Threat Analysis of Software Agents in Online Banking and Payments

Tamsanqa Ngalo, Hannan Xiao, Bruce Christianson
School of Computer Science
University of Hertfordshire
United Kingdom
Email: {t.ngalo, h.xiao, b.christianson}@herts.ac.uk

Ying Zhang
School of Computing and Engineering
University of West London
United Kingdom
Email: ying.zhang@uwl.ac.uk

***Abstract*—Software agents are the delegated subcontractors essential to connect the end-user to the bank and payment providers in a distributed service offering. This paper evaluates the key role that the different software agent types play to facilitate collaboration between clients and banks to perform online transactions. It highlights the threats and imminent risks that these software agents introduce in the chain as well as how these threats affect the trust relationship between principals. The discussed threats and resulting risks suggest vulnerabilities in the current software agent model which are beyond the bank and end users control. Both principals, the client and the service provider, are open to potential legal, security, quality of service, confidentiality and privacy compromises which influence the overarching trust relationship. There is resounding literature to illustrate advances that have been made to address the exposed challenges. However, a gap of misfortune remains where the software agent can act on its own accord exposing the contracting principals to internal and externally engineered threats thus tainting the trust relationship between these parties.**

1. Introduction

The transformation of banking and payment services from bricks and mortar to electronic channels has yielded some efficiencies to the industry, including financial and economic benefits. Online platforms offer reduced costs for servicing a large client base through a distributed network in the form of Internet Banking or e-commerce shopping and check-out platforms [1]. Customers can perform numerous banking and payment type transactions, such as money transfers; and bill payments, and purchase value-added services and view their balance, from anywhere in the world which they would otherwise have to perform inside a physical branch and at a fixed geographic location. Although security is of concern, the client sees benefits through convenience and accessibility, being able to transact from anywhere anytime [2].

Internet based servicing essentially brings the branch closer to a wider client base at a fraction of the cost in comparison to servicing a physical branch in all the respective client locations. To enable online transacting, the client engagement with the service provider occurs through a third-party agent in the form of a web browser, desktop or mobile application [3]. A client wishing to purchase goods from an online retailer on his computer must first launch a browser agent, such as Internet Explorer, and then shop through an online mall agent, such as Amazon, until they locate the right product and complete the sale. This remote anonymous collaboration between the client, agent and service provider presents many challenges, such as the client's ability to ascertain surety and confidence that he is transacting with an authentic provider who will deliver the acquired services. Additionally, the client wants to be satisfied that he's conducting the transaction through a reliable and secure agent channel that will treat his data with the expected protection and care. On the other hand, banks and payment providers use authentication to satisfy themselves that they are engaging with an authentic client [4]. Authentication is a process of verifying the users personal identity by means of matching personal and secondary credentials with the users profile, be it in person or remotely. There are some authentication models and methodologies available to service providers, from single-factor to multi-factor authentication schemes [5], [6].

From whatever perspective one looks at it, and whichever authentication system is employed, there is always the threat of intruders intercepting the internet session to steal the clients login credentials to perform malicious acts. Sometimes the intruder is agent itself misusing the collected transaction data, login credentials and personal data for other uses and gains outside the intended purpose [3]. Therefore, even when security measures like SSL and certificate protocols are an effective means to secure the channel and authenticate the agent there remains a potential data leak from the agent itself [7], [8]. Vamialis [9] poses a compelling analogy that if login credentials are synonymous to a house key, what is the likelihood of leaving the house key in an unsafe environment where any outsider could get access to it?

This research examines the threats posed by software agents in the online banking and payments context. We look at potential threats, challenges and vulnerabilities that such software agent model exhibits to both the client and service provider as well as table the current remedies and rating score of the threats. This paper is organised as follows: The following section discusses the different types of software agents involved in a typical internet banking and payment transaction. Then we provide an analysis of the threats posed by the defined software agents. Finally, we provide a summary and conclusions of our findings.

2. Software Agents in online Payments

The earliest definition of an agent can be derived from Carl Hewitt's proposal of the actor model' [10]. Hewitt defines an Actor as an independent control structure that acts on received messages and relays these messages to other Actors in the system through established protocol and having the following two characteristics and properties of operation: "the ACTION it should take when it is sent a message" and "its ACQUAINTANCES which is the finite collection of actors that it directly KNOWS ABOUT."

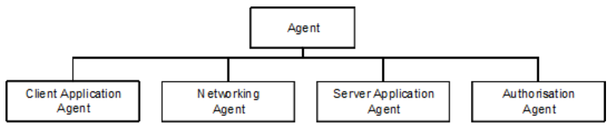
Nwana provides a modern definition of an agent and which is more descriptive and relevant to this researchs context [11]: "we define an agent as referring to a component of software and/or hardware which is capable of acting exactingly to accomplish tasks on behalf of its user."

