Research statement

Laurent Vanbever, ETH Zurich

November 2018

I conduct research in the area of computer networks, with a focus on **network management**. I design and build systems that allow people to safely operate large networks. A unifying theme in my work is to develop interfaces to monitor and actuate networks so as to achieve greater performance, robustness, and capabilities. To ensure deployability, I often develop these interfaces by revisiting existing mechanisms and sometimes, even abusing them. My research is inherently multi-disciplinary blending theoretical and algorithmic contributions with, among others, advances in program analysis and synthesis.

My most impactful research project since I joined ETH enables **declarative network programmability** within and between existing network infrastructures by either crafting routing announcements or by synthesizing routing configurations. This project and the associated publications have been awarded with multiple prizes by the research community and by network practitioners, including the SIGCOMM best paper award. Our recent works on **routing attacks** and their impact on anonymity networks and cryptocurrencies have also drawn considerable attention from both communities.

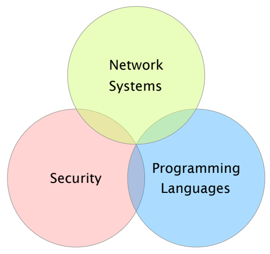


Figure 1: My main research area (network management) is inherently multi-disciplinary, leading to publications in multiple research communities.

Computer networking is a relatively young discipline of computer sciences and computer engineering which is concerned with the exchange of data between computer systems through a shared network. Despite its youth, networking has been and continues to be an extremely succesful and exciting field that brought us many of the technologies we use on a daily basis including the World Wide Web, emails, or video streaming. But the main achievement of the networking community is without any doubt the Internet itself, a "research experiment that escaped from the lab" and which went from interconnecting a handful of nodes in the early 70s to more than 3 billion today. This staggering growth is far from being over: over 1 billion extra Internet devices should be be connected by 2021.

Networking is not "all roses" though; there are many challenges lying ahead. In my humble opinion, one of the main challenges faced by our field lies in being able to effectively manage ever-growing network infrastructures carrying ever-growing traffic volumes. I see three main problems: *(i)* the lack of appropriate tools to monitor network infrastructures; *(ii)* the arcane low-level interfaces used by operators to manually specify network-wide behavior; and *(iii)* the use of suboptimal (and slow) control algorithms. In essence, all the basic building blocks of a well-managed system—the sensors, actuators and efficient control loops—are either missing or perfectible. Unsurprisingly, these problems lead to poorly performing and fragile network infrastructures where most of the traffic is routed suboptimally and where most of the downtimes are caused by humans, not equipment failures. Unfortunately, changing how network infrastructures work is also extremely difficult (especially in the Internet), a phenomenon commonly known as "Internet ossification".

Finding practical and creative ways to tackle these long-standing problems has been my "quest" for the last couple of years. The outcome of this quest is a network controller platform that can enforce high-level objectives in both existing and next-generation networks.

More specifically, my group and I have focused on three **main research themes**: *(i)* developing fast-yet-scalable **network measurement** techniques; *(ii)* developing fine-grained **control interfaces** that can adapt the network behavior in a declarative manner; and *(iii)* developing **control algorithms** that can drive the network in real-time in addition to analyzing its behavior. Figure 2 illustrates how a subset of my recent works fit in this context.

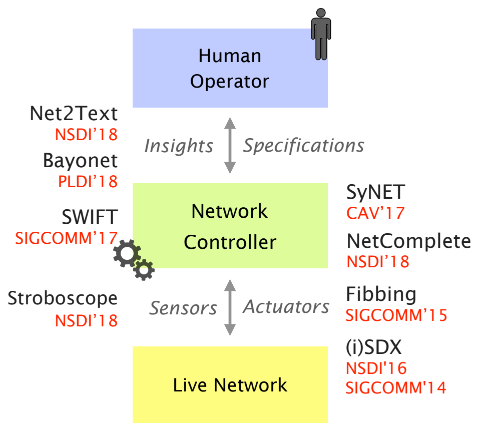


Figure 2: A subset of our recent works put in context.

In parallel, my group and I have been investigating how we can **secure network systems** by: *(i)* analyzing the effects of important attack vectors such as **routing attacks**; and *(ii)* developing techniques to let networks detect and defend themselves from these attacks.

