Quantizing Radio Link Data Rates to Create Ever-changing Network Conditions in Tactical Networks

ROBERTO RIGOLIN F. LOPES¹, JOHANNES LOEVENICH¹², PAULO H. RETTORE¹, SHARATH M. ESWARAPPA¹² AND PETER SEVENICH¹

¹Communication Systems (KOM), Fraunhofer FKIE, Bonn, Germany (e-mail: {roberto.lopes, johannes.loevenich, paulo.lopes.rettore, sharath.maligera.eswarappa, peter.sevenich}@fkie.fraunhofer.de)
²Institute of Computer Science IV, University of Bonn, Bonn, Germany (e-mail: {s6joloev, s6shmali}@uni-bonn.de)

Corresponding author: Roberto Rigolin F. Lopes (e-mail: roberto.lopes@fkie.fraunhofer.de).

This investigation was funded by BAAINBw (Federal Office of Bundeswehr Equipment, Information Technology and In-Service Support) and WTD81 (Bundeswehr Technical Center for Information Technology and Electronics).

ABSTRACT Several sources of randomness can change the radio link data rate at the edge of tactical networks. Simulations and field experiments define these sources of randomness indirectly by choosing the mobility pattern, communication technology, number of nodes, terrain, obstacles and so on. Therefore, the distribution of change in the network conditions is unknown until the experiment is executed. We start with the hypothesis that a model can quantize the network conditions, using a set of states updated within a time window, to define and control the distribution of change in the link data rate before the experiment is executed. The goal is to quantify how much variation in the link data rate a tactical system can handle and how long it takes to resume IP data-flows after link disconnections. Our model includes functions to combine patterns of change together, transforming one pattern into another, jumping between patterns, and creating loops among different patterns of change. We use exemplary patterns to show how the change in the data rate impacts other link metrics, such as latency and jitter. Our hypothesis is verified with experiments using VHF radios over different patterns of change created by our model. We compute the inter-packet latency of three types of IP data-flows (broadcast, unicast and overlay) to highlight the time to resume data-flows after long link disconnections. The experimental results also support the discussion on the advantages and limitations of our model, which was designed to test tactical systems using military radios.

INDEX TERMS Quantizing Link Data Rates, Ever-changing Network Scenarios, Tactical Networks

I. INTRODUCTION

Radio links at the edge of tactical networks are exposed to several sources of randomness that may affect the current link data rate [1]–[6]. For example, the sources of randomness are node mobility, type of terrain, physical obstacles, weather conditions, radio jamming by an adversary, physical/cyber attacks and so on. The combination of these random effects will define the mixed distribution of change in the network conditions observed during simulations, field experiments or military exercises. Therefore, the distribution of change in the network conditions is unknown until the experiment is executed. But with the beginning of the Scientific Revolution, Galileo Galilei (1564-1642) recommended that we "measure what can be measured, and make measurable what cannot be measured". Hence, our goal is to make the distribution of change in the link data rate measurable and controllable before the experiment is executed.

Given the wide range of military operations, we assume that the network conditions in tactical networks are difficult to anticipate and measure. One major question is how to make network conditions measurable and reproducible for quantitative comparisons, also including the element of chance (randomness) likely to happen in real-world scenarios such as hostile environments on battlefields. The present investigation introduces a model to define the distribution of change in the link data rate before the experiment is executed. The network conditions are quantized by discrete states representing the link data rate within a time window, which are updated by a combination of probability distributions. Our model has functions and algorithms to generate different
patterns of change to test the interoperability among user-facing nodes, tactical routers and military radios; referred as tactical system throughout this paper. Empowered by this abstract leap, we argue that tactical systems able to handle ever-changing link data rates have a higher probability of thriving during any military operation.

We start with the hypothesis that we can define a stochastic model to create sequences of link data rates to quantify the performance bounds of tactical systems, therefore, computing its robustness to changes in the network. Remember that robustness is the system’s ability to cope with both runtime errors and erroneous input [7]. Some of these errors are result of the system’s lack of sensing capabilities to identify changes in the link metrics, such as data rate, latency, jitter and packet loss. Unplanned link disconnection is the worst event in any communication scenario and emission control is an example of a planned one-way link disconnection; i.e. nodes can listen to radio signals but are not allowed to send data avoiding to reveal their position in the battlefield. For example, the system may not have an interface with radios and tactical routers, and may not implement multi-layer control loops to cope with changes in the network. The latter often relies on cross-layer contextual monitoring and analysis giving quantitative feedback to multi-layer control mechanisms that act in enforcement points adapting incoming and outgoing data-flows [8]–[11].

In this investigation, we enhance our stochastic model assuming that the underlying probability distribution of the link data rate is also changing as a function of time using inhomogeneous Markov chains [12]. First we define a set of exemplary link data rate patterns to study its effects on latency and jitter. Then, we construct three functions to transform one pattern of change into another, to jump among different patterns of change and to create loops among patterns of data rate change. Our hypothesis was verified with experiments in a Very High Frequency (VHF) network using real military radios (PR4Gs by Thales). We performed experiments with broadcast, unicast and overlay IP data-flows to characterize the system robustness to link disconnections over stable (non-stochastic baseline) and over ever-changing network conditions (stochastic). The network conditions created by our model can also be used to quantify the performance of reliable transport protocol and to test multi-layer store-and-forward mechanisms deployed at the edge of tactical networks.

The main contributions of this paper are the following:

- Discussion of experimental results quantifying the robustness of three types of IP data-flows in a VHF network using military radios.

The rest of this paper is organized as follows. Section II discusses the related investigations also developing tools or methodologies to test systems over different network scenarios. Section III describes our stochastic model to create different patterns of data rate change. Section IV introduces the functions to transform, jump and loop among different patterns of change. Section V discusses experimental results quantifying the latency observed in three types of data-flows in a VHF network. Section VI discusses the advantages and limitations of our model. Finally, Section VII concludes the paper and suggests future work. Table I lists the mathematical notation used throughout this paper.

II. RELATED WORKS

The performance evaluation of distributed systems gives quantitative feedback to identify run-time errors and to incrementally improve the system overall robustness. Therefore, network simulators/emulators and real test-beds are platforms to test the performance bounds of the system, providing arguments to identify the limitations of the system and guiding possible improvements. Since the communication scenarios are reproducible, the improvements in the system can be quantified and served as a feedback for further improvements. Many platforms to simulate/emulate network conditions are available nowadays [13], enabling the users to develop scenarios to test systems’ performance. However, the challenge is to model the distribution of change in the network conditions to evaluate the performance bounds of the system. Here, we are interested in testing tactical systems composed of a hierarchy of software and hardware components, and using radio communication technology at the edge of tactical networks.

In this investigation, we assume that the system is being tested using real military radios and tactical routers in a laboratory environment. For example, Fig. 1 lists the main
functional blocks from NATO’s Consultation, Command and Control (C3) taxonomy \[14, 15\] side-by-side with the military equipment used in the experiments reported in this investigation. In this figure, the hierarchy of functional blocks is in between the user-facing applications (A) and the tactical network (B) and is composed of five blocks, namely 0) radio(s), 1) communication services, 2) core enterprise services, 3) Community of Interest (COI) and 4) cross-layer information exchange (vertical). The latter couples together the military equipment hosting these functional blocks. The experiments in the present investigation used three equipment defined as follows:

- **User Node (2-3-4):** hosts the user-facing applications \[16\], COI services and core services, such as a queue of messages to differentiate messages by Quality of Service (QoS) parameters, such as priority, reliability and time of expire;
- **Tactical Router (1-2-4):** hosts routing services, core services and distributed control functionalities anticipated by Software Defined Networking (SDN);
- **Military Radio (0-1-4):** implements the physical layer and link layer, providing the fundamental IP connectivity, such as broadcast and multicast functionalities.

This figure also shows the three main problems partially addressed in the literature testing tactical systems \[10\]. The first challenge is to create user-generated data-flows from Command and Control (C2) systems to stress the underlying systems (Problem A). Second, how to create ever-changing network conditions including link disconnections that are reproducible for quantitative comparisons (Problem B). Lastly, how to handle the user data-flows A, given the network conditions B? Here, we are extending our solution to Problem B. The goal is to create network conditions to test the interface among the user-facing node, the tactical router and the military radio.

### A. BENCHMARK TOOLS

Commercial benchmark tools like NetSys-T \[17\], PassMark Advanced Network Test \[18\], Ixia network test solutions \[19\], Keysight network test solutions \[20\] are designed to evaluate the performance of systems emulating close-to-real network conditions and data-flows. These tools provide a large number of features and some of them can also introduce cyber attacks mixed with the user data-flows. However, most of these solutions have cost involved, address specific network characteristics and applications, and were developed to create test scenarios for high-speed network, such as the internet or intranets; therefore, not considering the edge of tactical networks. In programmable test platforms, one could use the model introduced in this investigation to define the distribution of change in the link data rate during performance tests.

There are also tools to create different types of user data-flow, such as **iperf**, Distributed Internet Traffic Generator (D-ITG) \[21\], MGEN \[22\] and academic tests like reported in \[23\]. The experiments discussed later in Section \[III-C\] used D-ITG to study the effects of the data rate change in the inter-packet latency and jitter.

### B. TACTICAL SYSTEMS

The tactical edge hosts mobile nodes using heterogeneous ad hoc networks providing radio links with low data rate and high latency \[24-26\]. Thus, the movement of nodes and the presence of physical obstacles on the way will constantly change the network topology and the link quality, which may include link disconnections (worst case). There are investigations combining simulation and emulation to create realistic military scenarios \[26, 27\] to test the whole software stack within tactical systems.

