David Hasselquist
Linköping University 2022-11-09

WASP Software Engineering: Assignment 2

1. Research topic

The use of a third-party cloud service results in a non-trusted point where many connections can be observed or manipulated by an attacker. This opens up attack vectors and reduces the trust placed on the infrastructure. In our research, we design and evaluate optimized solutions for secure communication and processing that reduce attack vectors and minimize the risks of outsourcing communication infrastructure to non-trusted cloud providers.

An important aspect is to push the performance tradeoff boundaries as far as possible within the context of communication use cases requiring different confidentiality, availability, and integrity levels. Therefore, we evaluate solutions along several dimensions and consider the perspectives of both attackers and service providers. In short, our research consists of:

- **Performance evaluation and system optimizations considering non-trusted autonomous clouds.** Here, we evaluate the security, reliability, and performance tradeoffs of current and future state-of-the-art solutions when placing some processing in one or more non-trusted autonomous clouds.

- **Future-proofing of state-of-the-art protocols and attacker models.** Here, we develop and compare retrofitted protocols and communication solutions for post-quantum cryptography in the context of autonomous clouds.

- **Automated and data-driven side-channel attacks.** Here, we use deep packet inspection and traffic analysis to extract sensitive information from encrypted network traces.

2.1 Security and Privacy

Most of the internet traffic is today encrypted with HTTPS over TCP/TLS or QUIC. While this increases users' privacy, it also presents challenges for network providers and others seeking to improve network performance and security.

Today, network operators deploy network intrusion detection systems (NIDS) to increase network security and monitor traffic for malicious activity or policy violations. With the increased use of end-to-end encryption on the internet, packets are not as easy to inspect, making encrypted network monitoring a non-trivial task. Furthermore, network operators can perform traffic analysis for load balancing, traffic prioritization, or traffic splitting based on the traffic type and contents. Similarly, operators still face the challenges of packet inspection as packets are encrypted.

Flow analysis, inspection of packet headers, and deep packet inspection (DPI) are today used to analyze encrypted network traffic. However, this requires heavy computation, especially when performed in real-time on high-speed networks with large volumes, velocity, and variety of traffic. Here, software engineering and design may play a significant role. To be able to process encrypted real-time network traffic, the processing speed is of great importance.

Furthermore, vast data volumes open up for AI/ML applications. Encrypted traffic analysis, especially in combination with AI/ML, is a well-researched area with many research challenges, and is also a domain with many commercial opportunities.

2.2 Continuous Deployment (Maintenance and Evolution)

Google has recently introduced QUIC as a next-generation transport-layer solution that addresses some shortcomings of the TCP protocol. It is a multi-layer transport protocol implemented in user-space, providing reliability on top of the UDP protocol. Some advantages of QUIC include encryption, rapid deployment, 0-RTT, and per-stream flow control. QUIC is globally deployed and used by many popular services, including Google search, Chrome, YouTube, Facebook, and Microsoft. The protocol has seen widespread adoption and is estimated to carry more than 10% of internet traffic today.

When designing software for an internet scale, many lessons can be learned from QUIC. One of the success factors of QUIC lies in one of its design goals: continuous and rapid deployment. By continuously deploying new versions, Google could quickly deploy security fixes, an essential aspect as QUIC provides end-to-end encryption and secure transport. Fast and continuous deployment also allowed them to experiment with various aspects and parameters, learn from experience, use the internet as a testbed, and adjust as needed.

Google was able to fully depreciate a previous version and deploy a new version in only a couple of months. Compared to TCP, such version changes could take years or even decades. This shows that when developing software applications at an internet scale, continuous and rapid deployment is of great importance. Continuous deployment on the internet scale is an important aspect with many research challenges. Today, many improvements on the internet never see practical use due to complex systems and interactions. It is, therefore, important to design future software systems with continuous deployment in mind.

2.3 Regulations and Compliance

Doing research on internet communication involves collecting network traffic that could contain sensitive user data. Even though much of the data is encrypted, raw network data and other metadata can still be highly sensitive. This makes it important to evaluate software performing data collection from a regulation and compliance point of view. To comply with various laws, one must start with the software and its design.

Data collection software must be designed to carefully handle sensitive data with minimization and anonymization. By anonymizing privacy-sensitive fields from network traces, a software system can function and follow various regulations and compliances. For example, an IP address 123.123.123.123 can be anonymized to 123.123.123.*, where the last octet is replaced with a wildcard. If more privacy is needed, the two last octets can be removed. This would ensure that a user cannot be directly identified while keeping key information that parts of an IP address reveal (e.g., geographical location).

Similar anonymization and minimization can also be made for MAC-addresses, providing user anonymity while preserving key information (e.g., hardware manufacturer). The data collection should be anonymized to the extent that it complies with the local regulations and the other data regulations (e.g., GDPR and CCPA).

Furthermore, the raw but anonymized and minimized data should be stored securely using a combination of physical and network security controls, restricting access to others. For example, data can be stored on campus servers following local regulations and compliance.

3. Future trends and directions of Software Engineering

In the future, I believe that much ML training will occur online in the cloud and happen in real-time in many software systems. One of the reasons is the emerging trend of cloud computing, where much computational power can be offloaded from constrained and smaller devices to the “unlimited” cloud. This will result in faster ML training and other processing than if performed locally on a small device with limited power.

Furthermore, with the emerging trend of edge and fog computing, where data centers are placed closer to the users, the latency of real-time data processing is significantly dropping. The need for distributed low-latency applications has emerged, and software engineering needs to adapt to this trend. However, not all software can easily be taken to the cloud and expect performance gains. Software engineering must adapt to these trends, and software needs to be built with scalability and elasticity in mind to be able to take full advantage of modern clouds.

In software development, cloud-native computing is a concept of building and running software applications that can take advantage of the distributed computing offered by many modern cloud models. These software applications are built with scale, elasticity, and flexibility in mind, requesting appropriate resources after load and demand. As one part of my research is to design and evaluate the performance of applications run in the cloud, it may be beneficial to design software in the context of cloud-native applications.

Furthermore, I think energy efficiency software will become more important in the future. Software accounts for a large share of today’s energy consumption, which is expected to increase even further. In the software engineering context, it is important to evaluate energy efficiency, as this might have a significant impact when used at scale. Also, with the increased usage of constrained IoT devices powered with batteries, software needs to be designed with energy consumption in mind. Finally, I think software security will become a more hot topic in the future. With the rising number of cyber attacks, the design of secure software is crucial.