2011 American Control Conference
on O'Farrell Street, San Francisco, CA, USA
June 29 - July 01, 2011

Key Establishment via Common State Information in Networked Control Systems

Husheng Li, Lifeng Lai, Seddik Djouadi and Xiao Ma

***Abstract*— Security is an important issue in networked control systems, but has not received sufficient attention. The fundamental step for realizing a security protocol for networked control systems is to establish a secret key between the sensor and controller. Traditional approaches for key establishment such as the Public Key Infrastructure (PKI) usually incur significant overhead. In this paper, the common information of the physical system state is exploited for the key establishment between the sensor and the controller. In this scheme, the controller takes an action that causes the system state to change, which can be observed by the sensor. The controller and the sensor will then exchange messages to find the common random bits in the predicted and observed system states, respectively. The secret key will be generated from the common bits. The theoretical bound for the rate of generating common bits is analyzed using information theoretic analysis. This key establishment scheme is implemented on a remote controlled inverted pendulum. Experiments show that the proposed algorithm can generate tens of common bits per second.**

I. Introduction

Networked control systems have been intensively studied for various applications such as unmanned vehicle networks and remote system state monitoring [2] [7]. Many aspects of networked control systems have been addressed, e.g., the impact of networking metrics like delay and packet loss on the observability and stability of the system, and the control strategy subject to communication delay or packet drop. However, few studies have considered the security aspect of networked control systems [1] [5] [6]. This is in a sharp contrast to the importance of security in control systems. In many cases, it is undesirable to leak the system state to an eavesdropper. Moreover, an intruder may break the information integrity and replace the sensor observations with false ones, thus causing instability of the whole system. Therefore, there is a pressing need to study the security of networked control systems. The cyber security of the smart grid [4], an example of networked control systems, has attracted significant attention [3].

A fundamental step in the security mechanism is to establish a common secret key in each transmission pair for the purposes of information privacy and message integrity. We require that the key establishment is dynamic and online, thus effectively combating the possible eavesdroppers. Traditional approaches involve a complicated mechanism of key distribution like Public Key Infrastructure (PKI), which incurs much overhead. However, if there is a common randomness between the transmission pair, which is unknown to other parties, the transmission pair can use this common randomness to establish a secret key. This approach has been applied in wireless communication systems, in which the wireless channel gain between the transmitter and receiver is used as the common randomness [9]–[12]. Such a scheme has been demonstrated by hardware experiments [8] [13] [14] [15].

H. Li, S. Djouadi and X. Ma are with the Department of Electrical Engineering and Computer Science, the University of Tennessee, Knoxville, TN, 37996 (email: husheng@eecs.utk.edu, djouadi@utk.edu, xma4@utk.edu). L. Lai is with the Department of Systems Engineering, University of Arkansas, Little Rock, AR, (email: lxlai@ualr.edu). This work was supported by the National Science Foundation under grants CCF-0830451 and ECCS-0901425.

Motivated by this common randomness based key establishment, we propose a mechanism to use the common information between a sensor and a controller, i.e., the system state, to establish the secrete key in a networked control system. In such a scheme, the controller takes an action which causes a change of the system state. The sensor reads the observations, which can also be predicted by the controller. If the noise is not strong and the prediction at the controller is precise, the observation at the sensor and the prediction at the controller should be similar. Then, both the observation and the prediction are quantized. Subsequently, a procedure of message exchange is carried out between the sensor and the controller to extract the common bits. Finally, the common bits are used for the secret key. Note that this requires an assumption that an eavesdropper is unable to measure the system state directly, which is valid in many situations. We also analyzed the theoretical bound for the common bit generation rate using information theoretic argument. The results show that common bits can still be generated even if the eavesdroppers has some partial information on the system observation.

The remainder of this paper is organized as follows. In Section II, the system model of the networked control system is introduced. The theoretical bound for the common bit generation rate is analyzed in Section III. The key establishment procedure is detailed in Section IV. An experiment, based on an inverted pendulum, is carried out in Section V. Finally, the conclusions are drawn in Section VI.

