PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In the **PATENT APPLICATION** of:	
Swoop IP Holdings LLC	
Application No.: 17/883,539	
Confirmation No.: 8678	Our File: SWO-PT005.4
Filed: August 08, 2022	
For: EMAIL-BASED E-COMMERCE	
Group: 3622	
Examiner: JOHN W VAN BRAMER	

DECLARATION UNDER 37 C.F.R. §1.132

(submitted electronically via Patent Center)
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

I, John P. Killoran, Jr., declare that:

I am a named inventor of the subject application and have personal knowledge of the facts set forth herein, except where stated otherwise, and in those instances I believe the facts stated to be true.

Contents

I. My Background

1. I am currently the Chief Executive Officer of Swoop In Technologies LLC. Swoop is a technology development company focused on infrastructure for a global trust network, including technologies relating to identity and payments across the internet. My work at Swoop has involved developing secure internet infrastructure for passwordless authentication, identity verification, and payment-related systems. According to my professional background, I have worked on technologies involving cryptography, email, QR technology, payments, identity, artificial intelligence, passwordless systems, and cybersecurity.

2. I have spent my career developing first-of-their-kind technologies directed to solving practical problems through scalable technical systems. I have been named on a substantial portfolio of issued U.S. patents and published applications relating to email-based e-commerce, email payment gateways, passwordless authentication, secure browser control, QR-code and biometric payment technologies, alternate payment controls, identity, and secure transaction processing. For example, I am the named inventor on U.S. Patent No. 9,704,148, titled "Email payment gateway for e-commerce," which identifies email, customer, SMTP, payment gateway, and UUID among its relevant technology terms. I am also the named inventor on U.S. Patent No. 11,288,713, titled "Sending funds via an email payment gateway," and U.S. Patent No. 11,410,143, titled "Email-Based E-Commerce."

3. My issued-patent work also includes technologies closely related to secure authentication and passwordless access. For example, U.S. Patent No. 11,968,195, titled "Email-based authentication for sign in and security," describes systems and methods for improving computer-network security, accessing secure accounts without passwords, and deterring automated attacks using email

authentication. These patents reflect my experience with systems that use email not merely as a communication medium, but as part of a security, identity, authentication, or transaction-control architecture.

4. My issued-patent work further includes payment-control and transaction-interface technologies. For example, U.S. Patent No. 11,416,829, titled "Myriad of payment methods with alternate payment controls," describes alternate payment controls and structured combinations of user selections for payment processing. Other patents and applications in my portfolio address email-based e-commerce using QR codes, barcodes, image recognition, alternate payment methods, and biometrics, including issued patents identified in the same patent family as U.S. Patent Nos. 11,182,790 and 12,125,034.

5. I hold a Bachelor of Science degree in Mechanical Engineering from Widener University, a Juris Doctor degree from Widener University School of Law, and completed a Masters Program in Philosophy at the University of New Mexico. My technical, legal, and business experience has included developing and commercializing systems that operate at the intersection of secure communications, identity, authentication, payments, and internet-based transaction workflows.

6. In connection with the technology described and claimed in the present application, I was involved in developing the architecture by which an email-originated transaction request can be generated, returned, authenticated, and conditionally executed. My role included identifying the technical risks presented when a transaction instruction is generated from an email client and returned through an email pathway rather than through a conventional authenticated web checkout session. Those risks include, for example, alteration of email contents, forwarding of email links, attempted spoofing, replay of old messages, use of stale offers, and loss of server-controlled web-session context.

7. Before and at the time of the invention, I was familiar with email campaign systems and the use of email to communicate transaction opportunities to recipients. I was also familiar with the practical limitations of email-originated transaction flows, including that a returned email message may be received outside the context of an authenticated web session and may need to be evaluated based on message contents, sender information, and server-side transaction data.

8. I was familiar with SMTP-based email transmission and the use of mailto hyperlinks to cause an email client to generate an email message. I understood that SMTP and mailto functionality could be used to generate and transport messages, but that those mechanisms do not themselves provide a transaction-specific authentication control packet, field-by-field validation of a returned order email, or expiration-based gatekeeping of transaction execution.

9. I was also familiar with customer and order databases used to store customer information, transaction information, campaign information, payment-related information, and order-status information. In developing the invention, I understood that database-stored information could be used to create server-side transaction context against which a returned email-originated order message could be validated.

10. I was familiar with authentication workflows used to decide whether a transaction request should be trusted before the system performs a transaction. In the context of the present application, the relevant workflow includes associating a UUID with an authentication control packet, embedding or including the UUID in a mailto-generated transaction path, receiving a returned SMTP order email, parsing fields of the returned email, comparing parsed values against stored authentication-control-packet fields, verifying expiration, and allowing order execution only when the returned message satisfies the required authentication checks.

11. I was also familiar with order confirmation and payment workflows. In particular, I understood that an e-commerce system may need to interact with payment processing systems, order fulfillment systems, and confirmation mechanisms. I also understood that when authentication of an email-originated order message fails, the system should not simply execute the order, but should instead deny, suspend, log, or route the transaction into an additional confirmation procedure.

12. Based on my experience, including my patent work in email-based e-commerce, email payment gateways, passwordless authentication, payment controls, identity, and secure transaction processing, a person of ordinary skill in the art would have understood that an email-originated transaction request presents technical risks that are different from those presented by a conventional web checkout flow. In a conventional web checkout, the transaction can occur within a server-controlled session. By contrast, in an email-originated transaction, the order instruction may be generated by an email client and returned via email infrastructure, which creates technical issues regarding whether the returned message remains associated with the intended customer, campaign, transaction parameters, and valid time window.

13. My testimony in this declaration is based on my personal knowledge of the invention, my role in developing the disclosed architecture, my experience commercializing and patenting technologies in related fields, and my experience with technologies relating to secure identity, passwordless authentication, email-based communication, payments, and internet transaction systems. I understand that an inventor may submit a declaration under 37 C.F.R. § 1.132 where the inventor has knowledge of the facts being asserted, and I provide this testimony for the purpose of explaining the technical context and the factual basis for how the claimed invention would have been understood by a person of ordinary skill in the art.

II. POSITA definition

14. I understand that, for purposes of evaluating how the claimed invention would have been understood, the relevant perspective is that of a person of ordinary skill in the art at the time of the invention. I also understand that the level of ordinary skill may be informed by the types of problems encountered in the art, the prior art solutions to those problems, the sophistication of the technology, and the educational background and practical experience of persons working in the field.

15. In my opinion, the relevant field of the invention includes email-based communications, e-commerce transaction systems, customer and order database systems, internet-based payment or order workflows, and secure transaction processing. The field also includes systems that use email messages or email-generated actions as part of an authentication, authorization, order submission, payment, confirmation, or transaction-execution process.

16. A person of ordinary skill in the art at the time of the invention would have had at least a bachelor's degree in computer science, computer engineering, electrical engineering, information systems, software engineering, or a related technical field, or equivalent practical experience, and at least two years of experience developing, implementing, maintaining, or evaluating systems involving email-based communications, web-based or email-based e-commerce, customer or order databases, payment or order-processing workflows, identity or authentication workflows, or secure network transaction systems.

17. A person of ordinary skill in the art also could have acquired the relevant knowledge through substantial practical experience rather than formal academic training. For example, a person with several years of hands-on experience designing, implementing, or managing email campaign systems, internet transaction systems, payment systems, authentication systems, or secure customer/order

processing systems could qualify as a person of ordinary skill even without a formal degree in computer science or engineering. Conversely, a person with a more advanced technical degree could require less practical experience.

18. The person of ordinary skill in the art would have been familiar with the basic operation of email systems, including the generation and transmission of email messages, the role of sender and recipient addresses, the use of subject and body fields, and the practical limitations of relying on email message contents or sender information alone for transaction security. Such a person also would have understood that email messages can be forwarded, modified, resent, delayed, or generated outside the server-controlled environment of a conventional web checkout session.

19. The person of ordinary skill in the art would have been familiar with Simple Mail Transfer Protocol, or SMTP, as a protocol for transmitting email messages, and would have understood that SMTP provides a mechanism for message transport but does not, by itself, provide transaction-specific authentication, verification of order parameters against server-side transaction context, or expiration-based gatekeeping of a commercial transaction.

20. The person of ordinary skill in the art would also have been familiar with mailto hyperlinks and the use of a mailto uniform resource identifier scheme to cause an email client to generate a pre-addressed email message. Such a person would have understood that a mailto hyperlink can populate fields such as a destination address, subject line, and body content, but that the mailto scheme itself does not determine whether a resulting email message should be trusted for order execution, whether the message corresponds to a valid customer or campaign, whether the message has been altered, or whether the transaction request remains unexpired.

21. The person of ordinary skill in the art would have understood that conventional web-based e-commerce systems often rely on server-controlled sessions, user accounts, cookies, shopping carts, payment pages, web forms, or other web application mechanisms to maintain transaction state and authenticate transaction activity. By contrast, an email-originated transaction generated from a mailto hyperlink and returned by SMTP presents a different technical situation because the order instruction is generated by an email client and returned through email infrastructure, rather than being submitted entirely within a server-controlled web session.

22. The person of ordinary skill in the art would have understood that this difference creates technical problems relating to how the receiving e-commerce system determines whether the returned email message remains associated with the intended customer, campaign, product or donation amount, member identifier, subject line, and time window. The person of ordinary skill also would have recognized risks associated with altered email body fields, altered subject lines, forwarded email links, mismatched sender addresses, replayed messages, stale offers, and returned messages that no longer correspond to valid server-side transaction context.

23. The person of ordinary skill in the art would have been familiar with the use of databases to store customer information, email addresses, transaction identifiers, campaign information, order information, payment-related information, and status information. Such a person also would have understood that server-side stored data can be used to validate information received in a later transaction request, including a transaction request received through an email pathway.

24. The person of ordinary skill in the art would have been familiar with identifiers, tokens, UUIDs, and other values used to associate a later-received message or request with previously stored server-side information. The person of

ordinary skill would have understood the difference between merely placing descriptive order information in a message and using an identifier to bind the message to stored server-side authentication or transaction context.

25. The person of ordinary skill in the art would also have been familiar with authentication and authorization concepts, including the use of multiple data fields to determine whether a request should be accepted or rejected. In the context of email-originated transaction processing, such a person would have understood that validation may involve comparing information extracted from a returned email message against information previously stored by the receiving system, rather than trusting the returned email message by itself.

26. The person of ordinary skill in the art would have understood that expiration information can be important in transaction-security systems. For email-originated transactions, expiration is particularly important because an email or mailto link may remain in an inbox, may be forwarded, or may be resent after the original campaign or offer should no longer result in transaction execution.

27. In my opinion, the definition above is appropriate for the claimed invention because the claims are not limited to ordinary commercial ordering activity. Rather, the claims address the technical manner in which an e-commerce system generates an email-originated transaction path, stores server-side authentication information, receives a returned SMTP order email, parses email fields, compares the parsed information against corresponding fields of an authentication control packet, verifies expiration, and permits order execution only when the returned email message satisfies the required authentication checks.

28. My opinions in this declaration would be the same under any reasonable definition of a person of ordinary skill in the art that includes practical familiarity with email-based communications, e-commerce systems, customer/order databases, and secure transaction processing. Under such a definition, a person of ordinary skill

in the art would have understood the technical context of the invention and would have recognized the distinction between merely using email to communicate order information and using an expiration-bound, server-side authentication control packet to determine whether an email-originated order message should be trusted for order execution.

III. State of the art at filing

29. At the time of the invention, e-commerce transactions were commonly performed through browser-based checkout systems. In a conventional browser-based e-commerce transaction, a customer typically interacted with a merchant website through a web browser, selected one or more items, placed the items into a shopping cart, and completed a checkout process through one or more web pages controlled by the merchant or by a payment processor. Persons working in the field understood that web applications commonly used session-management mechanisms to maintain state across multiple requests because HTTP is stateless. *See* The OWASP Foundation, *OWASP Testing Guide v3.0* 146–47 (2008).

