distributed using at least one of email, a website, or a social media platform. These limitations reflect that the tokenized transaction workflow may be initiated from different electronic distribution channels. In my opinion, a person of ordinary skill would understand that, regardless of the distribution channel used to present the offer, the technological improvement remains the e-commerce system's validation of the SMTP-based transaction message and token before processing the transaction.

190. Claims 6, 12, and 18 recite monitoring which tokens are in use based on entries in the token-use datastore. These limitations correspond to the persistent-state and replay-prevention aspects of the technological improvement. Monitoring tokens based on datastore entries allows the e-commerce system to determine whether tokens are available, used, invalidated, or otherwise ineligible for processing. A person of ordinary skill would understand this monitoring as part of the state-management architecture used to prevent duplicate or unauthorized processing of tokenized transaction instructions.

191. Claim 19 further narrows vendor-authorization verification by reciting verification of a message authentication code over at least the vendor identifier, transaction data, and expiration time using vendor authorization data that includes a vendor-specific cryptographic key, and rejecting the token when MAC verification fails. In my opinion, claim 19 reflects a concrete cryptographic implementation of the token-authenticator limitation. A person of ordinary skill would understand that verifying a MAC over vendor, transaction, and expiration information helps ensure

both the integrity of those token fields and the association of the token with a vendor-specific key recognized by the e-commerce system.

192. Claim 20 further narrows replay prevention by reciting that validating the token comprises generating a checksum of the token or decoded token, storing the checksum in the token-use datastore as a used-token record after validation succeeds and before transaction execution, and rejecting any subsequently received token that matches the used-token record. In my opinion, claim 20 reflects a concrete implementation of the claimed token-use datastore and one-time-use architecture. The checksum-based used-token record gives the e-commerce system persistent state for identifying later replay or reuse of the same token.

193. Taken together, the independent and dependent claims reflect the same practical application and technological improvement described above. The independent claims recite the core ordered security architecture for processing SMTP-based tokenized transaction messages. The dependent claims identify practical implementations involving vendor offer messages, multiple mailto links, multiple tokens, different transaction data, different distribution channels, token monitoring, MAC-based vendor verification, and checksum-based replay prevention.

194. In my opinion, a person of ordinary skill would understand these claim limitations as improving secure electronic transaction processing. The claims do not merely automate the commercial act of purchasing a product. They recite how an e-commerce system determines whether an SMTP-based transaction message should be trusted before execution. That determination is made through token decoding, expiration validation, one-time-use validation, token-use recording, sender-token matching, vendor-authenticator verification, and conditional transaction execution. These claim limitations correspond to a practical computer-security application for

processing tokenized transaction instructions received through an asynchronous email channel.

XV. Nexus Between the Claimed Features, Specification, and Technological Improvement

195. I understand that, for a declaration under 37 C.F.R. § 1.132 to be relevant to subject matter eligibility, there should be a nexus between the claimed invention and the evidence or technical explanation provided in the declaration. In my opinion, the technological improvement described in this declaration is reflected in the pending claims and is supported by the specification of the present application.

196. The table below identifies representative claim features that correspond to the technological improvement described above, the technical function performed by those features in the claimed SMTP-based transaction-processing architecture, and representative portions of the specification that support a person of ordinary skill's understanding of those features.

Claimed feature	Technical function in the SMTP transaction-security architecture	Representative specification support
Email message generated in response to activating a mailto link	Provides a structured SMTP-based transaction message that is generated from a link-driven transaction workflow, rather than arbitrary email correspondence	Present Application ¶¶ [0007], [0008], [0053], [0074]
Token generated for the mailto link	Creates a machine-readable transaction-authentication object associated with the mailto-generated transaction workflow	Present Application ¶¶ [0007], [0008], [0017], [0053], [0074]
Token comprising a vendor identifier	Identifies the vendor associated with the transaction token so that the e-commerce system can evaluate vendor authorization	Present Application ¶¶ [0028]-[0032], [0056], [0062], [0068]
Token comprising transaction data identifying a product identifier, price, quantity, or related transaction information	Binds machine-readable transaction context to the token so that the e-commerce system can determine what transaction the received SMTP message purports to initiate or confirm	Present Application ¶¶ [0027], [0034]-[0038], [0057]-[0067], [0074], [0077]
Token comprising an expiration time	Allows the e-commerce system to determine whether the token remains temporally valid, thereby addressing stale-message and stale-token risks in asynchronous SMTP transaction processing	Present Application ¶¶ [0039], [0044], [0064]
Token comprising a nonce or one-time-use value generated for the token	Provides a token-specific value that can be used by the e-commerce system to determine whether the tokenized transaction instruction has already been consumed or replayed	Present Application ¶¶ [0044], [0065], [0070]
Decoding the token to form a decoded token	Allows the e-commerce system to recover trusted token-derived	Present Application ¶¶ [0048], [0050]-

