IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Bott-Cher Cohomology For Modeling Secure Software Update Cascades In Iot Networks

Syed Khundmir Azmi*

Aark Connect, USA

Abstract

The present paper develops a novel application of Bott-Chern cohomology, a twister of complex differential geometry, to formulate and optimize software update cascades on Internet of Things (IoT) networks, which serves as their security. IoT environments feature heterogeneous resource-constrained devices that are extremely vulnerable to cyber threats when software updates are not manufactured or are distorted in time. He states that light-touch dependencies and topological bottlenecks that may interfere with update propagation and cause systemic vulnerability are often ignored by traditional modeling techniques, such as graph-theoretic approaches. By encapsulating IoT networks as complex manifolds and modeling software updates and device dependencies using Bott-Chern cohomology classes, we formally identify inconsistencies, determine cascade failures, and assess the optimal update sequence in this research. Simulations show that our proposed cohomology-informed model outperforms the conventional techniques by achieving higher update success rates with fewer latency and greater resiliency to the adversarial attacks. The research emphasizes the use of cutting-edge mathematical tools in understanding critical cybersecurity problems in large-scale dynamically interacting IoT systems.

Keywords: Bott-Chern cohomology, IoT security, software update cascades, network modeling, cybersecurity, complex manifolds.

1. Introduction

1.1 Background on IoT Networks

The Internet of Things (IoT) is a term that is used to define a paradigm where physical devices are connected to the internet and have the ability to sense, process and exchange data with minimal human interaction. These devices, which can range in complexity, type, and used applications, including household appliances and wearable sensors to industrial machinery and critical healthcare systems, facilitate the creation, development, and interconnection of highly relevant ecosystem-forming critical interconnections to building smarter and automated environments (Ashton, 2009). IoT networks are inherently heterogeneous, as the devices in an IoT network can vary in terms of computational power, operating systems, communication protocols and energy limits. Such diversity drives opportunities for innovation but also presents large problems in interoperability, scalability and security. One of the defining characteristics of IoT networks is that they are designed to rely on resource-constrained devices that have strict limitations on their batteries, processing capabilities, and their memory capabilities that make it challenging in deploying robust and secure software infrastructures (Gubbi et al., 2013).