Recent investigations in the literature \[28-42\] evaluate tactical systems (also called middleware, proxy, gateway or router) by simulating/emulating different network scenarios with conditions varying among three well-defined states, namely **disconnected, intermittent** and **limited**. However, most of the network scenarios are hard to reproduce because there is no precise definition of the distributions of change in the network conditions. Moreover, most of the experiments use non-stochastic means to create radio link disconnections. Therefore, these experiments reported in the literature motivated us to develop a stochastic model to create link disconnections, which is the main subject of the present investigation.

In \[43\], the authors discuss the challenges to periodically estimate or predict the current network state, proposing a solution to solve the problem. However, the proposed solution was not tested over non-deterministic network conditions including link disconnections. The authors in \[44\] developed network simulations and network field emulations to evaluate their messaging application at the edge of tactical networks. Their solution leverage the information shared by a proactive routing protocol used as an input for a forwarding algorithm at the application layer called GetCloser. They also proposed metrics to characterize and quantify the quality of experience among several simulations and real experiments.

Other studies reported in \[45, 46\] performed experiments using real network conditions (e.g. a convoy with few vehicles moving through a particular area) in order to test or propose services and applications. However, real deployments demands trained personnel and the military equipment \[47\]. Therefore, these experiments are difficult to reproduce, and are limited to a specific evaluation scenario which requires precise definition to quantify the performance of the tactical systems using the network. Moreover, hybrid experiments combining real and simulated/emulated network environments, are also developed to evaluate the system performance \[48\]. These solutions try to vary the network conditions using non-stochastic means to increase the accuracy quantifying performance metrics and determining when the system fails to perform.
TABLE 1: Definition of mathematical notations

| Notation | Definition |
|----------|------------|
| $B_{m}$ | Identity transition matrix $i$ for constant data rates |
| $B_{m}$ | Transition matrix $m$ from $B$ representing model $B$ |
| $B_{T}$ | Transformation matrix to change $B$ |
| $\mathfrak{B}$ | Set of transition matrices $B_{m}$ for $m \in \{1, \ldots, M\}$ |
| $\mathfrak{B}_{T}$ | Set of transformation matrices |
| $\delta(t)$ | Transformation step function mapping a time step $t$ to a number of time steps from $\Delta$ |
| $\Delta$ | Set of number of steps for transformations between patterns |
| $J_{i}$ | Pattern result of the jump $i$ |
| $L_{i}$ | Pattern result of the loop $i$ |
| $\lambda$ | Time distribution for state update |
| $N$ | Number of states in $S$ |
| $Model_{B}(B, \lambda)$ | Stochastic model to create patterns of change in the link data rate introduced in [10] |
| $Model_{B}(B, \Sigma, \lambda)$ | In-homogeneous Markov model to create patterns of change in the link data rate |
| $P_{1j}$ | Conditional probability of state $s_{i}$ to be chosen |
| $P_{i}$ | Next given that $s_{j}$ is the current state $(s_{i}|s_{j})$ |
| $\phi_{1}$ | Update function for the state $X_{t} = s_{i}$ of Markov chain |
| $\phi_{2}$ | Transformation function extending $\mathfrak{B}$ and $\theta$ to $\Sigma$ |
| $\phi_{3}$ | Update function for the in-homogeneous model |
| $\phi_{\text{sample}}$ | Sample function to sample a sequence of states $\Sigma$ |
| $s_{1}$ | State identifier |
| $S$ | Set of states of a Markov chain |
| $\sigma_{1}$ | State of the experiment at time $t$ |
| $\Sigma$ | Sequence of states $\{\sigma_{1}, \ldots, \sigma_{T}\}$ sampled by $\phi_{\text{sample}}$ |
| $\theta(t)$ | Transformation matrix function mapping a time step $t$ to the index $m \in \{1, \ldots, M\}$ of a transition matrix $B_{m} \in \mathfrak{B}$ |
| $\theta_{\text{trans}}(t)$ | Transformation matrix function mapping a time step $t$ to a transformation matrix $B_{T_{i}}$ in $\mathfrak{B}_{T}$ |
| $t$ | Point in time $\Sigma$ or $\Sigma_{\text{in}}$ |
| $\tau_{1}$ | Time distribution parameter at time $t$, defining how long the link will stay in a state |
| $T_{i}$ | Pattern result of the transformation $i$ |
| $\Sigma_{\text{in}}$ | Set of time steps $\{1, \ldots, T\}$ of an experiment |
| $\Sigma_{\text{out}}$ | Set of initial time steps $\{1, \ldots, T\}$ |
| $x_{1}, x_{2}$ | Random numbers between 0 and 1 |
| $X_{t}$ | Random variable at time $t$ |
| $X_{t}$ | Markov chain state vector at time $t$ |

C. PREVIOUS INVESTIGATIONS

In general, the recent literature shows limited quantitative evidence evaluating tactical systems over non-deterministic network conditions. Thus, we have been developing models to create ever-changing communication scenarios including the element of chance that can be reproduced for quantitative comparisons. In previous investigations [9], [10], we introduced two stochastic models to create communication scenarios with both user data-flows and network conditions changing independently. In [8], [10], [11] we reported experiments using the proposed models in a VHF network, the model to create network conditions is extended in this paper. The present investigation seeks to define a wide range of patterns of link data rate change to quantify the robustness of tactical systems. We develop our previous investigations further combining patterns of change in different ways using an enhanced version of our model. The goal is to evaluate the robustness of tactical systems handling a wide range of non-deterministic network conditions including link disconnections therefore complementing the state-of-the-art.

III. THE STOCHASTIC MODEL

In this section, we revisit our previous model to create patterns of link data rate change, reported in [10], which is the starting point for the enhanced model introduced later. In Section IV our stochastic model is composed of two distributions, one defining the link data rate and the other defining the time interval for updating the data rate. The distribution of link data rate is defined by the Markov chain $B$ and the time distribution for state update by $\lambda$, in short $Model_{B}(B, \lambda)$; we reuse the notation adopted in [10] for consistency among our previous papers.

A. DEFINITION

We defined the data rates supported by the modulations of our VHF radios (also referred as waveform in the literature) as states of the Markov chain $B$, as shown in Fig. 2. In this figure, the state 0 represents a radio link disconnection, state 1 the 0.6 kbps of nominal data rate and so on (for 1.2, 2.4 and 4.8 kbps) until state 5, which represents 9.6 kbps; the highest data rate supported by the radios. One can do a similar mapping to different types of military radio by using the nominal data rates supported by different radio modulations or waveforms. This mapping can be done even with communication technologies using different radio frequencies, such as High Frequency (HF) and Ultra High Frequency (UHF).

The motivation for this construction is to generate arbitrary sequences of data rates, represented by a finite set of states $|S| = N$ that can be reproduced for quantitative comparisons. Given any communication technology, the maximum nominal data rate is the best communication scenario $s_{N-1}$ (say state 5 in our case, see Fig. 2) and link disconnection is the worst scenario $s_{0}$ (state 0), because the node is out of the network and cannot communicate. Our model $Model_{B}(B, \lambda)$ can create all possible network conditions between these two extremes by quantizing these conditions into discrete states, which are updated by a combination of probability distributions. In the following we take advantage of this abstract leap to get closer and closer to the performance bounds of tactical systems by controlling the network conditions manipulating probability distributions within Markov chains.
Formally, the distribution of link data rate is defined by a sequences of data rates (states) to illustrate the discussion probabilities for all the states (\(B\) of a particular state filled with ones. By contrast, the \(B\):

\[
\begin{pmatrix}
0 & 1 & 2 & 3 & 4 & 5 \\
0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 \\
2 & 1 & 0 & 0 & 0 & 0 \\
3 & 1 & 0 & 0 & 0 & 0 \\
4 & 1 & 0 & 0 & 0 & 0 \\
5 & 1 & 0 & 0 & 0 & 0
\end{pmatrix}
\]

are ten exemplary patterns creating a range of network conditions from low (< 2 kbps) to high (> 4 kbps) data rates on average.

We assume that the variation in link data rate is the most important metric to evaluate multi-layer store-and-forward mechanisms in tactical systems because latency, jitter and packet loss are by products of fluctuations in the data rate (or vice-versa). Later in Section III-C we discuss quantitative experiments showing how the changes in the link data rate impacts both latency and jitter in a VHF network.

1) The update function

Let us assume that we have a transition matrix \(B\). The initial state \(X_0\) is arbitrarily defined, say \(X_0 = s_1 = 1\), and the next state is computed by the update function \(\phi_1\) in Eq. 2.

\[
X_{t+1} = \phi_1(X_t = s_i, x; B, \tau_{t+1}) = \begin{cases} s_0, & \text{for } x \in [p_{i0}, p_{i1}) \\ s_1, & \text{for } x \in [p_{i1}, p_{i2}) \\ s_2, & \text{for } x \in [p_{i2}, p_{i3}) \\ s_3, & \text{for } x \in [p_{i3}, p_{i4}) \\ s_4, & \text{for } x \in [p_{i4}, p_{i5}) \\ s_5, & \text{for } x \in [p_{i5}, 1] \end{cases}
\]

For example, let us assume that the current state is \(s_i\), then \(s_j\) is chosen as next state if \(x\) is in between \(p_{ij}\) and \(p_{ij+1}\) for \(j \in [0, ..., 5]\), where \(p_{ij+1} = 1\) for \(j = 5\). Since \(x\) is a random number between 0 and 1, and the lines of the transition matrix of a Markov chain always sum up to 1, the process will always compute a next state \(X_{t+1}\) following the
chain. Moreover, $\tau_{t+1}$ in $\phi_1$ changes according to the time distribution for state update $\lambda$. Thus $\lambda$ defines how long the link will stay in a given state, which represents a particular link data rate.