In terms of approach, nearly all of my works share common key ingredients such as network models, algorithms design, formal reasoning (including correctness proofs), and system implementations.

I intend to continue working on these themes in the future, placing greater emphasis on efficient data collection, fast-yet-stable control algorithms, and fine-grained actuation. To do so, I intend to leverage the new capabilities offered by next-generation network devices such as reprogrammable network ASICs and (Net)FPGAs.

Academic impact The results of most of my projects have been published at top networking conferences, namely: ACM SIGCOMM (2 papers since I started at ETH, 4 in total), USENIX NSDI (4 papers since I started at ETH) and ACM HotNets (4 papers since I started at ETH, 6 in total). Similarly to most computer science disciplines, conference proceedings are the best places to publish my work, both in terms of prestige and overall visibility.

In addition, multiple of our papers have been awarded with various prizes. Among others, our work on Fibbing[1] has won the **ACM SIGCOMM best paper award** while our work on the SDX platform[2] has won the **USENIX NSDI community award**.

[1] Vissicchio et al., SIGCOMM, 2015

[2] Gupta et al., NSDI, 2016

Real-world impact Nearly all the solutions we have developed either work today or offer significant benefits in the early stages of their deployments. Thanks to this philosophy, **four** of our papers have been awarded with an **IETF/IRTF Applied Networking Research Prizes** for "recent results in applied networking research that are relevant for transitioning into shipping Internet products and related standardization efforts". Some of the code we released is also used in production. For instance, the CloudRouter project[3] has integrated parts of our SDX codebase that it ships with their releases.

[3] `https://cloudrouter.org`

In addition to winning one of the IETF/IRTF ANRP prize, our work on routing attacks in the context of cryptocurrencies also had a considerable impact and got covered by a myriad of blog posts, technical articles[4], along with hundreds of mentions on social networks.

[4] Bitcoin.com, The Register, The Coin Telegraph, Naked Security, and The Morning Paper

*Declarative network programmability **today***

Computer networks are hard to manage. Given a set of high-level requirements (e.g., connectivity, security, reliability), networks operators have to coordinate the individual behavior of potentially thousands of devices running complex distributed routing protocols so that they, collectively, compute a compliant forwarding state. If this was not hard enough, operators need to coordinate these behaviors by relying on low-level configuration languages which vary, not only across vendors, but also across devices types. Not surprisingly, this complexity leads to many human mistakes, so many that they are currently responsible for the majority of the network downtimes.

The Software-Defined Networking (SDN) paradigm has recently emerged as a way to break this logjam by making network behaviors programmable from a single vantage point. SDN is predicated around two key characteristics. First, it logically separates the control-plane (which decides how to forward traffic) from the data-plane (which forwards the traffic accordingly). Second, it consolidates the different control-plane elements into one logically-centralized control program which relies on a standardized programming interface to provision the forwarding state in the network data-plane elements.

The tenets of SDN got tremendous traction in both academia and the industry. Yet, while almost everybody agreed on the benefits of centralized management, realizing and deploying SDN in practice turned out to be hard. Reasons include the need to deploy new hardware, the variability in how different hardware supported the (various) OpenFlow specifications, along with the challenges of designing scalable and reliable centralized network controllers. In 2018, 10 years after the original OpenFlow paper, few networks are "SDN-enabled", at least not the way it was originally intended.

During the last four years, our group has taken pioneering steps in bringing programmability within *and* between existing networks. First, we developed techniques to precisely control the network-wide behavior induced by distributed routing protocols by either injecting carefully crafted routing inputs (Fibbing[5]); or synthesizing low-level router configurations (SyNET[6], NetComplete[7]). Second, we introduced the concept of Software-Defined Internet eXchange Points (SDX) so as to precisely control inbound and outbound inter-domain traffic (SDX[8] and iSDX[9]).