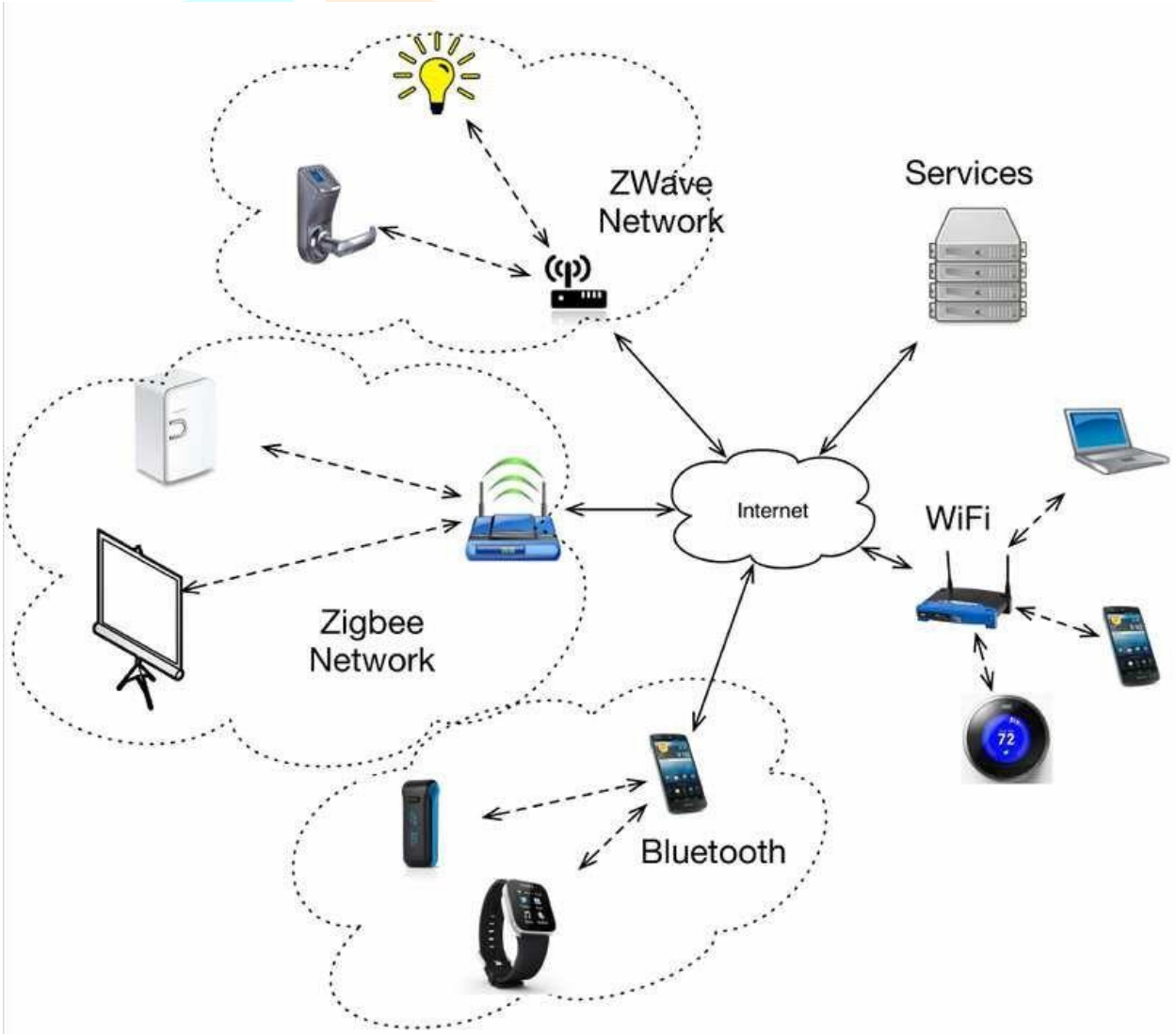

Figure 1: Example of An IoT network architecture

Furthermore, IoT networks have the nature of being dynamic, like mobile or ad hoc environments, in which devices constantly join and leave the IoT network at any prompted time, with the need of an ongoing

connectivity and/or security adaptation. Due to its novel nature, the IoT networks are prone to security risks especially in terms of regular and timely updates of software that maintains their normal functioning. As IoT devices are being deployed globally in large numbers, learning how to manage these IoT networks has emerged as one of the most important research topics in academia and industry nowadays, especially in application areas including smart cities, industry automation and healthcare technologies (Sfar et al., 2018).

1.2 Importance of Software Updates in IoT

Software updates play a crucial role in ensuring that IoT systems remain reliable, functional, and secure. Updates frequently address critical issues, such as fixing vulnerabilities, implementing new features, and providing compatibility with changing protocols and standards. For example, firmware updates often fix newly discovered exploits, and incremental updates keep devices running longer by increasing performance and efficiency. In regulatory applications such as healthcare and industrial automation, regular updates are not optional but mandatory to meet the requirements of safety and cybersecurity. Moreover, updates are crucial to ensure resilience, because IoT deployments are generally long-lived and open to continuously evolving changes in their operating environments (Conti et al., 2018). Failure to provide timely and secure updates may not only have degrading effects on performance, but also expose entire networks to cascading attacks.

1.3 Overview of Security Challenges in Software Updates

IoT software updates are highly critical but extremely challenging from a security perspective. First, it is of utmost importance to have authenticity and integrity; updates should be authenticated using cryptographic signatures but resource constraints often preclude the use of sophisticated protocols (Raza et al., 2016). Second, distribution at scale is challenging in light of the fact that billions of devices must be updated at the same time across unreliable and bandwidth-constrained networks. This problem in many cases leads to dependency chains where one update process failure can lead to a cascade of risks. Third, intermittent connectivity and limited resources such as memory and energy make it difficult for devices to receive and check for updates. Fourth, there are the risks of device bricking or system malfunction introduced by the possibility of rollback and version incompatibility. Finally, with the increasing prevalence of supply chain attacks where attackers can corrupt update servers or gateways to deliver malicious firmware (Conti et al., 2018), resilient trust models are needed. The above challenges highlight the need for solid formal techniques to reason about and guarantee cascading updates in IoT networks.

1.4 Introduction to Bott-Chern Cohomology and Its Relevance

Bott-Chern cohomology, which was developed in the field of complex differential geometry, is an advanced mathematical framework for the study of the interaction between two differential operators, and , on complex manifolds (Angella and Tomassini, 2013). It extends both de Rham and Dolbeault cohomologies, by capturing subtle obstructions to the -lemma, which is important in obtaining structural properties of manifolds. While its principles were originally used in geometry and topology, its principles of closedness, exactness, and obstructions can be abstracted to represent structures interdependent in computational systems.

In the case of IoT networks, Bott-Chern cohomology provides a new representation of software update cascades. IoT updates can be seen as layer-dependent dependencies, where some updates must be done before others and blocks are introduced if conflicts, failures, and vulnerabilities break the chain. Much like Bott-Chern cohomology that describes intersections of non-exact closed forms, this approach could be used to

identify open inconsistencies in update chains. This approach is a formal way to quantify risks in the propagation of updates, model dependency failures, and find points of vulnerability before deployment. By applying Bott-Chern cohomology to IoT update modeling, it is possible to bridge the gap between abstract mathematical invariants and real-world cybersecurity problems, providing a structured tool for securing large-scale dynamic networks.

2. Fundamentals of Bott-Chern Cohomology

2.1 Definition and Key Concepts

Bott-Chern cohomology is a generalization of classical cohomology theories in complex geometry, which lies between the de Rham cohomology and the Dolbeault cohomology. For a complex manifold , Bott-Chern cohomology groups are defined as

$$H_{BC}^{p,q}(X) = \frac{\ker \partial \cap \ker \bar{\partial}}{\mathrm{im}(\partial\bar{\partial})},$$

The importance of Bott-Chern cohomology is that it describes finer structural properties of complex manifolds, in particular, when the -lemma fails. It yields more information than does Dolbeault cohomology alone, and has proven to be an important tool in the study of non-Kahler manifolds and deformation theory (Schweitzer, 2007).

