npj | Quantum Information

www.nature.com/npjqi

ARTICLE OPEN

Check for updates

Secure quantum key distribution with a subset of malicious devices

Víctor Zapatero[1]✉ and Marcos Curty[1]

The malicious manipulation of quantum key distribution (QKD) hardware is a serious threat to its security, as, typically, neither end users nor QKD manufacturers can validate the integrity of every component of their QKD system in practice. One possible approach to re-establish the security of QKD is to use a redundant number of devices. Following this idea, we address various corruption models of the possibly malicious devices and show that, compared to the most conservative model of active and collaborative corrupted devices, natural assumptions allow to significantly enhance the secret key rate or considerably reduce the necessary resources. Furthermore, we show that, for most practical situations, the resulting finite-size secret key rate is similar to that of the standard scenario assuming trusted devices.

npj Quantum Information (2021)7:26 ; https://doi.org/10.1038/s41534-020-00358-y

INTRODUCTION

Quantum key distribution[1–4] (QKD) allows for information theoretically secure communications, unaffected by the long-term security weakening inherent to public-key cryptography[5,6]. Its security relies on fundamental physical principles and various assumptions, a crucial one being that the legitimate QKD users, say Alice and Bob, hold honest devices that stick to the QKD protocol and do not intentionally leak their private information to an eavesdropper (Eve). However, this strong assumption is probably unjustified, considering the amount of hardware and software Trojan horse attacks against conventional cryptographic systems reported in the past years[7–11]. After all, likewise conventional security hardware, QKD devices incorporate many sophisticated components typically provided by specialized companies, and neither QKD vendors nor users are capable of validating the security of all these components in practice[12]. However, a malicious component can totally compromise the security of QKD. Indeed, the fabrication process of QKD systems might provide Eve with plenty of opportunities to meddle with the QKD hardware, including both the optical equipment and the classical post-processing (CP) units. Moreover, Eve could even sidestep post-fabrication tests by arranging attack triggers that depend on a sequence of unlikely events[10,13].

Remarkably, not even device-independent (DI) QKD[14–18] can provide security against malicious devices, as shown in ref. [19]. It is the classical nature of the secret keys that makes QKD systems vulnerable to classical hacking in both the DI and the non-DI scenarios because classical keys are susceptible to copying.

A possible solution to foil malicious hardware and software in QKD was recently presented in ref. [20], and then experimentally demonstrated in ref. [21]. The triggering idea is that it might be more difficult for Eve to corrupt various devices than a single device; for example, if they originate from different providers. Therefore, one can use a redundant number of devices for both the raw key generation and the post-processing of QKD. As shown in ref. [20], under the assumption that the number of devices controlled by Eve is restricted, secure QKD is possible by combining verifiable secret sharing (VSS)[22–26]—whose essential building block is secret sharing[27,28], a standard technique in secure hardware design[29]—and privacy amplification (PA)[30,31]. Of course, both tools operate on top of DI and non-DI security analyses, which determine the secret key length that one can extract from the honest optical apparatuses.

However, a major limitation of the proposal in ref. [20] is that it is conceived for the case where all the corrupted devices fully obey a single Eve who can access their internal information and make them arbitrarily misbehave from the protocol. This scenario, which we refer to as the active-collaborative (AC) model, might be over-conservative in many practical situations. For instance, if Alice and Bob purchase devices from different vendors, it might be reasonable to expect that, even if they are corrupted, they do not collaborate, meaning that they do not share their private information with each other or cooperate in any way. Also, if the information delivered by a certain device is different from the one prescribed by the protocol, it might be detected by Alice and Bob a posteriori. In this sense, some QKD users might only request security against non-collaborative (rather than collaborative) or passive (rather than active) corrupted devices.

Crucially, when applied in more sensible corruption models like these, the proposal in ref. [20] provides no advantage at all with respect to the AC model. One major contribution of this work is to prove that some of these models actually enable a significant enhancement of the secret key rate, require fewer honest devices and classical communications than the AC model, or allow to remarkably diminish the post-processing time, a severe bottleneck in QKD. In particular, we introduce conditional VSS, a weaker version of VSS that is more suitable for the task of QKD. In addition, we present a general distributed QKD post-processing protocol appropriate for all the corruption models. Lastly, we evaluate the performance of two well-known QKD schemes in the presence of malicious devices. The simulations corroborate that notably improved non-asymptotic key rates can be reached by replacing the AC model by less conservative and probably more realistic models. Furthermore, in all the considered models, we find that the increased authentication cost of our protocol (compared to that of standard QKD post-processing) is negligible with respect to the secret key length for practical data block sizes and moderate numbers of corrupted devices.

[1]Escuela de Ingeniería de Telecomunicación, Department of Signal Theory and Communications, University of Vigo, Vigo, Spain. ✉email: vzapatero@com.uvigo.es

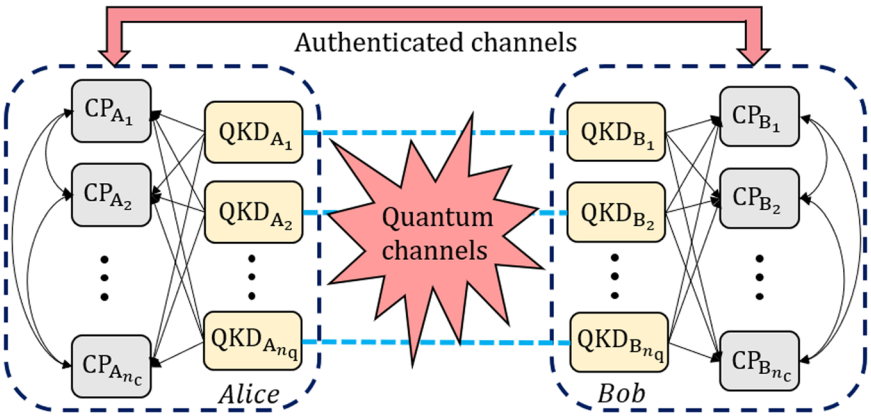

Fig. 1 QKD setup with multiple devices. Proposal of a QKD setup with redundant devices suggested in[20]. The areas surrounded by dashed lines define Alice's and Bob's labs. Alice's (Bob's) lab contains n_q QKD modules (yellow boxes). Each module of Alice is linked to a single module of Bob through a quantum channel (dashed blue lines), forming a so-called QKD pair, and t_q pairs are possibly malicious at most. In addition, Alice (Bob) holds n_c CP units (grey boxes), t_c of them being possibly malicious at most. In each lab, all the CP units are connected to each other and to all n_q local QKD modules via secure channels that provide both privacy and authentication (black solid arrows). Also, every unit of Alice is linked to every unit of Bob through an authenticated classical channel (all of them together symbolized by the red double-end arrow).

RESULTS

We start by describing the general formalism we consider. Without loss of generality, a standard QKD setup can be divided into two parts with separate roles: a QKD module and a CP unit. Alice's and Bob's QKD modules form a so-called QKD pair, whose role is to generate raw correlated data between the parties via quantum communication. Each module transfers its raw data to its local CP unit, and the two distant CP units distill a pair of secret keys from the raw data via coordinated classical post-processing and authenticated classical communication.