An exemplary implementation of the update function can be seen in Alg. 1. For simplicity we used an $N$-dimensional vector representation to describe the current state $X_t$ and next state $X_{t+1}$ of the Markov chain. More detailed $X_t$ has value 0 for all entries up to position $i$ which has value 1, describing that $X_t = s_i$. The same holds for $X_{t+1}$ and $j$ respectively. Using the current state $X_t$, a matrix $B$ holding the conditional probabilities of a Markov chain, a random seed and the state time parameter $\tau_i + 1$ as input, Alg. 1 calculates the next state $X_{t+1}$ of the Markov chain.

2) Exemplary sequences of states

We implemented the update function in Eq. 2 and executed the process twelve times using the different transition matrices in (1), which generated the twelve sequences of states plotted in Fig. 3. In this figure, each sequence has one hundred states out of six possible states (from 0 (red) to 5 (green)). The sequence starts from the bottom left of the plot and moves line by line from the left to the right until it reaches the top right corner. The three algebraic signs $\equiv$, $+$ and $-$ placed in between states denote "no change", "increment" and "decrement" in the data rate, respectively. We leverage the use of these three signs and the different colors to emphasize the changes in the link data rate generated by our model.

Notice that the left column in Fig. 3 lists stable network conditions defined by the $B_0_{....5}$ transition matrices in (1). $B_0$ represents the worst case scenario with 0 kbps for all states (Fig. 3a), and $B_5$ represents the optimum scenario with all states representing maximal nominal data rate 9.6 kbps (Fig. 3f). The right column lists the patterns of change created using the $B_{0.5}$ transition matrices in (1). The goal of these two columns plots is to show a striking comparison between non-stochastic and stochastic patterns of change quantizing network conditions into discrete states.

The properties of the transition matrices can be described as follows. $B_0$ has high probabilities for states 0 and 1, resulting in an average data rate of $\sim$0.4 kbps (Fig. 3b). $B_1$ has low probability of link disconnection with data rate of $\sim$1 kbps on average (Fig. 3c) and $B_2$ is very similar to $B_1$ but has zero probability for link disconnection ($\sim$1.5 kbps on average). $B_3$ represents a significant improvement in the network conditions by moving between states 2, 3 and 4 almost surely ($\sim$2.7 kbps on average). $B_4$ is a slightly modification of $B_3$ shifting equal probabilities to the states 5, 4 and 3 resulting in 4.8 kbps (Fig. 3e), and $B_5$ has equal probabilities ($1/6$) for all states with about 2.8 kbps on average (Fig. 3i).

These patterns create the foundation to generate more elaborated patterns of data rate change to quantify the robustness of tactical systems and are used as an input to our enhanced model introduced later in Section IV.

**FIGURE 3:** Six stable patterns $B_{0,...,5}$ (left column) and six changing patterns $B_{0,...,5}$ (right column) created with Model$_B$
TABLE 2: Data-flow metrics observed over stable and changing link data rates in a VHF network

| Metric                          | $B_0$ | $B_1$ | $B_2$ | $B_3$ | $B_4$ | $B_5$ | $B_6$ | $B_7$ | $P_1$ | $P_2$ |
|---------------------------------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| Total time (seconds)            | -     | 1493  | 2172  | 1490  | 1133  | 1081  | 570   | 609   | 358   | 370   |
| Total packets                   | 100   | 25    | 100   | 99    | 100   | 100   | 100   | 100   | 100   | 66    |
| Maximum latency                 | -     | 225.56| 36.90 | 29.57 | 22.39 | 23.17 | 18.36 | 17.81 | 13.88 | 18.00 |
| Average latency                 | -     | 58.48 | 21.97 | 15.08 | 11.47 | 10.77 | 5.77  | 6.14  | 3.63  | 3.75  |
| Standard deviation              | -     | 55.95 | 4.27  | 5.33  | 2.46  | 7.50  | 5.73  | 5.15  | 3.95  | 4.47  |
| Average jitter                  | -     | 39.67 | 3.25  | 2.93  | 1.71  | 5.97  | 5.37  | 4.69  | 3.51  | 3.74  |
| Bytes received                  | 0     | 32650 | 130600| 129294| 130600| 130600| 130600| 130600| 130600| 130600|
| Average bitrate                 | 0     | 0.17  | 0.48  | 0.69  | 0.92  | 0.96  | 1.83  | 1.71  | 2.91  | 2.82  |
| Average packet rate             | 0     | 0.016 | 0.04  | 0.06  | 0.08  | 0.09  | 0.17  | 0.16  | 0.28  | 0.27  |
| Packets dropped                 | 100   | 75    | 1     | 0     | 0     | 0     | 0     | 0     | 0     | 34    |

![Figure 4: Pendulum patterns $P_1$ and $P_2$](image)

(a) $P_1 \sim 3.42$ kbps ±2.82  
(b) $P_2 \sim 2.76$ kbps ±2.79

3) Non-stochastic patterns of change

We also created two non-stochastic sequences starting from state 5 and decreasing the data rate going step-by-step through all the states (9.6 kbps - 0 kbps) and back-and-forth in a pendulum pattern (9.6 kbps - 0 kbps - 9.6 kbps). Fig. 4a plots the pendulum $P_1$ with average data rate of $\sim 3.4$ kbps and Fig. 4b plots the pendulum $P_2$ with link disconnections reducing the average link data rate to $\sim 2.7$ kbps. Notice that these plots have no "equal" (=) signs between the states, which means that the link data rate is always decreasing or increasing by a factor of 2. Since these patterns are deterministic, it is easier to check if a tactical system can handle changes in the link data rate using feedback from the radio and tactical router. If a control mechanism can handle these changes (e.g. [10]), it has a higher probability of also handling the stochastic patterns.

C. QUANTIFYING LATENCY AND JITTER

1) Defining the metrics

Our model is varying the link data rate, however one could achieve similar results modeling other network metrics such as latency and jitter. Therefore, we want to quantify the effects of changing the data rate on latency and jitter in the VHF network used in all experiments reported in this paper.

Remember that data rate is the ratio between the amount of data, minus eventual data loss, over the time to send the data through a network link (here called latency), as defined in Eq. [3]. By consequence, latency is the ratio between the amount of data sent over the link data rate. And jitter is the absolute difference between the latency and the average latency, so it is also directly related to latency. Since we are measuring these metrics at the IP layer, we assume that the amount of data is the packet size and latency is the inter-packet latency.

\[
\text{data rate} = \frac{\text{data - loss}}{\text{latency}} \quad \Rightarrow \quad \text{latency} = \frac{\text{data}}{\text{data rate}} \quad (3)
\]

2) The experiments

We performed experiments over stable conditions using the six patterns in the left column of Fig. 3 (a,c,e,g,i,k), and over changing conditions using six patterns in the right column of Fig. 3 (b,d,f,h,j,l); using 10 seconds for update interval between states ($\tau_i = 10 \forall i \in \{1, ..., T\}$ sec). The goal was to compile quantitative evidence showing that the changes in the link data rate implies changes in the latency and, by consequence, in the jitter. This is a comparative study using stable conditions as a baseline against the changing conditions generated by our model.

Since our VHF radios have a 128 kB buffer, we sent 100 UDP (User Datagram Protocol) packets of 1264 Bytes each to avoid buffer overflow (i.e. 128 kB > 126 kB so the packets fit in the radio buffer) using the traffic generator tool called D-ITG (Distributed Internet Traffic Generator) as described in [21], [49]. We computed the one-way latency observed between two nodes using the logs collected at the receiver side, therefore not including the delay introduced by the radio buffer in the results.

![Figure 5: Parameters used at receiver (1) and sender (2)](image)

Fig. 5 lists the parameters used to set the D-ITG tool at the receiver (line 1) and at the sender (line 2). Notice that we set the tool to use the management network for signaling (–Si sets the interface and –Sda sets the IP address for signaling) and the VHF network for the UDP data-flows being measured in this study (i.e. using –i for the network interface connected to the radio network and –a for the radio IP address). The reason for splitting signaling and data-flow is that signaling uses the TCP protocol (Transmission Control Protocol) and the high latency in our VHF network caused timeouts in the socket, waiting to finish the TCP handshake.
where the white boxes represent stable conditions and yellow boxes represent changing conditions. Notice that the changing conditions generated a wider range of latency on average (compare the yellow boxes against the white ones) and the link disconnections created latency "outliers" in $B_3$ and $P_2$. 

Complementing these plots, Table 2 compiles all the metrics observed during experiments discussed in this section. $B_0$ is listed for completion and the frequent link disconnections in $B_0$ broke the radio IP routes, which means that the experiment could not be finished. During $B_1$ we observed $\sim 22$ seconds of average latency against $\sim 15$ seconds for $B_1$ which created better network conditions. From the pair $B_2$/$B_2$ to the pair $B_5$/$B_5$ we observed higher variations in both latency and jitter, when executing experiments with changing data rates.

IV. THE ENHANCED MODEL

In this section, we further develop our stochastic model to create more elaborated sequences of link data rates. Throughout this paper these sequences are also called patterns of change. For this purpose we define three functions to incrementally transform one pattern of change into another (Section IV-A1). Using the transformation functions we define a nested Markov chain to combine different patterns of change arbitrarily (Section IV-A2) and to create loops among two or more patterns of change (Section IV-A3). As a result, we can combine the twelve exemplary patterns in Fig. 3 using mathematical relations. The idea behind this enhanced model is to increase our control over both complexity and variation in these patterns of change to define experiments that also change as a function of time.