[5] Vissicchio et al., SIGCOMM, 2015

[6] El-Hassany et al., CAV, 2017

[7] El-Hassany et al., NSDI, 2018

[8] Gupta et al., SIGCOMM, 2014

[9] Gupta et al., NSDI, 2016

With **Fibbing**[4], we introduced a set of primitives which enable to precisely control the output of distributed routing protocols, i.e. the network forwarding behavior, by automatically synthesizing the routing messages taken as inputs by these protocols (see Figure 3). We designed and proved the correctness of several algorithms that compute optimized routing inputs given high-level requirements. One of our key contributions was to prove that Fibbing is expressive enough to induce any loop-free forwarding behavior. This was highly unexpected and recognized as an important achievement.

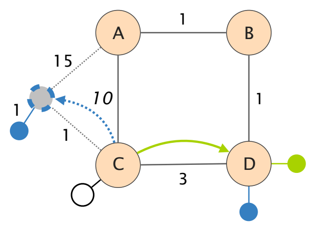


Figure 3: Fibbing in action. By injecting a fake node between *A* and *C* pointing to *A*, Fibbing manages to divert the blue traffic (and nothing else) upwards.

With **SyNET**[5], we introduced the network-wide configuration synthesis problem: Given a network specification $\mathcal{N}$, which defines the behavior of all routing protocols run by the routers, and a set $\mathcal{R}$ of requirements on the network-wide forwarding state, discover a configuration $\mathcal{C}$ such that the routers converge to a forwarding state compatible with $\mathcal{R}$. We captured the network behavior using Datalog and phrased the synthesis problem as an instance of finding an input for a Datalog program such that the program's fixed point satisfies the given properties. We then showed how to solve the synthesis problem by reducing it to satisfiability of SMT constraints. In addition to the problem formulation, our key contribution with SyNET was to show how to generate configurations for multiple routing protocols which depend on each other (e.g., BGP and OSPF).

With **NetComplete**[6], we explored a different style of configuration synthesis in which operators provide a sketch of the intended configurations (i.e., partial configurations with "holes") in addition to their high-level requirements. The job of the synthesizer is then to fill the holes such that the completed configurations are compliant with the requirements (if possible). The use of a partial configurations addresses two important challenges inherent to existing synthesis solutions such as SyNet: *(i)* it allows the operators to precisely control the shape of the synthesized configurations; and *(ii)* it allows the synthesizer to leverage the existing configurations to gain performance. Besides the novel setting, our main contribution with NetComplete was to show how to instantiate powerful synthesis techniques such as Counter-Example Guided Inductive Synthesis (CEGIS) to the network domain. To the best of my knowledge, NetComplete is currently the most advanced configuration synthesizer.

Finally, with **SDX**[7] and **iSDX**[8], we brought programmability to Internet routing via Internet eXchange Points (IXPs). More specifically, we showed how deploying programmable devices at IXPs can enable a lot of the performance and flexibility benefits promised by SDN in wide-area networks. Our main contribution with these papers was in designing a scalable controller platform that behaves correctly when interacting with BGP and possibly conflicting requirements coming from different neighbors.

Network analysis

Nowadays network operators struggle to answer even basic questions about their network behavior, for at least two reasons. First, and perhaps ironically given the amount of network data exchanged, collecting high-quality traffic statistics remains a major challenge as network devices are optimized for forwarding traffic not extracting information from it. Second, network operators have to manually bridge the large semantic gap separating these imperfect observations from the corresponding high-level insights that explain them.

Our group is currently tackling these two fundamental challenges by developing new network measurement and reasoning techniques.

We have already made three key contributions: *(i)* Stroboscope[10,11], a network monitoring approach based on deterministic packet sampling; *(ii)* Net2Text[12], a network "captioning" engine; and *(iii)* Bayonet[13], a probabilistic network reasoning engine.

[10] Tilmans et al., HotNets, 2016

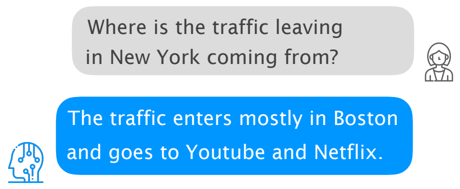
[11] Tilmans et al., NSDI, 2018

[12] Birkner et al., NSDI, 2018

[13] Gehr et al., PLDI, 2018

With **Stroboscope**[10,11], we introduced a network monitoring approach which enables fine-grained monitoring of *any* traffic flow by instructing routers to mirror millisecond-long traffic slices in a programmatic way. By coordinating packet mirroring across routers, Stroboscope can precisely track packets as they cross the network. Our key contributions included the design of efficient rules placement and minimization algorithms together with the design of a runtime system that can bound the amount of traffic monitored.