The focus of this work is the general scenario where not all the devices are trusted, thus forcing the parties to use a redundant number of them[20]. Throughout the paper, we shall consider that Alice and Bob share n_q QKD pairs (or simply "pairs"), and that each of them holds n_c CP units (or simply "units"). Similarly, we assume that up to t_q QKD pairs are corrupted (a QKD pair is corrupted when at least one of its modules is) and up to t_c CP units are corrupted per lab. Nevertheless, our results could be easily adapted to contemplate different numbers of honest and corrupted units in each lab. For $j = 1, \ldots, n_q$, Alice's (Bob's) module QKD_{A_j} (QKD_{B_j}) is connected to all her (his) units $\{CP_{A_l}\}_{l=1}^{n_c}$ ($\{CP_{B_{l'}}\}_{l'=1}^{n_c}$) via secure channels, that is, channels that provide both privacy and authentication. Also, all of Alice's (Bob's) units are pairwise connected by secure channels too. Since all these links take place within Alice's (Bob's) lab, in practice security could be enforced by using, say, physically protected cables. Similarly, the CP_{A_j} are connected to the $CP_{B_{l'}}$ by authenticated classical channels. Lastly, as usual, a quantum channel fully accessible to an eavesdropper links QKD_{A_j} to its partner QKD_{B_j}. A schematic of this QKD setup is given in Fig. 1.

AC corruption

In the first place, let us briefly summarize the proposal in ref. [20], which establishes the security of QKD in the AC model using the setup of Fig. 1. On the one hand, given that $n_q > t_q$, PA allows to "remove" not only the information Eve gains through her intervention in the quantum channel (as it is done in standard QKD post-processing), but also the information she learns from the corrupted QKD pairs. On the other hand, given that $n_c > 3t_c$[26], VSS enables an honest QKD module to split a raw key into shares and redundantly allocate them among its local CP units for distributed post-processing. Crucially, the properties of VSS may guarantee the secrecy and the correctness of the final keys reconstructible by Alice and Bob at the end of this post-processing.

Before we analyse alternative corruption models, it is convenient to tight up some few loose ends affecting the proposal in ref. [20]. In the first place, it requires the execution of $n_q + 1$ separate PA steps to distill a secret key. On the contrary, in section "Alternative corruption models" we show that a single PA step suffices, which actually applies to all possible corruption models (see also section "Distributed QKD post-processing protocol" for a distributed post-processing protocol that implements PA in a single step).

In the second place, the use of standard VSS assures that the post-processing is resilient to the misbehaving of the CP units at the price of relying on simulated broadcast, better known as byzantine agreement[32]. However, this is a very stringent task: it requires the exchange of an exponentially increasing number of classical messages, say $C \sim O(n_c^{t_c})$, among the units that want to reach the agreement[32]. What is more, the achieved resiliency is probably not relevant for QKD. After all, Eve has unrestricted access to the quantum channel and thus may induce the abortion of the QKD protocol at will. For these reasons, in all corruption models we replace VSS by a weaker cryptographic primitive, namely, conditional VSS (defined in the "Methods" section), which circumvents simulated broadcast by simply allowing the CP units to abort the protocol. As seen in section "Alternative corruption models" below, this replacement is not only advantageous in the AC model, but whenever actively corrupted CP units are considered, whether they collaborate or not.

Alternative corruption models

In what follows, we address various adversarial scenarios alternative to the AC model. In particular, three looser non-mixed corruption models exist: passive and collaborative (PC), active and non-collaborative (AN) and passive and non-collaborative (PN), where non-collaboration is obviously only defined if multiple corrupted devices exist. Importantly, we decouple the analysis of the different corruption models for the QKD modules and the CP units, such that the results we present for the QKD modules do not assume a specific model for the CP units and vice versa. In addition, we maintain the general QKD setup presented in Fig. 1.

Let us discuss the QKD modules first. In virtue of the *privacy* of conditional/standard VSS (see the "Methods" section), a distributed QKD post-processing protocol using VSS guarantees that the

extractable secret key length does not depend on the corruption model of the CP units, but only on that of the QKD pairs. What is more, let us assume for now that the parties select the AC model as their preferred model for the QKD pairs. For $j = 1, \ldots, n_q$, the j-th QKD pair runs an independent QKD session. As shown in Supplementary Note 1, for $n_q = t_q + 1$ (minimum valid choice of n_q for a given t_q), the ϵ_{cor}-correct, ϵ_{sec}-secret key length/extractable via one-step PA from all these sessions is given by

$$l = \left\lfloor \min_j \{h_\varepsilon^j - \lambda_j\} - \log_2\left(\frac{1}{\hat{\epsilon}_{cor}\epsilon_{PA}^2\delta}\right)\right\rfloor, \tag{1}$$

where h_ε^j is a hypothetical lower bound on the ε-smooth min-entropy of Bob's j-th raw key conditioned on the information held by Eve up to the parameter estimation (PE) step, and the smooth parameter ε depends on the PE procedure. As explained in Supplementary Note 1, the term "hypothetical" here refers to the fact that the information delivered by corrupted QKD modules cannot be trusted. Similarly, λ_j is the public syndrome information required for the reconciliation of the j-th pair of raw keys, and $\hat{\epsilon}_{cor} = \epsilon_{cor} - \epsilon_{AU}$ for a pre-agreed authentication error ϵ_{AU}, such that $\epsilon_{AU} < \epsilon_{cor}$ and $\epsilon_{AU} < \epsilon_{sec}$. Lastly, ϵ_{PA} is the error probability of the PA step and $\delta > 0$ is arbitrary, such that

$$\epsilon_{sec} \geq 2\varepsilon + \delta + \epsilon_{PA} + \epsilon_{AU}. \tag{2}$$

As one would expect, from Eq. (1) we see that, if a single honest QKD pair exists, "a single key" can be extracted from all n_q raw keys in the AC model. Notably, the generalization of Eq. (1) to $n_q - t_q > 1$ is straightforward.

Now, let us address the alternative models, PC, AN and PN. As long as the malicious QKD pairs are collaborative, an omniscient Eve could learn all the information they hold about the keys, and as long as they are active, they can deliver untrustworthy protocol information unsuitable for correct PE. Hence, although for different reasons, the intermediate scenarios PC and AN cannot lead to an enhancement of the secret key length with respect to the AC model: they also require to remove all the key material that comes from corrupted QKD pairs via PA, thus demanding $n_q > t_q$ as well. In particular, the extractable key length for $n_q = t_q + 1$ in the PC (AN) corruption model is given by Eq. (1) too.

In the PN corruption model, one assumes an independent Eve per malicious QKD pair that does not collaborate with the eavesdroppers possibly controlling the other pairs. Moreover, passivity implies that corrupted pairs deliver trustworthy protocol information that allows to quantify the "ignorance" (in secret bits) that the Eves possibly corrupting other pairs have about their raw data. Thus, it suffices to remove the information held by the most knowledgeable eavesdropper via PA in order to provide security against all of them. As a consequence, secure QKD is possible even if all the QKD pairs are corrupted in the PN model, that is, even if $n_q = t_q$. In this setting, one can show that the ϵ_{cor}-correct, ϵ_{sec}-secret key length extractable via one-step PA (see Supplementary Note 2) is given by

$$l = \left\lfloor \min_v \sum_{j\neq v}^{n_q} \left\{H_{min}^\varepsilon(s_B^j|E_v) - \lambda_j\right\} - \log_2\left(\frac{1}{\hat{\epsilon}_{cor}\epsilon_{PA}^2\delta^{n_q-1}}\right)\right\rfloor, \tag{3}$$

where $H_{min}^\varepsilon(s_B^j|E_v)$ denotes the ε-smooth min-entropy of Bob's j-th sifted key, s_B^j, conditioned on the information E_v held by the v-th eavesdropper (i.e. the one that corrupts the v-th QKD pair, with $v = 1, \ldots, n_q$). The remaining parameters were introduced in Eq. (1), and the secrecy parameter now satisfies

$$\epsilon_{sec} \geq (n_q - 1)(2\varepsilon + \delta) + \epsilon_{PA} + \epsilon_{AU}. \tag{4}$$

Remarkably, Eq. (3) trivially outperforms Eq. (1) for any given $t_q > 1$ (and we recall that non-collaboration is only defined in this case).