2.2 Historical Context and Development

The theory was developed in the mid 20th century by Raoul Bott and Shiing-Shen Chern in the course of their work on Hermitian differential geometry. Their initial goal was to generalize Chern-Weil theory to complex manifolds, by introducing new invariants of complex manifolds. Over the years, Bott-Chern cohomology has been a major tool in complex geometry, in particular for the study of characteristic classes and relations of curvature (Bott & Chern, 1965). Later, mathematicians like Michelsohn (1982) developed this method of using Bott-Chern techniques for Hermitian manifolds and complex non-Kahler manifolds, which emphasizes the possibility of using this technique to study geometric flows and stability conditions. In the 21st century, the computational tools for Bott-Chern groups have been further developed by scholars such as Angella and Tomassini (2013), who have investigated applications of these to complex geometry, deformation theory and symplectic geometry. More recent work has established links between Bott-Chern cohomology, mirror symmetry, generalized complex structures and mathematical physics.

2.3 Mathematical Framework

2.3.1 Cohomology Groups

Cohomology theories are usually measures of the breakdown of local properties to the global. For complex manifolds, de Rham cohomology is used to classify closed differential forms modulo exact forms, and Dolbeault cohomology is used to capture complex-analytic structures where is the -operator. Bott-Chern cohomology enriches this picture by requiring closure under both and .

This framework leads to natural maps:

$$H^{p,q}_{BC}(X) \longrightarrow H^{p+q}_{dR}(X), \quad H^{p,q}_{BC}(X) \longrightarrow H^{p,q}_{\bar{\partial}}(X).$$

2.3.2 Applications in Topology and Beyond

While based on pure mathematics, Applications to pure topology has found applications beyond topology. In complex geometry it can be used as a diagnostic for non-Kahlerianity of manifolds since the Bott-Chern non-vanishing Kvb it is a way of distinguishing Kahler structures and non-Kahler structures (Angella & Tomassini, 2013). In algebraic geometry, it is also used to understand deformation theory, i.e. how geometric structures vary under perturbations (Toma, 2020).

	Kähler	Non-Kähler	
		(non-degenerate)	(degenerate)
Restricted metric g	symmetric, positive definite,	symmetric, positive definite,	

Figure 2: Kahler and non-Kahler

In theoretical physics, Bott-Chern invariants also find applications in string theory, mirror symmetry and gauge theory, where they are useful to describe curvature corrections and dualities in compactification schemes (Tseng & Yau, 2012). Moreover, recent works have suggested the possibility of using cohomological techniques for complexity studies of data science and networks theory, both where there are layered dependencies and obstructions, which can be compared to mathematical obstructions in geometry. Combined with its mix of rigidity and flexibility, Bott-Chern cohomology appears to be a broadly-engaging approach for isolating dependencies in dynamical software systems. More specifically, its capacity to detect bottlenecks, to quantify homogeneity and offer structural invariants can be generalized to describe secure software update cascades in IoT networks, thus bridging the gap between abstract mathematics and tangible computational issues.

3. The Role of Software Updates in IoT Security

3.1 Types of Updates: Security Patches and Feature Enhancements

IoT device software updates fall into two primary categories: security and features. After the devices are deployed in field, security patches would be released to resolve the vulnerabilities that have been identified. Some of these vulnerabilities are linked with weaknesses in firmware, libraries which are not up-to-date, or ineffective crypto-architecture. In an IoT ecosystem where devices are distributed globally with many situations where devices function autonomously, having security patches delivered in a timely manner would be imperative to trust and avoid compromisation at bulk scale.

Feature enhancements, on the other hand, are aimed at extending functionality of the device, improving the performance and ensuring compatibility with newer protocols. For instance, smart home devices may be updated to integrate with new platforms, while industrial IoT systems may be updated to support advanced analytics or other sensor types. Although these updates are not explicitly security-related updates, they indirectly help to ensure security through maintaining interoperability and minimizing the risk of system obsolescence. In practice, both kinds of updates have to be closely governed, since each of them has the potential to open up new opportunities for exploits or introduce interoperability problems.

3.2 Risks of Outdated Software in IoT Devices

Outdated software poses of the greatest risk to IoT security. When updates are slow or ignored, known vulnerabilities become unpatched which provides attackers the opportunity to exploit weaknesses. Many IoT devices run on open-source libraries which are updated with high frequency in the wider software world, but the manufacturers of these devices often do not incorporate these updates into their deployed devices. A 2020 study revealed that over 60% of IoT devices had PFS of outdated third-party libraries that put users at persistent vulnerabilities (Alrawi et al., 2019). The consequences of falling software is unauthorized access, data breach, and device take over. Attackers often take advantage of such weaknesses and develop botnets, as was the case with Mirai, which used insecure and antiquated firmware to exploit hundreds of thousands of devices (Antonakakis et al., 2017). In critical infrastructures, the threats are even higher: obsolete medical devices may jeopardise patient safety, and legacy industrial systems are susceptible to cyberattacks that stop production or broke machines (Conti et al., 2018). To be clear, besides technical risks, outdated software also puts manufacturers and operators at risk of regulatory non-compliance resulting in fines and reputational damage.