With **Net2Text**[12], we introduced the concept of network captioning which is akin to image captioning in which models generate dense natural language descriptions of images. Net2Text takes as input a query expressed in natural language together with low-level network measurements and automatically produces succinct summaries, also in natural language, which efficiently capture network-wide semantics (see Figure 4 for an example). Our key contribution with Net2Text was to phrase the captioning problem as an optimization problem that aims to balance coverage (i.e., how much state does the summary describe) and explainability (i.e., how much information does the summary provide). As this problem is NP-hard, we developed approximation algorithms which generate summaries in few seconds, with marginal loss of quality.


Where is the traffic leaving in New York coming from?

The traffic enters mostly in Boston and goes to Youtube and Netflix.

Figure 4: Net2Text in action. Given a query expressed in natural language, Net2Text automatically summarizes low-level network observations (e.g., forwarding tables and statistics) into short and insightful summaries also expressed in natural languages.

With **Bayonet**[13], we introduced a reasoning engine which can reason about network behaviors under uncertainty and ensure that probabilistic properties are met. Examples include ensuring that the probability of network congestion is below a threshold or performing Bayesian reasoning to ensure that supposedly random load-balancing is performing adequately. Our key contribution was to phrase the probabilistic network reasoning problem as inference in existing probabilistic languages. Doing so enabled us to leverage existing inference systems and to offer a flexible and expressive interface to operators.

Data-driven network control

Today the Internet routing system does not optimize for performance and often ends up directing network traffic along suboptimal paths. Even worse, the Internet routing system is slow at computing these decisions, and will get slower as it continues to grow.

Our group is tackling these challenges by developing the concept of **data-driven routing** in which network devices gather insights over a large corpus of data and adapt their decisions accordingly. Our first key contribution in this domain was SWIFT[14], a predictive fast-reroute framework which analyzes control-plane data to estimate the total extent of a failure and preventively reroute traffic.

[14] Holterbach et al., SIGCOMM, 2017

With **SWIFT**[14], we tackled the two fundamental root causes behind slow Internet routing convergence by relying on: *(i)* inference algorithms to address slow outage notification by predicting the overall extent of the remote outage out of few control-plane messages; and *(ii)* a hierarchical forwarding table along with pre-computed data-plane tags to quickly reroute the affected traffic towards unaffected backup paths. Taken together, these techniques enabled us to reduce the convergence time to $\sim$2 seconds, independently of the number of affected prefixes. We also proved that, under reasonable assumptions, SWIFT will not cause any forwarding loops and will only be beneficial, irrespective of the set of routers deploying it.

New domain: Network security

Computer networks are hostile environments in which new attacks are reported literally every day. Each of these attacks can lead to possible loss of connectivity, reduced performance or violation of privacy. Until a few years ago, these network attacks mostly originated from the Internet, fostering research into building better and faster security appliances (e.g., firewalls) sitting at the edge of the networks. Nowadays, network attacks are also often performed by insiders, acting directly from within the network. With respect to Internet-originated attacks, in-network attacks are harder to protect from as they can come from anywhere in the network (not just from few Internet-facing links) and can take many different forms (host-based, link-based or device-based).

Our group is currently investigating the extent of various network threats together with how we can build secure and more robust network systems. Thus far we have made two key contributions. First, we demonstrated the effects of routing attacks on various applications such as anonymity networks (Raptor[15,16]) and cryptocurrencies (Bitcoin[17,18]). Second, we started to develop various "self-defense" techniques which enable networks to automatically detect and mitigate insider attacks (NetHide[19], iTap[20]).