Table 1. Conditional VSS with non-mixed corruption.

n_c, R, r	Active	Passive
Collaborative	$n_c = 3t_c + 1$ $R = 2t_c + 1$ $r = \binom{n_c-1}{t_c}$ $n_c = 2t_c + 2$	$n_c = t_c + 1$ $R = 1$ $r = 1$ $n_c = 2$
Non-collaborative	$R = 2t_c + 1$ $r = n_c - 1$	$R = 1$ $r = 1$

Minimum resources of a distributed QKD post-processing protocol based on conditional VSS, depending on the corruption model of the CP units. While n_c is the total number of units per party, R is the redundancy of each raw key share and r is the number of key shares managed per CP unit from each of its local QKD modules. The number t_c of possibly corrupted units per lab is at least two for the non-collaborative models (AN and PN), as non-collaboration is only defined in this case.

In what follows, we discuss the CP units. Although the corruption model of the CP units does not affect the extractable key length, l, it determines the necessary resources to securely implement a distributed post-processing using conditional VSS: the number of units per party, n_c, the number R of copies per share of raw key to be delivered by any given QKD module, and the total number of raw key shares managed per CP unit, say r, originating from a given QKD module. On the one hand, n_c and R determine the necessary classical communications both between labs and inside each lab, and the total authentication cost of the former, say l_{AU}. On the other hand, r strongly affects the post-processing time, a usual concern in the performance of QKD. In Table 1 we list the minimum values of n_c, R and r required for distributed QKD post-processing, depending on the corruption model of the CP units.

The entries of the table follow from the requirements of conditional VSS and are established in Proposition 1 in the "Methods" section (see Supplementary Note 3 for a proof of this proposition). As we observe, all the restricted models allow to reduce the resources with respect to the AC model. For instance, note that the number r of shares per unit grows exponentially with n_c for a fixed fraction of corrupted units in the AC model. This might lead to prohibitively long post-processing times even for small values of n_c. Nevertheless, this problem disappears if one assumes that the possibly corrupted units are non-collaborative, thus moving to the AN model. Also in this model, it is worth noting that conditional VSS tolerates $n_c = 2t_c + 2$, while standard VSS would still require $n_c = 3t_c + 1$, a constraint imposed by the necessity to allow for simulated broadcast.

Within the passive models (PC and PN), the distributed post-processing has the extra advantage that the PE and the lab-to-lab classical communications can be conducted by a single CP unit per lab. On the contrary, the active models require the participation of $R = 2t_c + 1$ units per lab for these tasks, in order to assure the presence of a majority of honest units.

Remarkably, in the "Methods" section, we formulate a distributed QKD post-processing protocol adequate for all the corruption models, matching the entries of Table 1 in each case. The security of this protocol, established in Proposition 3 (see section "Distributed QKD post-processing protocol"), is proven in Supplementary Note 4 combining conditional VSS with a standard QKD security analysis.

Lastly, as stated above, the corruption model of the CP units also determines the authentication cost, l_{AU}, of the distributed post-processing. The classical communications require to select R distinct CP_{A_i} and R distinct $CP_{B_{i'}}$, such that each of the former pre-shares a dedicated pool of secret key bits with each of the latter

for authentication purposes. Thus, denoting the common size of every *key pool* by $|k|$, it follows that

$$l_{\mathrm{AU}} = R^2 \times |k|, \tag{5}$$

where R is given in Table 1 for each model. A possible estimation of $|k|$ using a typical authentication scheme[33] is presented in Supplementary Note 5. Within this scheme, the authentication cost of a message scales logarithmically with its length, suggesting that for most practical situations $l_{\mathrm{AU}} \ll l$, as we shall corroborate in the next section.

Performance evaluation

To complete the analysis, we calculate explicit secret key rates in various significant corruption models, and in the finite key regime. The secret key rate is defined as

$$K = \frac{l - l_{\mathrm{AU}}}{n_{\mathrm{q}} N}, \tag{6}$$

where we recall that l (l_{AU}) is the extractable secret key length (authentication cost) and n_{q} (N) is the number of QKD pairs (number of transmission rounds per pair). For illustration purposes, l_{AU} is computed according to the classical communications of the distributed post-processing presented in the "Methods" section.

For concreteness, we assume the same corruption model for the QKD modules and the CP units, a natural supposition in practice. Moreover, we restrict ourselves to the extreme corruption models, AC and PN, as the intermediate scenarios (AN and PC) do not allow to enhance the secret key rate, disregarding the authentication cost (see section "Alternative corruption models"). We also assume that Alice and Bob use the minimum number of devices that allows for $K > 0$, which depends on the corruption model they consider. For AC corruption, this means that they agree on the number $t_{\mathrm{q}} \geq 0$ ($t_{\mathrm{c}} \geq 0$) of malicious QKD pairs (CP units per lab) they want to be protected against, and use $n_{\mathrm{q}} = t_{\mathrm{q}} + 1$ pairs ($n_{\mathrm{c}} = 3t_{\mathrm{c}} + 1$ units per lab). Alternatively, for PN corruption, they use $n_{\mathrm{q}} = 2$ QKD pairs and $n_{\mathrm{c}} = 2$ CP units per party, which suffices to achieve $K > 0$ even if all the devices are possibly malicious (see section "Alternative corruption models").

We consider two practical QKD protocols with decoy states: an efficient measurement-device-independent quantum key distribution (MDI-QKD) scheme[34] with three decoy intensities in the basis X (devoted to PE) and one signal intensity in the basis Z (devoted to key distillation), and the standard decoy-state BB84 scheme[35] with three decoy intensities per basis. Detailed analyses of these protocols are provided in Supplementary Notes 6 and 7, respectively. For each protocol, we compute estimates of l (given by Eq. (1) for the AC model and by Eq. (3) for the PN model) and l_{AU} (given by Eq. (5)), by setting the observables to their expected values according to respective channel models described in the cited Supplementary Notes. These channel models depend on various common experimental parameters: the efficiency of the photo-detectors, set to $\eta_{\mathrm{det}} = 65\%$, their dark count probability, set to $p_{\mathrm{d}} = 7.2 \times 10^{-8}$ (both values matching the recent MDI-QKD experiment reported in ref. [36]) and the polarization misalignment, set to say $\delta_{\mathrm{mis}} = 0.08$, for illustration purposes. Moreover, in both the MDI-QKD and the BB84 schemes, the weakest decoy intensity is set to $\omega = 10^{-3}$ for the numerics. In each case, we optimize the remaining protocol inputs (i.e. intensity settings, and basis and decoy probabilities) to maximize K as a function of the channel loss between Alice and Bob.