Software agent performs computational tasks and routine work on behalf of the user or employing entity [11]. Software agents are the bridge, the channel and the essential go-between that brings the client and service provider together into a virtual marketplace, bank branch, advertising catalogue or social platform.

2.1. Types of Software Agents

There are different software agent types in the path of a transaction when considering online payments and banking. A typical transaction between a client and their bank would be conducted via the web browser on a public network which in turn encapsulates the virtual payment as the transaction transverses from the server to server, router to router until its final destination. Following Nwana [11], we identify the agent types depicted in Fig. 1, which are classified according to the role that they play in the processing of a banking and payment transaction between the user and service provider.

Figure 1. Agents classification in online payments



Client Application Agent. This software agent acts as the users main interface into online hosted banking and payment services. In the context of this research, the web browser is the most commonly used client agent. However, mobile applications are becoming more and more prevalent as seen in the likes of mobile banking and marketplace applications. The penetration of internet based user resources and the exponential growth of cloud services such as Gmail accounts, Facebook, PayPal has warranted the need to share access to these resources amongst different applications. Authentication protocols such as OAuth enable third party applications to become software agents and consume hosted services from another service provider [12]. In addition to access to the actual website, OAuth is a typical example of the interface function performed through the client application agent when used to facilitate a payment, for example through PayPal or a Two Factor Authentication (2FA) provider [5].

Networking Agent. This is the software and firmware code that packages the payment data message into the many different messaging, network and encryption protocols. For example, the web browser embeds a networking agent that captures the application and user data to/from HTML and may go as far as to negotiate an SSL session with the destination application server on behalf of the user [3]. User self-securing schemes such as The Onion Routing (TOR) also become networking agents as they wrap and unwrap the onion on its way to and from the final destination [13].

Server Application Agent. This is the web application, which resides on the banks servers, and may execute on the users browser to facilitate the online banking and payment transactions. The key attribute of a server application agent is its mobility or ability to be hosted centrally and be executed remotely. For example, the amazon.com website is a server application agent which collaborates with the browser client application agent to enable the trading of goods on that website.

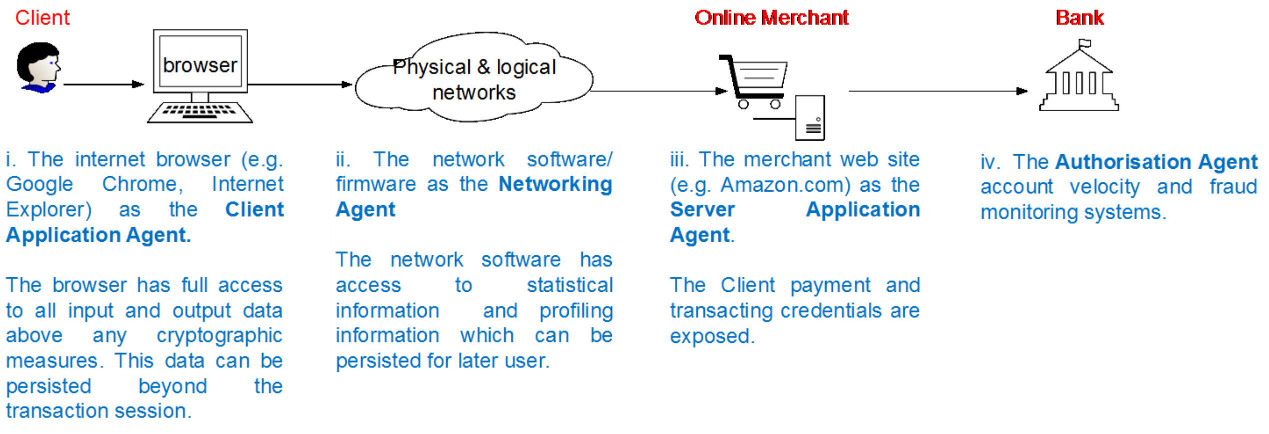
Authorisation Agent. These are the account velocity management and fraud monitoring systems that evaluate pre and post transaction data to perform some intelligent logic to notify and trigger a management or operational decision. For example, when a payment transaction takes place the bank and payment providers pass transaction data into their account management authorisation agent software which in turn would report if certain parameters have been exceeded to alarm of a potential fraud or if the users account balance is sufficient to honour the transaction.

2.2. A Typical Case Study

The defined agent types can be illustrated through a typical online purchase transaction as depicted in Fig. 2. It must be noted, however, that this use-case describes a simplified process flow and that software agents can take on multiple agency roles, depending on their function in the chain. For example, a typical web browser can act as both a client application and a networking agent.