30. In those conventional web-based systems, the merchant's server or associated payment system ordinarily maintained transaction context during the checkout process. For example, the system could maintain a server-side session, a shopping-cart state, a logged-in user account, a cookie-based session identifier, or another browser-mediated mechanism to associate the customer, selected items, pricing, payment information, shipping information, and checkout status. Contemporaneous OWASP guidance identified an online shopping cart as a typical example in which the application must track user identity, user profile, selected products, quantities, individual prices, discounts, and related transaction state over the user's session. *See id.* at 147.

31. In a conventional web checkout flow, authentication and transaction state could be managed within the web application environment. The web server could determine whether the customer was logged in, whether the customer's session was still active, whether the shopping cart contents matched the server-side state, whether the transaction was submitted through the expected checkout page, and whether the payment process remained within a controlled workflow. Although such systems could still have security risks, the web application ordinarily had direct control over the checkout sequence and could maintain server-side state throughout the transaction. Contemporaneous OWASP guidance recognized that session-related data is vital to application security and that tampering with cookies may result in session hijacking, privilege escalation, or unauthorized influence over application operations. *See id.* at 147–48. NIST's December 2011 electronic-authentication guidance likewise recognized that remote authentication and transactions over open networks, including e-commerce transactions, presented technical challenges requiring risk-based authentication and transaction-security controls. *See* William E. Burr et al., Nat'l Inst. of Standards & Tech., Special Publ'n 800-63-1, *Electronic Authentication Guideline* vi–vii (Dec. 2011).

32. An email-originated transaction generated from a mailto hyperlink presents a materially different technical environment. A mailto hyperlink can cause a user's email client to generate an email message addressed to a designated recipient and prepopulate fields such as the subject line and body. RFC 6068 explains that the mailto URI scheme identifies resources reached using Internet mail and allows mail header fields and message bodies to be provided. *See* M. Dürst et al., *The "mailto" URI Scheme* §§ 1–2, RFC 6068, at 3–5 (Oct. 2010). Once generated, however, the resulting email message is transmitted through email infrastructure rather than through the original merchant's controlled web checkout session. RFC 6068 states that resolving a mailto URI does not cause an immediate interaction with a server;

instead, the client creates a message with fields set as defaults, and the user can edit the message, send the message unedited, or choose not to send it. *See id.* § 3, at 7.

33. In an email-originated transaction, the e-commerce system receiving the returned order email may receive the contents of the email message and message-related information, such as sender information, recipient information, subject-line information, body-field information, and email-transport information. The receiving system does not necessarily receive the same type of continuous session state that would exist in a conventional web checkout. SMTP is directed to transferring electronic mail reliably and efficiently, and SMTP mail may pass through intermediate relay or gateway hosts on its path from sender to recipient. *See* J. Klensin, *Simple Mail Transfer Protocol* § 1.1, RFC 5321, at 5 (Oct. 2008). Accordingly, the returned email may arrive after a delay, may be generated by a local email client, may pass through intermediate mail systems, and may be received outside the context of the email that originally caused the transaction opportunity to be presented.

34. A person of ordinary skill in the art would have understood that SMTP and mailto functionality provide useful message-generation and message-transport mechanisms, but they do not, by themselves, solve the transaction-security problem created when an order instruction is generated outside a controlled web checkout session. SMTP transports email messages, while mailto assists in composing a message and setting default fields. *See* RFC 5321, *supra*, § 1.1, at 5; RFC 6068, *supra*, §§ 1–3, at 3–7. But neither SMTP nor the mailto URI scheme inherently provides a server-side transaction authentication packet, field-by-field validation of returned order content against stored transaction context, or expiration-based gatekeeping before order execution.

35. As a result, a person of ordinary skill in the art would not have assumed that a returned email message should be trusted merely because it was received at

the e-commerce system. RFC 5321 expressly recognizes that SMTP mail is inherently insecure because users can create messages that may trick recipients into believing the messages came from somewhere else, and because SMTP mail inherently cannot provide authentication or integrity checks at the transport level. *See* RFC 5321, *supra*, § 7.1, at 75. The receiving system would therefore still need to determine whether the email message corresponds to the intended customer, campaign, product or donation amount, member identifier, subject line, sender address, and valid time window. Without a mechanism for making those determinations, the system could execute an order based on an email message that is stale, altered, forwarded, replayed, or otherwise not properly bound to the original transaction context.

36. One technical risk is message alteration. In an email-originated transaction, information placed in the body or subject line of a generated email may be visible to, or modifiable by, the user or by software operating on the user's device before the message is sent. This is consistent with RFC 6068's statement that, after a mailto URI is resolved, the user can edit the message before sending it. *See* RFC 6068, *supra*, § 3, at 7. For example, product identifiers, quantities, donation amounts, campaign identifiers, member identifiers, or subject-line information could be changed before the message is transmitted back to the e-commerce system. If the receiving system merely trusted the received email contents, an altered message could cause execution of a transaction that does not correspond to the original transaction opportunity.

37. A related technical risk is subject-line or body-field mismatch. In a mailto-generated order message, different pieces of transaction information may be placed in different portions of the email message, including the body field and subject field. RFC 6068 recognizes that the "subject" header field and "body" are among the limited fields generally expected to be understood by mailto resolvers,

but also explains that mailto header-field name/value pairs do not directly correspond to final mail-message headers and that mail clients create new fields as they would for any new message. *See id.* §§ 2–4, at 5–8. A person of ordinary skill in the art would have understood that these fields are not inherently self-authenticating. If the receiving system does not verify that the returned subject line and body-field parameters correspond to the server-side transaction context, the received email may not reliably indicate that the customer selected the intended transaction option from the original email.

38. Another technical risk is forwarding. Emails and mailto hyperlinks can be forwarded from one recipient to another. A forwarded email may allow a different user or a different email account to activate a link that was originally generated for a particular customer, member, campaign, offer, or transaction. In that circumstance, the returned order email may include some information copied from the original message, but the sender of the returned message may not be the intended customer or member associated with the original server-side transaction context. This risk is exacerbated by the fact that a mailto URI does not create an immediate server interaction, but instead creates a user-editable message through the user's mail client. *See id.* § 3, at 7.

39. Another technical risk is spoofing or sender mismatch. A returned email may appear to come from a particular address, or may include transaction data associated with a particular customer, but the receiving system still needs to determine whether the sender address and the returned transaction fields correspond to the customer and transaction information that the system previously stored. RFC 5321 recognized before the time of the invention that SMTP mail could be spoofed and could not provide authentication or integrity checks at the transport level. *See* RFC 5321, *supra*, § 7.1, at 75. A person of ordinary skill in the art therefore would have understood that relying only on visible email content or an asserted sender

address would not provide the same type of transaction-state control available in a server-managed checkout session.

40. Another technical risk is replay. An email-generated transaction request can be resent or reused after the original transaction opportunity was created. NIST defined a replay attack as an attack in which an attacker replays previously captured messages between a legitimate claimant and verifier to masquerade as that claimant to the verifier or vice versa. *See* NIST SP 800-63-1, *supra*, at 13. NIST further explained that replay resistance can be achieved where protocols use nonces, challenges, or timeliness data to prove the freshness of a transaction. *See id.* at 71. Because email messages may remain in an inbox or sent folder for an extended period of time, a user or another actor may attempt to resend a prior order email or reactivate an old mailto link. Without expiration-based validation, the system may be unable to distinguish a current transaction request from a stale or replayed request that should no longer result in order execution.

41. Stale offer execution is a related problem. Email campaigns may be time-limited, price-limited, inventory-limited, or otherwise tied to a particular campaign window. A mailto link in an initial email may remain clickable after the campaign has expired or after the associated price, donation amount, item availability, or business rules have changed. In the web-session context, contemporaneous OWASP guidance recognized that session tokens should be destroyed or made unusable and that proper controls should be enforced server-side to prevent later reuse; OWASP also stated that "everything should be checked server-side" and that resetting session cookies to previous values should not restore application access. *See* OWASP Testing Guide v3.0, *supra*, at 133–35. A person of ordinary skill in the art would have understood that an e-commerce system using email-originated transaction requests similarly needs a way to determine whether a

returned order email remains within a valid time window before the system performs order execution.

42. Another technical problem is the loss of web-session context. In a browser-based checkout, the system may maintain a live session that ties the customer's actions to a server-side transaction state. In an SMTP/mailto-based workflow, the order instruction is generated by an email client and returned through email infrastructure. That returned email is not inherently tied to a live authenticated web session, a server-side shopping cart session, or a continuous browser-mediated checkout flow. NIST recognized that authentication and data-transfer security depend on the relationship between an authentication protocol and the subsequent session protocol used to transfer data, including whether authentication is bound to the data-transfer session. *See* NIST SP 800-63-1, *supra*, at 71. Therefore, the receiving e-commerce system must use another mechanism to bind the returned message to the original customer, campaign, transaction parameters, and validity period.

43. The technical problem addressed by the claimed invention arises from this difference in architecture. The problem is not merely that a customer may place an order. The problem is how an e-commerce system can securely determine whether an order email generated from a mailto hyperlink and returned through SMTP should be trusted for transaction execution when the message is received outside a conventional server-controlled checkout session. This technical problem is consistent with the recognized need, in remote authentication and transaction systems, to resist replay, session hijacking, manufacture or modification of assertions, and misuse of credentials or transaction messages over open networks. *See id.* at vi–vii, 71; *see also* NIST SP 800-63-1, *supra*, at 8, 11, 13.

44. A person of ordinary skill in the art would have understood that the claimed invention addresses that problem by creating and using server-side

authentication context for the email-originated transaction. In particular, the system associates a UUID with an authentication control packet that includes transaction-specific fields such as the client identifier, email campaign identifier, item or donation amount, subject line, customer email address, member identifier, and timestamp including an expiration date. This approach is consistent with the contemporaneous web-security principle that sensitive transaction or session state should be validated against trusted server-side controls rather than accepted merely because client-visible data is presented. *See* OWASP Testing Guide v3.0, *supra*, at 133–35, 147–48. The returned email is then parsed and evaluated against that stored authentication context before order execution is permitted.

45. In my opinion, the authentication control packet is technically significant because it provides a server-side reference against which a later-received SMTP order email can be validated. Rather than relying only on the fact that an email message was received, the system evaluates whether the returned email corresponds to the previously stored transaction context. This changes the information flow of the email-originated transaction: the system first creates server-side authentication context, then receives a mailto-generated SMTP message, then parses returned message fields, then verifies those fields against the stored context, and only then conditionally performs order execution. RFC 4122 separately cautioned that UUIDs should not be treated as security capabilities whose mere possession grants access; the claimed architecture does not rely on mere UUID possession, but instead uses the UUID in conjunction with the server-side authentication control packet, field comparison, and expiration verification. *See* P. Leach et al., *A Universally Unique IDentifier (UUID) URN Namespace* § 6, RFC 4122, at 16 (July 2005).

46. The expiration-date verification is also technically significant. Because email messages and mailto links can persist indefinitely in a user's email account, a

receiving e-commerce system needs a mechanism for determining whether the returned email remains timely. NIST recognized that replay resistance may be achieved by freshness mechanisms, including nonces, challenges, or timeliness data, allowing a verifier to detect old messages that do not correspond to the current authentication session. *See* NIST SP 800-63-1, *supra*, at 71. The claimed expiration verification prevents the system from executing an order merely because an old email or old link was later used. Instead, the system verifies that the UUID remains unexpired based on the expiration date before authenticating the transaction and permitting order execution.

47. The claimed architecture therefore addresses risks that arise specifically from using email as the transaction-originating pathway. A returned SMTP order email can be delayed, forwarded, modified, resent, spoofed, or separated from the original server-controlled transaction context. RFC 6068 confirms that a mailto URI results in creation of a user-editable message rather than an immediate server interaction, and RFC 5321 confirms that SMTP does not provide transport-level authentication or integrity for the user and message path as a whole. *See* RFC 6068, *supra*, § 3, at 7; RFC 5321, *supra*, § 7.1, at 75. The claimed system mitigates those risks by requiring the returned email to match server-side authentication information and by requiring the UUID to remain unexpired. In this way, the system imposes a security gate between receipt of the email-generated order message and execution of the transaction.