A. DEFINITION OF FUNCTIONS

1) Transformations among patterns

Let us assume that we have a set of integers $\mathfrak{T}_{\text{init}} \subseteq \mathbb{Z} = \{1, \ldots, T\}$, where $T$ represents the number of different time steps in an experiment and a set $\mathcal{B}$ of initial matrices $B_1, \ldots, B_M$, holding the conditional probabilities of $M$ different Markov chains representing patterns of change in the network. Moreover, let $\theta : \mathfrak{T}_{\text{init}} \to \{1, \ldots, M\}$ and $\delta : \mathcal{T} \to \Delta$ be two functions, mapping time steps $t$ either to the unique index $m \in \{1, \ldots, M\}$ of a matrix $B_m \in \mathcal{B}_{\text{init}}$, s.t. $B_{\theta(t)} = B_m$ or to a number of transformation steps $\delta(t)$ defined by a set $\Delta \subseteq \mathbb{N}$. It should be noted that initially $\mathfrak{T}_{\text{init}} = \mathfrak{T} = \{1, \ldots, T\}$ does not necessarily apply. $\mathfrak{T}_{\text{init}} \neq \mathfrak{T}$ is the case if the number of steps $\delta$ differs from the simple mapping $\delta(t) = 1 \forall t \in \mathfrak{T}$. In the following we describe a process to extend the mapping $\theta$ (update function) to be well defined on $\mathcal{T} = \{1, \ldots, T\}$, s.t. we have a complete description of our experiment in terms of $\theta, \delta$ and a set of matrices $\mathcal{B}$ over time $\mathfrak{T}$.

Given a number of steps $\delta(t)$, we can transform one pattern of change into another by increasing or decreasing the average link data rate incrementally. This means, if we want to transform $B_{\theta(t)}$ into $B_{\theta(t+\delta(t))}$ in $\delta(t)$ steps (number of interactions within a time window), we subtract $B_{\theta(t)}$ from

3) Initial experimental results

A comparison between the latency over stable conditions and over changing link data rates is shown in four plots at Fig. 6. Fig. 6a plots five blue-ish curves representing the latency over $B_{1..5}$ (stable data rates) and Fig. 6b plots five red-ish curves representing the latency observed over $B_{1..5}$ (changing data rates). One can see the more regular latency curves observed during stable conditions in (a) getting more irregular over changing conditions in (b). This point is highlighted in (c) with the two curves for the two pendulum patterns, where $P_1$ (black curve) varies up and down as an inverse function of the link data rate and $P_2$ (green curve) has latency peaks (> 45 sec) during the link disconnections. We observed that the radio combines two packets when the data rate is above 2.4 kbps resulting in latency close to zero in these plots. Finally, Fig. 6d plots the box-plot for all the experiments,
$B_{θ(4+δ(t))}$ and divide by the number of steps ($δ(t) > 0$) as defined in Eq. 4 and vice-versa, where $B_{θ(t)}$ and $B_{θ(4+δ(t))}$ define arbitrary probability distributions like the transition matrices in $B$. The resulting transformation matrix $B_{T_{1}}$ can be added $δ(t)$ times to $B_{θ(t)}$ to transform $B_{θ(t)}$ into $B_{θ(4+δ(t))}$. For example, Fig. 7a shows the sequence of states starting with the probability distribution from $B_{3}$, which is transformed into $B_{0}$ in 8 steps ($δ(1) = 8$, and the resulting $B_{T_{1}}$ is shown in $B$); i.e. for each two lines in the figure we changed the initial matrix $B_{3}$ towards transforming it into the $B_{0}$ matrix. This process degraded the link data rate as a function of time.

$$B_{T_{1}} = B_{θ(4+δ(t))} - B_{θ(t)}$$

(4)

$$0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0$$

$$1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0$$

$$2 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0$$

$$3 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0$$

$$4 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0$$

$$5 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0$$

(5)

As a natural result we can describe the update process for changing the conditional probabilities as a function of time $θ_{trans}$ : $\mathcal{T} \rightarrow \mathcal{B}_{T}$ by mapping each $t \in \{1, \ldots, T\}$ to $B_{T_{t}} \in \mathcal{B}_{T}$, where $\mathcal{B}_{T}$ is the set of transformation matrices defined in Eq. 4. Moreover, we can add the matrices $B_{θ(t)} + B_{T_{t}}$ to $\mathcal{B}$, update integer $M$ to fulfill $M = |\mathcal{B}|$ and extend $θ$ to a mapping $\mathcal{T} \rightarrow \{1, \ldots, M\}$ s.t. $B_{θ(t+1)} = B_{θ(t)} + B_{T_{t}}$ with $δ(t) > 1$ respectively. This transformation process can also be interpreted as function

$$φ_{2} : (\mathcal{B}, θ, δ, \mathcal{T}_{init}) \rightarrow (\mathcal{B}, θ, \mathcal{B}_{T}, \mathcal{T})$$

(6)

mapping archetype $(\mathcal{B}, θ, δ, \mathcal{T}_{init})$ to image $(\mathcal{B}, θ, \mathcal{B}_{T}, \mathcal{T})$ extending $\mathcal{B}$ and $θ$ to $\mathcal{T} = \{1, \ldots, T\}$. Alg. 2 shows an exemplary implementation of the transformation function $φ_{2}$ assuming that we can access the matrices in $\mathcal{B}$ and $\mathcal{B}_{T}$ using a key-value data structure.

Once $\mathcal{B}$ and $θ$ are extended to $\mathcal{T} = \{1, \ldots, T\}$, we can sample a sequence of patterns $Σ = \{σ_{1} = X_{1}, \ldots, σ_{T} = X_{T}\}$, characterizing the communication scenario of an experiment, by calling update function $φ_{1}$ for each time step $t \in \mathcal{T}$ w.r.t. the current state $X_{t}$, pattern matrix $B_{θ(t)}$, random seed and state time distribution parameter $τ$.

$$φ_{sample}(X_{0}, \mathcal{B}, θ, \mathcal{B}_{T}, Σ) \rightarrow Σ = (σ_{1}, \ldots, σ_{T})$$

(7)

Alg. 2 shows one way to implement the sample function in Eq. 7 using update function $φ_{1}$ by re-using the StateUpdate Algorithm (Alg. 1).

Another way of transforming the probability distribution for a state update is to use a pre-defined transformation matrix like $B_{T_{2}}$ in $B$ to increase the probability of one state (say zero) and decreasing the probability of another state (say five). Notice that the transformation matrix needs to be symmetric with its lines summing up to zero, therefore, guaranteeing that the lines in the transition matrix still sums to one. More precisely, we can reduce the probability of system state $X_{t} = s_{t} \in \{0, \ldots, 5\}$ in $δ(t)$ steps (scaling factor), so we divide $B_{T_{1}}$ by $δ(t)$ as defined in Eq. 4. Here $B_{θ(4+δ(t))}$ is the transformed matrix at time $t + δ(t)$, $B_{θ(t)}$ is the transition matrix at time $t$, $B_{T_{t}}$ is the predefined transformation matrix and $δ(t)$ is the number of transformation steps. An example for $X_{2} = 4$ is plotted in Fig. 7b. In this figure, at the bottom there are several appearances of state 4 (green) which is incrementally replaced by state 0 (red), therefore, also degrading the link data rate as a function of time.

$$B_{θ(t+δ(t))} = B_{θ(t)} + \frac{B_{T_{t}}}{δ(t)}$$

(8)

Algorithm 2: Sample

Input: $X_{0}, \mathcal{B}, θ, δ, T, λ, seed$

Output: Sequence of states $Σ = \{σ_{1} = X_{1}, \ldots, σ_{T} = X_{T}\}$

Initialization:

1: $Σ \leftarrow \{\}$
2: $N \leftarrow X_{0}.length$
3: $X_{t} \leftarrow X_{0}$
4: for $t = 1$ to $T$ do
5: $m \leftarrow θ(t)$
6: $\mathcal{B} \leftarrow \mathcal{B}[m]$
7: $τ \leftarrow λ(t)$
8: $X_{t} \leftarrow StateUpdate(X_{t}, \mathcal{B}, seed, τ)$
9: for $i = 0$ to $N − 1$ do
10: if $X_{t}[i] == 1$ then
11: $Σ \leftarrow Σ \cup \{i\}$
12: end if
13: end for
14: end for
15: return $Σ$
Algorithm 3: Pattern Transformation

Input: \( \mathcal{B}, \theta, \delta, \overline{\mathcal{X}}_{\text{init}} \)
Output: \( \mathcal{B}, \mathcal{B}_T, \theta \)

Initialization:
1. \( \overline{\mathcal{X}} \leftarrow \overline{\mathcal{X}}_{\text{init}} \)
2. \( \mathcal{B}_T \leftarrow \{ \} \)
3. \( M \leftarrow |\mathcal{B}| \)
4. for \( t \in \overline{\mathcal{X}}_{\text{init}} \) do
5. \( \mathcal{B}_T \leftarrow (\mathcal{B}[\theta(t)] - \mathcal{B}[\theta(t + \delta(t))]) / (\delta(t)) \)
6. if \( (\delta(t) > 1) \) then
7. for \( (j = 1 \) to \( \delta(t) - 1) \) do
8. if \( ((\mathcal{B}[\theta(t)] + j \mathcal{B}_T) \notin \mathcal{B}) \) then
9. \( \theta(t + j) \leftarrow M + 1 \)
10. \( \mathcal{B} \leftarrow \mathcal{B} \cup (\mathcal{B}[\theta(t)] + j \mathcal{B}_T) \)
11. else
12. Find \( m \in M \) s.t. \( \mathcal{B}[m] = \mathcal{B}[\theta(t)] + j \mathcal{B}_T \)
13. \( \theta(t + j) \leftarrow m \)
14. end if
15. \( \mathcal{B}_T \leftarrow \mathcal{B}_T \cup j \mathcal{B}_T \)
16. \( M \leftarrow M + 1 \)
17. end for
18. end if
19. \( \mathcal{B}_T \leftarrow \mathcal{B}_T \cup \mathcal{B}_T \)
20. end for
21. \( \overline{\mathcal{X}} \leftarrow \{1, ..., T\} \)
22. return \( \mathcal{B}, \theta, \mathcal{B}_T, \overline{\mathcal{X}} \)

Note that one needs to adopt a round strategy to keep both the transition matrix \( B_{\theta(t)} \) and the transformation matrix \( B_{T_t} \) consistent during the update process. That is, the rows of \( B_{\theta(t)} \) must sum to one and the rows of \( B_{T_t} \) must sum to zero.