Claimed feature	Technical function in the SMTP transaction-security architecture	Representative specification support
identifying a third-party vendor and a customer email address	transaction, vendor, and customer information for use in later validation and authentication checks	[0051], [0060]-[0068], [0072]
Validating the token by determining that the expiration time has not elapsed	Prevents stale tokenized transaction messages from being processed after the token or offer is no longer valid	Present Application ¶¶ [0039], [0044], [0064]
Validating the token by determining, using a token-use datastore, that the nonce or one-time-use value has not previously been used	Provides server-side, stateful replay prevention for tokenized SMTP transaction instructions	Present Application ¶¶ [0044], [0065], [0070]
Recording use of the nonce or one-time-use value in the token-use datastore after successful validation and before transaction execution	Updates persistent token-use state before transaction execution so that later attempts to use the same token, nonce, one-time-use value, checksum, or related group value may be detected and rejected	Present Application ¶¶ [0044], [0065], [0070]
Authenticating the sender by determining that the sender email address matches the customer email address identified by the decoded token	Binds SMTP sender information to customer identity information recovered from the decoded token, thereby preventing transaction execution based solely on an apparent sender address or solely on token content	Present Application ¶¶ [0048], [0060], [0068], [0072]
Determining that the third-party vendor is authorized to generate the token by verifying a token-authenticator bound to the token	Provides a technical trust mechanism for determining whether the token originated from, or was authorized by, a vendor recognized by the e-commerce system	Present Application ¶¶ [0054], [0056], [0068], [0069], [0078]

Claimed feature	Technical function in the SMTP transaction-security architecture	Representative specification support
Token-authenticator comprising a cryptographic signature, message authentication code, or encrypted token portion	Provides cryptographic or equivalent technical evidence of token integrity, token authenticity, and vendor authorization	Present Application ¶¶ [0017], [0054], [0056], [0068], [0069], [0078]
Performing the transaction only when sender authentication, token validation, and vendor-authorization verification are successful	Gates transaction execution on multiple coordinated security checks directed to the SMTP message, token state, customer identity, and vendor authorization	Present Application ¶¶ [0048], [0068]-[0072], [0074], [0078]
Notifying the third-party vendor of successful transaction processing	Completes the transaction-processing workflow after the e-commerce system has validated and processed the tokenized SMTP transaction message	Present Application ¶¶ [0008], [0074], [0078]
Monitoring which tokens are in use based on entries in the token-use datastore	Maintains token-use state for determining whether tokens remain available, have been used, have been invalidated, or are otherwise ineligible for additional transaction processing	Present Application ¶¶ [0044], [0065], [0070]
MAC over at least the vendor identifier, transaction data, and expiration time using a vendor-specific cryptographic key	Provides a concrete implementation of token-authenticator verification that protects vendor identity, transaction context, and expiration information against unauthorized alteration	Present Application ¶¶ [0054], [0056], [0061]-[0064], [0068]
Checksum of the token or decoded token stored as a used-token record	Provides a concrete implementation of server-side replay prevention by creating a persistent record that can be used to reject later matching tokens	Present Application ¶ [0070]

197. In my opinion, the claim features identified above are not isolated business steps. They are coordinated technical features that define how the e-commerce system processes a tokenized SMTP transaction message before transaction execution. The specification describes the corresponding architecture and explains how tokens may be generated, transmitted, decoded, authenticated, checked for validity, checked for prior use, and processed by an e-commerce system.

198. A person of ordinary skill in the art would understand the cited portions of the specification to support the technical improvement described in this declaration. In particular, the specification supports the use of a token generated for a mailto-based transaction workflow, the inclusion of transaction, vendor, customer, expiration, and security-related information in the token, the use of cryptographic or equivalent token-authentication mechanisms, the verification of sender email information against decoded token information, and the maintenance of token-use state to prevent repeated use of the same token or related token group.

199. In my opinion, the claimed features correspond to a practical computer-security application for SMTP-based transaction processing. The e-commerce system does not merely receive an email and perform a commercial transaction. Rather, the e-commerce system receives an SMTP message containing a tokenized transaction instruction and routes that message through token decoding, expiration validation, one-time-use validation, token-use recording, sender-token authentication, vendor-authenticator verification, and conditional transaction execution.

200. The technological improvement described in this declaration is therefore reflected in the pending claims and supported by the specification. The improvement is the e-commerce system's stateful, cryptographically verifiable, sender-bound processing of tokenized SMTP transaction messages. A person of

ordinary skill would understand that improvement as enhancing the security, integrity, and reliability of electronic transaction processing through an asynchronous email channel.

XVI. Opinion Regarding Technological Improvement, Without Legal Conclusion

201. I understand that the question of patent eligibility under 35 U.S.C. § 101 is a legal question. I am not offering a legal opinion on whether the pending claims are patent eligible. My opinions in this declaration are offered from a technical perspective, based on my experience with email-based e-commerce systems, tokenized transaction processing, and secure electronic transaction workflows.

202. From that technical perspective, it is my opinion that a person of ordinary skill in the art would understand the specification and pending claims to describe an improvement to secure electronic transaction processing. The specification describes a vendor-token architecture in which a vendor-side system may generate transaction tokens and an e-commerce system may later receive, decode, validate, authenticate, and process those tokens. Present Application ¶¶ [0007], [0017], [0027], [0050]-[0056], [0068]-[0072]. A person of ordinary skill would not understand that architecture as merely a statement that email may be used to buy goods or services. Rather, a person of ordinary skill would understand the architecture as addressing technical issues that arise when SMTP messages are used to carry machine-readable transaction instructions.