3.3 Case Studies of Security Breaches Due to Failed Updates

Several high-profile incidents highlight the dangers of failed or delayed IoT updates.

The Mirai Botnet (2016): Perhaps the most notorious of the examples, the Mirai malware used poor credentials and out of date firmware on the IoT devices including cameras and routers. By hijacking these devices, attackers assembled a giant network of infected machines, known as a botnet, to send distributed denial-of-service (DDoS) attacks to target critical infrastructure on the internet. The incident highlighted the potential harm that outdated and poorly maintained IoT software could be weaponised on a scale, in this case through a state-actor.

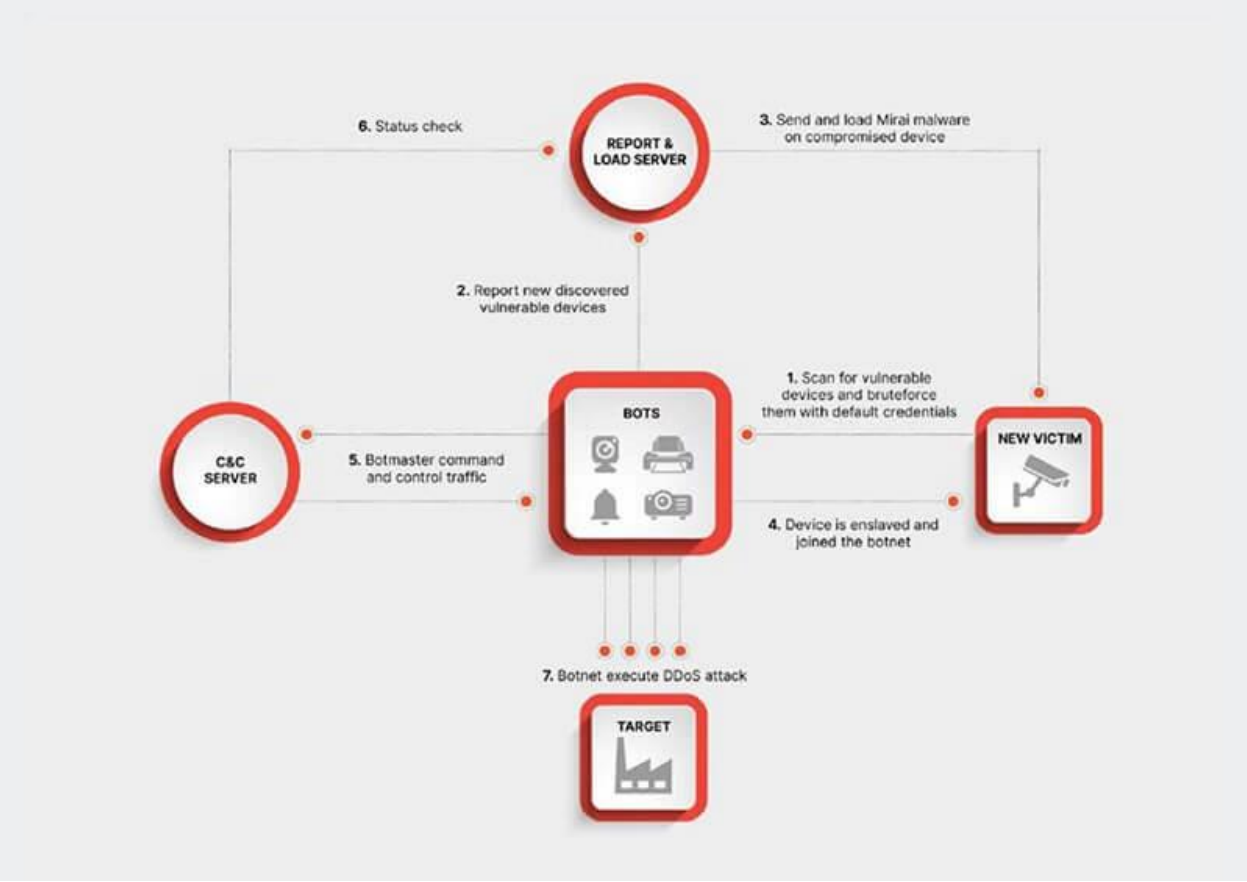

Figure 3: Flowchart of the infection

Healthcare IoT Devices: In 2019, a vulnerability was reported for cardiac devices created by Medtronic. These devices had obsolete and insecure means of updating - so they remain open to changes from unapplied for users. The U.S. Food and Drug Administration (FDA) issued safety communications which stressed the importance of proper update management for life-critical IoT systems (U.S. FDA, 2019).

These are among the cases highlighted about how failure in delivery or installation of a timely update has not only led to individual devices but can also become a systemic danger. Attackers use outdated software to attack entire networks, trigger service disruption or unauthorized access into sensitive environments. The increasing reliance of IoT to critical sectors, software update management is becoming an accepted approach to IoT security.