For the finite key analysis, we select a post-processing block size of M bits. Then, for every value of the channel loss, we choose the smallest number of transmission rounds per QKD pair, N, that assures that all n_{q} sifted keys reach this block size except with a probability of, say $\gamma_{\mathrm{sift}} = 5 \times 10^{-3}$, according to the channel model.

Regarding the error correction (EC) leakage, we assume the typical model $\left|sy(s_{\mathrm{B}}^{j})\right| = M f_{\mathrm{EC}} h(E_{\mathrm{tol}})$ for every EC syndrome, where $f_{\mathrm{EC}} = 1.16$ is the efficiency of the EC protocol, $h(\cdot)$ is the binary entropy function, and E_{tol} is a pre-fixed threshold quantum bit error rate (QBER). In particular, E_{tol} is an upper bound on the QBER that any pair of sifted keys can reach according to the channel model, except with an error probability of $\gamma_{\mathrm{EC}} = 5 \times 10^{-3}$.

Finally, the security parameters are set to $\epsilon_{\mathrm{cor}} = \epsilon_{\mathrm{sec}} = 10^{-8}$ and $\epsilon_{\mathrm{AU}} = 5 \times 10^{-9}$. As shown in Supplementary Note 5, ϵ_{AU} determines the individual authentication error probability γ_{AU} via $\epsilon_{\mathrm{AU}} = (t_{\mathrm{c}} + 1)^2 (n_{\mathrm{q}} + 1)\gamma_{\mathrm{AU}}$ ($\epsilon_{\mathrm{AU}} = (n_{\mathrm{q}} + 1)\gamma_{\mathrm{AU}}$) in the presence of actively (passively) corrupted CP units. Given ϵ_{sec} and ϵ_{AU}, the remaining parameters, ϵ_{PA} and δ, entering the extractable key length, l, are determined by imposing a common value, γ_{sec}, for every error term that contributes to $\hat{\epsilon}_{\mathrm{sec}} = \epsilon_{\mathrm{sec}} - \epsilon_{\mathrm{AU}}$ (given by Eqs. (2) and (4)). In particular, from the PE procedure presented in Supplementary Note 6 (Supplementary Note 7), it follows that $\gamma_{\mathrm{sec}} = \hat{\epsilon}_{\mathrm{sec}}/48$ ($\gamma_{\mathrm{sec}} = \hat{\epsilon}_{\mathrm{sec}}/20$) in the MDI-QKD (BB84) scheme within both the AC and the PN scenarios, where we used the fact that $n_{\mathrm{q}} = 2$ in the latter case.

Adhering to all the above, in Fig. 2, we plot the secret key rate as a function of the total channel loss for the MDI-QKD scheme,

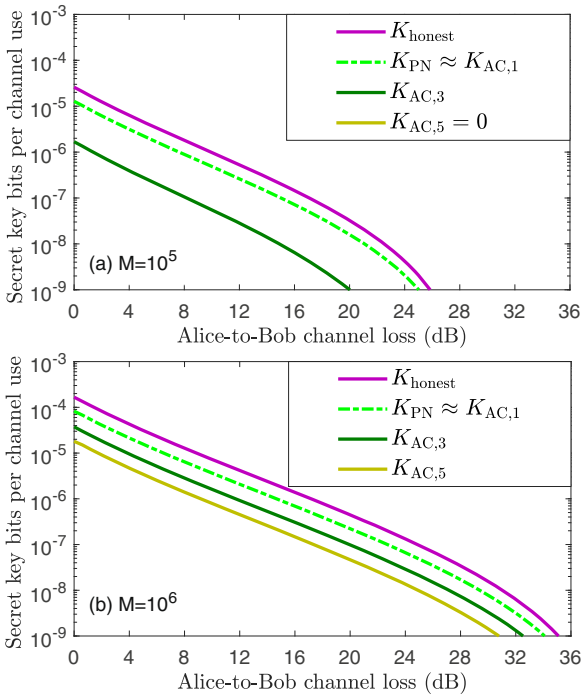

Fig. 2 Performance evaluation. Secret key rate, K of a decoy-state MDI-QKD scheme[34] in various adversarial scenarios with malicious devices, as a function of the total channel loss between Alice and Bob (assumed to be at the same distance of the untrusted measurement node). Two finite block sizes are considered, **a** $M = 10^5$ and **b** $M = 10^6$, and the authentication cost is computed according to the distributed post-processing protocol of the "Methods" section. In both figures, the purple line is the secret key rate in the standard scenario—where each party holds one QKD module and one classical post-processing (CP) unit, both of them trusted—and green lines denote different corruption models. In particular, the dashed-dotted phosphorescent line is the secret key rate assuming passive and non-collaborative corrupted devices, which requires the use of two QKD pairs and two CP units per lab (all of them being possibly malicious) to provide security. A more conservative scenario is represented by the solid non-phosphorescent green lines, which assume active and collaborative corrupted devices. These lines further assume the same number, say t, of malicious QKD pairs and malicious CP units per lab, which requires the use of at least $n_{\mathrm{q}} = t + 1$ QKD pairs and $n_{\mathrm{c}} = 3t + 1$ CP units per party to provide security. Specifically, the dark (light) green line corresponds to $t = 3$ ($t = 5$).

considering that Alice and Bob are at the same distance of the central untrusted node. Similarly, the secret key rate of the BB84 scheme is plotted in Supplementary Fig. 4. In both cases, for illustration purposes, two different block sizes are considered, $M \in \{10^5, 10^6\}$. Within the AC corruption model, for concreteness, we only address the symmetric case $t_q = t_c = t$, such that $n_q = t + 1$ and $n_c = 3t + 1$. Hence, we use the notation $K_{AC,t}$ ($l_{AC,t}$) for the secret key rate (length) secure against t corrupted devices of each kind in this model. Similarly, K_{PN} (l_{PN}) denotes the secret key rate (length) in the PN model, which, as explained above, unambiguously requires $n_q = n_c = 2$. Lastly, K_{honest} (l_{honest}) denotes the secret key rate (length) in the standard situation where each party holds one trusted QKD module and one trusted CP unit, that is, $K_{honest} = K_{AC,0}$ ($l_{honest} = l_{AC,0}$).

The conclusions gathered from Fig. 2 are readily understood in view of the results of section "Results". In the first place, for both $M = 10^5$ and $M = 10^6$, we find that $K_{PN} \approx K_{AC,1}$ to a precision that cannot be distinguished in the figure. This follows from the fact that, in both cases, two raw keys are generated (as $n_q = 2$) and the parties need to remove the information from one of them via PA. Indeed, comparing Eqs. (1) and (2) with Eqs. (3) and (4), one observes that

$$l_{PN} = l_{AC,1}, \tag{7}$$

that is, the secret key lengths coincide exactly for fixed security parameters, fixed experimental inputs (N and E_{tol}) and average observables. Thus, the minuscule difference between K_{PN} and $K_{AC,1}$ comes from the authentication cost, as $l_{AU} \propto R^2$ with $R = 2t + 1$ ($R = 1$) in the AC (PN) model.

The same argument relates $K_{AC,t}$ and $K_{honest}/(t + 1)$ for all t. On the one hand, for the specifications above,

$$l_{AC,t} = l_{honest} \tag{8}$$

for all t, which corresponds to the key material coming from the honest QKD pair. On the other hand, in the presence of t malicious QKD pairs, the extraction of the above key length requires the generation of $t + 1$ raw keys in the AC model. Thus, from Eq. (6), it follows that

$$\frac{K_{honest}}{t+1} - K_{AC,t} = \frac{\delta l_{AU}}{(t+1)N} \tag{9}$$

in the simulations, where δl_{AU} denotes the extra authentication cost of the AC model with $t_q = t_c = t$, compared to the honest scenario. Due to the factor N^{-1} in the right-hand side of Eq. (9), larger block sizes lead to smaller differences $K_{honest}/(t + 1) - K_{AC,t}$.