In our use-case, a client seeking to purchase a book from an e-commerce merchant such as Amazon.com conducts the transaction through software agents that fulfil different roles on behalf of the client. The client launches their browser, enters the e-commerce URL and shops for their desired book. Once selected, the client is prompted for payment

Figure 2. A typical online payment transaction


Client
browser
Physical & logical networks
Online Merchant
Bank
i. The internet browser (e.g. Google Chrome, Internet Explorer) as the Client Application Agent.
The browser has full access to all input and output data above any cryptographic measures. This data can be persisted beyond the transaction session.
ii. The network software/ firmware as the Networking Agent
The network software has access to statistical information and profiling information which can be persisted for later user.
iii. The merchant web site (e.g. Amazon.com) as the Server Application Agent.
The Client payment and transacting credentials are exposed.
iv. The Authorisation Agent account velocity and fraud monitoring systems.

card details through the check-out process which in-turn communicates with the client's bank to fulfil payment.

Step i. The client initiates the Internet browser application, such as Chrome or Internet Explorer. The browser client application agent is responsible for interpreting and presenting the HTML pages to the user as well as invoking the networking agent when necessary to request and post pages to/from the host server application. The browser application serves the web pages of the online store and relays the user purchase and session data to the retailer or marketplace server through the networking agent.

Step ii.The networking agent collects the user and application data messages and its purpose is to cipher, transport and deliver these messages to the host server (and vice versa) utilising the underlying network communication and infrastructure which itself may comprise of a collection of other networking agents, such as the popular OpenSSL (https://www.openssl.org/). The data message is decoded and encoded to conform to the respective systems messaging and encryption protocols as it transverse the network until it reaches the destination.

Step iii.The e-commerce web server encapsulates the navigation, application and check-out process and it renders the web pages remotely on the clients internet browser. During the check-out process, the e-commerce website assumes the payment function or delegates this to a processor, such as PayPal (www.paypal.com) or Amazon Payments (https://payments.amazon.co.uk/), which takes over the capturing of the card or payment instrument details and performs the actual payment authorization with the respective issuing bank. The capturing of card payment data is facilitated through the client application agent (the browser) which serves as the interface agent between the merchant, the bank and the end user.

Step iv. Finally, the bank receives the payment request to authorise the transaction against the client's account balance. As part of this process, the authorisation agent validates the payment information for fraud monitoring and performs velocity tests to detect any anomalies in the client's transacting patterns and behaviour to alert the bank.

3. Software Agents Threat Modelling Tools

The prevalence of the internet of things and the fast adoption of mobile devices is continually placing more internet capable computers around people. Ubiquitous usage of computers where users roam and consume the nearest fixed or mobile online device to access their cloud resources is now a reality. Software agents are the dominating tools and common place to access and consume online services and as such require serious attention and consideration to privacy and personal data practices [3], [14].

The Microsoft STRIDE threat model is used to categorise the software agent threats while the DREAD system is applied to rate the severity of identified threats [15], [16].

STRIDE Threat Model. The model is applied and classified into the following payments and banking related threats.

S = Spoofing/Deception, the threat of perpetrators ability to illegally access a users banking account or payment profile to transact as if they were the rightful owner of that account, profile or payment instrument.

T = Tampering, the threat of tampering with the user or provider data, be it during the transaction or data stored in the provider's database.

