Review of *”A Case for Remote Attestation in Programmable Dataplanes”*

Will Thomas

This paper [**prog˙data**] primarily focuses on the security of programmable dataplanes, and how we can improve the security of programmable dataplanes through aemote attestation. I specifically chose this paper because I work in the KU SLDG lab under Professor Perry Alexander. In SLDG, we work on designing and formalizing the semantics of Copland (a domain specific language and protocol used for remote attestation), as well as a Copland Attestation Manager (an environment which will manage attestation protocols). This paper specifically refers to Copland, and cites Prof. Alexander et. als. work in this area, which I thought was a rather neat extension of our classes studies into real research.

First, we must unpack what a ”programmable dataplane” is. A programmable dataplane best follows the definition of Software Defined Networking (SDN) that we gave in class. The SDN is implemented remotely, giving many benefits to view the system overall from a centralized point of view. The reason it refers to a programmable **dataplane** rather than control plane is that although the control plane is where the SDN is implemented, the features that it controls are dataplane specific. Some of the benefits this paper proposed as having come from programmable dataplanes are the ability to optimize resource use, and patch equipment from a centralized location. However, this paper also brings up the issues that can arise from using a programmable dataplane. The primary issue is that programmable dataplanes can act as a single point of failure for the security of a system. If the integrity of the programmable dataplane (PD) is compromised, then all the dataplanes that it controls (a large collection of routers) will be compromised as well. This is specifically brought up in the case of what is known as the ”Athens Affair”, which was a cyberattack targeting high-ranking officials in the Greek government. The attack took place by compromising a programmable dataplane, which in turn allowed for eavesdropping on Greek government officials. This is where the techniques of Remote Attestation play a role in mitigating the risk that a PD is compromised.

Remote Attestation is a set of techniques that help provide evidence to the validity of a system and show that it is not compromised. This is done by measuring certain components of a system from a core ”Root of Trust” that is assumed to be safe. For example, the TPM (Trusted Platform Module) of a system may be a ”Root of Trust” that we can assume will not be compromised without a sufficiently powerful adversary. If the adversary is powerful enough to corrupt the ”Root of Trust” then all gathered evidence could not be trusted as well. Copland is a domain specific language and protocol used to carry out Remote Attestation. [**cop˙tutor**] Copland allows for the ”Copland phrase” writer to write an arbitrarily strict attestation over the system it wishes to attestion. This will then be managed by the Copland Attestation Manager (which has been built and formalized in Coq within the KU SLDG lab), which will ensure that the necessary measurements take place and the evidence is then passed along in accordance to the Copland phrase. A benefit of Copland and remote attestation in general that is highlighted within the paper is ”RA can be used to enable dynamic assessments of network security characteristics through automated generation, collection, and evaluation of rigorous evidence of trustworthiness” [**prog˙data**]. If we are able to correctly implement Remote Attestation, we can at the very least constrain the actions our adversary can take to compromise out programmable dataplanes [**confining**]. The specific methodologies to maximize the constraints on our adversary are a major research topic in remote attestation, but it is generally agreed that layered attestations starting from the ”Root of Trust” to the highest (and most insecure) levels of a system provides the largest chain of evidence as to the security of a system [**layered˙attest**].

Now that we have a better understanding of Remote Attestation in general, we can look at some of the highlighted use cases from this paper. The use cases specifically cover how to secure a programmable dataplane through remote attestation.

The first proposed use case is to help ensure that the configuration of a programmable dataplane has not been compromised. In the ”Athens Affair”, the programmable dataplane was modified to run a separate program than was intended, which was ultimately due to a misconfiguration that was allowed to take place. Remote attestation would be able to avoid accepting network traffic when a packet that is sent does not have a sufficient evidence chain that the configurations of switches it has passed through. If for example the packet takes a different path, or the configuration does not meet a certain ”golden value” of safe switches, then the packet will be assumed compromised.