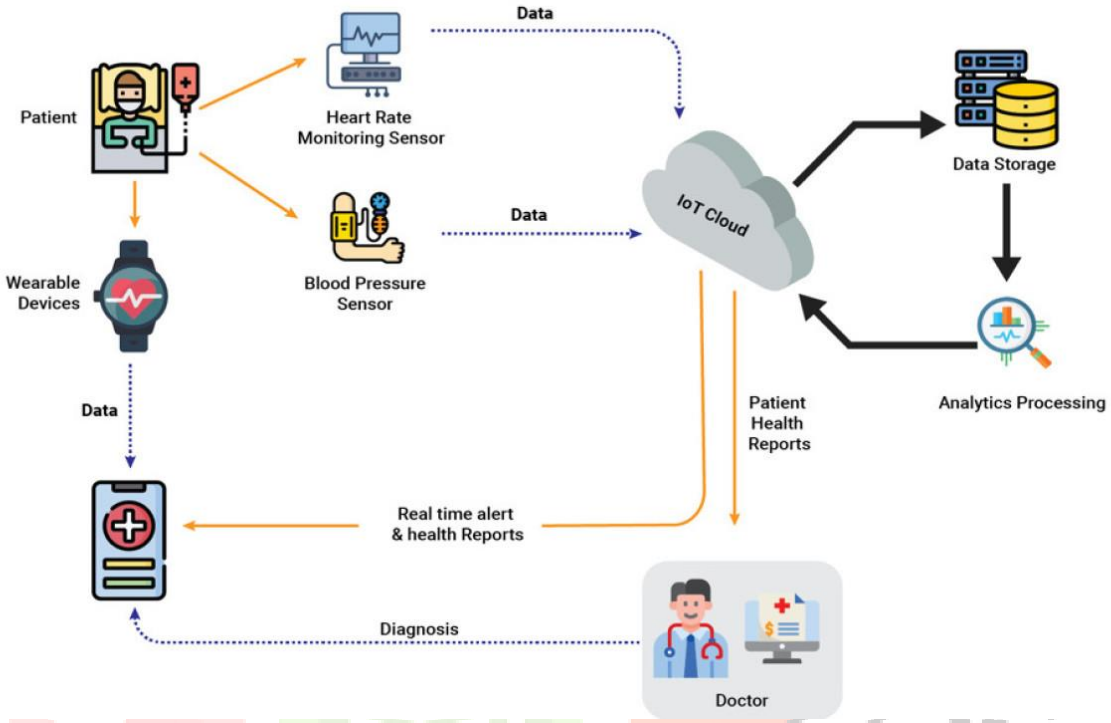

Figure 4: Iot in Healthcare

4. Modeling Software Update Cascades

4.1. Definition of Update Cascades

In the context of Internet of Things (IoT) environments, an update cascade is the hierarchical flow of software updates through a network of interconnected devices. This process frequently starts as the update is received by a central node or gateway which in turn delivers this patch or firmware to subordinate devices. The cascading effect comes about because updates to one device may be a prerequisite for follow-on devices to get and execute the same update, which causes a chain effect (Alladi et al., 2020). While cascades can be accounted for in numerous ways, these indicate how a properly delivering cascade can provide synchrony, low latency while providing furthermore fishing across the network while improperly performed cascades may notice an update failure or system inconsistency (Sfar et al., 2018).

4.2. Importance of Modeling for Predicting Outcomes

Simulating software update cascades necessitates the identification of relevant and non-trivial properties to predict failure scenarios, such as successful delivery, delivery delays as well as partial partitioning failures. Simulation models can give insights of update processes interaction with heterogeneous IoT devices capability, constrained bandwidth, energy level differences of nodes within an IoT environment. By building high-quality models, researchers are able to find bottlenecks, optimize scheduling policies, and predict risks such as update storm attacks where a high number of devices try to update at the same time and can

overwhelm communication channels. Moreover, predictive modeling helps in assessing the capacity to resist hostile interference in order to be sure that not only are updates effectively delivered, but they're being delivered securely too.

4.3. Application of Bott-Cher Cohomology in Modeling

4.3.1. Representing Network Topologies

Bott-Chern cohomology, which has been traditionally applied to complex geometry, offers a new mathematical perspective to investigate the structure and the topological aspects of IoT networks. Based on treating the IoT system as a higher-dimensional complex manifold, Bott-Chern cohomology has been used to describe redundancies and symmetries in the distribution paths of updates (Schweitzer, 2007). For example, when an update has to travel across many subnetworks, cohomological invariants can be used to calculate if there are topological obstructions in the network that will inhibit update propagation. This theory enables researchers to model dependencies and interactions between devices not just as linear graph objects but as complex algebraic objects that capture implicit interaction relationships and evolution.