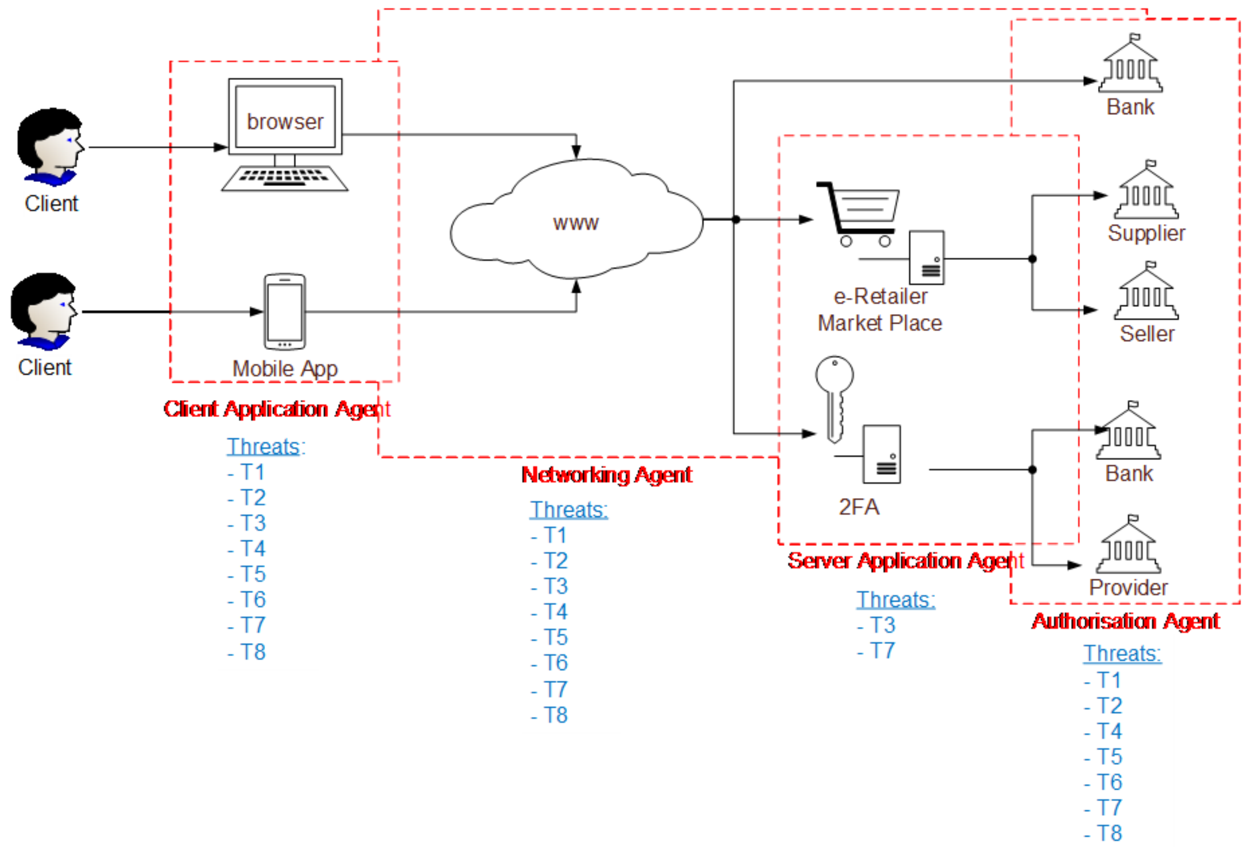
R = Repudiation, the threat of exposing the payment provider or bank to false claims of liability.

I = Information Disclosure, the threat of unwillingly exposing data to external parties and intruders.

D = Denial of service, the threat of disrupting the legitimate user or provider from accessing their account or completing the intended transaction or service in whole or in part.

E = Escalation of Privileges, the threat of granting unauthorised access and rights to the banking account, payment profile and instrument such that these impending users can transact and act on the account as if fully trusted.

Figure 3. Presentation of threats and areas of vulnerability



DREAD Risk Model. The risk assessment model is applied to each identified threat by assigning a severity score of between 1 and 10 to the DREAD review components. The final score of every identified threat is divided by 5 to obtain an average scoring between 1 and 10. 1 indicates a low risk while 10 indicates a high risk. The DREAD components are:

D = Damage What is the prevalence of the problem and how much damage can it cause?

R = Reliability How likely is the threat to succeed?

E = Exploitability How easy is it to exploit the threat?

A = Affected Users How many users does the threat affect?

Di = Discoverability How easy is it to discover and fix? A threat not likely to be discovered and/or hard to fix scores high.

4. Threats Analysis of Payments Software Agents

The case study described in section 2.2 is used to exemplify the threats to which each software agent type exposes on a typical online payment and banking transaction. Figure 3 is a graphical representation of the identified threats and areas of vulnerabilities. Table 1 is a listing of the identified threats as analysed and prioritised per the DREAD rating. The mitigations will be discussed in the next section.

The STRIDE Denial of Service (DoS) threat was not considered in most of the analysed threats as it is the belief of the authors that such an attack is relatively common and easy to deploy nowadays. It is therefore assumed that any of the discussed threats have a good potential to result in a successful DoS attack. We argue, though, that a DoS threat is of specific significance in the Man-in-the-browser threat as it not only denies the rightful user access to services but also threatens to alter the transaction data such that an illegitimate recipient may benefit from the transaction instead of the intended recipient.