2) Jumps among patterns

Let us define another transformation function \( \phi_3 \) to "jump" from one probability distribution into another, using a set of \( M \) Markov chains \( \mathcal{B} = B_1, ..., B_M \) as states of another Markov chain \( B_J \). The resulting model \( \text{Model}_B(B, \overline{\mathcal{X}}, \lambda) \) is illustrated in Fig. 8. In this figure, state zero is \( B_0 \), state one is \( B_1 \), state two is \( B_2 \) and so on until state five which is \( B_5 \) (\( \mathcal{B} = \{B_0, B_1, B_2, B_3, B_4, B_5\} \)); these transition matrices were previously defined in [1] and plotted in Fig. 3. Therefore, the probability distributions in [1] are used by the update function \( \phi_3 \) as defined in Eq. [9]. For example, given the current system state \( X_t = s_i \) and two random numbers \( x_1, x_2 \in [0, 1] \), the next state \( X_{t+1} = s_j \) is computed by determining the next step transition matrix \( B_{t+1} \) matching \( x_2 \) and the probability intervals from the \( B_J \) distribution and calling the update function \( \phi_3 \) with sampling parameter \( x_1 \), transition matrix \( B_{t+1} \) and time distribution \( \tau_{t+1} \sim \lambda \) afterwards. This is called in-homogeneous Markov chain because the probability distribution may change as a function of time (i.e. \( t \in \overline{\mathcal{X}} = 1, ..., T \)) [12].

Given a set of matrices \( \mathcal{B} = B_1, ..., B_M \), an initial state vector \( \overline{X}_0 \) and time distribution \( \lambda \subseteq \mathbb{N} \) we can compute a sequence of states \( \Sigma = \{\sigma_1, ..., \sigma_T\} \) over time \( \overline{\mathcal{X}} = \{1, ..., T\} \) following the in-homogeneous model \( \text{Model}_B(B, \overline{\mathcal{X}}, \lambda) \) by first sampling \( |T| \) transition matrices using the \( B_J \) distribution and applying the Sample algorithm (Alg. 2) afterwards. To this end, let \( B_t \) be the transition matrix that we sample at time \( t \), then we set \( \theta(t) = B_t \) to guarantee that \( \theta \) is a well defined mapping \( \overline{\mathcal{X}} \rightarrow \{1, ..., T\} \) after the sampling process.

Moreover, we define \( \delta : \overline{\mathcal{X}} \rightarrow \Delta \) to be the simple mapping \( \delta(t) = 1 \) \( \forall t \in \overline{\mathcal{X}} \), where \( \Delta = \{1\} \). That is, the process will change the state distribution in one step that we call "jump".

Now we can apply Sample algorithm (Alg. 2) with input \( \overline{X}_0 \), \( \theta, \delta, T, \lambda \) and a random seed to get a sequence \( \Sigma \) representing

Algorithm 4: Jump

Input: \( \overline{X}_0, \mathcal{B}, B_J, \theta, T, \lambda, \text{seed} \)
Output: Sequence of states \( \Sigma = \{\sigma_1 = X_1, ..., \sigma_T = X_T\} \)

Initialization:
1. \( M \leftarrow |\mathcal{B}| \)
2. \( m \leftarrow \text{randomNumber(seed,} 1, M) \)
3. \( B_t \leftarrow \mathcal{B}[m] \)
4. \( \overline{X}_0 \leftarrow \overline{X}_0 = 0 \)
5. \( \overline{X}_{\text{inh}}[m] \leftarrow 1 \)
6. \( \theta(1) \leftarrow m \)
7. for \( t = 1 \) to \( T \) do
8. \( x \leftarrow \text{randomNumber(seed,} 0, 1) \)
9. \( \overline{X}_t \leftarrow \overline{X}_{\text{inh}} \leftarrow B_J \)
10. for \( t = 0 \) to \( M - 1 \) do
11. if \( (\overline{X}_{\text{inh}}[i] = 1) \) then
12. \( \theta(t) \leftarrow i \)
13. end if
14. end for
15. \( \delta(t) \leftarrow 1 \)
16. \( \Sigma \leftarrow \text{Sample} \left(\overline{X}_0, \mathcal{B}, \theta, \delta, T, \lambda, \text{seed}\right) \)
17. return \( \Sigma \)
“jumps” among patterns $\mathcal{B} = B_1, \ldots, B_M$ defined in $\phi_1$ (Eq. [3]). For the sake of completeness, however, it should be noted that we could also generate “smooth jumps” among patterns by applying $\phi_2$ on $(\mathcal{B}, \theta, \delta, \Phi_{init})$ (Alg. [5]) before we use the Sample algorithm (Alg. [2]).

An algorithm returning a sequence $\Sigma$ that represents “jumps” among patterns is given in Alg. [4]. This algorithm requires an initial vector $X_0$, a set of matrices $\mathcal{B}$, a matrix holding the probability distribution for the outer Markov chain $B_f$, a place holder for function $\theta$, the maximum number of time steps $T$ of the experiment, a time distribution $\lambda$ for the step length of the system states and a random seed.

Using this inputs the algorithm computes $\Sigma$ by first sampling $T$ transition matrices, updating $\theta$ and calling the Sample algorithm (Alg. [2]) as described before.

Fig. 9a illustrates the output of our “jump” function plotting a sequence of states $J_1$ starting with the best conditions in $B_{s(max)}$ and goes towards $B_{s(min)}$ jumping (every three lines) through $B_4$, $B_3$, $B_2$ and $B_1$ in this sequence. Thus, using a deterministic way of choosing stochastic patterns. Our goal is to use such a pattern to compute what is the variation in link data rate that a tactical system (say radio, router and node) can handle before losing IP connectivity. Because any system will stop exchanging messages or IP packets somewhere between $B_5$ and $B_0$ in the figure. And we want to know what is the maximum variation in data rate that the tactical system can handle, and by consequence also quantifying the system robustness to changes in the network.

This model can also create extreme variations in the link data rate by using another Markov chain to define the probability distribution for state update. For example, using equal probabilities among the six states (say from $B_5$ in [1]) one can create the sequence of states in Fig. 9b where each line was generated by a different probability distribution; let us call it $J_2$ to differentiate from the first example $J_1$. Later in Section IV-B we compare all the patterns of data rate change plotted in this paper.

3) Loops among patterns

Using functions $\phi_2$ and $\phi_{sample}$ we can create a loop of $M > 2$ probability distributions $\mathcal{B} = B_1, \ldots, B_M$. The construction goes as follows. Assuming that a time distribution $\Delta$ and corresponding function $\delta : \Phi_{init} \rightarrow \Delta$ for the number of transformation steps is given, we define the initial set of $M + 1$ different time steps $\{t_1, \ldots, t_{M+1}\} = \Phi_{init}$ s.t. each pair of consecutive time steps $t_m$ and $t_{m+1} \in \Phi_{init}$ fulfills the constraint $t_m - t_{m+1} = \delta(t_m)$. This guarantees that $\phi_2$ generates a well defined mapping $\theta$ on $\Phi$. Moreover, we define function $\theta : \Phi_{init} \rightarrow \{1, \ldots, M\}$ s.t. $\theta(t_m) = m \forall m \in \{1, \ldots, M\}$ and $\theta(t_{M+1}) = 1$. Now we can apply $\phi_2$ on $(\mathcal{B}, \theta, \delta, \Phi_{init})$ to update $\mathcal{B}$ by adding the matrices $B_{\theta(t_m)} + B_{\theta(t_m)} \cdot \delta(t_m) - 1 \cdot B_{\theta(t_{m+1})}$ for all time steps $t_m$ s.t. $\delta(t_m) > 1$ and extend $\theta$ to a mapping from $\mathcal{B}$ to $\Phi = \{1, \ldots, t_{M+1}\}$.

As a result, the computation starts with the first distribution and goes interactively transforming the distributions into another until it arrives at the last distribution. Then, it gets back to the first distribution in the same manner reversing the process. Loops can also be implemented with jumps earlier defined in Section IV-A2 but the transition between the distribution is done in one single step which may create abrupt changes in the link data rate (as in $J_2$ in Fig. 9b).

Loops among different patterns can also be used to quantify the system robustness to changes in the network conditions. We are particularly interested in computing how long it takes to recover from failure created by random occurrences of radio link disconnections. The idea is to create a loop between a pattern with high data rates (good conditions) and another with low data rates, including link disconnections, that surely breaks the system within a particular time window.