48. A person of ordinary skill in the art would have understood that this is different from merely using email to advertise a product or receive an order. The claimed invention does not simply instruct a generic computer to complete a sale. It defines a particular email-originated transaction architecture in which the e-commerce system conditions order execution on validation of a returned SMTP order email against stored server-side authentication context and expiration

information. This is consistent with recognized security principles that state-changing or transaction-related requests should not be accepted merely because they are presented to a server; the system should verify that the request is authorized, current, and bound to the correct server-side context. *See* NIST SP 800-63-1, *supra*, at 71; OWASP Testing Guide v3.0, *supra*, at 133–35, 147–48.

49. In my opinion, the technical problem and solution described above would have been apparent to a person of ordinary skill in the art from the specification and claims. The specification describes UUIDs, authentication control packets, selectable email hyperlinks or buttons, returned emails, parsing of email content, comparison of returned information against stored authentication information, positive or negative security qualification, and transaction continuation or denial based on the result. The claims reflect that disclosed architecture by requiring the mailto-generated order email, UUID, authentication control packet, field comparison, expiration verification, authentication based on the comparison, and conditional order execution. The cited technical references confirm that the problem arises from the technical characteristics of mailto message creation, SMTP message transport, lack of inherent SMTP authentication/integrity, replay risk, and the need for server-side state or context to authenticate transaction-related requests. *See* RFC 6068, *supra*, §§ 1–4, at 3–8; RFC 5321, *supra*, §§ 1.1, 7.1, at 5, 75; NIST SP 800-63-1, *supra*, at 8, 11, 13, 71; RFC 4122, *supra*, § 6, at 16; OWASP Testing Guide v3.0, *supra*, at 133–35, 147–48.

50. I understand that the Examiner has characterized the claims as being directed to advertising, marketing, and sales activities. From a technical perspective, however, a person of ordinary skill in the art would have understood that the claimed limitations address how the receiving e-commerce system technically determines whether an email-originated transaction request is authentic, current, and bound to the intended server-side transaction context before allowing order execution. That

determination is a technical security problem arising from the use of an SMTP/mailto transaction pathway outside a conventional web checkout session, not merely a business rule for completing a sale.

IV. The Claimed Architecture and How It Changes Information Flow

51. I understand that the Examiner has characterized the claimed invention as using generic computer components and a standard mailto URI scheme to perform advertising, marketing, or sales activity. From a technical perspective, I do not contend that SMTP, mailto hyperlinks, databases, processors, memory, or network interfaces were newly invented. Rather, my opinion is that a person of ordinary skill in the art would have understood the claimed invention as using those components in a specific architecture that changes how information flows through an email-originated transaction before the system permits order execution. The specification describes an e-commerce system including an e-commerce database, a security module, a message processing module, and an email interface module that cooperate to generate, transmit, receive, parse, authenticate, and process email-based transactions. Application, ¶¶ [0036], [0040]-[0043].

52. In a conventional mailto-based interaction, a mailto hyperlink may cause an email client to generate a message with a destination address, subject line, and body content. That functionality, by itself, does not create a server-side authentication context, does not verify whether the returned message corresponds to a particular customer or campaign, does not determine whether the returned message has been altered, and does not determine whether the transaction opportunity remains unexpired. The claimed architecture adds those controls by creating a server-side authentication control packet before the returned SMTP order email is received, embedding or including a UUID in the mailto-generated transaction path, parsing the returned order email, verifying returned fields against the stored

authentication control packet, enforcing expiration, and allowing order execution only after authentication succeeds. Application, ¶¶ [0052]-[0058], [0068]-[0069], [0074]-[0084].

53. In my opinion, that architecture changes the information flow of the transaction. The system does not merely send an advertisement and receive an order. Instead, before the order email is received, the e-commerce system stores authentication context in the database. The mailto-generated email then carries a UUID and selected transaction parameters back to the e-commerce system. After the order email is received, the system parses the returned email and validates it against the server-side authentication context. Only after the returned message is authenticated does the system perform order execution. That ordered flow is materially different from trusting an email message simply because it was received at the e-commerce system. Application, ¶¶ [0054]-[0058], [0074]-[0080], [0153]-[0155].

54. As one example, an initial email may be generated for Customer A with a mailto hyperlink associated with Campaign X, Product Y, Quantity 2, Subject Line Z, and an expiration date. If the mailto hyperlink is activated, the returned SMTP order email is not accepted merely because it includes Product Y and Quantity 2. Instead, the e-commerce system checks whether the returned UUID, product or amount information, campaign information, subject line, sender email address, and expiration information correspond to the stored authentication control packet. If the email was forwarded to Customer B, altered to change the quantity, or resent after expiration, the comparison or expiration check may prevent order execution. A person of ordinary skill in the art would have understood that this example reflects the claimed architecture because the returned email is treated as an input to authentication rather than as a self-validating order instruction.

A. Server-Side Authentication Control Packet

55. The server-side authentication control packet is central to the claimed architecture. The application describes generating a UUID, associating the UUID with authentication information, creating an authentication control packet, and storing the UUID and authentication control packet for later use in validating a returned email. Application, ¶¶ [0054]-[0058], [0068]-[0069]. A person of ordinary skill in the art would have understood that this packet creates a stored server-side reference for later validating a returned SMTP order email.

56. The authentication control packet is not inherent in SMTP or in the mailto URI scheme. SMTP can transport an email message, and a mailto hyperlink can cause an email client to generate a message, but those mechanisms do not themselves create a server-side packet that binds together the intended customer, campaign, transaction amount, subject line, member identifier, sender-related information, and expiration window. The specification describes an authentication control packet including fields such as UUID, client identifier, email campaign, item or donation amount, email address, subject line, member, timestamp, and additional fields for authentication or verification. Application, ¶¶ [0057]-[0067].

57. In my opinion, a person of ordinary skill in the art would have understood this as a technical change in the transaction architecture. The transaction is not executed based solely on the contents of the returned email. The returned email is instead treated as an input to a verification architecture. The e-commerce system determines whether that input matches a previously stored authentication control packet. This changes the role of the email from a mere order instruction into a message that must be authenticated against server-side transaction context before the system acts on it. Application, ¶¶ [0053], [0074]-[0080].

58. The authentication control packet also addresses the loss of conventional web-session context. In a web checkout, the server may maintain a live session tying the user, cart, pricing, and checkout flow together. In the claimed email-originated transaction, the returned order email arrives through SMTP rather than through a continuous web checkout session. The authentication control packet substitutes for that missing live checkout context by storing the transaction-specific information needed to determine whether the returned email remains associated with the intended customer, campaign, item or donation amount, subject line, member identifier, and expiration window. Application, ¶¶ [0054]-[0067], [0076]-[0080].

B. UUID Embedded in the Mailto-Generated Message

59. The UUID provides the link between the returned SMTP order email and the stored authentication control packet. The specification describes generating UUIDs, associating the UUIDs with authentication information, storing the UUIDs in the database, embedding UUIDs in selectable hyperlinks or buttons, and later parsing returned emails for those UUIDs. Application, ¶¶ [0037]-[0039], [0052]-[0058], [0068]-[0069], [0086]-[0087]. A person of ordinary skill in the art would have understood that the UUID allows the e-commerce system to associate the later-received order email with the correct server-side authentication context.

60. In the claimed architecture, the UUID is not merely an order number or a visible transaction code. It functions as a linking value between the returned SMTP order email and a preexisting server-side authentication control packet. This allows the e-commerce system to validate the returned email against information that was stored before the return email was received. Application, ¶¶ [0054]-[0058], [0076]-[0079].

61. The application expressly supports the opaque nature of the UUID. It explains that UUIDs delivered in an email may be opaque and that the authentication

information associated with the UUID is not derivable from parsing the UUID itself. Application, ¶ [0054]. That architecture is technically significant because the UUID does not itself need to expose the underlying customer, campaign, amount, subject-line, member, or expiration information. Instead, the UUID allows the system to locate or identify the trusted server-side packet against which the returned email will be verified. Application, ¶¶ [0054]-[0058], [0068]-[0069], [0076]-[0077].

62. This reduces reliance on the returned email message as the source of truth. If the returned email contains visible transaction information, that information may be altered, forwarded, or resent. In the claimed architecture, the system does not simply trust those returned values. The UUID links the returned message to stored server-side context, and the system then performs a comparison between the returned information and corresponding packet fields. The information flow therefore runs from server-side packet creation, to UUID-linked message generation, to returned-email parsing, to server-side verification, rather than from received email directly to order execution. Application, ¶¶ [0053]-[0058], [0074]-[0080].

63. A person of ordinary skill in the art would have understood that this is different from merely putting descriptive order information in the body of an email. Descriptive information in an email may tell the recipient what a user appears to be ordering. The claimed UUID-linked authentication control packet architecture determines whether the returned email should be trusted for order execution. Application, ¶¶ [0054]-[0058], [0076]-[0080].

C. Parsing the Returned SMTP Order Email

64. The claimed parsing steps are part of the authentication architecture, not merely routine reading of an order. The application describes returned emails being parsed for embedded identifiers and other authentication information. Application, ¶¶ [0053], [0074]-[0077]. It also describes order email bodies

containing fields such as product identifiers, customer identifiers, quantity, and campaign identifiers. Application, ¶¶ [0105]-[0108], [0121], [0124]-[0125].

65. In my opinion, a person of ordinary skill in the art would have understood that the parsing steps serve a technical verification purpose. The e-commerce system parses the returned email because the system must determine whether the message corresponds to the server-side authentication context that was created before the returned email was received. The parsing therefore extracts data needed for authentication, including the UUID, transaction parameters, and subject line. Application, ¶¶ [0053], [0074]-[0077].

66. The sender email address is also part of the verification architecture. The specification describes authentication control packet fields including an email address and describes comparing parsed return-email information against authentication information associated with the UUID. Application, ¶¶ [0057]-[0067], [0076]-[0079]. A person of ordinary skill in the art would have understood that this comparison helps determine whether the returned message originated from the expected customer email address rather than from an unintended recipient, a forwarded link, or a mismatched account.

67. The parsing architecture is important because a mailto-generated order email can be created outside the e-commerce system's direct control. The returned email may include body content, subject-line content, and sender information, but the receiving system must decide whether those values should be trusted. The claimed architecture uses parsing as the first step in that validation process. Application, ¶¶ [0074]-[0080], [0117]-[0121].

68. Stated differently, the parsing steps are not merely extracting order details so that a sale can be completed. They are extracting authentication inputs from a returned SMTP message so that the e-commerce system can compare those

inputs against a trusted authentication control packet and determine whether order execution should be permitted. Application, ¶¶ [0053], [0076]-[0084].

D. Field-by-Field Verification

69. The field-by-field comparison is the gatekeeping mechanism that changes the system's behavior. The application describes selecting parameters from the authentication control packet, comparing those parameters with parsed return-email information, and continuing the security qualification process until selected parameters have been reviewed and qualified. Application, ¶¶ [0076]-[0079]. A person of ordinary skill in the art would have understood that this comparison verifies more than the existence of an order. It verifies whether the returned email remains consistent with the intended transaction context.

70. The comparison checks whether the returned email corresponds to the intended customer. By verifying the sender email address against the customer email address or corresponding packet information, the system can identify a mismatch between the expected customer and the email account from which the order message is received. This is important because email links can be forwarded or reused, and the sender of the returned message may not be the intended customer. Application, ¶¶ [0057]-[0067], [0076]-[0080].