4.1. Threat Analysis of Client Application Agent

This is the local application resident on the users computer or mobile device. This is the most used and frequent type of software agent that the user interacts with directly when he/she wants to access online payment and banking services. In this case, the user invokes their web browser on their computer or mobile device to perform an online purchase. The user types the web address (URL) of the

TABLE 1. Threat Analysis

No.	Threat	Consequence	STRIDE Category	DREAD Rating	Current Mitigation
T1	Internal software attack, such as Man in the browser attack	User personal, account, payment and login credentials are leaked out which may result in a compromise of the user account.	S, T, R, I, D, E	9	Resolved through user education of dangerous agents, whitelisting or blacklisting of the harmful software agents as well as applying fuzzy logic to detect and disable the malicious actors from accessing the system [17], [18]
T2	Software agent leaking users information	User personal, account, payment and login credentials are leaked out which may result in a compromise of the user account.	S, R, I, E	8	Agents ask the user for permission to access and use the collected personal information for profiling and other purposes [19]
T3	User unawareness and general lack of knowledge	User login credentials and personal data are exposed and may be used to attempt access to the account	S, R, I, E	8	Resolved through user education of dangerous agents, whitelisting or blacklisting of the harmful software agents as well as applying fuzzy logic to detect and disable the malicious actors from accessing the system [17], [18]
T4	Interception and data alterations	This is the threat towards data integrity. User personal, account, payment and login credentials are leaked out which may result in a compromise of the user account.	S, R, I, D, E	7	Employ symmetric or asymmetric encryption solutions to secure the channel between the software agent and the service provider [14].
T5	Bad service, representation, miscommunication and threat to the brand, reputation and trust relationship between the user and the bank or payment provider.	The clients lose trust in the bank or service provider thus turning their business elsewhere. Clients may distrust the electronic channels and migrate back to the expensive brick and mortar channels. The reputation of the bank or service provider is tainted.	S, T, R, I, E	7	Currently, resolved through user education; whitelisting or blacklisting of the harmful software agents as well as applying fuzzy logic to detect and disable the malicious actors from accessing the system [17], [18]. We argue that this threat is not addresses adequately mitigated at present.
T6	Decompiling, repackaging and simulation of the software agent code or application.	User login credentials are exposed. The user transaction data can be modified in the background to benefit fraudsters.	S, T, R, I, E	6	Using cryptographic schemes to sign the original code and scan and match the certificate for originally and integrity. Whitelisting of trusted applications.
T7	External access through vulnerable operating environment.	The agent is compromised, and client and provider data is extracted for malicious use as well as access to the user's account and other payment information.	S, R, I, E	6	Take precaution and measures to use good coding practices to secure the application for the environment [19], [20], [21].
T8	Non-compliance with industry standards and regulations such as Payment Card Industry (PCI) standard as well as privacy laws.	Data protection and security measures are weakened, and the database and personal information is exposed and can be leaked for malicious use.	S, T, R, I, E	6	Publishing of standards as well ongoing audits to ensure software agent providers comply. However, some service providers with small transaction volumes are not subjected to the same level of scrutiny. User education of harmful agents; whitelisting or blacklisting of the dangerous software agents as well as applying fuzzy logic to detect and disable the malicious actors from accessing the system [17], [18].

e-commerce merchant into the browser, and the browser application takes over the underlying HTTP protocol POST and GET exchange until the merchant's marketplace is fully loaded.

The user interacts with the merchant's online store inside the browser application. The Browser application has full and clear unencrypted access to all the data input fields such as client selections, preferences, login credentials, billing address, residential address, current geographic location, account balance, transaction history, spending patterns, account settings, user profile and payment card information. Some of this client personal information is input directly by the user while others are retrieved from the merchants stored database.

Furthermore, the browser application runs and operates inside an environment such as Microsoft Windows, Android or similar which may have direct access to the browser internal storage area. The following threats have been identified which expose the user and payment or bank service provider to vulnerabilities:

T1. User data inside the web browser or client application agent is handled and presented in clear unencrypted form which enables internal software attacks such as man-in-the-browser (MITB) that exploit, capture and share private and sensitive information with the outside world without the users permission or knowledge [18]. This can be achieved by injecting malicious code through JavaScript, browser extensions and web views.

T2. The browser application can leak information for advertising, mining, user profiling and malicious use outside the intended application. This leaked information can be used to gain unauthorised access to the user's bank or payment account to syphon data for marketing or fraud purposes as well as the possible escalation of privileges for illicit use [22], [23].

T3. The unsuspecting user, and due to general lack of knowledge, can install malicious browser plug-ins and accept weak settings as well as incorrectly granting access to the application to mine personal information or obtain unwarranted access to the users banking account. This can further extend to other threats using personal user information to launch phishing attacks [24].

T4. The underlying objective of these defined threats within the client application agent is the access to user authentication credentials and personal information. User data may be intercepted, leaked and modified resulting in unintended results or unauthorised access to account and payment information from the bank or service provider through the agent [17], [25].

T5. The browser or client application agent acts as the middle man and virtual interface between the client and the bank or payment service provider. This relationship presents a threat of mistreating the customer or providing inferior service through bad presentation, miscommunication or poor service quality. All these factors threaten the brand, reputation and, most importantly, the trust relationship between the user and bank or payment provider [26], [27].