[15] Vanbever et al., HotNets, 2014
[16] Sun et al., USENIX Security, 2015
[17] Apostolaki et al., IEEE S&P, 2017
[18] Apostolaki et al., NDSS, 2019
[19] Meier et al., USENIX Security, 2018
[20] Meier et al., SOSR, 2017

In our works on **routing attacks**, we quantified the possible impact of large-scale routing attacks on anonymity and cryptocurrencies networks. Using actual routing and application data, we demonstrated that routing attacks can have devastating consequences such as allowing attackers to deanonymize Tor users[15,16] or prevent the *entire* Bitcoin network to function[17]. In addition to uncovering this new vector of attacks, we also investigated how to prevent them. In our most recent paper, **SABRE**[18], we described the design of a secure and scalable Bitcoin relay network which relays blocks worldwide through a set of connections that are resilient to routing attacks.

In our works on **self-defending networks**, we developed techniques to obfuscate important network properties from attackers by leveraging network programmability. With **iTap**[19], we showed how to anonymize internal network communications by rewriting packet headers at the network edge. As rewriting all headers does not scale, our key contribution was to rewrite headers in a way that guarantees strong anonymity while, at the same time, scaling the control-plane (number of events) and the dataplane (number of flow rules). With **NetHide**[20], we showed how to obfuscate entire network topologies preventing attackers from gathering topology information and use it to craft attacks. Our key contribution was to phrase the network obfuscation problem as a multi-objective optimization problem that allows for a flexible tradeoff between security and usability.

The future: "Deep" Network Programmability

I intend to continue working on the above themes along with investigating the possibilities offered by **In-Network Computing**, a new paradigm which leverages the capability of programmable network hardware (both inside the network and at the edge) to perform computation at line rate. How to best leverage "deep" network programmability and for what purpose is a promising research area. In the following, I briefly highlight a few topics I would like to explore.

Machine Learning meets Networks One area I plan to look at deals with leveraging in-network computing to learn more representative and up-to-date traffic models. Having such traffic models is key to many applications such as anomaly detection. Today, training precise models is hard as the only source of data often consists of randomly sampled packets collected asynchronously. In addition, I also intend to explore how to leverage programmability to perform inference at line rate instead of having to divert traffic to middleboxes.

Self-Driving Networks Another area I intend to explore deals with the ability to offload control plane tasks to the network hardware, enabling it to directly detect and react to network events (e.g., failures). Doing so would be beneficial for at least two reasons. First, it would enable to massively speed up routing computations, as modern network hardware can process billions of packets per second. Second, it would enable traffic-aware routing in which traffic is rerouted, not according to control plane signals, but according to data plane ones. As control planes signals are often slow to propagate[14], I think this can drastically help. Overall, I believe this direction is highly promising as attested by a recent HotNets paper we wrote on the subject[21].

[21] Molero et al., HotNets, 2018

Managing Programmable Networks Another problem I intend to study deals with how to manage these deeply programmable networks. A key challenge here comes from the fact that programmable networks maintain state which might be critical to maintain to ensure application correctness. Important questions include how to allocate this state across applications (or possibly share it) and how to migrate it across devices (e.g., to support scale-in/out operations). Similarly to self-driving networks, we started to scratch the surface in a recent short paper published at SOSR[22].

[22] Luo et al., SOSR, 2017

References

Click on the title to access the corresponding PDF. All my publications, talks, and CV are also available online on `https://vanbever.eu`.

Maria Apostolaki, Aviv Zohar, and Laurent Vanbever. Hijacking Bitcoin: Routing Attacks on Cryptocurrencies In *Security and Privacy (IEEE S&P'17), 2017 IEEE Symposium on*, pages 375–392. IEEE, 2017. IETF/IRTF ANRP Award .

Maria Apostolaki, Gian Martin, Jan Müller, and Laurent Vanbever. SABRE: Protecting Bitcoin against Routing Attacks. In *Proceedings of the 26th Network and Distributed System Security Symposium (NDSS'19)*, 2019.

Rüdiger Birkner, Dana Drachlser-Cohen, Laurent Vanbever, and Martin Vechev. Net2Text: Query-Guided Summarization of Network Forwarding Behaviors. In *Proceedings of the 15th USENIX Symposium on Networked Systems Design and Implementation (NSDI'18)*. USENIX, 2018.

Ahmed El-Hassany, Petar Tsankov, Laurent Vanbever, and Martin Vechev. Network-Wide Configuration Synthesis. In *Proceedings of the 2017 International Conference on Computer Aided Verification (CAV'17)*. Springer, 2017.