71. The comparison checks whether the returned email corresponds to the intended campaign. By verifying the campaign identifier against the authentication control packet, the system can determine whether the returned order email is associated with the correct email campaign. This matters because different campaigns may have different offers, timing, pricing, donation options, or member associations. Application, ¶¶ [0060]-[0067], [0105]-[0108], [0121], [0140].

72. The comparison checks whether the returned email corresponds to the intended item or donation amount. The application describes authentication control

packet fields including item or donation amount, and it also describes order-email body fields including product identifiers, quantities, and campaign identifiers. Application, ¶¶ [0061], [0105]-[0108], [0121], [0138]-[0140]. This prevents the system from executing an order simply because an email contains an item or amount. Instead, the returned value must correspond to the stored server-side authentication context.

73. The comparison checks whether the returned email corresponds to the intended subject line. The specification describes the authentication control packet as including a subject-line field and describes use of email subject and body information in generated and returned emails. Application, ¶¶ [0064], [0094]-[0096], [0103]-[0108]. Subject-line verification is technically meaningful in a mailto-generated email because the subject field is a separate field from the body field and can carry information used to associate the returned message with a particular campaign, transaction type, or order-processing path.

74. The comparison checks whether the returned email corresponds to the intended member or customer association. The specification describes member information, customer information, email addresses, and authentication-control-packet fields that can be used to bind the returned email to the person or account for whom the original transaction opportunity was generated. Application, ¶¶ [0043]-[0048], [0057]-[0067]. That binding is important because an email-originated transaction does not inherently occur within a live authenticated web session.

75. Finally, the comparison checks whether the UUID remains within the valid time window. The specification identifies expiration-date information as authentication information associated with the UUID and also describes a timestamp field in the authentication control packet. Application, ¶¶ [0054], [0066]. This prevents order execution based on stale or replayed email-originated transaction messages.

76. In my opinion, the field-by-field verification makes the claimed system materially different from a system that merely receives an order email and fulfills the order. The claimed system does not use the returned email as a self-validating order instruction. It treats the returned email as a message that must be authenticated by matching extracted fields to corresponding fields in a server-side packet, including expiration information. Application, ¶¶ [0053]-[0058], [0076]-[0084].

E. Expiration Enforcement

77. Expiration enforcement is a particularly important aspect of the claimed architecture. Email messages can remain in a recipient's inbox, sent folder, archive, or forwarded chain indefinitely. A mailto hyperlink in an old email may remain clickable long after the original offer, campaign, price, donation opportunity, or member-specific transaction should no longer be valid. The application describes expiration-date information associated with UUID-based authentication information. Application, ¶ [0054].

78. The application supports the expiration-bound authentication architecture by describing authentication information associated with a UUID, including an expiration date, and by identifying a timestamp field in the authentication control packet. Application, ¶¶ [0054], [0066]. A person of ordinary skill in the art would have understood that this prevents the system from executing an order merely because an old email or old mailto-generated message is later presented to the e-commerce system.

79. The expiration check is not merely a business rule about when an offer ends. In the claimed architecture, expiration is part of the technical authentication process. The system determines whether the UUID associated with the returned email remains valid before authenticating the transaction. If the UUID is expired, the transaction should not proceed through the normal order execution path because

the returned email no longer corresponds to a valid authentication window. Application, ¶¶ [0053]-[0054], [0077]-[0084].

80. This expiration-based gatekeeping addresses replay and stale-link risks. A replayed email, a resent order email, or a stale mailto link may contain the same visible data that once corresponded to a valid transaction opportunity. Without expiration verification, the receiving e-commerce system might not distinguish the stale message from a current one. By verifying expiration before authentication and order execution, the claimed system imposes a technical validity check between message receipt and transaction execution. Application, ¶¶ [0054], [0066], [0076]-[0084].

81. From a technical perspective, the expiration-bound authentication control packet is not merely a generic instruction to complete an order. It is a specific security mechanism for an email-originated transaction path. The application describes positive and negative security qualification, transaction continuation, transaction denial, and further measures when comparison or authentication fails. Application, ¶¶ [0053], [0077], [0080]-[0085].

F. Conditional Order Execution Based on Authentication

82. The claimed architecture culminates in conditional order execution. The application describes continuing or denying a transaction based on security qualification and performing the order execution procedure when the required checks are successful. Application, ¶¶ [0080]-[0084], [0153]-[0155]. This means that the system does not execute the order merely because an email was received. The system executes only after the returned email has passed the required authentication checks.

83. This conditional execution changes the operational relationship between receipt of an email and completion of a transaction. In a simple email-order

system, receipt of an email may itself be treated as the user's order. In the claimed system, receipt of the email initiates a validation process. The returned message must be parsed, matched against the authentication control packet, checked for sender and field consistency, and verified as unexpired. Only then does the system perform order execution. Application, ¶¶ [0053], [0074]-[0084], [0153]-[0155].

84. A person of ordinary skill in the art would have understood that this architecture provides a technical security gate for SMTP/mailto-based transaction processing. The gate is implemented through the ordered relationship among the stored authentication control packet, UUID, mailto-generated order email, parsed email fields, field-by-field comparison, expiration verification, authentication result, and conditional order execution. Application, ¶¶ [0054]-[0058], [0068]-[0069], [0074]-[0084].

85. The system therefore changes both how information flows and when the system acts. Information does not flow directly from initial email to returned order email to order execution. Instead, the system establishes server-side authentication context before the returned email is received; the returned email supplies extracted values that are used for verification; and order execution is permitted only after the returned values are validated against that server-side context. Application, ¶¶ [0054]-[0058], [0074]-[0084], [0153]-[0155].

86. In my opinion, a person of ordinary skill in the art would have understood this as an improvement to the security and reliability of email-originated transaction processing. The improvement is not the invention of SMTP or mailto. The improvement is the architecture that uses SMTP and mailto in a constrained way: a returned SMTP order email generated from a mailto hyperlink is not trusted for order execution unless it matches an expiration-bound authentication control packet stored by the e-commerce system. Application, ¶¶ [0053]-[0058], [0068]-[0069], [0074]-[0084].

87. This architecture is consistent with the application's description of UUID generation, opaque UUID usage, authentication control packets, embedded identifiers in selectable email hyperlinks or buttons, returned-email parsing, parameter comparison, positive or negative security qualification, and transaction continuation or denial based on authentication results. Application, ¶¶ [0052]-[0058], [0068]-[0069], [0074]-[0085], [0086]-[0087].

88. For these reasons, my opinion is that a person of ordinary skill in the art would have understood the claimed invention as addressing a technical security problem arising from email-originated transaction processing. The claims do not merely use a generic computer to complete a sale. They define an ordered technical architecture for deciding whether a returned SMTP order email generated from a mailto hyperlink should be trusted before the e-commerce system executes the transaction. Application, ¶¶ [0053]-[0058], [0074]-[0084], [0153]-[0155].

V. Claimed Improvement to Technology

A. Improvement to SMTP/Mailto Transaction Security

89. In my opinion, a person of ordinary skill in the art would have understood that the claimed invention improves the security and reliability of email-originated transaction processing. The improvement is not that SMTP, mailto hyperlinks, UUIDs, databases, or processors were newly invented. Rather, the improvement is the specific architecture by which the e-commerce system uses those components to create a verification layer for an order email generated from a mailto hyperlink and returned through SMTP before the system performs order execution.

90. SMTP and mailto functionality perform different, limited technical roles. SMTP provides a mechanism for transferring email messages. *See* J. Klensin,

Simple Mail Transfer Protocol § 1.1, RFC 5321, at 5 (Oct. 2008). A mailto URI allows an email client to generate a message with certain fields, such as destination, subject, and body information, but resolving a mailto URI does not itself cause an immediate interaction with a server, and the user may edit the generated message before sending it. *See* M. Dürst et al., *The "mailto" URI Scheme* §§ 1–3, RFC 6068, at 3–7 (Oct. 2010). Those standards confirm that SMTP and mailto, standing alone, do not provide the claimed server-side authentication control packet, field-by-field verification, or expiration-based gatekeeping before transaction execution.

91. A person of ordinary skill in the art also would have understood that SMTP email presented recognized security limitations. RFC 5321 expressly recognized that SMTP mail is inherently insecure because users can create messages that appear to come from another source, and that SMTP mail inherently cannot provide authentication or integrity checks at the transport level. *See* RFC 5321, *supra*, § 7.1, at 75. That limitation is technically significant in the context of the claimed invention because the claimed transaction begins with a mailto-generated email message and returns through email infrastructure rather than through a continuous server-controlled checkout session.

92. The claimed invention addresses that problem by imposing an authentication architecture between receipt of the returned email and execution of the transaction. The application describes generating UUIDs, embedding or including UUIDs in selectable email links or buttons, associating UUIDs with authentication information, storing an authentication control packet, parsing returned emails for UUIDs and other authentication information, comparing parsed information with selected fields of the authentication control packet, and generating positive or negative security qualification indications. Application, ¶¶ [0052]-[0058], [0068]-[0069], [0074]-[0085].

93. A person of ordinary skill in the art would have understood that this architecture improves SMTP/mailto transaction security because it prevents the e-commerce system from executing an order merely because an email was received. Instead, the returned SMTP order email is treated as an input to a verification process. The system parses the returned email, identifies the UUID and associated parameters, verifies those values against server-side authentication information, verifies that the UUID remains within its valid time window, and executes the transaction only if the authentication process succeeds. Application, ¶¶ [0053]-[0058], [0076]-[0084], [0153]-[0155].

94. The authentication control packet is technically significant because it creates server-side transaction context for an email-originated transaction. In a conventional web checkout, transaction state may be maintained through a server-side session, shopping cart, logged-in account, or similar browser-mediated workflow. In the claimed email-originated transaction, the returned order email is not submitted through that same continuous web-session path. The authentication control packet therefore provides a stored reference for determining whether the returned email remains associated with the intended customer, email campaign, item or donation amount, subject line, member identifier, and validity period. Application, ¶¶ [0054]-[0067], [0076]-[0080].

95. This improvement is consistent with contemporaneous web-security principles. Before the effective filing date, OWASP recognized that web applications use session management to maintain user identity and transaction state across multiple requests, and identified online shopping carts as examples in which applications track user identity, selected products, quantities, prices, discounts, and related transaction state. *See* The OWASP Foundation, *OWASP Testing Guide v3.0* 146–48 (2008). A person of ordinary skill in the art would have understood that an email-originated order flow lacks the same continuous browser-session context and

therefore requires a different mechanism for binding a later-received message to trusted server-side transaction information.

96. The claimed UUID-based architecture provides that mechanism. The UUID does not merely describe what the user appears to be ordering. Instead, it links the returned SMTP order email to a server-side authentication control packet. The application expressly describes UUIDs delivered in an email as opaque and explains that the associated authentication information is not derivable from parsing the UUID itself. Application, ¶ [0054]. A person of ordinary skill in the art would have understood that this is important because the returned email message is not the sole source of truth. The UUID permits the e-commerce system to associate the returned message with trusted server-side authentication context.

97. The claimed field-by-field comparison also provides a technical improvement. The system verifies returned email information against corresponding fields of the authentication control packet, including fields such as the UUID, client identifier, email campaign identifier, item or donation amount, subject line, email address, member identifier, and timestamp. Application, ¶¶ [0057]-[0067], [0076]-[0079]. A person of ordinary skill in the art would have understood that this comparison reduces the risk that altered, mismatched, forwarded, spoofed, stale, or replayed email messages will trigger order execution.

98. This field-by-field verification is materially different from merely receiving an email order. In a simple email-order workflow, the system may treat the email as the order instruction. In the claimed architecture, the email is not self-authenticating. The system checks whether the returned message still matches the preexisting authentication control packet before it executes the order. Application, ¶¶ [0053], [0076]-[0084]. That verification changes how the system treats the returned message: receipt of the email initiates authentication, not execution.

99. The expiration aspect of the authentication control packet is another important technological improvement. Email messages and mailto links can persist in a user's inbox, archive, sent folder, or forwarded email chain for an indefinite period of time. The claimed architecture addresses that problem by associating expiration information with the UUID and by requiring that the UUID be verified as unexpired before authentication and order execution. Application, ¶¶ [0054], [0066], [0076]-[0084].