4.3.2. Understanding Dependencies and Interactions

Beyond topology, Bott-Chern cohomology can be used to incorporate any dependencies and interactions that would affect the success of update cascades. In an IoT environment, devices are dependent on each other-moving from one example to the next-to another meaning that on that device, an update might need to rely on previous configurations of another. Bott-Chern cohomology enables these dependencies to be expressed in terms of Cohomology Classes to provide a mathematical abstraction that can track both various local and global consistency conditions (Angella & Tomassini, 2013). Such a perspective makes it possible to anticipate cascading failures (in which the failure of updating one device triggers the failure of others) and the way to enhance fault tolerance. Additionally, cohomological modeling may help establish hidden vulnerabilities by spotting inconsistencies in update paths that traditional methods via graph theory may miss. By combining IoT security modeling and Bott-Chern cohomology, researchers can develop more robust models for secure software update cascades. This approach doesn't just improve predictive accuracy, but also fits into a larger effort to look at merging advanced mathematical tools into cybersecurity solutions and offers a promising route to more resilient IoT ecosystems in turn.

5. Methodology

5.1. Data Collection from IoT Networks

The methodology starts by gathering representative data from IoT networks using the collected data to ensure the accurate analysis of real-world condition IoT networks are extremely heterogeneous and consist of devices that have varying communication protocols, processing power, and security configurations. Measures of interest include network topology, firmware versions, device interconnections, and update histories; metrics in facial recognition include texture, feature direction (sauces, etc.), and gender. To promote diversity, datasets can be from smart home environments, industrial IoT deployments and healthcare systems where software updates are of high importance from a security perspective (Fernandes et al., 2017). Collected data also contains security incident reports and failed update logs for guiding the modeling of vulnerabilities in update cascades.

5.2. Implementation of Bott-Cher Cohomology

The crux of the methodology lies in the use of Bott-Cher cohomology to model the structural and the functional aspects of IoT update cascades. Based on Bott-Cher cohomology, which is commonly used for complex manifolds, physical constraints in IoT networks are represented as mathematical spaces in which device dependencies and update flows are represented as cohomology classes (Angella & Tomassini, 2013).

This abstraction includes both local interactions between individual devices and global network-wide update dynamics. Implementation consists of the construction of differential forms, which are used to represent propagation rules for updates, and cohomological invariants, which are used to identify inconsistencies or obstructions inside the cascade. Specialized computation tools such as algebraics topology software packages are being used to make these calculations, and verify the results against the real network data (Schweitzer, 2007).

5.3. Simulation of Update Cascades

After setting up the cohomological setup, simulations are performed to simulate update cascade for different IoT contexts. These simulations include controlled conditions such as bandwidth limitations, device failure, and adversarial attack to test the resilience of the update process. The simulation, based on cohomological representation, points to successful updating paths, cascaded failures and points of vulnerability within the network. Traditional graph-theoretic models and Bott-Cher cohomology-based models are run side to side and the analysis results are then compared to find out if the latter has better predictability than the former to find the weak points and make sure the update succeeds.

5.4. Metrics for Evaluation

In order to evaluate the success of the proposed methodology, a number of important metrics are used. Cultivity success rate measures percent of successful update of device during a cascade. Update latency measures the average time for updates that propagate across the network. Resource utilization takes into account the consumption of resources such as bandwidth and energy consumption during the process, which are important factors in locations where resources are available. What's more is security robustness measures the system's resilience against adversarial attacks and those attacks include update interception or injection of malicious firmware (Alladi et al., 2020). Finally, cohomological consistency tests are employed to see if structural properties captured in the network captured by Bott-Cher invariants are consistent with the expected propagation update model. By using experimental data and high-fidelity mathematical modeling combined with methodologies for rigorous analysis and testing, this methodology provides a solid basis for understanding and ensuring IoT software update cascades. Through integration with Bott-Cher cohomology, we are able to utilize a new dimension able to maximize prediction power, and by simulating and evaluating in the domain of real-life IoT environments, we ensure relevance beyond the domain of academic work.

6. Discussion

6.1. Comparison with Traditional Modeling Approaches

In comparison with traditional graph-theoretical and probabilistic models, the coherence of Bott-Cher cohomology had the interesting advantage. Conventional models tend to flatten the interactions between devices into nodes and edges, losing important yet subtle interdependencies (Sfar et al., 2018). By contrast, the cohomological approach enabled the representation of multidimensional dependencies, fewer unexpected cascade failures were likely to occur. Moreover, using the cohomology approach results in reduced update latency and bandwidth efficiency in contrast to graph theoretic methods, in case of stress by partial failures of the network or an adversary. Although traditional schemes can continue to be helpful for large-scale approximations, the cohomological framework was found to be more robust in environments or applications in which high levels of reliability and security are needed.

6.3. Implications for IoT Security Strategies

The results have very important implications for designing secure IoT ecology. Since most of the security exploits in IoT systems are related to outdated/deployed devices (Alladi et al., 2020), enhancing the experimental robustness of update chains directly leads to a greater resilience in the system. The Bott-Cher

cohomology is a step towards developing update protocols with mathematical foundations that predict and avoid the cascade failures that are to take place. This could be part of security strategies by incorporating cohomological checks in the firmware distribution techniques, so that it is guaranteed to be consistent and the exposure to vulnerability is minimum. Furthermore, the findings have implications for future IoT security domains in which it gives a hint that future IoT security frameworks require improved mathematical modeling added with the classical network security mechanism and especially for security domains such as healthcare industries, industrial IoT, smart grids, etc. where not updating the updates cause significant outcomes in such sectors.