For example, Fig. 10a plots a loop with 200 states from $B_4$ to $B_1$ and back to $B_4$. Notice that there is a slight change of patterns in the middle of the plot. We can make the loop transition more drastic replacing $B_1$ with $B_0$. Thus, Fig. 10b starts with $B_4$, changes to $B_0$ and gets back to $B_4$. Frequent
link disconnections (state 0 in the figure) will break the underlining IP routes computed by proactive routing protocols, which will be reestablished after the network conditions improve. Such loops were designed to compute how long it takes to break and recover from challenging conditions over ever-changing data rates, as discussed later in Section V.

B. COMPARING THE PATTERNS OF CHANGE

One can visually compare the diversity of patterns created by the three functions introduced in this section by examining the plots in the figures (T1 and T2, J1 and J2) and (J1 and J2); also paying attention to the average data rate and standard deviation computed with the 200 states in each figure. These plots illustrate the versatility and control of our stochastic model generating different patterns of data rate change to quantify the system robustness.

This comparison gets more concrete plotting the average data rate per line (there are 10 states per line in each plot) to highlight the diversity of variations as a function of time, see the curves in the three plots in Fig. 11. The top plot has T1 (black curve) against T2 (blue curve) with similar data rates, but T1 has a wider variation starting with higher data rates and finishing with lower data rates on average. In the middle plot, J1 (black curve) starts with the best conditions and moves step by step towards complete link disconnection (0 kbps). In sequence, J2 (red curve) jumps randomly among the patterns of change with greater standard deviation at each point in the curve. Finally, L1 (black curve) and L2 (green curve) have similar patterns of change but L2 has lower datarates in the middle of the pattern (say lines from 10 to 15). Notice that there are 20 points per curve meaning that each point is the average data rate of ten states or one line of the sequences with 200 states, and the error bars plot the standard deviation from the mean.

Complementing, Table 3 lists the average datarate, standard deviation and states distribution (%) for all the patterns of change discussed in this paper. The states distribution column lists the percentage of each of the six states composing the twenty-two patterns in this paper. These patterns were arbitrarily defined to illustrate the definitions throughout this paper, and some of them were used in the experiments reported in the next section.

V. EXPERIMENTAL RESULTS

In this section, we describe the experimental setup and discuss results over non-stochastic and stochastic patterns of link data rate generated with our enhanced model Model2(B, Σ, λ) introduced earlier in Section IV. The goal is to verify if these patterns of change can reveal the limitations of three types of IP data-flows in a VHF network.

A. NETWORK SETUP

Our experimental network has a data plane composed of four VHF radios (1) each connected to a router (2) and then to a laptop (3), as illustrated in Fig. 12. In this figure, the radio antennas are wired and connected to four coaxial relays (0) in a laboratory environment. The relay was used to "cut" the antenna cable, of a particular radio for a given time window distributed with respect to λ (say 10 seconds), to implement the link disconnection (state 0 in our model). We performed experiments between two nodes acting as sender (laptop 1) and receiver (laptop 2). During the experiments, the link data rate between the two nodes is changing following one of the patterns defined earlier in sections III or IV. Our network also has a data acquisition server (4) connected to the nodes.

| Pattern | Data rate ± SD | States distribution (%) | Fig. |
|---------|----------------|-------------------------|------|
| B1      | 0.00 ± 0.00    | 100 0 0 0 0 0           | 7a   |
| B2      | 0.39 ± 0.28    | 70 30 0 0 0 0           | 7b   |
| B3      | 0.60 ± 0.09    | 100 0 0 0 0 0           | 7c   |
| B4      | 1.02 ± 0.29    | 28 72 0 0 0 0           | 7d   |
| B5      | 1.20 ± 0.00    | 0 0 100 0 0 0           | 7e   |
| B6      | 1.47 ± 0.75    | 28 35 37 0 0 0         | 7f   |
| B7      | 2.40 ± 0.00    | 0 0 0 100 0 0           | 7g   |
| B8      | 2.71 ± 1.53    | 0 2 35 71 0 0           | 7h   |
| B9      | 3.10 ± 0.00    | 0 0 0 0 100 0           | 7i   |
| B10     | 3.85 ± 2.84    | 0 0 0 3 33 32           | 7j   |
| B11     | 4.80 ± 0.00    | 0 0 0 0 0 100           | 7k   |
| B12     | 9.60 ± 0.00    | 0 0 0 0 0 100           | 7l   |
| B13     | 2.84 ± 3.17    | 19 0 0 0 0 17           | 7m   |
| T1      | 3.42 ± 2.82    | 12 25 25 25 13          | 11a  |
| T2      | 2.76 ± 2.79    | 12 20 20 20 10          | 11b  |
| J1      | 2.28 ± 3.04    | 20 20 16 14 12 18      | 11c  |
| J2      | 3.55 ± 3.10    | 24 1 1 29 31 14        | 11d  |
| J3      | 4.31 ± 4.18    | 22 10 14 14 6 36       | 11e  |
| J4      | 4.39 ± 4.04    | 17 13 14 9 13 34       | 11f  |
| L1      | 3.92 ± 3.63    | 11 12 10 15 24 28      | 11g  |
| L2      | 3.77 ± 3.64    | 36 5 5 23 18 22        | 11h  |
| L3      | 3.67 ± 3.94    | 30 5.6 12.6 14.4 9.4 28| 11i  |
| L4      | 1.76 ± 1.84    | 39 0.6 18.4 20 22 0    | 11j  |
via a management network to collect all the logs plotted and a data acquisition server (4).

FIGURE 12: VHF network with wired antennas hosting relays (0), radios (1), routers (2), user-facing nodes (3) and a data acquisition server (4)

Regarding the implementation of ModelB, we implemented the algorithms described in [IV] as well as a script to change the radio modulation following a given pattern using the radio’s SNMP (Simple Network Management Protocol) interface. The link disconnection was implemented controlling one relay between the two radios used in the experiment; i.e. the second relay was not used in this configuration. The investigation started with experiments over non-stochastic network conditions to define the disconnection time bounds for the broadcast/unicast/overlay IP data-flows and to develop our intuition to interpret results over ever-changing conditions. The sender was sending three types of IP packets using broadcast/unicast/overlay addresses every 5 seconds and we monitored both sender and receiver in Fig [12]. The three IP data-flows are defined as follows:

- **Broadcast**: allows one-to-many data-flows using the last IP in a domain, e.g. 192.168.1.255/24 or 192.168.255.255/16. This is a common functionality in LANs, and is also implemented by the VHF radios in our network setup;
- **Unicast**: allows one-to-one data-flows relying on the availability of IP routes among the nodes, which are configured statically among the tactical routers in our network setup creating a fully connected topology;
- **Overlay**: allows one-to-one data-flows through an overlay IP network using routes created dynamically by OLSRv2, which is configured to use the broadcast address for neighbor discovery. We visualized the direct dependency between overlay and broadcast in the latency plots, discussed later in this section.

Thus, we are performing a comparative study using the basal radio functionalities, broadcast and unicast, as a baseline for our own overlay configuration. Table 4 summarizes the experimental setup listing the radio type and main configuration parameters used during the experiments, that we reported in this paper.

### B. RESULTS OVER NON-STOCHASTIC CONDITIONS

1) Pendulum pattern \( P_1 \)

First, we performed an experiment using the pendulum pattern \( P_1 \) to verify if the three data-flows have similar latency over network conditions without link disconnections. Fig. 13a plots the latency\( y_{\text{ups}} \), buffer occupancy\( y_{\text{mid}} \) and data rate\( y_{\text{down}} \) observed over \( T = 200 \) states from \( P_1 \). Since this pattern has no link disconnection, it serves as a baseline to show that the three data-flows have similar latency in our VHF network. The radio buffer was shared by the three data-flows and had similar usage during the whole experiment. Note that the buffer occupancy is plotted with the state colors to visualize the inverse relation between data rate and buffer occupancy more clearly; i.e. the higher the data rate, the lower the buffer occupancy.

2) Link disconnection

In sequence, we performed another non-stochastic experiment including a link disconnection interrupting the three data-flows. The experiment started with a constant link data rate (9.6 kbps) followed by a link disconnection lasting for a pre-defined time interval (say from 20 to 180 seconds). The goal was to define the minimum disconnection time to break the radio’s IP routes, which is about 180 seconds (~3 minutes) for our VHF radios.

Fig. 13b plots the latency\( y_{\text{ups}} \), the buffer\( y_{\text{mid}} \) and the link data rate\( y_{\text{down}} \) as a function of time for the three data-flows, namely broadcast\( y_{\text{black}} \), unicast\( y_{\text{gray}} \) and overlay\( y_{\text{red}} \). There are four points highlighted in this plot, identifying the start (\( t_1 \)) and end (\( t_2 \)) of the link disconnection, marking when the broadcast and unicast flows are resumed (\( t_3 \)), and indicating when the overlay flow is resumed (\( t_4 \)). These four points in time are defined as follows:

- **(t_1)** Link disconnection begins: the point in time identifying that the relay is activated, "cutting" the wired antenna between the sender and receiver (state 0 or 0 kbps). The grey dashed line (vertical) shows the exact point in time;
- **(t_2)** Link disconnection ends: the point in time identifying that the relay is deactivated, reconnecting the two radios using the maximum nominal data rate (state 5 with 9.6 kbps). The green dotted line shows the point in time.
when the data rate changed from zero to 9.6 kbps as also plotted in the bottom of Fig. [13B].