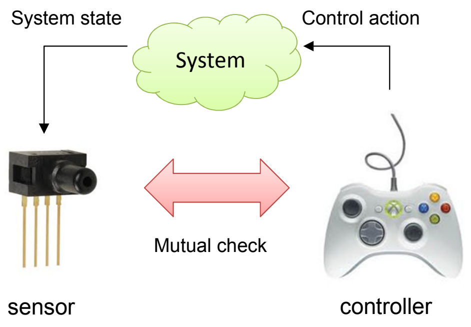

Fig. 1: An illustration of the key establishment procedure.

II. System Model

We consider the following linear dynamic system:

$$\begin{aligned} \mathbf{x}(t+1) &= \mathbf{A}\mathbf{x}(t)+\mathbf{B}\mathbf{u}(t)+\mathbf{w}(t) \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t)+\mathbf{n}(t), \end{aligned} \tag{1}$$

where $\mathbf{x}$ is an N-vector and represents the system state, $\mathbf{y}$ is an M-vector representing the observations and $\mathbf{u}$ is a K-vector action taken by the controller. Both vectors $\mathbf{w}$ and $\mathbf{n}$ are the perturbations to the linear system. We assume that the eavesdropper is also able to observe the system state via

$$\mathbf{z}(t)=\mathbf{D}\mathbf{x}(t)+\mathbf{m}(t), \tag{2}$$

in which $\mathbf{D}$ is the observation matrix for the eavesdropper and $\mathbf{m}$ is the noise.

Both the sensor and the controller will generate a key based on the signal it sends and receives. A key rate R_{key} is said to be achievable if for each $\epsilon > 0$, there exists n_0 such that for each $N \geq n_0$ we have that

$$\begin{aligned} \Pr(K_A \neq K_B) &\leq \epsilon, \\ \frac{1}{N}H(K_A) &\geq R_{key}-\epsilon, \\ \frac{1}{N}I(K_A;\mathbf{Z}) &\leq \epsilon, \quad \text{and} \\ H(K_A) &\geq \log|K_A|-\epsilon. \end{aligned} \tag{3}$$

Here, the first requirement says that the keys generated at the sensor and the controller should be the same with a high probability. The second requirement specifies the key rate. The third requirement says that the attacker obtains negligible information about the generated keys. Finally, the fourth requirement says that the generated key should be uniformly distributed.

III. Theoretical Bound

In this section, we develop theoretical bounds on the rate of keys that can be generated from a control system.

A. No Attacker Observation Case

We first consider the case in which the attacker is not able to observe the system state, i.e., $\mathbf{D}=\mathbf{0}$. In this case, one can model the dynamic between the control center and the sensor as the following communication system: the transmitter is the control center which sends $\mathbf{u}(t)$, the receiver is the sensor which receives

$$\begin{aligned} \mathbf{y}(t+1) &= \mathbf{C}\mathbf{x}(t+1)+\mathbf{n}(t+1) \\ &= \mathbf{C}(\mathbf{A}\mathbf{x}(t)+\mathbf{B}\mathbf{u}(t)+\mathbf{w}(t))+\mathbf{n}(t+1) \\ &= \mathbf{C}\mathbf{B}\mathbf{u}(t)+\mathbf{C}\mathbf{A}\mathbf{x}(t)+\mathbf{C}\mathbf{w}(t)+\mathbf{n}(t+1). \end{aligned} \tag{4}$$

In (4), the term $\mathbf{CBu}(t)$ is the signal, the term $\mathbf{Cw}(t)+\mathbf{n}(t+1)$ is the noise. The term $\mathbf{CAx}(t)$ can also be modeled as noise. However, the receiver has partial knowledge about this term obtained from

$$\mathbf{y}(t)=\mathbf{C}\mathbf{x}(t)+\mathbf{n}(t). \tag{5}$$

Hence, along serving the control purposes, the controller can use $\mathbf{u}(t)$ to send a secret key to the sensor. For an input distribution on $\mathbf{u}$, the following key rate is achievable

$$R_k = I(\mathbf{u};\mathbf{y}(t),\mathbf{y}(t+1)), \tag{6}$$

in which $I(x;y)$ denotes the mutual information between x and y.