100. A person of ordinary skill in the art would have understood the expiration check as a replay-resistance and stale-message control, not merely a business preference about offer timing. Before the effective filing date, NIST described replay attacks as attacks in which previously captured messages are replayed, and explained that replay resistance can use nonces, challenges, or timeliness data to prove transaction freshness. *See* William E. Burr et al., Nat'l Inst. of Standards & Tech., Special Publ'n 800-63-1, *Electronic Authentication Guideline* 13, 71 (Dec. 2011). In the claimed invention, the expiration date associated with the UUID performs a similar technical role: it helps the e-commerce system determine whether the returned SMTP order email is still fresh enough to be trusted for transaction execution.

101. RFC 4122 also supports the technical significance of not relying solely on possession of a UUID. RFC 4122 cautions that UUIDs should not be used as security capabilities because a predictable or known UUID should not by itself authorize access. *See* P. Leach et al., *A Universally Unique IDentifier (UUID) URN Namespace* § 6, RFC 4122, at 16 (July 2005). The claimed architecture is consistent with that principle because it does not rely merely on receipt of a UUID. Instead, the UUID is associated with a server-side authentication control packet, and the system verifies multiple returned email fields and expiration information before authenticating the transaction. Application, ¶¶ [0054]-[0067], [0076]-[0084].

102. The claimed invention also improves the reliability of email-originated transaction processing by accounting for the technical fragility of email identity and message contents. SPF was later standardized because preexisting email protocols did not restrict what a sending host could claim in SMTP identity fields. *See* S. Kitterman, *Sender Policy Framework (SPF) for Authorizing Use of Domains in Email, Version 1* § 1, RFC 7208, at 4–5 (Apr. 2014). Although RFC 7208 post-dates the effective filing date, it confirms the technical problem already reflected in RFC 5321: baseline SMTP did not itself provide reliable sender authentication. *See* RFC 5321, *supra*, § 7.1, at 75.

103. The claimed invention does not attempt to solve email security by modifying SMTP itself or by creating a new mailto URI standard. Instead, it imposes a transaction-specific verification layer at the e-commerce system. The receiving system evaluates the returned email in view of stored server-side authentication information, including the UUID, customer email address, campaign information, subject line, member information, transaction parameters, and expiration information. Application, ¶¶ [0054]-[0067], [0076]-[0084]. That architecture allows the system to use ordinary SMTP/mailto functionality while adding authentication constraints that SMTP and mailto do not provide by themselves.

104. In my opinion, this is a technological improvement because it changes the security posture of the system. Without the claimed architecture, a returned SMTP email could be treated as an order instruction based primarily on its contents and apparent sender. With the claimed architecture, the returned email must pass a server-side authentication process before order execution. The system therefore reduces the risk that an altered, spoofed, forwarded, replayed, or stale email-generated message will result in execution of an unauthorized or invalid transaction. Application, ¶¶ [0053], [0074]-[0085], [0153]-[0155].

105. I understand that the Office Action discusses the expiration-bound authentication-control-packet features. From a technical perspective, those features are important to the claimed architecture because they provide a freshness and validity check for an email-originated transaction request before the system permits order execution. A person of ordinary skill in the art would have understood that this expiration-bound verification changes the system's treatment of a returned SMTP order email from a self-validating order instruction into a message that must satisfy a server-side authentication condition before execution

B. Improvement Reflected in the Claims

106. The technological improvement described above is reflected in the pending claims. I understand that the pending independent claims recite storing, in a database, information including a unique identifier for the e-commerce transaction and an email address of a customer. A person of ordinary skill in the art would have understood this as establishing stored server-side information used to authenticate a later-received email-originated transaction request. Application, ¶¶ [0036], [0043]-[0048], [0054]-[0058].

107. The pending independent claims also recite configuring a mailto hyperlink that conforms to a mailto uniform resource identifier scheme and that, when activated, generates an order email message addressed to an email address of an e-commerce system. The mailto hyperlink includes, in a body field, a UUID and additional parameters, and includes, in a subject field, a subject line. The application describes mailto hyperlinks that generate return or order emails, including examples in which body fields include product, customer, quantity, and campaign information, and subject/body information is used in email-based transaction processing. Application, ¶¶ [0094]-[0097], [0103]-[0108], [0117]-[0125].

108. The claims further recite that the UUID is associated in the database with an authentication control packet. The application describes generating UUIDs, associating UUIDs with authentication information, and creating and storing authentication control packets that include fields such as UUID, client identifier, email campaign, item or donation amount, email address, subject line, member, timestamp, and other fields. Application, ¶¶ [0054]-[0067], [0068]-[0069]. A person of ordinary skill in the art would have understood that this limitation reflects the server-side authentication context that enables the returned SMTP order email to be verified.

109. The claims further recite transmitting an initial email message via SMTP and receiving, via SMTP, an order email message generated by activation of the mailto hyperlink. The application describes transmitting initial email messages that include selectable hyperlinks or buttons, receiving return emails generated by user selection, and using email interface and message-processing modules to process those messages. Application, ¶¶ [0040]-[0041], [0071]-[0074], [0117]-[0121], [0141]-[0143].

110. The claims further recite parsing the body field of the order email message to extract the UUID and additional parameters, and parsing the subject field to extract the subject line. The application describes parsing returned emails for embedded identifiers and other authentication information, and describes order email fields containing transaction-related parameters such as product identifiers, customer identifiers, quantities, and campaign identifiers. Application, ¶¶ [0053], [0074]-[0077], [0105]-[0108], [0121], [0124]-[0125].

111. The claims further recite performing a comparison in which the UUID, additional parameters, subject line, and sender email address are verified against corresponding fields of the authentication control packet. The application describes comparing the UUID and other authentication information from the returned email

against selected parameters of the authentication control packet, continuing until selected parameters have been reviewed, and generating a positive or negative security qualification indication. Application, ¶¶ [0053], [0057]-[0067], [0076]-[0080].

112. The claims further recite verifying the UUID as unexpired based on the expiration date. The application describes expiration-date information as authentication information associated with the UUID and also describes timestamp information in the authentication control packet. Application, ¶¶ [0054], [0066]. A person of ordinary skill in the art would have understood that this limitation reflects the freshness or timeliness control that reduces the risk of replayed or stale email-originated transaction requests.

113. The claims further recite authenticating the e-commerce transaction based on the comparison and performing the order execution procedure only on a condition that the e-commerce transaction is authenticated. The application describes positive and negative security qualification, transaction continuation or denial, and order execution when the relevant checks are satisfied. Application, ¶¶ [0080]-[0085], [0153]-[0155]. A person of ordinary skill in the art would have understood that this limitation reflects the security gate between receiving the email-generated order message and executing the transaction.

114. My opinions also apply to dependent claims 19 and 20. Claim 19 further reflects the technical verification architecture by requiring the product identifier and quantity extracted from the body field to be verified against corresponding fields of the authentication control packet. Claim 20 further reflects the technical gatekeeping architecture by requiring the system to refrain from performing order execution when expiration or verification fails and to store a record associated with the UUID and the failure condition. These limitations are consistent with the specification's description of comparing returned email information against

authentication-control-packet fields and continuing, denying, or taking additional measures based on the security qualification result. Application, ¶¶ [0053], [0057]-[0067], [0076]-[0085].

115. Taken together, these claimed limitations reflect the technological improvement described above. The claims do not merely recite the desired result of completing an e-commerce transaction. They recite a particular architecture and information flow in which the e-commerce system creates or stores authentication context, sends an initial email containing a mailto hyperlink, receives a mailto-generated SMTP order email, parses the returned email, verifies the extracted values against a server-side authentication control packet, verifies expiration, authenticates based on the comparison, and executes the order only if the returned email satisfies the authentication checks. Application, ¶¶ [0052]-[0069], [0071]-[0087], [0094]-[0097], [0103]-[0108], [0117]-[0125], [0153]-[0155].

116. In my opinion, a person of ordinary skill in the art would have understood that the claimed combination improves the security and reliability of SMTP/mailto transaction processing by adding a server-side verification layer that SMTP and mailto do not themselves provide. That improvement is reflected directly in the claims through the UUID, authentication control packet, parsing, field-by-field comparison, expiration verification, authentication, and conditional order execution limitations.

117. In my opinion, the technological improvement is reflected in the claims at least by the ordered combination of: storing the UUID and customer email address; configuring the mailto hyperlink to include a UUID and parameters; associating the UUID with an authentication control packet; receiving the generated SMTP order email; parsing the body and subject fields; comparing parsed fields and sender email address against corresponding authentication-control-packet fields; verifying expiration; authenticating based on the comparison; and performing order

execution only if authenticated. Application, ¶¶ [0054]-[0067], [0074]-[0084], [0153]-[0155].

118. A person of ordinary skill in the art would have understood that this ordered combination changes how the e-commerce system treats a returned SMTP email. The email is not accepted as a valid order merely because it reaches the e-commerce system. Instead, the email is evaluated against server-side authentication context and expiration information before execution. That change improves the security and reliability of the email-originated transaction pathway.

119. For these reasons, my opinion is that the claimed invention provides a technological improvement to SMTP/mailto-based transaction processing. It allows an e-commerce system to use ordinary email-generation and email-transport mechanisms while imposing transaction-specific authentication constraints that those mechanisms do not natively provide. The result is an email-originated transaction system in which returned order emails are authenticated against server-side transaction context before order execution occurs.

VI. Why the Improvement Is Not Merely the Business Result of Selling Goods

120. I understand that the Examiner has characterized the claimed invention as involving advertising, marketing, and sales activities. From a technical perspective, however, a person of ordinary skill in the art would not have understood the claimed invention as merely automating the business result of selling goods. The claims are directed to a specific way of determining whether an email-originated transaction request should be trusted before the e-commerce system performs order execution.

121. The technical problem addressed by the claimed invention exists because the transaction instruction is generated and transmitted through email infrastructure. A mailto hyperlink can cause an email client to generate an email

message, and SMTP can transport that message. But the resulting order email is not submitted through a continuous, server-controlled web checkout session. Instead, the order instruction is generated by an email client and returned through email infrastructure. That architecture creates technical risks relating to alteration, forwarding, spoofing, replay, stale links, and loss of server-controlled transaction context. RFC 6068 confirms that resolving a mailto URI does not cause an immediate interaction with a server, but instead creates a message that the user may edit before sending. *See* M. Dürst et al., *The "mailto" URI Scheme* § 3, RFC 6068, at 7 (Oct. 2010). RFC 5321 likewise recognizes that SMTP mail is inherently insecure because users may create messages that appear to come from another source and because SMTP mail cannot inherently provide authentication or integrity checks at the transport level. *See* J. Klensin, *Simple Mail Transfer Protocol* § 7.1, RFC 5321, at 75 (Oct. 2008).

122. The claimed solution is therefore not simply "selling a product by email." The claimed architecture imposes a technical authentication gate between receipt of the returned SMTP order email and execution of the transaction. The application describes UUIDs, authentication control packets, returned-email parsing, comparison of parsed information with authentication information, positive or negative security qualification, and transaction continuation or denial based on the result. Application, ¶¶ [0052]-[0058], [0068]-[0069], [0074]-[0085].

123. In a system that merely automates a sale, an order message might be treated as an instruction to complete the commercial transaction. In the claimed architecture, by contrast, receipt of the email does not itself complete the transaction and does not itself cause order execution. The returned email is treated as an input to an authentication process. The e-commerce system parses the returned email, extracts the UUID and other parameters, compares those extracted values with corresponding fields of a stored authentication control packet, verifies expiration,

and only then authenticates the transaction and performs order execution. Application, ¶¶ [0053], [0076]-[0084], [0153]-[0155].