Finally, since $K_{AC,t} \propto (l_{honest} - l_{AU})$ and $l_{AU} \propto (2t + 1)^2$ in the AC model, $K_{AC,t}$ vanishes for any given block size if a large enough number of CP units is considered, as eventually $l_{AU} > l_{honest}$. This is the case for $M = 10^5$ and $t = 5$ in Fig. 2.

DISCUSSION

QKD security today requires every QKD component to be honest and follow the protocol steps. Nevertheless, our experience in classical cryptography indicates that this might be very hard to certify in practice. Even in the DI setting, where the QKD devices are often referred to as uncharacterized black boxes, it is mandatory to assure that, beyond the reception of quantum signals from an untrusted source, the only interaction these boxes have with the outside world is the exchange of inputs and outputs with the legitimate parties. This assumption, despite weak, is still very hard to verify. Fortunately, as pointed out in ref. [20], one can protect QKD against malicious equipment by using redundant devices to combine VSS with PA, an approach that we follow in this work.

However, a major limitation of the proposal in ref. [20] is that it relies on simulated broadcast, a very high-priced task in terms of total communication, especially for large numbers of CP units. What is more, the scheme presented in ref. [20] requires the execution of $n_q + 1$ PA steps, where n_q is the total number of QKD pairs. In this work, we eliminate the limitation of simulated broadcast and show that a single PA step suffices, thus turning the approach in ref. [20] into practical.

Moreover, the proposal in ref. [20] assumes that the malicious devices may actively deviate from the protocol and collaborate with each other, which is probably over-pessimistic. For instance, an archetypical security breach consists of a malicious item implanted by an eavesdropper in an honest apparatus, leading to a passively corrupted device that may leak private information but sticks to the protocol prescriptions. Likewise, if the devices originate from different vendors, it is reasonable to expect that possibly corrupted apparatuses do not collaborate. In this work, we show that very natural assumptions like these allow to achieve a better performance than the AC model, both in terms of secret key rate and necessary resources.

Also, it is often stated in the QKD community that one could simply bitwise add modulo 2 (XOR) the final keys generated by different QKD systems to defeat malicious equipment. Although this alternative may assure the privacy of the output, it has the major problem of generally requiring more devices than actually necessary to establish security, due to the non-distributed post-processing. For instance, note that not only the QKD module but also the CP unit in any given QKD system learns the raw key in the XOR approach, leading to a double-trouble situation where one must contemplate the worst possible combination of modules and units to guarantee the privacy of the raw key material. Similarly, the XOR approach does not prevent an actively malicious unit from jeopardizing the post-processing of the raw key generated by its module.

Furthermore, we would like to note that secret-sharing-type techniques are, in fact, the standard tool to guarantee security against untrusted devices. For instance, it is the adopted solution in modern hardware secure modules[37–39]. Likewise, similar ideas to those we present here may be deployed in QKD to relax the security assumptions in trusted node network architectures, such that one can establish the security of the final keys even if some intermediate nodes are compromised[40].

Another contribution of this work is to evaluate the finite secret key rate of practical QKD schemes in the presence of malicious devices, for different corruption models and accounting for the authentication cost of the redundant classical communications. Particularly, based on our theoretical results, we devise an efficient distributed QKD post-processing protocol adequate for all the corruption models we examine. The simulations confirm that our techniques may achieve finite secret key rates comparable to those of standard QKD with trusted devices. Putting it all together, this work is a fundamental step towards the development of practical QKD systems secure against malicious devices possibly sabotaged by a third party, a major threat against classical cryptography today that cannot be put aside in the quantum-safe era.

METHODS

Conditional VSS

Here, we introduce a modified version of the VSS scheme presented in ref. [26] that contemplates the possibility of aborting, thus providing a weaker cryptographic primitive than standard VSS. For this reason, we refer to it as conditional VSS.

We consider a scenario with one possibly dishonest dealer, D, and a set of n parties, $\mathbb{P} = \{P_1, \ldots, P_n\}$, t of which are possibly corrupted. In this setting, a conditional VSS scheme is a pair of protocols, (Share, Reconstruct), satisfying three properties: *privacy*, *conditional commitment* and *conditional correctness* (defined below). In full generality, Share and Reconstruct run as follows. During Share, D distributes an input m among the n parties, which pairwise perform consistency tests on their common information via secure channels and possibly abort. Upon non-abortion of

Share, during Reconstruct the parties collaborate to retrieve m. The defining properties of conditional VSS are given below:

1. *Privacy:* If D is honest, the information obtained by any set of t or less parties prior to Reconstruct is independent of m.
2. *Conditional commitment*: Upon non-abortion of Share, Reconstruct yields the same output for all non-actively corrupted parties.
3. *Conditional correctness*: Upon non-abortion of Share, if D is honest the common output of all non-actively corrupted parties is the input m.

Regarding the parties, all four non-mixed corruption models presented in the main text shall be addressed: AC, AN, PC and PN. However, we do not restrict to any of them yet. Also, note that the set of non-actively corrupted parties includes all the parties (and not only the honest ones) in the passive models. As for the dealer, D is said to be dishonest if it may distribute incorrect/inconsistent information about his input to the parties or directly reveal it to them. In particular, this means that if the QKD modules belong to the PN model, even corrupted modules are honest dealers.

In what follows, we describe a pair of protocols, (Share, Reconstruct), that depend on various settings, and such that adequate choices of these settings confer the pair the category of a conditional VSS scheme. We remark that the adequacy of some given settings depends on the corruption model one assumes for the parties. Also, the protocol definitions below assume that the parties and the dealer do not misbehave, whether or not these protocols are robust against active corruption. The dealer's input m is assumed to be a binary string, and we recall that the symbol "$\oplus$" denotes bitwise XOR. In addition, this operation is generalized to a pair of strings with different lengths by padding the shortest one with as many zeros as necessary for the lengths to match. This said, Share runs as follows:

1. D uses a q-out-of-q SS scheme to split a message m into q random shares, by selecting the first $q-1$ shares m_i at random and then choosing $m_q = m \oplus m_1 \oplus \ldots \oplus m_{q-1}$.
2. For $i = 1, \ldots, q$, D sends m_i to all the parties in a certain subset, say $\sigma_i \subsetneq \mathbb{P}$, via secure channels. If any of these parties does not receive the share, she takes a zero bit string as default share.
3. If $|\sigma_i| > 1$, all pairs of parties in σ_i perform a consistency test: they send each other their copies of m_i over secure channels to check if they are equal. If any party finds an inconsistency, she aborts the protocol.

Importantly, abortion proceeds in two steps: the aborting party sends an abortion order to all other parties, and each receiving party resends the order to all the rest. Upon reception of an abortion order, the parties abort. Step two assures that the non-actively corrupted parties always abort collectively. Upon non-abortion of Share, Reconstruct runs as follows:

1. All pairs of parties send each other their shares through authenticated channels.
2. For $i = 1, \ldots, q$, each party uses MV to reconstruct the share m_i, and then obtains $m = \oplus_{i=1}^{q} m_i$.