T6. Intruders can decompile and repackage the browser application with inserted malicious code. Software Development Kits (SDK) and libraries are readily available on the internet making it easy for anyone to develop and deploy an application simulating an authentic software agent. Open source and the connectedness of expert resources around the world has made it easier to outsource parts and portions of software functions to open source communities. Open source projects such as OpenSSL and TOR are open to recompilations, repackaging and application simulation [13].

T7. The operating environment, such as Android and Windows, can also present its vulnerabilities which can extend onto the browser application, therefore, exposing the data and operations performed by the client through the software agent to external threats. This can be through means of inter-application communication or other security exploitations like memory access, weak permissions management and low-level API access [19], [20], [21].

T8. The browser application, being generic by nature of its design, does not place emphasis and priority on compliance with banking and payments industry standards and regulations such as PCI standard and published legal regulations [14], [21]. Similarly, non-conformance with privacy laws and data retention policies in the different jurisdictions and countries exposes the service provider to legal risks [9].

4.2. Threat Analysis of Networking Agent

Software agents in their majority rely on the networking agent to ensure reliable and, at most times secure, delivery of the message contents between the agent and host or another agent party in between. This is the local software agent responsible for packaging the messages between the browser and the e-commerce merchant web server. This software is embedded as firmware in the form of dedicated micro-controllers inside networking equipment routers and switches as well as suites of applications installed at the Internet Service Provider (ISP). In addition to relaying or routing the messages, the networking agent can profile the user as well as gain access to some or all the clear unencrypted data. The following threats have been identified for the type of software agent:

T1. Networking agents may present Internal software attack risks, such as the OpenSSL heartbeat vulnerability which lead to the Heartbleed attack.

T2. The software responsible for packaging and message transportation has capabilities to leak information and profile both the client and principal for usages, frequency of transacting and possibly leak the payment transaction data.

T3. Of the level at which the networking agent operates, they cannot be easily detectable and as such users are not aware of their existence or even their expected call to action.

T4. The networking agent can be intercepted, and man-in-the-middle attacked through insecure networks and infrastructure. A typical example is the MITM Heartbleed attack on the widely used OpenSSL, which risks any and all applications that relied on it for security on the channel.

T5. The networking agent can give substandard service by throttling or setting lower priority to the e-commerce client and merchant thus providing an inferior service. The unknowing client assumes this to be the service of the merchant which threaten the brand, reputation and most importantly the trust relationship of the bank or payment provider.

T6. Open source and the connectedness of expert resources around the world has made it easier to outsource parts and portions of software functions to open source communities. Open source projects such as OpenSSL and TOR are open to recompilations, repackaging and application simulation [13].

T7. The operating environment, such as Linux and Windows, can also present its own vulnerabilities which can extend onto the networking agent software, therefore, exposing the data and operations performed by the client through the software agent to external threats. The success of the Target data breach was due to environmental vulnerabilities that were compromised through the heating, ventilation and air-condition system.

T8. The internet service provider software can unduly maintain some of the logs and transportation data in conflict with banking and payments industry standards and regulations this further extends to non-conformance with privacy laws and data retention policies in the different jurisdictions.

4.3. Threat Analysis of Server Application Agent

This is the principal e-commerce retailers web site application that contains the logic, content and processes to advertise and sell the product. This can also be the banks virtual branch in the form of Internet Banking. The web server is responsible for guiding the client through the purchase transaction process up to the point of taking payment, which could be a process undertaken by or delegated to the client application agent. Although by in large it is controlled and maintained by the principal (the bank or payment service provider), the server application agent executes remotely on the clients computer or mobile device inside the client application agent (the browser) which presents the following threats.

T3. Due to a general lack of knowledge to make the right choices, the unsuspecting user may install malicious browser plug-ins, helpers and extensions, accept weak settings as well as incorrectly grant access to the application to mine personal information or obtain unwarranted access to the users banking account. This can extend further to other personal information threats such as phishing attacks [24].

T7. The operating environment, such as Android and Windows present vulnerabilities to the e-commerce virtual mall or Internet Banking virtual branch, therefore, exposing the data and operations performed by the client through the software agent to external threats such as memory leaks, weak permissions management and data access [19], [20], [21].

4.4. Threat Analysis of Authorisation Agent

These are the account velocity, fraud monitoring and alert systems deployed by the bank or payment provider. These can also be user profiling software that guides the merchant or bank for marketing purposes.

T1. Interfacing with or embedding external or third party software exposes the authorisation agent to Internal software attack risks such as injection of malicious code for predictable outcomes.

T2. The software responsible for alerting against fraud and malicious actions or guiding the bank towards preventative measures can also capture the resulting outcomes for marketing and other unintended purposes.