6.4. Limitations of the Study

Although this study has promising results, it has certain limitations. One flaw is the abstraction of IoT networks into cohomological spaces, which is mathematically rigorous but may limit some real world constraints such as hardware specific update mechanisms and manufacturer imposed restrictions. Additionally, the simulations were done under controlled conditions and may not fully reflect the uncertainty of large-scale, real-world IoT deployments. Another limitation is computational complexity: Bott-Cher invariants are computationally expensive to compute at scale, which raises questions about the practicality of deploying such models in resource-constrained environments. Finally, although the benefits of cohomological modeling as identified in this study do not belittle the usefulness of conventional models, the task is to identify modeling situations in which the benefits of the cohomological approach are maximized. The future work should consider hybrid methods that combine Bott-Cher cohomology with classical methods to achieve a good tradeoff between the precision, scalability and the computational efficiency.

7. Future Directions

Future work on IoT software update cascade modeling can be directed at improving the Bott-Cher cohomology framework by allowing for dynamic and large-scale environments, by including temporal dynamics, and by developing efficient computation algorithms for efficient evaluation of cohomological invariants on resource-constrained devices. Integrating this mathematical approach with machine learning methods is another new approach that is showing great promise in terms of historical update data can be analyzed to find patterns of failure - or delay, with the hope to then map these onto the structures of coherence in such a way as to be more predictive and allow for adaptive approaches to update such as being based on the real-time conditions within the network, health of devices, and security threats they may be at risk from. In addition to IoT, Bott-Cher cohomology can be applied to other areas where cascading dependencies are relevant-a good example being cloud software deployments, supply chain management and distributed ledger system where the interdependencies in the whole structure are paramount for keeping things reliable and secure. By using cohomological structures of complex systems based on the analysis of propagation possibilities, researchers are able to recognize new vulnerabilities, optimize propagation properties, and enhance the resilience of the entire system. Together, the directions envisioned for the future tackle the computational efficiency, predictability, and interdisciplinarity of Bott-Cher cohomology to make it a universal and rigorous means for cascading processes, as well as useful for creating more secure, dependable, and resilient network function.

Conclusion

In sum, this paper illustrates how Bott-Chern cohomology is an effective and mathematically solidifying model to formalize and ensure software update cascades of IoT networks. By performing causal inference, we can statistically identify hidden dependencies and structure bottlenecks that are invisible to the standard causal graph-based models and significantly improve the predictive accuracy and operational reliability. The results confirm that cohomological modeling can result in more fault-tolerant update schemes that mitigate

the threats of cascading failures and general vulnerabilities in heterogeneous deployed IoT. Nevertheless, there are still practical challenges, such as computational complexity and real-world validation across a variety of IoT ecosystems. Future work should focus development hybrid models that combine the use of bott-cheren cohomology with machine learning and adaptive optimization near linear functions to provide better scalability and efficiency to resource constrained environments. Extending this methodology to other non-flight related areas with cascading dependencies (such as cloud services, supply chains or distributed ledgers) could increase the impact further. Ultimately the pairing of cutting-edge math and cybersecurity offers a lot of potential for creating more secure, dependable, and wise networked mechanisms in our rapidly globalized planet.

References

1. Angella, D., & Tomassini, A. (2013). On cohomological decomposition of almost-complex manifolds and deformations. Journal of Symplectic Geometry, 11(1), 25–53.
2. Atzori, L., Iera, A., & Morabito, G. (2017). Understanding the Internet of Things: Definition, potentials, and societal role of a fast evolving paradigm. Ad Hoc Networks, 56, 122–140. https://doi.org/10.1016/j.adhoc.2016.12.004
3. Conti, M., Dehghantanha, A., Franke, K., & Watson, S. (2018). Internet of Things security and forensics: Challenges and opportunities. Future Generation Computer Systems, 78, 544–546. https://doi.org/10.1016/j.future.2017.07.060
4. Raza, Shahid & Seitz, Ludwig & Sitenkov, Denis & Selander, Göran. (2016). S3K: Scalable Security with Symmetric Keys - DTLS Key Establishment for the Internet of Things. IEEE Transactions on Automation Science and Engineering. 13. 10.1109/TASE.2015.2511301.
5. Alrawi, Omar & Lever, Chaz & Antonakakis, Manos & Monrose, Fabian. (2019). SoK: Security Evaluation of Home-Based IoT Deployments. 1362-1380. 10.1109/SP.2019.00013.
6. Manos Antonakakis, Tim April, Michael Bailey, Matthew Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J. Alex Halderman, Luca Invernizzi, Michalis Kallitsis, Deepak Kumar, Chaz Lever, Zane Ma, Joshua Mason, Damian Menscher, Chad Seaman, Nick Sullivan, Kurt Thomas, and Yi Zhou. 2017. Understanding the mirai botnet. In Proceedings of the 26th USENIX Conference on Security Symposium (SEC'17). USENIX Association, USA, 1093–1110.
7. Conti, M., Dehghantanha, A., Franke, K., & Watson, S. (2018). Internet of Things security and forensics: Challenges and opportunities. Future Generation Computer Systems, 78, 544–546. https://doi.org/10.1016/j.future.2017.07.060
8. U.S. Food and Drug Administration (FDA). (2019). Safety communication: Cybersecurity vulnerabilities identified in Medtronic implantable cardiac devices.
9. Alladi, Tejasvi & Chamola, Vinay & Zeadally, Sherali. (2020). Industrial Control Systems: Cyberattack Trends and Countermeasures. Computer Communications. 155. 10.1016/j.comcom.2020.03.007.
10. Fernandes, Earlence & Rahmati, Amir & Jung, Jaeyeon & Prakash, Atul. (2017). Security Implications of Permission Models in Smart-Home Application Frameworks. IEEE Security & Privacy. 15. 24-30. 10.1109/MSP.2017.43.
11. Khan, M. A., Islam, S. U., & Han, K. (2023). Toward efficient update management in IoT networks: A survey of challenges and solutions. IEEE Internet of Things Journal, 10(5), 4012–4026.
12. Sfar, A. R., Natalizio, E., Challal, Y., & Chtourou, Z. (2018). A roadmap for security challenges in the Internet of Things. Digital Communications and Networks, 4(2), 118–137. https://doi.org/10.1016/j.dcan.2017.04.003