124. The specific technical character of the claimed architecture is reflected in the authentication control packet. The application describes an authentication control packet that may include a UUID, client identifier, email campaign identifier, item or donation amount, email address, subject line, member, timestamp, and additional fields. Application, ¶¶ [0057]-[0067]. These fields are not merely a statement of a desired commercial result. They provide server-side authentication context that allows the e-commerce system to determine whether the returned email message remains associated with the intended customer, campaign, transaction amount, subject line, member, and valid time window.

125. The claimed architecture therefore addresses a technical trust problem that arises before any commercial result is performed. The question answered by the claimed architecture is not merely "what did the customer buy?" The more fundamental technical question is "should this returned SMTP message be trusted as a valid transaction request?" The claimed system answers that question by validating the returned email against server-side authentication context and expiration information before order execution is permitted.

A. The Claimed Invention Does Not Merely Automate a Sale

126. A person of ordinary skill in the art would have understood that automating a sale and securing an email-originated transaction request are different technical concepts. A sale may be completed by many different mechanisms, including a web checkout page, a phone order, an in-person point-of-sale system, or a manually reviewed email. The claimed invention is specifically concerned with the technical risks introduced when an order instruction is generated through a mailto hyperlink and returned through SMTP.

127. In a conventional browser-based checkout, a merchant's server may maintain session context throughout the transaction. That session context may associate the user, cart contents, pricing, payment information, and checkout state. By contrast, in the claimed SMTP/mailto transaction flow, the order email is generated by a user's email client and returned outside a continuous browser session. Because of that different architecture, the receiving system needs a mechanism to determine whether the returned email still corresponds to the original server-side transaction context. Application, ¶¶ [0040]-[0043], [0054]-[0067], [0074]-[0080].

128. The claimed invention supplies that mechanism. The system associates the UUID with an authentication control packet before the returned order email is processed. Application, ¶¶ [0054]-[0058], [0068]-[0069]. The order email is then parsed to extract the UUID and other authentication information, and the extracted information is compared to the authentication control packet. Application, ¶¶ [0053], [0074]-[0079]. That sequence is a technical authentication architecture, not simply a commercial instruction to complete an order.

129. This distinction is important because the business result could be the same in many systems: a customer receives an offer and a transaction is completed. But the technical means are different. The claimed means require the e-commerce system to validate an email-originated request against stored server-side authentication information and expiration data before execution. The technological improvement lies in that verification architecture, not in the fact that the transaction may involve a sale, donation, or other commercial action.

130. The same point applies even though the claims refer to an "e-commerce transaction" and "order execution." Those terms identify the transaction environment in which the security architecture operates. They do not eliminate the technical nature of the authentication process. The application describes a message-processing and security process in which returned email information is parsed,

compared, qualified, accepted, denied, or routed to further measures depending on the security result. Application, ¶¶ [0053], [0074]-[0085], [0149]-[0155].

131. In my opinion, a person of ordinary skill in the art would have understood that the claimed invention changes the system's operation before the commercial result occurs. The system must first determine whether the returned SMTP order email is authentic, current, and bound to the expected server-side transaction context. Only after that determination does the system proceed to order execution. Application, ¶¶ [0076]-[0084], [0153]-[0155].

B. The Technical Security Workflow Is Independent of the Commercial Item

132. The technical security workflow is independent of the particular commercial item, donation amount, or transaction object. The claimed authentication process would operate in substantially the same technical manner whether the returned email relates to a product order, a donation, a membership transaction, a campaign-specific authorization, or another controlled email-originated transaction request. What matters technically is not the commercial nature of the item or donation, but whether the returned SMTP message matches the server-side authentication context and remains within its validity period.

133. The application itself reflects that broader technical character. It describes item or donation amounts, product identifiers, customer identifiers, campaign identifiers, member information, subject lines, email addresses, timestamps, and additional fields used as part of authentication or verification. Application, ¶¶ [0057]-[0067], [0105]-[0108], [0121], [0138]-[0140]. These fields allow the system to validate the returned email message against stored context regardless of whether the transaction involves goods, donations, membership activity, or campaign-specific responses.

134. For example, the same authentication architecture can validate a product-order email by checking the UUID, customer email address, campaign identifier, product identifier, quantity, subject line, and expiration information against the authentication control packet. Application, ¶¶ [0057]-[0067], [0105]-[0108], [0121]. The architecture can also validate a donation-related email by checking the UUID, campaign identifier, donation amount, member identifier, subject line, sender email address, and expiration information against the stored packet. Application, ¶¶ [0057]-[0067], [0138]-[0140].

135. In each case, the technical question is the same: whether the returned SMTP message should be trusted. The claim does not depend on the economic value of the item, the type of product, the charitable nature of the donation, or the business rules of a campaign. The claim depends on technical validation of the returned message against server-side authentication information and expiration data.

136. This is why the improvement is not merely the business result of completing a sale. A business rule might specify that a customer may buy an item, donate an amount, or join a membership program. The claimed architecture specifies how the system determines whether an email-generated transaction request is technically valid before the system acts. That distinction would have been apparent to a person of ordinary skill in the art.

137. The extrinsic evidence supports this distinction. SMTP and mailto standards describe message transport and message-generation functionality, not server-side authentication of transaction requests. *See* RFC 5321, *supra*, § 1.1, at 5; RFC 6068, *supra*, §§ 1–3, at 3–7. Security guidance and standards separately recognize technical mechanisms for resisting replay, validating freshness, binding requests to expected context, and preventing misuse of client-visible tokens. *See* William E. Burr et al., Nat'l Inst. of Standards & Tech., Special Publ'n 800-63-1, *Electronic Authentication Guideline* 13, 71 (Dec. 2011); P. Leach et al., *A*

Universally Unique IDentifier (UUID) URN Namespace § 6, RFC 4122, at 16 (July 2005).

138. The fact that the returned email may ultimately cause a commercial transaction does not change the technical role of the authentication workflow. Payment systems, order fulfillment systems, and confirmation systems may all be downstream consequences of a successful authentication result. But the claimed improvement occurs before those downstream steps: the e-commerce system decides whether the returned email should be accepted as a valid transaction request. Application, ¶¶ [0076]-[0084], [0153]-[0167].

139. A person of ordinary skill in the art would therefore have understood the claimed architecture as a security-control architecture for email-originated transactions, not merely as a way to sell a product. The same architecture could be used whenever an email-generated instruction must be validated against server-side context before a controlled action is performed.

C. The Architecture Changes the Execution Condition

140. The claimed architecture changes the condition under which the e-commerce system executes the transaction. The system does not perform order execution merely because it receives an order email. Instead, the system performs order execution only after the returned email has been authenticated based on comparison with the authentication control packet and verification that the UUID remains unexpired. Application, ¶¶ [0076]-[0084], [0153]-[0155].

141. This changed execution condition is technically significant. In a less secure email-order system, a received email might be sufficient to begin fulfillment or payment processing. In the claimed system, the received email is not sufficient by itself. The email must be parsed, its fields must be extracted, the extracted

information must be compared against stored packet fields, and the UUID must be verified as unexpired. Application, ¶¶ [0053], [0057]-[0067], [0074]-[0084].

142. The authentication control packet functions as the system's trusted reference for deciding whether the returned email should trigger execution. The returned email provides candidate information; the authentication control packet provides the stored context against which that candidate information is evaluated. This changes the execution condition from "email received" to "email received, parsed, matched to stored authentication context, verified as unexpired, and authenticated." Application, ¶¶ [0054]-[0058], [0076]-[0084].

143. That change is especially important because email messages can persist, be forwarded, be modified, or be resent. A returned email may contain information that once corresponded to a valid transaction, but that does not mean the transaction should still be executed. The expiration check ensures that the UUID remains within a valid time window before execution. Application, ¶¶ [0054], [0066], [0076]-[0084]. This is consistent with NIST's recognition that replay resistance can be supported by freshness or timeliness data. *See* NIST SP 800-63-1, *supra*, at 71.

144. The changed execution condition also addresses alteration and mismatch risks. A body field may contain a product identifier, campaign identifier, member identifier, or amount. A subject field may identify a campaign, transaction path, or order-processing context. A sender address may identify the account from which the message was sent. The claimed comparison verifies those fields against corresponding fields in the authentication control packet before execution. Application, ¶¶ [0057]-[0067], [0076]-[0079], [0105]-[0108].

145. In this architecture, failure of authentication changes system behavior. The application describes negative security qualification, transaction denial, and further measures when the comparison is unfavorable or authentication fails.

Application, ¶¶ [0053], [0077], [0080]-[0085]. Thus, the system's operation is conditional on the technical authentication result. The same email that might otherwise appear to request an order does not cause execution if the authentication checks fail.

146. The claimed execution condition also helps explain why the improvement is not merely commercial. A business rule might define what goods are sold, what price applies, or whether a campaign is active. The claimed system defines a technical gate that determines whether an email-generated instruction is sufficiently authenticated to allow the system to act. That gate is implemented through UUID association, server-side packet storage, returned-email parsing, field-by-field verification, expiration checking, and conditional execution.

147. This architecture is consistent with the general security principle that possession of a token or identifier should not alone authorize an action. RFC 4122 cautions that UUIDs should not be used as security capabilities whose mere possession grants access. *See* RFC 4122, *supra*, § 6, at 16. The claimed architecture follows that principle because it does not rely solely on the UUID appearing in the returned email. Instead, the system uses the UUID with corresponding packet fields, parsed email parameters, sender information, subject-line information, and expiration information before authenticating and executing the transaction. Application, ¶¶ [0054]-[0067], [0076]-[0084].

148. The architecture is also consistent with technical approaches used in other security-sensitive systems, where identifiers and requests are bound to expected server-side context before acceptance. For example, OAuth threat-model guidance describes exact matching of redirect URIs and binding authorization codes to client identifiers to reduce misuse. *See* T. Lodderstedt et al., *OAuth 2.0 Threat Model and Security Considerations* §§ 4.4.1.3, 4.4.1.8, 5.2.3.4, RFC 6819, at 24–25, 31, 54 (Jan. 2013). Although OAuth is a different technology, a person of ordinary

skill in the art would have recognized the same technical principle: a received request should be checked against server-side context and expected fields before the system accepts it as valid.

149. In my opinion, the claimed architecture is not simply a business rule for completing a sale. The claims require a specific technical mechanism for deciding whether an email-originated transaction request should be trusted before order execution is permitted. That mechanism is reflected in the claimed UUID, authentication control packet, returned-email parsing, field-by-field comparison, expiration verification, authentication result, and conditional order execution.

150. A person of ordinary skill in the art would have understood that the commercial result is downstream of the technical authentication mechanism. The system first determines whether the returned email is valid. Only after that determination does the system perform the order execution procedure. Application, ¶¶ [0076]-[0084], [0153]-[0155]. The claimed improvement therefore lies in the architecture and operation of the email-originated transaction system, not merely in the business objective of selling goods or completing donations.

151. For these reasons, my opinion is that the claimed invention should be understood as addressing a technical security problem in SMTP/mailto-based transaction processing. The claimed architecture does not merely automate a sale. It changes the system's information flow and execution condition by requiring a returned email-generated transaction request to be authenticated against expiration-bound server-side transaction context before execution is allowed.

VII. Response to Examiner's "Generic Computer" Characterization

152. I understand that the Examiner has characterized the claimed invention as using a general-purpose computer, generic computer components, SMTP, and a standard mailto URI scheme. I do not contend that processors, memory, databases,

network interfaces, SMTP, UUIDs, or mailto hyperlinks were themselves newly invented. I also do not contend that the invention requires a new SMTP protocol, a new mailto URI standard, or a new type of general-purpose computer.

153. My opinion is that a person of ordinary skill in the art would have understood the claimed invention as using known components in a particular architecture that changes the information flow and execution condition of an email-originated transaction. The invention is not directed to the mere presence of a processor, memory, database, network interface, SMTP, or mailto hyperlink. Rather, the invention is directed to how those components are arranged and used to authenticate a returned SMTP order email before the e-commerce system performs order execution.