In general, in order for MV to be well defined, the output must be set to a default value in case of a tie. Nevertheless, ties never occur for the adequate choices of the parameters n and q and the subsets σ_i we present next.

Proposition 1 *Let t be the maximum number of corrupted parties, and let $\{T_1, \ldots, T_{\binom{n}{t}}\}$ be any ordered list of all possible combinations of t parties. Under the following settings, (Share, Reconstruct) defines a conditional VSS scheme*:

1. $n = 3t+1$, $q = \binom{n}{t}$ *and* $\sigma_i = \mathbb{P}/T_i$ *(AC corruption).*
2. $n = 2t+2$, $q = n$ *and* $\sigma_i = \mathbb{P}/P_i$ *(AN corruption).*
3. $n = t+1$, $q = n$ and $\sigma_i = P_i$ *(PC corruption).*
4. $n = 2$, $q = n$ and $\sigma_i = P_i$ *(PN corruption).*

What is more, the above settings are optimal in the number of parties.

The reader is referred to Supplementary Note 3 for a proof of Proposition 1. Also, note that, by definition of R (see section "Alternative corruption models"), we have that $R = |\sigma_i|$ for all i.

Finally, we remark that the above conditional VSS scheme enables secure multiparty computation of linear functions of the shared private input in a very simple way. Let $L(\cdot)$ be the linear function to be computed on m. Upon non-abortion of Share, each party applies L to its shares of m, in so obtaining shares of $L(m)$. Since this step requires null communication, *privacy*, *conditional commitment* and *conditional correctness* are trivially maintained.

Generation of random bit strings (RBSs)

Distributed QKD post-processing also relies on the possibility to generate unbiased random bit strings (RBSs) of a pre-fixed length L among n parties, when up to t of them are possibly corrupted. Here, we describe an RBS generation protocol suitable for the active corruption models, AC and AN, that builds on conditional VSS to safeguard the randomness of its output string (the passive models shall be addressed afterwards).

Let us set the total number of parties, n, the total number of shares, q, and the subsets of parties, σ_i, as specified in Proposition 1 for the considered model (AC or AN). The RBS generation protocol runs as follows:

1. For $k = 1, \ldots, t+1$, P_k creates a random L-bit string, R_k, and distributes it among all n parties (including itself) using Share. If, for some k, Share aborts, the RBS generation protocol aborts. If a party receives any share whose length differs from L, she aborts.
2. Upon non-abortion of step 1, the parties use Reconstruct to obtain R_k for all $k = 1, \ldots, t+1$. Then, each of them individually calculates $R = \oplus_{k=1}^{t+1} R_k$.

Proposition 2 *The RBS generation protocol outputs a common random L-bits string for all non-actively corrupted parties.*

The reader is referred to Supplementary Note 3 for a proof of Proposition 2.

Finally, using the standard notion of passivity given in the main text, one can avoid the use of conditional VSS for RBS generation in the passive models (PC and PN). Instead, any given unit can generate the strings directly, and such strings are truly random by assumption.

Distributed QKD post-processing protocol

Making use of our theoretical results, here we present a distributed QKD post-processing protocol based on conditional VSS that is appropriate for all non-mixed corruption models introduced in section "Results". We refer to it simply as Protocol.

In the first place, the parties agree on the corruption models they assume for the QKD modules and the CP units (which might be different in general), and also select the numbers t_q and t_c of corrupted devices they want to be protected against. In case they choose AC, AN or PC corruption (PN corruption) for the modules, they must hold $n_q = t_q + 1$ ($n_q = 2$) QKD pairs in total—given that they stick to the rule of using the minimum valid amount of devices—and the secret key length l is given by Eq. (1) (Eq. (3)). Similarly, they provide themselves with as many CP units as specified in Table 1 for their preferred model. Coming next, they agree on a correctness (secrecy) parameter, ϵ_{cor} (ϵ_{sec}), and a total authentication error ϵ_{AU}, such that $\epsilon_{\text{AU}} < \epsilon_{\text{cor}}$ and $\epsilon_{\text{AU}} < \epsilon_{\text{sec}}$.

For $j = 1, \ldots, n_q$, the pair $(\text{QKD}_{\text{A}_j}, \text{QKD}_{\text{B}_j})$ runs a QKD session to generate the basis Z raw key strings, $(r_{\text{A}}^j, r_{\text{B}}^j)$, to be kept private, and some non-private protocol information, $(\text{info}_{\text{A}}^j, \text{info}_{\text{B}}^j)$, typically including the basis and intensity settings, detection events and so on. Crucially, $(\text{info}_{\text{A}}^j, \text{info}_{\text{B}}^j)$ includes all the raw key material required for PE. The post-processing procedure (namely, Protocol) starts next and is described below. Although the description assumes that the possibly corrupted devices do not deviate from the protocol steps, Protocol is indeed secure against active eavesdroppers, as established in Proposition 3 below. Finally, although not explicitly stated, in case of abortion, the aborting party must notify the other party. This said, Protocol runs as follows.

Let us focus on, say, the j-th QKD pair:

1. *Distribution of data*: QKD_{A_j} (QKD_{B_j}) distributes shares of its raw key r_{A}^j (r_{B}^j) among the CP_{A_i} following the Share protocol of a conditional VSS scheme (see section "Conditional VSS") for the selected corruption model of the CP units. We denote the set of units that receive the i-th share of r_{A}^j (r_{B}^j) by σ_i^{A} (σ_i^{B}), which without loss of generality is common for all $j = 1, \ldots, n_q$. In addition, QKD_{A_j} (QKD_{B_j}) sends the protocol information info_{A}^j (info_{B}^j) to all $\text{CP}_{\text{A}_i} \in \sigma_1^{\text{A}}$ ($\text{CP}_{\text{B}_{i'}} \in \sigma_1^{\text{B}}$), and the latter perform a consistency test on this data: they pairwise check that their copies of info_{A}^j (info_{B}^j) match via authenticated channels. If a CP_{A_i} ($\text{CP}_{\text{B}_{i'}}$) finds an inconsistency, it aborts the protocol (see the Share protocol in the section devoted to conditional VSS for the two-step abortion procedure we consider).
2. *Sifting*: Each $\text{CP}_{\text{A}_i} \in \sigma_1^{\text{A}}$ sends its copy of info_{A}^j to all $\text{CP}_{\text{B}_{i'}} \in \sigma_1^{\text{B}}$, which individually apply majority voting (MV) to decide on a single copy.

Then, the $\mathrm{CP}_{\mathrm{B}_{i'}} \in \sigma_1^{\mathrm{B}}$ forward some sifting information, sift^j, computable from the pair $(\mathrm{info}_{\mathrm{A}}^j, \mathrm{info}_{\mathrm{B}}^j)$, to the $\mathrm{CP}_{\mathrm{B}_{i'}} \notin \sigma_1^{\mathrm{B}}$, which apply MV too. Using sift^j, every $\mathrm{CP}_{\mathrm{B}_{i'}}$ discards some key bits from their shares of r_{B}^j to obtain shares of the sifted key, s_{B}^j. Alternative sifting schemes that require to discard random subsets of the data could easily be adapted by including an RBS generation protocol (see section "Generation of random bit strings (RBSs)").