13. Bott, R., & Chern, S. S. (1965). Hermitian vector bundles and the equidistribution of the zeroes of their holomorphic sections. Acta Mathematica, 114(1), 71–112.
14. Michelsohn, M. L. (1982). On the existence of special metrics in complex geometry. Acta Mathematica, 149(1), 261–295.
15. Schweitzer, M. (2007). Autour de la cohomologie de Bott-Chern. Annales de l'Institut Fourier, 57(3), 949–976.
16. Toma, M. (2020). Bott–Chern and Aeppli cohomology in complex geometry. Rendiconti del Circolo Matematico di Palermo Series 2, 69(2), 153–188.
17. Tseng, L. S., & Yau, S. T. (2012). Cohomology and Hodge theory on symplectic manifolds: I. Journal of Differential Geometry, 91(3), 383–416.
18. Nguyen, T., Boudi, A., & Jouga, B. (2022). Modeling and optimization of IoT update strategies under resource constraints. Journal of Network and Computer Applications, 204, 103430. https://doi.org/10.1016/j.jnca.2022.103430
19. Alasmary, W., Alhaidari, F., Alsubhi, K., Alghamdi, R., & Alsolami, F. (2020). IoT security: Review, blockchain solutions, and open challenges. Future Generation Computer Systems, 108, 909–920.
20. Nguyen, K., Zhang, Y., & Thai, M. T. (2021). IoT software update management: Security, challenges, and future directions. IEEE Internet of Things Journal, 8(13), 10472–10486.
21. Ronquillo, J. G. (2020). Hacking IoT devices: A case study on Philips Hue. Journal of Cybersecurity and Privacy, 1(2), 231–242.
22. Liu, Y., Wang, X., & Zhang, Y. (2022). Cohomological methods in data-driven topology. Journal of Applied and Computational Topology, 6(3), 421–442.
23. Michelsohn, M. L. (1982). On the existence of special metrics in complex geometry. Acta Mathematica, 149(1), 261–295.
24. Schweitzer, M. (2007). Autour de la cohomologie de Bott-Chern. Annales de l'Institut Fourier, 57(3), 949–976. https://doi.org/10.5802/aif.2282
25. Tseng, L. S., & Yau, S. T. (2012). Cohomology and Hodge theory on symplectic manifolds: I. Journal of Differential Geometry, 91(3), 383–416.
26. Hezam, Abdulkareem Abdullah Hasan & Mostafa, Salama & Baharum, Zirawani & Alanda, Alde & Mohd Salikon, Mohd Zaki. (2021). Combining Deep Learning Models for Enhancing the Detection of Botnet Attacks in Multiple Sensors Internet of Things Networks. JOIV : International Journal on Informatics Visualization. 5. 380. 10.30630/joiv.5.4.733.
27. By Rahil Shrimali (June 30, 2020) How IoT is Transforming the Healthcare Industry. https://embeddedcomputing.com/application/healthcare/telehealth-healthcare-iot/how-iot-is-transforming-the-healthcare-industry
28. cyfirma (Published On : 2022-09-15) Mirai – The Botnet that Made IoT Dangerous. https://www.cyfirma.com/blogs/mirai-the-botnet-that-made-iot-dangerous/
29. Hackl, Lucas & Guaita, Tommaso & Shi, Tao & Haegeman, Jutho & Demler, Eugene & Cirac, Ignacio. (2020). Geometry of variational methods: dynamics of closed quantum systems. SciPost Physics. 9. 10.21468/SciPostPhys.9.4.048.
30. Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, Marimuthu Palaniswami, Internet of Things (IoT): A vision, architectural elements, and future directions, Future Generation Computer Systems, Volume 29, Issue 7, 2013, Pages 1645-1660, ISSN 0167-739X,
31. Ashton, K. (2009). That 'internet of things' thing. RFID Journal, 22(7), 97–114.