154. The technological improvement does not depend on the novelty of any individual component taken in isolation. The improvement lies in the ordered combination by which the e-commerce system creates or stores server-side authentication context, includes a UUID in a mailto-generated transaction path, receives a returned SMTP order email, parses returned email fields, validates those returned fields against a stored authentication control packet, enforces expiration, and conditionally gates order execution. Application, ¶¶ [0052]-[0058], [0068]-[0069], [0074]-[0085], [0153]-[0155].

155. A person of ordinary skill in the art would have understood that this architecture is not the same as merely using a generic computer to receive and process an email order. In a simple email-order system, the email may be treated as the order instruction. In the claimed architecture, the returned SMTP order email is not treated as self-authenticating. Receipt of the email initiates a technical validation process in which the email is parsed and compared against server-side authentication information before the system determines whether order execution is permitted. Application, ¶¶ [0053], [0076]-[0084].

156. The specification describes the relevant architecture in technical terms. It describes generating UUIDs, associating UUIDs with authentication information, creating and storing authentication control packets, embedding or including UUIDs in selectable email hyperlinks or buttons, parsing returned emails for UUIDs and other authentication information, comparing returned information with selected authentication-control-packet fields, and generating positive or negative security qualification indications. Application, ¶¶ [0052]-[0058], [0068]-[0069], [0074]-[0085].

157. The authentication control packet is a central part of this architecture. The specification describes an authentication control packet including fields such as UUID, client identifier, email campaign, item or donation amount, email address, subject line, member, timestamp, and additional fields. Application, ¶¶ [0057]-[0067]. A person of ordinary skill in the art would have understood that this packet provides server-side transaction context against which the later-received SMTP order email can be validated. That server-side packet is not provided by SMTP itself and is not provided by the mailto URI scheme itself.

158. The use of the UUID is also not merely a generic data-storing or data-transmitting step. The specification describes UUIDs delivered in an email as opaque and explains that associated authentication information is not derivable from parsing the UUID itself. Application, ¶ [0054]. In that architecture, the UUID links the returned email to server-side authentication context without itself exposing the authentication-control-packet information. A person of ordinary skill in the art would have understood that this reduces reliance on visible email contents as the sole source of transaction authority.

159. That point is technically important because baseline SMTP and mailto were not designed to provide the claimed security architecture. SMTP transfers email messages. *See* J. Klensin, *Simple Mail Transfer Protocol* § 1.1, RFC 5321, at

5 (Oct. 2008). A mailto URI allows an email client to compose a message with certain fields, but resolving a mailto URI does not itself cause an immediate interaction with a server, and the user may edit the generated message before sending it. *See* M. Dürst et al., *The "mailto" URI Scheme* § 3, RFC 6068, at 7 (Oct. 2010). A person of ordinary skill in the art would therefore have understood that a separate transaction-authentication architecture was needed if an e-commerce system was to rely on returned SMTP messages to initiate order execution.

160. The claimed architecture supplies that separate transaction-authentication layer. The returned email is parsed for the UUID and other parameters. Application, ¶¶ [0053], [0074]-[0077]. Those values are compared against authentication information stored in the authentication control packet. Application, ¶¶ [0076]-[0079]. The transaction is continued or denied based on the resulting security qualification. Application, ¶¶ [0080]-[0085]. In my opinion, this ordered relationship among the components changes how the e-commerce system operates.

161. The expiration feature further distinguishes the claimed architecture from ordinary computer use. The specification describes expiration-date information associated with the UUID and a timestamp field in the authentication control packet. Application, ¶¶ [0054], [0066]. The expiration check prevents an old, forwarded, resent, or stale mailto-generated order email from being treated as a valid current transaction request merely because the email reaches the e-commerce system. That is a technical gate between message receipt and order execution.

162. This is not merely the use of a database to store information. The database stores authentication context that is later used to evaluate whether a returned SMTP order email should be trusted. In other words, the database is not simply a passive repository of customer or order records. It stores authentication-control-packet fields against which returned email fields are verified before the

system decides whether to execute the order. Application, ¶¶ [0054]-[0067], [0076]-[0080].

163. Nor is this merely the use of a processor to perform generic processing. The processor implements a particular ordered sequence: store authentication context, configure the mailto-generated message path, receive the SMTP order email, parse body and subject fields, compare extracted values and sender information against stored packet fields, verify expiration, authenticate based on the comparison, and conditionally execute or deny the transaction. Application, ¶¶ [0052]-[0058], [0074]-[0085], [0153]-[0155]. A person of ordinary skill in the art would have understood that this ordered sequence defines the security behavior of the e-commerce system.

164. Nor is this merely the use of a network interface to transmit and receive data. The network interface participates in a controlled email transaction flow in which a returned SMTP message is not accepted at face value. The returned message is instead passed into an authentication process that determines whether the message remains bound to server-side transaction context. Application, ¶¶ [0040]-[0043], [0071]-[0084].

165. Nor is this merely the use of a standard mailto hyperlink. The specification describes mailto hyperlinks and selectable email controls that include embedded identifiers or parameters used in the later authentication process. Application, ¶¶ [0086]-[0087], [0094]-[0097], [0103]-[0108], [0124]-[0125], [0140]. The technical point is not that the mailto URI scheme itself was invented. The technical point is that the mailto-generated email is integrated into an authentication architecture in which the returned email is checked against server-side packet fields before execution.

166. In my opinion, a person of ordinary skill in the art would have understood that the claimed architecture is an ordered arrangement of known

components that performs a specific security function. The system pre-generates or stores server-side authentication context. It links the returned email to that context using a UUID. It receives the email through SMTP. It parses returned fields. It validates the returned fields against packet fields. It enforces expiration. It conditionally gates execution. Application, ¶¶ [0052]-[0069], [0074]-[0085], [0153]-[0155].

167. I understand that the claimed architecture may be implemented using one e-commerce system rather than requiring multiple separately claimed computing devices. From a technical perspective, that does not change my opinion. The improvement is not dependent on the number of physical devices recited in the claim. The improvement lies in the ordered arrangement and operation of the system components and data structures: the system establishes server-side authentication context, links a returned SMTP order email to that context using a UUID, parses returned email fields, compares those fields against corresponding authentication-control-packet fields, verifies expiration, and gates order execution based on the authentication result. A person of ordinary skill in the art would have understood that this changes how the e-commerce system treats a returned SMTP order email, regardless of whether the functions are performed by one server, multiple servers, or distributed components.

168. The advantages of that arrangement are technical. The architecture reduces reliance on the apparent contents of an email message, reduces the risk that altered fields will trigger order execution, reduces the risk that a forwarded message will be accepted for the wrong sender, reduces the risk that a stale or replayed email will be executed, and allows the e-commerce system to use ordinary email infrastructure while adding transaction-specific authentication controls. Application, ¶¶ [0053]-[0058], [0076]-[0085].

169. From a technical perspective, the authentication-control-packet fields, verification, and expiration enforcement define how the e-commerce system changes its treatment of a returned SMTP order email. The system does not proceed directly from email receipt to order execution. It proceeds from email receipt to packet-based validation, expiration checking, authentication, and then conditional execution. Application, ¶¶ [0053], [0076]-[0085], [0153]-[0155].

170. The relevant architecture is reflected in how the system organizes and controls information flow through the transaction pathway. The system creates server-side authentication context before the returned email is received and then uses that context to authenticate the later-received SMTP message before execution. Application, ¶¶ [0054]-[0069], [0074]-[0084]. The returned email is therefore not the sole source of authority for the transaction; it is an input that must be validated against previously stored server-side information.

171. In my opinion, the claimed invention changes the information flow from a direct email-order path into a server-side authentication path. The system sends an initial email with a mailto control, receives a returned SMTP order email, parses the returned message, compares parsed values against an authentication control packet, verifies expiration, determines an authentication result, and conditionally executes the order. Application, ¶¶ [0052]-[0069], [0071]-[0087], [0153]-[0155].

172. My factual opinion is that a person of ordinary skill in the art would have understood the specification as describing a technological improvement to email-originated transaction processing because the specification adds server-side authentication context, field-by-field validation, and expiration-based gatekeeping to an SMTP/mailto transaction path. Application, ¶¶ [0052]-[0069], [0074]-[0085].

173. The expiration-bound authentication control packet is not merely incidental to the use of a computer. It determines whether a later-received email-

generated order request remains valid for execution. The specification describes expiration-date information associated with the UUID, a timestamp field in the authentication control packet, positive and negative security qualification, and transaction continuation or denial based on the result of authentication. Application, ¶¶ [0053]-[0054], [0066], [0076]-[0085].

174. In my opinion, a person of ordinary skill in the art would not have understood the claimed invention as merely saying "apply e-commerce using a computer." The claims recite a specific technical process for evaluating whether a returned SMTP order email generated from a mailto hyperlink should be trusted. That process depends on the ordered combination of a UUID, server-side authentication control packet, parsed email fields, field-by-field comparison, expiration verification, and conditional execution. Application, ¶¶ [0052]-[0069], [0074]-[0085], [0153]-[0155].

175. The claimed invention uses known components, but it uses them in a particular security architecture that changes the information flow and execution conditions of email-originated transactions. The returned SMTP message is not accepted merely because it is received. It must be authenticated against stored server-side transaction context and expiration information before the e-commerce system performs order execution. Application, ¶¶ [0053], [0076]-[0085], [0153]-[0155].

176. My opinion would be the same even if every individual component of the system were considered known. A person of ordinary skill in the art would have understood that the technological contribution lies in the claimed ordered combination and the resulting change in system operation. The system moves from simple email receipt and order fulfillment to packet-based authentication and expiration-controlled execution of email-originated transactions.

177. In my opinion, that ordered combination improves the security and reliability of SMTP/mailto-based transaction processing. It allows ordinary email infrastructure to be used for transaction initiation while addressing technical risks that ordinary email infrastructure does not resolve by itself, including altered email contents, sender mismatch, forwarding, replay, stale links, and loss of server-controlled checkout context. Application, ¶¶ [0053]-[0058], [0074]-[0085].

VIII. Nexus Between Testimony, Specification, and Claims

178. The opinions in this declaration are directed to the claimed invention because the pending claims recite the same technical architecture discussed above. In particular, the independent claims describe an SMTP/mailto transaction path in which an initial email includes a mailto hyperlink, activation of that hyperlink generates an order email message, the order email is received by the e-commerce system via SMTP, the returned email is parsed, extracted fields are compared against a stored authentication control packet, the UUID is verified as unexpired based on an expiration date, and order execution occurs only if the transaction is authenticated.

179. The opinions in this declaration are also grounded in the specification. I do not rely on the extrinsic references discussed herein to add features to the disclosure. I refer to those materials only to explain the technical context in which a person of ordinary skill in the art would have understood the specification and claims.

180. The specification describes an e-commerce system that includes an e-commerce database, a security module, a message processing module, and an email interface module. Application, ¶¶ [0036], [0040]-[0043]. The specification further describes the e-commerce system generating and transmitting advertisement emails, receiving return emails, parsing email content, performing security qualification, and

continuing or denying transactions based on the results of that qualification. Application, ¶¶ [0071]-[0085].

181. My testimony regarding UUIDs is supported by the specification's discussion of generating UUIDs, storing generated UUIDs, associating UUIDs with authentication information, and using UUIDs embedded in or associated with selectable email controls. Application, ¶¶ [0037]-[0039], [0052]-[0058], [0068]-[0069], [0086]-[0087]. The specification also explains that UUIDs delivered in email may be opaque and that the authentication information associated with a UUID is not derivable merely by parsing the UUID. Application, ¶ [0054]. That disclosure supports my opinion that the UUID is not merely descriptive order information. Rather, it serves as a linking value that connects a returned email message to server-side authentication context.