T4. Data within the authorisation agent domain can be intercepted and manipulated for malicious gain. In May 2016, over 8 million was fraudulently withdrawn from cash machines from a South African bank [28] in a space of a few hours without the awareness from the banks alert and monitoring systems.

T5. The authorisation agent can miscalculate a condition and incorrectly grant/deny access to the end user. The South African heist incident reported in [28] is a typical example. This can further be supported by the growing number of Ransomware incidents where institutions databases are concealed to deny service, however, what is more pertinent to this research is the unpredictable response and judgement of the authorisation agent in such an event.

T6, T7. Due to their autonomy and delegation to third party providers, authorisation agents are susceptible to operating system threats as well as recompilations, repackaging and application simulation [13], [29].

T8. The authorisation agent can maintain logs and data outside its mandate with the principal bank therefore in violation of the industry norms and standards.

4.5. Agent Threats Comparison

Table 2 summarises and compares the threats identified in the respective agent types. This comparison illustrates the prevalence of threats 1 through to 8 in the Client Application, Networking and the Authorisation Agents indicating grave challenges in the present implementation of software agent relationships in internet banking and mobile payments. The Server Application agent observes the user impersonation and external threats due to environmental vulnerabilities such the recent spread of Ransomware attacks. The Authorisation agent does not record the impersonation threat as it operates within a trusted environment on data that should already have been pre-authenticated. The underlying outcome is that none of the categorised agent types are exempt from any serious threats.

TABLE 2. COMPARISON OF THREATS ON DIFFERENT SOFTWARE AGENTS

Agent Type	T1	T2	T3	T4	T5	T6	T7	T8	T9
Client App.	✓	✓	✓	✓	✓	✓	✓	✓	✓
Networking	✓	✓	✓	✓	✓	✓	✓	✓	✓
Server App.			✓					✓	
Authorisation	✓	✓		✓	✓	✓	✓	✓	✓

5. Conclusion

Software agents are delegated subcontractors essential to connect the end-user to the bank and payment providers in a distributed service offering. Trust is a vital component in the continued acceptance and maintenance of reliable remote payment and banking of which these delegated software agents are key to its accomplishment. This paper evaluated the critical role that the different software agent types play to facilitate collaboration between clients and banks to perform online transactions. It highlights the threats and impending risks that these software agents introduce in the chain as well as how these threats affect the trust relationship between principals. Some risks have questioned the bank or payment provider's execution of its obligations to the client, the legal and regulatory bodies and its business mandate. The discussed threats and resulting risks expose vulnerabilities in the current software agent model which are beyond the bank and the end users control. Both the client and bank are open to liabilities and security risks as well as confidentiality and privacy compromises affecting the trust relationship.

This security vulnerable environment makes it difficult, if not impossible, for the banks to guarantee the quality and service offering due to the possibility of interception of user credentials and personal data that can occur anywhere in the distribution, including inside the agent itself, which by virtue of reliability or continued use may have won the users confidence. We identify a gap of misfortune to both principals (the bank and client) emanating from the presented inadequacies in the existing software agent models. An opportunity exists for a different software agent that is representative of both the client and the service provider's interests to forge and maintain trust relationships between these parties in open distributed computer systems.

References

[1] F. Ecer, "Performance evaluation of internet banking branches via a hybrid mcdm model under fuzzy environment." *Economic Computation & Economic Cybernetics Studies & Research*, vol. 49, no. 2, pp. 200 – 218.

[2] M. Wu, C. Jayawardhena, and R. Hamilton, "A comprehensive examination of internet banking user behaviour: evidence from customers yet to adopt, currently using and stopped using." *Journal of Marketing Management*, vol. 30, no. 9-10, pp. 1006 – 1038.

[3] J. Zhu and B. C. Desai, "User agent and privacy compromise," in *Proceedings of the Eighth International C* Conference on Computer Science & Software Engineering*, ser. C3S2E '15, 2008, pp. 38–45.

[4] M. Just and D. Aspinall, "On the security and usability of dual credential authentication in uk online banking," in *2012 International Conference for Internet Technology and Secured Transactions*, Dec 2012, pp. 259–264.

[5] J. C. Liou and S. Bhashyam, "A feasible and cost effective two-factor authentication for online transactions," in *The 2nd International Conference on Software Engineering and Data Mining*, June 2010, pp. 47–51.

[6] B. Schneier, "Two-factor authentication: Too little, too late," *Commun. ACM*, vol. 48, no. 4, pp. 136–, Apr. 2005. [Online]. Available: http://doi.acm.org/10.1145/1053291.1053327

[7] S. D. RAMCHURN, D. HUYNH, and N. R. JENNINGS, "Trust in multi-agent systems," *The Knowledge Engineering Review*, vol. 19, no. 1, p. 125, 2004.