203. The improvement is reflected in the claims through an ordered combination of technical operations. The claims require an email message generated in response to activation of a mailto link, a token generated for that mailto link, decoding of the token by the e-commerce system, expiration validation, nonce or

one-time-use validation, token-use datastore recording, sender authentication based on decoded token information, vendor-authenticator verification, and conditional transaction execution only when the required checks are successful. In my opinion, a person of ordinary skill would understand that ordered combination as a transaction-security protocol for determining whether an SMTP-based transaction message should be trusted and processed.

204. Each component of that ordered combination contributes to the overall technological improvement. Token generation for a mailto link provides a structured machine-readable transaction object associated with a particular link-driven transaction workflow. Token decoding permits the e-commerce system to obtain transaction, customer, and vendor information from the token rather than relying only on visible email text or apparent sender information. Expiration validation addresses stale-token risk. Nonce or one-time-use validation addresses replay or duplicate-use risk. Recording token use in a token-use datastore gives the e-commerce system persistent state for future replay prevention. Sender authentication binds the SMTP message to the customer identity identified in the decoded token. Vendor-authenticator verification binds the token to a vendor authorized by the e-commerce system to generate transaction tokens. Conditional transaction execution ensures that the transaction is performed only after the required security checks are satisfied.

205. In my opinion, this improves the security of SMTP-based e-commerce transaction processing. The claimed architecture reduces the risk that an e-commerce system will process stale messages, replayed tokens, duplicate token submissions, messages sent from an email address inconsistent with the decoded token, or tokens generated by unauthorized vendors. The specification describes these types of technical controls by disclosing token expiration, token groups, checksum storage

after token use, prevention of multiple use of the same token, sender email verification, public-key cryptography, MAC authentication, vendor-specific keys, key-validity confirmation, vendor-authorization verification, and key revocation. Present Application ¶¶ [0044], [0048], [0054], [0056], [0064]-[0072].

206. In my opinion, this also improves the reliability of SMTP-based e-commerce transaction processing. Email messages may be asynchronous, persistent, resent, forwarded, delayed, or otherwise received outside the original transaction context. A system that merely receives an email containing product or customer information may not reliably determine whether the message should still be honored. The claimed architecture improves reliability by requiring the e-commerce system to evaluate token freshness, token-use state, sender-token consistency, and vendor authorization before the transaction is performed.

207. The claimed architecture further improves the integrity of SMTP-based e-commerce transaction processing. By using token-authenticator verification and vendor authorization data, the e-commerce system can determine whether a token is associated with a vendor authorized to generate tokens. By using token-use state, the e-commerce system can determine whether a token or related one-time-use value has already been consumed. By using decoded customer email information, the e-commerce system can determine whether the sender of the SMTP message corresponds to the customer identified in the token. These controls help preserve the integrity of the transaction-processing system by ensuring that the system processes only tokenized transaction instructions that satisfy the required technical checks.

208. In my opinion, the claimed improvement is rooted in computer-network and transaction-processing technology rather than merely a commercial practice. The relevant improvement is not the commercial idea that a customer may purchase goods or services. The relevant improvement is how an e-commerce system

processes SMTP-based transaction messages that contain machine-readable tokens. The claims change the processing path for such messages by requiring token decoding, token validation, token-use datastore interaction, sender-token authentication, vendor-authenticator verification, and conditional transaction execution.

209. A person of ordinary skill would understand that these features change both the information flow and the state-management architecture of the e-commerce system. The information flow is changed because an incoming SMTP message is not routed directly to transaction execution merely because it contains transaction information. Instead, the message and token are routed through security-processing steps before the transaction is executed. The state-management architecture is changed because the e-commerce system records token-use information and uses that stored information to determine whether later token submissions should be rejected.

210. For these reasons, as a technical matter, I believe a person of ordinary skill in the art would understand the specification and pending claims to describe an improvement to secure electronic transaction processing. The improvement is reflected in the ordered claim limitations and is directed to the way the e-commerce system evaluates and conditionally processes SMTP-based tokenized transaction messages. This is a technological improvement to secure transaction processing, not merely the implementation of a commercial transaction using email.

XVII. Conclusion

211. I declare that all statements made herein of my own knowledge are true and that all statements made herein on information and belief are believed to be true.

212. I further declare that I have reviewed the contents of this declaration and that the statements set forth herein are based on my personal knowledge, my

technical experience, my review of the materials identified above, and my understanding of how a person of ordinary skill in the art would have understood the specification and claims at the relevant time.

213. I understand that willful false statements and the like are punishable by fine or imprisonment, or both, under 18 U.S.C. § 1001. I further understand that willful false statements may jeopardize the validity of the application or any patent issuing thereon.

214. I declare under penalty of perjury that the foregoing is true and correct.

Signed by:
CF083E7D39D5499...

John P. Killoran, Jr.

DATE: 6/1/2026 | 3:29 PM EDT