3. *PE*: Using $(\mathrm{info}_{\mathrm{A}}^j, \mathrm{info}_{\mathrm{B}}^j)$, each $\mathrm{CP}_{\mathrm{B}_{i'}} \in \sigma_1^{\mathrm{B}}$ computes a hypothetical lower bound h_ε^j (see Supplementary Notes 1 and 2 for the details) on the ε-smooth min-entropy of s_{B}^j conditioned on the information held by an eavesdropper up to the PE step, for a certain ε that depends on the PE procedure.

Once steps 1 to 3 are implemented for $j = 1, \ldots, n_{\mathrm{q}}$, all $\mathrm{CP}_{\mathrm{B}_{i'}}$ construct their shares of the concatenated sifted key $s_{\mathrm{B}} = [s_{\mathrm{B}}^1, \ldots, s_{\mathrm{B}}^{n_{\mathrm{q}}}]$, such that the kth share of s_{B} is simply given by the concatenation of the kth share of s_{B}^1, the kth share of s_{B}^2 and so on. In addition, from all n_{q} values h_ε^j, every $\mathrm{CP}_{\mathrm{B}_{i'}} \in \sigma_1^{\mathrm{B}}$ computes a lower bound l on the secret key length extractable from s_{B} via PA. If a $\mathrm{CP}_{\mathrm{B}_{i'}} \in \sigma_1^{\mathrm{B}}$ finds $l \leq 0$, it aborts the protocol. Otherwise, the post-processing proceeds as follows:

4. *RBS generation:* Every $\mathrm{CP}_{\mathrm{B}_{i'}} \in \sigma_1^{\mathrm{B}}$ forwards l to the $\mathrm{CP}_{\mathrm{B}_{i'}} \notin \sigma_1^{\mathrm{B}}$, which apply MV. All $\mathrm{CP}_{\mathrm{B}_{i'}}$ perform an RBS generation protocol to select two random 2-universal hash functions, h_{EV} and h_{PA}, respectively devoted to error verification (EV) and PA.
5. *Information reconciliation*: Every $\mathrm{CP}_{\mathrm{B}_{i'}}$ computes its shares of the string of concatenated syndromes, $sy_{\mathrm{B}} = [sy(s_{\mathrm{B}}^1), \ldots, sy(s_{\mathrm{B}}^{n_{\mathrm{q}}})]$, and the EV tag $h_{\mathrm{EV,B}} = h_{\mathrm{EV}}(s_{\mathrm{B}})$. Here, $sy(\cdot)$ is a linear function specified by an EC protocol for a pre-agreed QBER. Altogether, the $\mathrm{CP}_{\mathrm{B}_{i'}}$ reconstruct sy_{B} and $h_{\mathrm{EV,B}}$ via the Reconstruct protocol of a conditional VSS scheme (see section "Conditional VSS"). Each $\mathrm{CP}_{\mathrm{B}_{i'}} \in \sigma_1^{\mathrm{B}}$ sends the following items to every $\mathrm{CP}_{\mathrm{A}_i} \in \sigma_1^{\mathrm{A}}$:
   (a) The total sifting information, $\{\mathrm{sift}^j\}_{j=1}^{n_{\mathrm{q}}}$.
   (b) The syndrome information, sy_{B}, a description of h_{EV} and the EV tag, $h_{\mathrm{EV,B}}$.
   (c) A description of h_{PA}.

For all three items, each $\mathrm{CP}_{\mathrm{A}_i} \in \sigma_1^{\mathrm{A}}$ applies MV to decide on a single copy. Then, it forwards $\{\mathrm{sift}^j\}_{j=1}^{n_{\mathrm{q}}}$, h_{EV} and h_{PA} to the $\mathrm{CP}_{\mathrm{A}_i} \notin \sigma_1^{\mathrm{A}}$ (which apply MV too), and every $\mathrm{CP}_{\mathrm{A}_i}$ sifts its shares of the raw keys r_{A}^j to obtain shares of the concatenated sifted key $s_{\mathrm{A}} = [s_{\mathrm{A}}^1, \cdots, s_{\mathrm{A}}^{n_{\mathrm{q}}}]$. Following the EC protocol, all $\mathrm{CP}_{\mathrm{A}_i}$ compute their shares of the concatenated syndrome string, $sy_{\mathrm{A}} = [sy(s_{\mathrm{A}}^1), \ldots, sy(s_{\mathrm{A}}^{n_{\mathrm{q}}})]$, and jointly reconstruct it via the Reconstruct protocol of a conditional VSS scheme. From sy_{B} and sy_{A}, each $\mathrm{CP}_{\mathrm{A}_i} \in \sigma_1^{\mathrm{A}}$ computes the error pattern $\hat{e}$ and updates its copy of the first share of s_{A} by XOR-ing it with $\hat{e}$. Thus, by construction, Alice's corrected key is $\hat{s}_{\mathrm{A}} = s_{\mathrm{A}} \oplus \hat{e}$. Then, all $\mathrm{CP}_{\mathrm{A}_i}$ compute their shares of the EV tag $h_{\mathrm{EV,A}} = h_{\mathrm{EV}}(\hat{s}_{\mathrm{A}})$ and jointly reconstruct it via the Reconstruct protocol of a conditional VSS scheme. Finally, every $\mathrm{CP}_{\mathrm{A}_i} \in \sigma_1^{\mathrm{A}}$ checks that $h_{\mathrm{EV,A}} = h_{\mathrm{EV,B}}$. Otherwise, it aborts the protocol.

6. *PA*: In case of not aborting, every $\mathrm{CP}_{\mathrm{A}_i}$ ($\mathrm{CP}_{\mathrm{B}_{i'}}$) computes its shares of the final key $k_{\mathrm{A}} = h_{\mathrm{PA}}(\hat{s}_{\mathrm{A}})$ ($k_{\mathrm{B}} = h_{\mathrm{PA}}(s_{\mathrm{B}})$).

In Supplementary Note 4, we prove that the following security claim holds for all (non-mixed) corruption models of the QKD modules and the CP units.

Proposition 3 *Suppose that Protocol does not abort. Then, Alice and Bob can unambiguously determine unique ϵ_{cor}-correct and ϵ_{sec}-secret final keys.*

Importantly, the determination of such final keys by Alice and Bob can be done by simply applying MV on the key shares held by their respective CP units, followed by an XOR operation. More generally, in the presence of actively corrupted units, the $\mathrm{CP}_{\mathrm{A}_i}$ ($\mathrm{CP}_{\mathrm{B}_{i'}}$) can forward their final shares to a local key management layer[41,42]. There, they could be stored in distributed memories or employed for applications such as message encryption, which in turn can be performed share-wise too.