182. My testimony regarding authentication control packets is likewise supported by the specification. The specification expressly describes an authentication control packet associated with a UUID. Application, ¶¶ [0054]-[0058], [0068]-[0069]. It further describes the authentication control packet as including fields such as UUID, client identifier, email campaign, item or donation amount, email address, subject line, member, timestamp, and additional fields. Application, ¶¶ [0057]-[0067]. Those disclosures support my opinion that the authentication control packet creates server-side transaction context against which a later-received SMTP order email can be validated.

183. My testimony regarding embedded identifiers in email buttons or hyperlinks is tied to the specification because the specification describes selectable hyperlinks or buttons in emails and describes embedding UUIDs or identifiers in those email controls. Application, ¶¶ [0052], [0058], [0068]-[0069], [0086]-[0087], [0124], [0140]. The specification also describes mailto hyperlinks that include fields such as destination, subject, and body information and that generate return or order

emails when selected. Application, ¶¶ [0094]-[0097], [0103]-[0108], [0117]-[0121]. Together, those disclosures support my opinion that the invention uses a mailto-generated transaction path while adding authentication context and verification that are not supplied by a mailto hyperlink alone.

184. My testimony regarding parsing of returned emails is also grounded in the specification. The specification describes parsing return emails for embedded identifiers and other authentication information. Application, ¶¶ [0053], [0074]-[0077]. It also provides examples of returned email bodies containing transaction-related information, including product identifiers, customer identifiers, quantities, and campaign identifiers. Application, ¶¶ [0105]-[0108], [0121], [0124]-[0125]. Those disclosures support my opinion that parsing is not merely reading an order. It is the extraction of authentication inputs from a returned SMTP message so that the e-commerce system can determine whether the message should be trusted for execution.

185. My testimony regarding comparison of parsed information with authentication information is supported by the specification's description of selecting parameters from the authentication control packet and comparing those parameters against information parsed from the return email. Application, ¶¶ [0076]-[0079]. The specification further describes using that comparison to determine whether selected parameters have been reviewed and qualified, and whether the security qualification succeeds or fails. Application, ¶¶ [0077]-[0080]. That disclosure supports my opinion that the claimed system performs field-by-field validation rather than treating the returned email as a self-validating order instruction.

186. My testimony regarding positive and negative security qualification is also tied to the specification. The specification describes positive and negative security qualification indications. Application, ¶¶ [0078]-[0085]. It explains that,

when the comparison or other security qualification is favorable, the transaction may continue, and when the comparison is unfavorable, the authentication process may terminate, additional measures may be taken, or the transaction may be denied. Application, ¶¶ [0077], [0080]-[0085]. These disclosures support my opinion that the system's behavior changes based on the technical authentication result.

187. My testimony regarding expiration and stale-message control is supported by the specification's disclosure of expiration-date information associated with the UUID and a timestamp field in the authentication control packet. Application, ¶¶ [0054], [0066]. That disclosure supports my opinion that the claimed architecture determines whether a returned email-generated transaction request remains within a valid time window before the system allows order execution.

188. My testimony regarding denial, further measures, or confirmation when authentication fails is also supported by the specification. The specification describes negative security qualification, transaction denial, failure emails, and additional action when authentication is not successful. Application, ¶¶ [0077], [0080]-[0085]. The specification also describes an order confirmation procedure, including transmitting a confirmation email message, receiving a confirmation response email message, and performing order execution when the response information indicates that the order is confirmed. Application, ¶¶ [0149]-[0155], [0169]-[0186]. These disclosures support my opinion that the claimed system does not simply execute every returned email order. Instead, it changes its behavior depending on whether the returned email is authenticated.

189. My testimony regarding transaction completion only after authentication is supported by the specification's description of continuing the transaction when security qualification is successful and performing the order execution procedure to complete the order when the relevant checks are satisfied. Application, ¶¶ [0080]-[0084], [0153]-[0155]. That disclosure supports my opinion

that the claimed architecture imposes a security gate between receipt of a returned SMTP order email and execution of the transaction.

190. The testimony in this declaration is also tied directly to the pending claim limitations. The independent claims require storing information including a unique identifier and customer email address, configuring a mailto hyperlink that generates an order email addressed to the e-commerce system, including a UUID and additional parameters in the body field, including a subject line in the subject field, associating the UUID with an authentication control packet, transmitting an initial email via SMTP, receiving a returned order email via SMTP, parsing the body and subject fields, comparing the parsed information and sender email address against corresponding packet fields, verifying expiration, authenticating based on the comparison, and performing order execution only if authentication succeeds.

191. Each technical opinion expressed above corresponds to one or more of those claim limitations. My opinions about the technical problem correspond to the claim's use of a mailto-generated order email returned via SMTP outside a conventional server-controlled checkout session. My opinions about server-side authentication context correspond to the claim's authentication control packet. My opinions about message validation correspond to the parsing and comparison limitations. My opinions about replay and stale-message risk correspond to the timestamp and expiration-date limitations. My opinions about changed system behavior correspond to the conditional order-execution limitation.

192. The extrinsic technical materials discussed above are also tied to the claimed invention. I do not rely on those materials to add new inventive features to the specification. Rather, they provide technical context for how a person of ordinary skill in the art would have understood SMTP, mailto hyperlinks, UUIDs, replay resistance, session context, server-side validation, and email-originated security risks at the time of the invention. That context supports my opinion that the

specification's UUID-linked, expiration-bound authentication-control architecture would have been understood as a technical security architecture for email-originated transaction processing.

193. RFC 5321 supports my testimony that SMTP provides email transport but does not itself provide the transaction-specific sender authentication or message-integrity assurances needed to trust a returned email order for execution. *See* J. Klensin, *Simple Mail Transfer Protocol* §§ 1.1, 7.1, RFC 5321, at 5, 75 (Oct. 2008). RFC 6068 supports my testimony that a mailto URI generates a user-editable email message rather than an immediate authenticated server interaction. *See* M. Dürst et al., *The "mailto" URI Scheme* § 3, RFC 6068, at 7 (Oct. 2010). Those technical facts are tied to the claims because the claims specifically use SMTP and a mailto hyperlink as the transaction-originating pathway.

194. NIST guidance regarding replay resistance and freshness supports my testimony that the expiration-date limitation performs a technical function in the claimed architecture. NIST described replay attacks as involving replay of previously captured messages and described freshness mechanisms, including nonces, challenges, and timeliness data, as ways to resist replay. *See* William E. Burr et al., Nat'l Inst. of Standards & Tech., Special Publ'n 800-63-1, *Electronic Authentication Guideline* 13, 71 (Dec. 2011). That technical context is tied to the claims because the claims require verifying that the UUID is unexpired based on the expiration date before order execution occurs.

195. RFC 4122 supports my testimony that the claimed architecture does not rely merely on possession of a UUID. RFC 4122 cautioned that UUIDs should not be treated as security capabilities whose mere possession grants access. *See* P. Leach et al., *A Universally Unique IDentifier (UUID) URN Namespace* § 6, RFC 4122, at 16 (July 2005). That technical context is tied to the claims because the pending claims do not merely require receiving a UUID. They require comparing the UUID,

additional parameters, subject line, sender email address, and expiration information against the stored authentication control packet before authenticating the transaction.

196. OWASP's web-session guidance supports my testimony concerning the technical difference between a conventional web checkout flow and an email-originated transaction flow. OWASP described session management as a way for web applications to maintain user identity and transaction state across requests, including shopping-cart state, product selections, quantities, prices, discounts, and related information. *See* The OWASP Foundation, *OWASP Testing Guide v3.0* 146-48 (2008). That technical context is tied to the claims because the claimed invention supplies server-side authentication context for an SMTP/mailto transaction path that is not inherently part of a continuous web checkout session.

197. My testimony is not intended to add limitations that are absent from the specification or claims. Rather, it explains how a person of ordinary skill in the art would have understood the disclosed and claimed architecture. The specification discloses the UUID, authentication control packet, embedded identifiers, returned-email parsing, comparison of parsed information to stored authentication information, positive and negative security qualification, expiration information, and conditional transaction continuation or denial. Application, ¶¶ [0052]-[0069], [0074]-[0085], [0153]-[0155]. The pending claims recite those same concepts in the context of an SMTP/mailto e-commerce transaction.

198. For these reasons, there is a direct nexus between the testimony in this declaration, the specification, and the pending claims. The testimony addresses the same technical architecture that the specification describes and that the claims recite: a returned SMTP order email generated from a mailto hyperlink is not trusted merely because it is received, but is parsed and authenticated against an expiration-bound, server-side authentication control packet before the e-commerce system performs order execution.

IX. Conclusion and Declaration

199. For the reasons discussed above, my opinion is that a person of ordinary skill in the art would have understood the claimed invention as addressing a technical security problem arising from email-originated transaction processing. The problem is not merely how to complete an e-commerce transaction. The problem is how an e-commerce system can determine whether a returned SMTP order email generated from a mailto hyperlink should be trusted for order execution when that returned email is received outside a continuous, server-controlled web checkout session.

200. A person of ordinary skill in the art would have understood that SMTP and mailto functionality do not themselves provide the claimed transaction-specific authentication architecture. SMTP provides email transport, and a mailto hyperlink can cause an email client to generate a message. But those mechanisms do not themselves provide a server-side authentication control packet, field-by-field comparison of returned email values against stored authentication information, expiration-based verification of a UUID, or conditional order execution based on the result of that authentication process.

201. The specification describes, and the claims reflect, a particular architecture for addressing those limitations. In that architecture, the e-commerce system associates a UUID with server-side authentication information, stores an authentication control packet, includes or embeds the UUID in a mailto-generated transaction path, receives a returned SMTP order email, parses the returned email for authentication inputs, compares the parsed information against corresponding fields of the stored authentication control packet, verifies that the UUID remains unexpired, and performs order execution only when the e-commerce transaction is authenticated. Application, ¶¶ [0052]-[0069], [0074]-[0085], [0153]-[0155].

202. In my opinion, the UUID and authentication control packet are technically significant because they change the role of the returned email message. The returned email is not treated as a self-validating order instruction. Instead, the returned email is treated as a message that must be validated against server-side transaction context before the system acts on it. That validation reduces reliance on visible email contents or apparent sender information alone and provides a mechanism for detecting altered, mismatched, forwarded, stale, or replayed transaction requests.

203. The expiration-based verification is also technically significant. Email messages and mailto links can remain available long after the original transaction opportunity was created. By requiring verification that the UUID remains unexpired based on an expiration date, the claimed architecture adds a freshness check between receipt of the returned email and execution of the transaction. A person of ordinary skill in the art would have understood that this feature reduces the risk that a stale or replayed email-generated transaction request will cause order execution.

204. In my opinion, the claimed invention therefore changes both how information flows through the system and when the system performs order execution. Information does not flow directly from email to returned order email to order execution. Instead, the system establishes server-side authentication context, receives and parses a returned SMTP order email, validates returned values against that context, verifies expiration, and only then conditionally executes the order. Application, ¶¶ [0053]-[0058], [0074]-[0085], [0153]-[0155].

205. My opinions are based on my personal knowledge of the invention, my role in developing the disclosed architecture, my experience with email-based communications, secure identity, authentication, payments, and internet transaction systems, and my understanding of how a person of ordinary skill in the art would have interpreted the specification and claims at the time of the invention. I do not

offer these opinions as legal conclusions. Rather, I provide them to explain the technical context, technical problem, and technical improvement reflected in the disclosed and claimed architecture.

206. For the reasons stated in this declaration, a person of ordinary skill in the art would have understood the claimed invention as providing a technical improvement to SMTP/mailto-based transaction processing. The improvement lies in the expiration-bound, server-side authentication-control architecture that determines whether a returned email-generated transaction request should be trusted before order execution is permitted.

207. I declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true. I further declare that these statements were made with the knowledge that willful false statements and the like are punishable by fine or imprisonment, or both, under 18 U.S.C. § 1001, and that such willful false statements and the like may jeopardize the validity of the application or any patent issuing therefrom.

Signed by:
CE083E7D39D5499...

John P. Killoran, Jr.

DATE: 5/12/2026 | 12:52 AM EDT