The second use case focuses on uses evidence gathered along a path as a security feature. Each switch and router along a network path will have a specific key that it can sign the evidence with as it is passed along the route. This will then be evaluated by the end-user (either host or client) and compared to whatever value it should be. Only if the evidence packet has been passed along a specifically required path will the output be decrypted properly.

The third use cases uses the path evidence as a tag. This is specifically used to help a programmable dataplane make ”authorization decisions that affect on intra- or inter-domain handling of traffic” [**prog˙data**]. The evidence that can be generated along the path of a intra-domain route will look different than the evidence from an inter-domain route. A network administrator could foreseeably want to treat these packets differently, and would know with a high-degree of certainty that the evidence (generated by Remote Attestation) points to a packet either being intra- or inter-domain.

The fourth use case is to use evidence as a form of documentation. This brought up a very interesting case of the ability of a government entity to actually compromise a civilian or business computer in order to get rid of malware. It is becoming a more common situation in which a government entity (upon receiving evidence of malware) will intervene and manually remove the malware [**doj˙hack**]. This is allowed via a court order. The use case for remote attestation referred to in this paper would be to take evidence that a system is in fact compromised to allow a government entity to remove it. Additionally, they suggest that a chain of evidence could be established showing that a system that has government intervention will only be modified in the ways prescribed by the court order. Thus evidence is used both to justify the court order, and to ensure the court order is fulfilled exactly as promised.

The final use case mentioned in this paper is cross-referenced attestation. This use cases seems quite similar to the third use case in that it is primarily used for making decisions regarding intra- vs. inter-domain routing. It goes a bit more in-depth as to explain that intra-domain requests (even though they are likely more implicitly trustworthy) must also provide evidence as to the state of their implementations. If the implementation of a system is not verified, it will be blocked and not allowed either into the domain, or to move within the domain.

The paper [**prog˙data**] proceeds to do in-depth explanations of Remote Attestation and Copland, which have already been addressed in this review. It then attempts to extend the Copland language via three new operators. These operators are motivated as helping make Copland more general in cases when the network topology may not be known. The operators are inherited from NetKAT [**net˙kat**], which is a formal system for network verification. The operators are $\forall$, $\blacktriangleright$, and $\overset{*}{\Rightarrow}$. The $\forall$ operator provides an abstraction for Places, which are the Copland equivalent of targets. Rather than explicitly naming the Places, this allows for a more general form of it to be used. Although this operator does seem beneficial from a high-level, it is actually unnecessary based on the Copland semantics. The places (as they are written in a Copland phrase) are considered maximally abstract. It is only after a process known as Negotiation (part of the Copland protocol) that the places are no longer abstract and represent a concrete place within a collection of systems. The extension of Copland with $\forall$ would certainly help make it more explicit that places are abstract, but is not necessary when a strict interpretation of the semantics is used. $\blacktriangleright$ and $\overset{*}{\Rightarrow}$ has interesting use cases as extensions to Copland. It is foreseeable that one may wish to abstract the path that they take within a remote attestation, as it certainly allows for greater generality. While the benefits seem quite obvious, the possible downsides are more nuanced and come into the fundamentals of remote attestation itself. Copland as a protocol and language has been defined to give the user absolute control of their data and where it goes. This is to ensure that some of the fundamental tenets of remote attestation are not violated [**principles**]. The addition of $\blacktriangleright$ and $\overset{*}{\Rightarrow}$ would not in themselves necessarily compromise the principles of remote attestation, but may cause someone to unintentionally violate them by using an operator so powerful. It is certainly worth further investigation into the extension of the language this way, especially since this seems to be the way that people are using it in real-life use cases.

Overall, this paper provided a great chance to learn more about programmable dataplanes, or as they could also be referred to Software Defined Networking (SDN). It also showed the real-life benefits of active research towards Copland and Remote Attestation going on at the University of Kansas. It proposed extensions to Copland, which seem to have merit and should definitely be reflected upon for future iterations of the language, especially to allow usability in more general purposes.