Received: 28 June 2020; Accepted: 4 December 2020;
Published online: 05 February 2021

REFERENCES

1. Bennett, C. H. & Brassard, G. Quantum cryptography: public key distribution and coin tossing. In *Proc. IEEE International Conference on Computers, Systems & Signal Processing* 175–179 (IEEE, New York, Bangalore, 1984).
2. Scarani, V. et al. The security of practical quantum key distribution. *Rev. Mod. Phys.* **81**, 1301 (2009).
3. Lo, H.-K., Curty, M. & Tamaki, K. Secure quantum key distribution. *Nat. Photonics* **8**, 595 (2014).
4. Xu, F., Ma, X., Zhang, Q., Lo, H.-K. & Pan, J.-W. Secure quantum key distribution with realistic devices. *Rev. Mod. Phys.* **92**, 025002 (2020).
5. Diffie, W. & Hellman, M. New directions in cryptography. *IEEE Trans. Inf. Theory* **22**, 644–654 (1976).
6. Rivest, R. L., Shamir, A. & Adleman, L. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* **21**, 120–126 (1978).
7. Gligor, V. D. *A Guide to Understanding Covert Channel Analysis of Trusted Systems*, Vol. 30 (National Computer Security Center, 1994).
8. Zander, S., Armitage, G. & Branch, P. A survey of covert channels and countermeasures in computer network protocols. *IEEE Commun. Surv. Tutor.* **9**, 44–57 (2007).
9. Prevelakis, V. & Spinellis, D. The Athens affair. *IEEE Spectr.* **4**, 26–33 (2007).
10. Yang, K., Hicks, M., Dong, Q., Austin, T. & Sylvester, D. A2: Analog malicious hardware. In *IEEE Symposium on Security and Privacy* 18–37 (IEEE, 2016).
11. Robertson, J. & Riley, M. The big hack: how China used a tiny chip to infiltrate US companies. *Bloomberg Businessweek* 4 (2018).
12. Adee, S. The hunt for the kill switch. *IEEE Spectr.* **45**, 34–39 (2008).
13. Becker, G. T., Regazzoni, F., Paar, C. & Burleson, W. P. Stealthy dopant-level hardware trojans. In *International Workshop on Cryptographic Hardware and Embedded Systems*, 197–214 (Springer, Berlin, Heidelberg, 2013).
14. Mayers, D. & Yao, A. C. C. Quantum cryptography with imperfect apparatus. In *Proceedings of the 39th Annual Symposium on Foundations of Computer Science*, 503–509 (1998).
15. Acín, A. et al. Device-independent security of quantum cryptography against collective attacks. *Phys. Rev. Lett.* **98**, 230501 (2007).
16. Vazirani, U. & Vidick, T. Fully device-independent quantum key distribution. *Phys. Rev. Lett.* **113**, 140501 (2014).
17. Arnon-Friedman, R., Dupuis, F., Fawzi, O., Renner, R. & Vidick, T. Practical device-independent quantum cryptography via entropy accumulation. *Nat. Commun.* **9**, 459 (2018).
18. Miller, C. A. & Shi, Y. Robust protocols for securely expanding randomness and distributing keys using untrusted quantum devices. *J. ACM* **63**, 33 (2016).
19. Barrett, J., Colbeck, R. & Kent, A. Memory attacks on device-independent quantum cryptography. *Phys. Rev. Lett.* **110**, 010503 (2013).
20. Curty, M. & Lo, H.-K. Foiling covert channels and malicious classical post-processing units in quantum key distribution. *npj Quantum Inf.* **5**, 14 (2019).
21. Li, W. et al. Experimental quantum key distribution secure against malicious devices. Preprint at https://arxiv.org/abs/2006.12863 (2020).
22. Chor, B., Goldwasser, S., Micali, S. & Awerbuch, B. Verifiable secret sharing and achieving simultaneity in the presence of faults. In *Proc. of the 26th Annual Symposium on Foundations of Computer Science (FOCS'85)*, 383–395 (IEEE Computer Society, Los Alamitos, 1985).
23. Cramer, R., Damgård, I. B. & Nielsen, J. B. *Secure Multiparty Computation and Secret Sharing* (Cambridge Univ. Press, New York, 2015).
24. Ben-Or, M., Goldwasser, S. & Wigderson, A. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, 1–10 (1988).
25. Chaum, D., Crépeau, C. & Damgard, I. Multiparty unconditionally secure protocols. In *Proceedings of the Twentieth annual ACM Symposium on Theory of computing*, 11–19 (1988).
26. Maurer, U. Secure multi-party computation made simple. *Discret. Appl. Math.* **154**, 370–381 (2006).
27. Shamir, A. How to share a secret. *Commun. ACM* **22**, 612–613 (1979).
28. Blakley, G. R. Safeguarding cryptographic keys. In *Proc. of the AFIPS 1979 National Computer Conference (NCC'79)*, 313–317 (AFIPS Press, New Jersey, 1979).
29. Mitra, S., Wong, H.-S. P. & Wong, S. Stopping hardware Trojans in their tracks. *IEEE Spectrum* **20** (2015) https://spectrum.ieee.org/semiconductors/design/stopping-hardware-trojans-in-their-tracks.
30. Bennett, C. H., Brassard, G. & Robert, J. M. Privacy amplification by public discussion. *SIAM J. Comput.* **17**, 210–229 (1988).
31. Tomamichel, M., Schaffner, C., Smith, A. & Renner, R. Leftover hashing against quantum side information. *IEEE Trans. Inf. Theory* **57**, 5524–5535 (2011).
32. Lamport, L., Shostak, R. & Pease, M. The Byzantine generals problem. *Trans. Program. Lang. Syst.* **4**, 382–401 (1982).
33. Krawczyk, H. LFSR-based hashing and authentication. In *Advances in Cryptology—CRYPTO'94, Lecture Notes in Computer Science*, Vol. 893, 129–139 (Springer, 1994).
34. Zhou, Y. H., Yu, Z. W. & Wang, X.-B. Making the decoy-state measurement-device-independent quantum key distribution practically useful. *Phys. Rev. A* **93**, 042324 (2016).
35. Lim, C. C. W., Curty, M., Walenta, N., Xu, F. & Zbinden, H. Concise security bounds for practical decoy-state quantum key distribution. *Phys. Rev. A* **89**, 022307 (2014).

36. Yin, H. L. et al. Measurement-device-independent quantum key distribution over a 404 km optical fiber. *Phys. Rev. Lett.* **117**, 190501 (2016).
37. Thales Group. nShield Solo HSMs. https://www.thalesesecurity.com/products/general-purpose-hsms/nshield-solo.
38. Gemalto. Hardware Security Modules. https://safenet.gemalto.com/dataencryption/hardware-security-modules-hsms/.
39. Amazon Web Services. AWS CloudHSM. https://aws.amazon.com/cloudhsm/ (2021).
40. Salvail, L. et al. Security of trusted repeater quantum key distribution networks. *J. Comput. Secur.* **18**, 61–87 (2010).
41. Peev, M. et al. The SECOQC quantum key distribution network in Vienna. *N. J. Phys.* **11**, 075001 (2009).
42. Sasaki, M. et al. Field test of quantum key distribution in the Tokyo QKD Network. *Opt. Express* **19**, 10387–10409 (2011).

ACKNOWLEDGEMENTS

We thank Liu Zhang Chen-Da for useful discussions on verifiable secret-sharing and secure multiparty computation. We acknowledge the support from the Spanish Ministry of Economy and Competitiveness (MINECO), the Fondo Europeo de Desarrollo Regional (FEDER) through the grant TEC2017-88243-R and the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement no. 675662 (project QCALL) for financial support. V.Z. gratefully acknowledges support from a FPU scholarship from the Spanish Ministry of Education.

AUTHOR CONTRIBUTIONS

M.C. conceived the initial idea and triggered the consideration of this research project. V.Z. made the theoretical analysis and performed the numerical simulations, with inputs from both authors. M.C. and V.Z. analysed the results and prepared the manuscript.

COMPETING INTERESTS

The authors declare no competing interests.

ADDITIONAL INFORMATION

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41534-020-00358-y.

Correspondence and requests for materials should be addressed to V.Z.

Reprints and permission information is available at http://www.nature.com/reprints

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.