\((t_3)\) The broadcast and unicast data-flows are resumed: the point in time identifying that the IP packets from the broadcast and unicast data-flows were resumed at the receiver. These two data-flows are combined because we observed similar latency during experiments with our VHF radios. Other radios may have different latency for broadcast and unicast, so \(t_3\) would be split into two points in time. Moreover, multicast IP can also introduce distinct latency demanding the addition of another \(t_{\text{multicast}}\):

\((t_4)\) Overlay data-flow is resumed: the point in time where the IP packets from the overlay dataflow are resumed at the receiver;

Therefore, \(t_2\) minus \(t_1\) gives the link disconnection time (180 seconds), \(t_3\) minus \(t_2\) gives the time to resume the radio IP connectivity (\(\sim 226\) seconds or \(\sim 3.77\) minutes) and \(t_4\) minus \(t_3\) computes the additional time delay introduced by our overlay configuration (\(\sim 76\) seconds). Notice that we can reduce the latency for the overlay by reducing the “hello” interval but increasing the data overhead for neighbor discovery; this configuration uses less than 5\% of the maximum nominal data rate, supported by our radios. Also note that there is an annotation \(<\text{flush}>\) marking the point in time when the radio flushed its buffer during link disconnections (see the middle plot in all figures in this section).

In sequence, we discuss the experiments using the patterns of change created by our enhanced model to quantify the data-flows robustness to changes in the link data rate.

C. RESULTS OVER EVER-CHANGING DATA RATES

We repeated the experiment with the three data-flows over six patterns generated by our model, namely two transformations \((T_1\) and \(T_2\)), two jumps \((J_1\) and \(J_2\)) and two loops \((L_1\) and \(L_2\)). In every figure in this section, we plot inter-packet latency (seconds) at the receiver node, as a function of time.
for three data flows against the buffer occupancy (kBytes) and link data rate (kbps) observed at the sender. We set the time interval for state update to 10 seconds for the six experiments with $T = 200$ states each, summing up 2000 seconds of total time per experiment. If one or more data-flows were broken, the link was set to 4.8 kbps and we waited until the data-flows were resumed, which is the case for two patterns $J_1$ and $L_1$, plotted in figures 15a and 16a. Since 10 is eighteen times smaller than 190 seconds, the pattern of change must have at least eighteen zero states in sequence to recreate the disconnection time used in the previous experiment over non-stochastic conditions (Section V-B2), which is the case for two patterns $J_1$ and $L_1$, plotted in figures 15a and 16a.

1) Transformations $T_1$ and $T_2$

The two transformations start with an average data rate higher than 2.4 kbps and goes step-by-step decreasing the data rate also introducing link disconnections, see the stack of plots side-by-side in Fig. 14. On the left side, Fig. 14a plots the latency as a function of time for the first transformation $T_1$ with two peaks reaching less than 240 seconds of inter-packet latency, meaning that the IP routes were not broken until the frequency of link disconnections increased in the second half of the experiment (i.e. time > 1000 sec). Therefore, the latency is similar for the three data flows except for three points where the overlay packets experienced higher latency (see the red curve in Fig. 14b). However, the second transformation $T_2$ created two peaks with less than 200 seconds latency and a higher peak (< 600 seconds) when the link disconnections started to be more frequent towards the end of the experiment, as plotted in Fig. 14b.

2) Jumps $J_1$ and $J_2$

The two jumps created dissimilar network conditions as shown in the two stack of plots in Fig. 15. The first jump $J_1$ in Fig. 15a has three data flows with similar inter-packet latency until the end of the experiment, where the link disconnections start to happen more frequently (see plot at the bottom of the
TABLE 5: Average latency for three data-flows (columns) over six different patterns of change (rows)

| Pattern | Broadcast | Unicast | Overlay |
|---------|-----------|---------|---------|
| T1      | 19.38 ± 32.8 | 1065    | 10.24 ± 31.8 |
| T2      | 14.11 ± 26.3 | 547     | 9.24 ± 30.8 |
| J1      | 11.33 ± 27.8 | 500     | 7.79 ± 19.9 |
| J2      | 9.43 ± 12.8 | 231     | 6.06 ± 15.1 |
| L1      | 11.12 ± 31.2 | 415     | 7.56 ± 25.9 |
| L2      | 19.43 ± 19.9 | 910     | 12.88 ± 72.1 |

By contrast, Fig. 15b shows the three data flows over random jumps in J2 creating peaks of latency after two long link disconnections (100 sec each). Notice that the overlay data-flow accumulated more than 290 seconds (about 4.8 min) of latency at the end of the experiment, against ~220 seconds (3.6 min) for the other two data flows. For J1 the numbers are higher but the proportional difference similar.

3) Loops L1 and L2
The two loops L1 and L2 differ in the network conditions created in the middle of the experiment, where L2 has more link disconnections, as shown in the two stack of plots in Fig. 16. On the left, Fig. 16a plots the first loop L1 with a maximum latency of about 630 seconds (10.5 minutes), indicating that the variation in the data rate restarted the neighbor discovery for overlay (i.e. the routes expired). However, the second loop L2 broke the three data flows around 500 seconds of experiment (x axis) which were resumed after 1700 seconds in the experiment. Overlay had a peak latency of about 1122 seconds, at least 200 seconds (~3.3 minutes) higher than broadcast and unicast, as plotted in Fig. 16b. These numbers suggest that our overlay configuration is significantly less robust to changes in the network conditions.

4) Packet loss for the three data-flows
Fig. 17 plots the packet loss (%) for the three data-flows over the six patterns discussed in this section. Note that broadcast and unicast have similar loss during all the experiments, but overlay has significantly lower loss than the other two. This happens because the dynamic routes expired during the long disconnections so the IP routes are removed from the operating system not allowing the node to send packets to the radio. This is an advantage of having an overlay network because the proactive neighbor discovery will detect link disconnections and topology changes. In case of an alternative link to the receiver, the routing protocol would automatically adjust the routing table to use the other link.

5) Summary of results
Table 5 summarizes the latency observed in the three data-flows (columns) during the six experiments (rows), discussed in this section, namely two transformations (T1 and T2), two jumps (J1 and J2) and two loops (L1 and L2). Complementing, Fig. 18 plots together the range of inter-packet latency observed during the six experiments for comparison against each other. Reinforcing the previous observation we suggest that the overlay data-flows have higher latency during patterns of change with several disconnections like J1, J2 and L2. Notice that the strip-charts also highlight the diversity of link data rates created by our model resulting in lower latency shown in Fig. 18d and three times higher latency in figures 18a and 18c (see the different scale in the x axis).

VI. DISCUSSION
The wide range of military operations and the sensitivity of military deployments motivated us to develop the model introduced in this paper. Military operations can be highly unpredictable and tactical systems need to perform even in the presence of an active adversary in both physical and...
digital domains. Hence, it is challenging to design tactical systems robust to the wide range of conditions in these military operations. Recent literature and military exercises (e.g. NATO’s Coalition Warrior Interoperability eXercise (CWIX) [51]) lack mathematical arguments and quantitative evidence suggesting that tactical systems can handle ever-changing communication scenarios.

Therefore, our models were designed to test the interface among the user-facing node, tactical router and radio platform in a laboratory environment during the system’s development phase or during military exercises like CWIX, but before field experiments. Remember that the interfaces among the different equipment will support the system’s ability to handle changes in the network conditions. Moreover, we developed the discussion throughout this paper comparing non-stochastic network conditions against the stochastic conditions created by our models. This presentation is to emphasize our criticism of recent literature reporting experiments with military radios using non-stochastic means to create network scenarios including link disconnections.

Military personnel usually goes through a tough training program dealing with extreme physical and mental conditions. In an analogy between humans and machines [52], we designed our model to create a tough testing program for tactical systems deployed at the edge of tactical networks. We invite the reader to look again at the patterns of change in this paper as a "set of obstacles" that every new version of a tactical system needs to go through before stepping into the battlefield. Empowered by this abstract leap to define network conditions, we claim that tactical systems, which are able to handle the patterns of change introduced by our model, have a higher probability of thriving during any military operation together supporting humans and robots.

The results in this paper evaluated the performance bounds of IP data-flows, therefore any tactical system using these radios cannot do better than the lower bounds defined by the broadcast and unicast data-flows. We plan to use the experimental results reported here to discuss improvements, with the radio manufacturer, in both configuration and interface with the military radios. We want to know if we can tune the neighbor discovery in the radio in run-time and if a cross-layer mechanism can take control of the flush in the radio buffer that happens during link disconnections.

Finally, we leverage the fact that this journal does not have a page limit to plot two loops with 500 states each as shown in Fig. [19]. On the left, Fig. [19a] the sequence has loops jumping from good conditions to complete disconnection back-and-forth but with increasing number of steps (i.e. $\Delta = 1, 2, 3, 4, 5$). On the other side, Fig. [19b] the sequence of loops starts in good conditions and goes step-by-step transforming into complete disconnection and back-and-forth also increasing the number transformation steps (i.e. same $\Delta$ as $L_3$). These two sequences of patterns can be used in a fully automate process to quantify the robustness of tactical systems, illustrating the flexibility of our model.

![Figure 19: Two sequences of loops $L_3$ and $L_4$](image)

**VII. CONCLUSION**

We have been developing the mathematical tools to quantify the robustness of tactical systems over ever-changing communication scenarios with both user-generated data-flows and network conditions changing independently. In this investigation, we enhanced a stochastic model to create different patterns of link data rate change in a VHF network. We assumed that the network conditions can be quantized in states updated within a time window. The goal was to create a wide range of stochastic network conditions including link disconnection to quantify the robustness of tactical systems to changes in the radio link data rate.