[8] Y.-J. Hu, "Some thoughts on agent trust and delegation," in *Proceedings of the Fifth International Conference on Autonomous Agents*, ser. AGENTS '01, 2001, pp. 489–496.

[9] A. Vamialis, "Online service providers and liability for data security breaches." *Journal of Internet Law*, vol. 16, no. 11, pp. 23 – 33.

[10] "Viewing control structures as patterns of passing messages," *Artificial Intelligence*, vol. 8, no. 3, pp. 323 – 364, 1977.

[11] H. S. Nwana, "Software agents: an overview," *The Knowledge Engineering Review*, vol. 11, no. 3, p. 205244, 1996.

[12] B. Leiba, "Oauth web authorization protocol," *IEEE Internet Computing*, vol. 16, no. 1, pp. 74–77, Jan 2012.

[13] P. Syverson, "A peel of onion," in *Proceedings of the 27th Annual Computer Security Applications Conference*, ser. ACSAC '11, 2011, pp. 123–137.

[14] L. Nosrati and A. M. Bidgoli, "Security assessment of mobile- banking," in *2015 International Conference and Workshop on Computing and Communication (IEMCON)*, Oct 2015, pp. 1–5.

[15] Z. Yang and Z. Zhang, "The study on resolutions of stride threat model," in *2007 First IEEE International Symposium on Information Technologies and Applications in Education*, Nov 2007, pp. 271–273.

[16] D. LeBlanc. (2007) Dreadful. [Online]. Available: https://blogs.msdn.microsoft.com/david_leblanc/2007/08/14/dreadful/

[17] J. Yu and T. Yamauchi, "Access control to prevent attacks exploiting vulnerabilities of webview in android os," in *2013 IEEE 10th International Conference on High Performance Computing and Communications*, 2013, pp. 1628–1633.

[18] "Man-in-the-browser-cache: Persisting https attacks via browser cache poisoning," *Computers & Security*, vol. 55, pp. 62 – 80, 2015.

[19] "Permission based android security: Issues and countermeasures," *Computers & Security*, vol. 43, pp. 205 – 218, 2014.

[20] S. Fahl, M. Harbach, M. Oltrogge, T. Muders, and M. Smith, "Hey, you, get off of my clipboard," in *Financial Cryptography and Data Security*, A.-R. Sadeghi, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 144–161.

[21] M. El-Serngawy and C. Talhi, "Captureme: Attacking the user credential in mobile banking applications," in *2015 IEEE Trustcom/BigDataSE/ISPA*, vol. 1, Aug 2015, pp. 924–933.

[22] I. Leontiadis, C. Efstratiou, M. Picone, and C. Mascolo, "Don't kill my ads!: Balancing privacy in an ad-supported mobile application market," in *Proceedings of the Twelfth Workshop on Mobile Computing Systems &#38; Applications*, ser. HotMobile '12. New York, NY, USA: ACM, 2012, pp. 2:1–2:6. [Online]. Available: http://doi.acm.org/10.1145/2162081.2162084

[23] A. Sadeghi, H. Bagheri, and S. Malek, "Analysis of android inter-app security vulnerabilities using covert," in *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, vol. 2, May 2015, pp. 725–728.

[24] A. Vishwanath, "Habitual facebook use and its impact on getting deceived on social media," *Journal of Computer-Mediated Communication*, vol. 20, no. 1, pp. 83–98, 2015.

[25] Z. S. Al-Salloum and S. D. Wolthusen, "Threat analysis model of an agent-based vulnerability mitigation mechanism using bayesian belief networks," in *2011 IEEE Network Science Workshop*, June 2011, pp. 144–151.

[26] Y. Qu, W. Rong, Y. Ouyang, H. Chen, and Z. Xiong, "Social aware mobile payment service popularity analysis: The case of wechat payment in china," in *Advances in Services Computing*, L. Yao, X. Xie, Q. Zhang, L. T. Yang, A. Y. Zomaya, and H. Jin, Eds. Cham: Springer International Publishing, 2015, pp. 289–299.

[27] M. S. Sohail and I. M. Al-Jabri, "Attitudes towards mobile banking: are there any differences between users and non-users?" *Behaviour & Information Technology*, vol. 33, no. 4, pp. 335–344, 2014.

[28] D. Demetriou and A. Laing. (2016) Thieves steal 8.8m in ATM bank heist in Japan, The Telegraph. [Online]. Available: http://www.telegraph.co.uk/news/2016/05/23/thieves-steal-88m-in-atm-bank-heist-in-japan/

[29] B. Christianson, "Delegation and not-so smart cards," in *Security Protocols*, B. Christianson, B. Crispo, W. S. Harbison, and M. Roe, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 158–167.