Ahmed El-Hassany, Petar Tsankov, Laurent Vanbever, and Martin Vechev. NetComplete: Practical Network-Wide Configuration Synthesis with Autocompletion. In *Proceedings of the 15th USENIX Symposium on Networked Systems Design and Implementation (NSDI'18)*. USENIX, 2018.

Timon Gehr, Sasa Misailovic, Petar Tsankov, Laurent Vanbever, Pascal Wiesmann, and Martin Vechev. Bayonet: Probabilistic Inference for Networks. In *Proceedings of the 39th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI'18)*, pages 586–602. ACM, 2018.

Arpit Gupta, Laurent Vanbever, Muhammad Shahbaz, Sean Donovan, Russ Clark, Nick Feamster, Jennifer Rexford, and Scott Shenker. SDX: A Software Defined Internet Exchange. In *Proceedings of the 2014 ACM SIGCOMM Conference (SIGCOMM'14)*. ACM, 2014.

Arpit Gupta, Robert MacDavid, Rüdiger Birkner, Marco Canini, Nick Feamster, Jennifer Rexford, and Laurent Vanbever. An Industrial-Scale Software Defined Internet Exchange Point. In *Proceedings of the 13th USENIX Symposium on Networked Systems Design and Implementation (NSDI'16)*, 2016. USENIX NSDI Community Award .

Thomas Holterbach, Stefano Vissicchio, Alberto Dainotti, and Laurent Vanbever. SWIFT: Predictive Fast Reroute. In *Proceedings of the 2017 ACM SIGCOMM Conference (SIGCOMM'17)*, Los Angeles, CA, USA, 2017.

Shouxi Luo, Hongfang Yu, and Laurent Vanbever. Swing State: Consistent Updates for Stateful and Programmable Data Planes. In *Proceedings of the Symposium on SDN Research (SOSR'17)*, pages 115–121. ACM, 2017.

Roland Meier, David Gugelmann, and Laurent Vanbever. iTAP: In-network Traffic Analysis Prevention Using Software-Defined Networks. In *Proceedings of the Symposium on SDN Research (SOSR'17)*, ACM SOSR '17, pages 102–114, New York, NY, USA, 2017. ACM. CyCon Junior Scholar Award .

Roland Meier, Petar Tsankov, Vincent Lenders, Laurent Vanbever, and Martin Vechev. NetHide: Secure and Practical Network Topology Obfuscation. In *Proc. USENIX Security Symposium*. USENIX Association, 2018.

Edgar Costa Molero, Stefano Vissicchio, and Laurent Vanbever. Hardware-Accelerated Network Control Planes. In *Proceedings of the 17th ACM Workshop on Hot Topics in Networks (Hotnets'18)*, pages 120–126. ACM, 2018.

Y. Sun, A. Edmundson, L. Vanbever, O. Li, J. Rexford, M. Chiang, and P. Mittal. RAPTOR: Routing Attacks on Privacy in Tor. In *Proc. Usenix Security*, August 2015.

Olivier Tilmans, Tobias Bühler, Stefano Vissicchio, and Laurent Vanbever. Mille-Feuille: Putting ISP traffic under the scalpel. In *Proceedings of the 15th ACM Workshop on Hot Topics in Networks (HotNets'16)*, pages 113–119. ACM, 2016.

Olivier Tilmans, Tobias Bühler, Ingmar Poese, Stefano Vissicchio, and Laurent Vanbever. Stroboscope: Declarative Network Monitoring on a Budget. In *Proceedings of the 15th USENIX Symposium on Networked Systems Design and Implementation (NSDI'18)*. USENIX, 2018.

Laurent Vanbever, Oscar Li, Jennifer Rexford, and Prateek Mittal. Anonymity on quicksand: Using BGP to compromise TOR. In *Proceedings of the 13th ACM Workshop on Hot Topics in Networks (HotNets'14)*, page 14. ACM, 2014.

Stefano Vissicchio, Olivier Tilmans, Laurent Vanbever, and Jennifer Rexford. Central Control Over Distributed Routing. In *Proceedings of the 2015 ACM SIGCOMM Conference (SIGCOMM'15)*, 2015. ACM SIGCOMM Best Paper Award IETF/IRTF ANRP Award .