We started this investigation creating a range of patterns of change using a previous model to study the effects of data rate changes on latency and jitter. Since the link data rate is the input for the model, we did experiments over non-stochastic and stochastic patterns of data rates to study the effects of the changes introduced by our model in the latency and jitter in a VHF network. The experimental results showed the inverse
relation between data rate and latency (i.e. the higher the data rate the lower the latency and vice-versa), and jitter is the variance in the latency.

Then, we enhanced the model defining three functions to combine the basic patterns together. The first function transforms one pattern of change into another one. The second function uses patterns of change as states of an inhomogeneous Markov chain jumping among different patterns of change within nested Markov chains. Finally, the third function creates loops among patterns of change combining features of the other two functions. We compared exemplary patterns of data rate change generated by our model to highlight its flexibility and reproducibility. We discussed experimental results in a VHF network quantifying the latency observed with three types of IP data-flows, namely broadcast, unicast and overlay.

These experiments exposed the limits of IP data-flows after link disconnection and the additional latency introduced by neighbor discovery in our overlay configuration. We learned that there is a significant delay to resume IP flows after disconnections longer than 3 minutes. This fact suggests that the network topology expires during long disconnections resulting in re-starting the neighbor discovery process among the radios.

Given the diversity of patterns of change introduced in this investigation, we argue that tactical systems designed to handle these conditions have a higher probability of thriving during any type of network conditions in military operations. Moreover, the extreme network conditions created by our model can be used to test the performance bounds of the interfaces between radio, tactical router and user-facing node. The results discussed here can be used as quantitative feedback for further improvements for multi-layer control loops adapting the tactical system to the current network conditions.

As a future work, we plan to quantify the robustness of reliable transport protocols over ever-changing conditions from both user data-flows and link data rates, computing its performance re-transmitting IP packets lost during link disconnections. We are particularly interested in designing and evaluating multi-layer control loops adapting its metrics to the ever-changing conditions, and also reusing previous experiences in tactical networks within machine learning models. We also plan to extend our model to larger networks to create ever-changing network topologies in simulated/emulated environments.

ACKNOWLEDGMENT

This investigation was funded by Bundeswehr: BAAINBw (Federal Office of Bundeswehr Equipment, Information Technology and In-Service Support) and WTD81 (Bundeswehr Technical Center for Information Technology and Electronics).

The patterns in this paper were inspired by the work of M. C. Escher (1898 - 1972) and B. Mandelbrot (1924 - 2010).

REFERENCES

[1] H. Wang, B. Crilly, W. Zhao, C. Autry, and S. Swank, “Implementing Mobile Ad Hoc Networking (MANET) over legacy tactical radio links,” in IEEE Military Communications Conference (MILCOM), Orlando, FL, USA, Oct 2007, pp. 1–8.

[2] N. Suri, E. Benvenuti, M. Tortonesi, C. Stefanelli, J. Kovach, and J. Hanna, “Communications middleware for tactical environments: Observations, experiences, and lessons learned,” IEEE Communications Magazine, vol. 47, no. 10, pp. 56–63, October 2009.

[3] G. Elmasry, “A comparative review of commercial vs. tactical wireless networks,” IEEE Communications Magazine, vol. 48, no. 10, pp. 54–59, October 2010.

[4] L. Li, P. Vigneron, C. Brown, M. Shi, and T. Kunz, “On designing bandwidth constrained mobile tactical networks for complex terrains,” IEEE Communications Magazine, vol. 50, no. 2, pp. 188–194, February 2012.

[5] M. Tortonesi, A. Morelli, C. Stefanelli, R. Kohler, N. Suri, and S. Watson, “Enabling the deployment of COTS applications in tactical edge networks,” IEEE Communications Magazine, vol. 51, no. 10, pp. 66–73, October 2013.

[6] R. Amin, D. Ripplinger, D. Mehta, and B. N. Cheng, “Design considerations in applying disruption tolerant networking to tactical edge networks,” IEEE Communications Magazine, vol. 53, no. 10, pp. 32–38, October 2015.

[7] IEEE, “IEEE standard glossary of software engineering terminology,” IEEE Std 610.12-1990, pp. 1–84, Dec 1990.

[8] R. R. F. Lopes, P. H. Balaraju, A. T. Silva, P. H. Rettore, and P. Sevenich, “Experiments with a queuing mechanism over ever-changing datarates in a VHF network,” in IEEE Military Communications Conference (MILCOM), Norfolk, VA, USA, November 2019, pp. 712–717.

[9] R. R. F. Lopes, P. H. Balaraju, and P. Sevenich, “Creating and handling ever-changing communication scenarios in tactical networks,” in 15th International Conference on the Design of Reliable Communication Networks (DRCN), Coimbra, Portugal, March 2019, pp. 67–74.

[10] R. R. F. Lopes, P. H. Balaraju, P. H. Rettore, and P. Sevenich, “Queuing over ever-changing communication scenarios in tactical networks,” IEEE Transactions on Mobile Computing, pp. 1–15, June 2020.

[11] P. H. Balaraju, P. H. Rettore, R. R. F. Lopes, S. M. Eswarappa, and J. Loevenich, “Dynamic adaptation of the user data flow to the changing data rates in VHF networks: An exploratory study,” in 11th IEEE International Conference on Network of the Future (NoF), Bordeaux, France, Oct 2020, pp. 1–9.

[12] O. Häggeström, Finite Markov Chains and Algorithmic Applications, ser. London Mathematical Society Student Texts. Cambridge University Press, 2002.

[13] B. Linkleter. (2020) Open-source routing and network simulation. Accessed: 2020-07-07. [Online]. Available: https://www.keysight.com/us/en/solutions/network-test.html

[14] I. Hanany, “Architectural work for modeling and simulation combining the NATO architecture framework and C3 taxonomy,” vol. 14, pp. 139–158, 11 2016.

[15] A. Council, “NATO Unclassified C3 Taxonomy Baseline,” https://www.nato.int/nato_static_fl2014/assets/pdf/pdf_2019_09/20190912_190912-C3-Taxonomy-baseline.pdf, 2019, [Online; accessed 16-July-2020].

[16] I. Evans, B. Ewy, M. Swink, S. Pennington, D. Siquieros, and S. Earp, “TIGR: the tactical ground reporting system,” IEEE Communications Magazine, vol. 51, no. 10, pp. 42–49, October 2013.

[17] B. E. Systems. (2020) Benchmark electronic: Network systems lab. Accessed: 2020-07-07. [Online]. Available: http://www.benchmark-electronics.com/network-systems-lab.html

[18] P. A. Network. (2020) Passmark advanced network test. Accessed: 2020-07-07. [Online]. Available: https://www.passmark.com/products/performanceest/pt_advnet.php

[19] I. Network. (2020) Ixia network test solutions. Accessed: 2020-07-07. [Online]. Available: https://www.ixiacom.com/solutions/network-test-solutions

[20] K. Network. (2020) Keysight: Network test solutions. Accessed: 2020-07-07. [Online]. Available: https://www.keysight.com/us/en/solutions/network-test.html

[21] A. Botta, A. Dainotti, and A. Pescapé, “A tool for the generation of realistic network workload for emerging networking scenarios,” Computer Networks, vol. 56, no. 15, pp. 3531 – 3547, 2012.
ROBERTO RIGOLIN F. LOPES (S’09, M’13) is a scientist at Fraunhofer FKIE in Bonn, Germany. Sitting with the Communication Systems Department (KOM), he has been attacking problems in Computer Networks and Distributed Systems with particular interest in the performance bounds of tactical systems over ever-changing communication scenarios. He is puzzled with Nature so the model introduced here was inspired by the early developments of Quantum Mechanics. His education/experience as a scientist includes three universities in Brazil (UFMT, UFSCar and USP), one in the Netherlands (UTwente), one in Norway (NTNU) and many books on Physics, Mathematics and Philosophy.

JOHANNES LOEVENICH is currently pursuing his masters in Computer Science at University of Bonn, Germany. He is working as a student assistant in Communication Systems Department (KOM) at Fraunhofer FKIE in Bonn, Germany. He received Bachelors degree in Computer Science at 2017 from Rheinische Friedrich-Wilhelms-Universität Bonn, Germany. One year later he also received a Bachelors degree in Mathematics from the same university. His research interests include Computer Systems, Distributed Systems, Tactical Networks, Data Science and Game Theory.

PAULO H. L. RETTORE is a scientist at Fraunhofer FKIE in Bonn, Germany. He received his BSc. and MSc. degrees in Computer Science in 2009 and 2012, respectively. He got his Ph.D. degree in Computer Science at Federal University of Minas Gerais (UFMG) in 2019. His research interests include Computer Networks, Distributed Systems, Tactical Networks, Ubiquitous Computing, Internet of Things, Intelligent Transportation Systems and Smart Mobility.

SHARATH MALIGERA ESWARAPPA is currently pursuing his masters in Computer Science at University of Bonn, Germany. He is working as a student assistant in Communication Systems Department (KOM) at Fraunhofer FKIE in Bonn, Germany. He received Masters degree in Embedded Systems from Manipal University, India in 2014 and Bachelor of Engineering Degree in Information Science from Visvesvaraya Technological University, India in 2012. His research interests include Computer Networks, Distributed Systems, Data Engineering and Tactical Networks.

PETER SEVENICH is the head of the Robust Heterogeneous Network group at the Communications Systems Department at Fraunhofer FKIE in Bonn, Germany. Peter got his degree in Physics from University of Bonn, Germany. He has participated in the conception and execution of several national/international projects developing military communications. His research interests include IP routing in Tactical Networks, Service-Oriented Architectures, Software Defined Networks and Software Defined Radios.