We have

$$\begin{aligned} R_k &= I(\mathbf{u};\mathbf{y}(t),\mathbf{y}(t+1)) \\ &= I(\mathbf{u};\mathbf{y}(t),\mathbf{y}(t+1)-\mathbf{A}\mathbf{y}(t)) \\ &\geq I(\mathbf{u};\mathbf{y}(t+1)-\mathbf{A}\mathbf{y}(t)) \\ &\doteq I(\mathbf{u};\tilde{\mathbf{y}}), \end{aligned} \tag{7}$$

In this case, the interaction between the $\mathbf{u}$ and $\tilde{\mathbf{y}}$ can be modeled as a MIMO channel

$$\tilde{\mathbf{y}}=\mathbf{C}\mathbf{B}\mathbf{u}(t)+\tilde{\mathbf{n}}, \tag{8}$$

in which $\tilde{\mathbf{n}}=\mathbf{C}\mathbf{w}(t)+\mathbf{n}(t+1)-\mathbf{A}\mathbf{n}(t)$ is the noise with covariance matrix

$$\mathbf{\Sigma}_{\tilde{n}}=\mathbf{C}\mathbf{\Sigma}_w\mathbf{C}^T+\mathbf{\Sigma}_n+\mathbf{A}\mathbf{\Sigma}_n\mathbf{A}^T.$$

To maximize $I(\mathbf{u};\tilde{\mathbf{y}})$, one should choose the distribution of $\mathbf{u}$ to be Gaussian input, which allows us to achieve the key rate of

$$\begin{aligned} R_k &= \max_{\mathbf{\Sigma}_u}\log\det\left(\mathbf{C}\mathbf{B}\mathbf{\Sigma}_u\mathbf{B}^T\mathbf{C}^T+\mathbf{\Sigma}_{\tilde{n}}\right)-\log\det\left(\mathbf{\Sigma}_{\tilde{n}}\right) \\ &= \max_{\mathbf{\Sigma}_u}\log\det\left(\mathbf{C}\mathbf{B}\mathbf{\Sigma}_u\mathbf{B}^T\mathbf{C}^T+\mathbf{\Sigma}_n\right) \\ &\quad -\log\det\left(\mathbf{C}\mathbf{\Sigma}_w\mathbf{C}^T+\mathbf{\Sigma}_n+\mathbf{A}\mathbf{\Sigma}_n\mathbf{A}^T\right) \end{aligned} \tag{9}$$

However, in practice, $\mathbf{u}$ also needs to serve for control purposes, hence the distribution on $\mathbf{u}$ is not necessarily Gaussian. Nevertheless, for any distribution, the key rate $I(\mathbf{u};\tilde{\mathbf{y}})$ is achievable.

B. General Case

In general, the attacker can also observe a noisy version of the system state

$$\begin{aligned}\mathbf{z}(t+1) &= \mathbf{D}\mathbf{x}(t+1)+\mathbf{m}(t+1)\\ &= \mathbf{D}\mathbf{B}\mathbf{u}(t)+\mathbf{D}\mathbf{A}\mathbf{x}(t)+\mathbf{D}\mathbf{w}(t)+\mathbf{n}(t+1).\end{aligned}$$

Here, $\mathbf{DBu}(t)$ is the signal term, $\mathbf{Dw}(t)+\mathbf{n}(t+1)$ is the noise term, $\mathbf{DAx}(t)$ is the interference term. The eavesdropper also observes $\mathbf{Dx}(t)+\mathbf{m}(t)$ which provides side-information about the interference term. Now, the scenario can be modeled as a MIMO wiretap channel [16]. For this MIMO wiretap channel the following secrecy rate is achievable

$$R_k = [I(\mathbf{u};\mathbf{y}(t),\mathbf{y}(t+1)) - I(\mathbf{u};\mathbf{z}(t),\mathbf{z}(t+1))]^+, \quad (10)$$

in which $[x]^+ = \max\{0,x\}$. To achieve this secrecy rate, one should use the stochastic encoding technique [17]. More specifically, for each key value K, we associate it with a group of codewords. When we decide to send key k, we randomly select a codeword from the group associated with the key. This additional randomness induce uncertainty at the eavesdropper, while the receiver can decode this randomness, and recover the key value.

We can further simplify (10). From (7), we know that $I(\mathbf{u};\mathbf{y}(t),\mathbf{y}(t+1)) \geq I(\mathbf{u};\tilde{\mathbf{y}}(t))$. Define

$$\tilde{\mathbf{z}}(t+1) = \mathbf{DBu}(t)+\mathbf{Dw}(t)+\mathbf{n}(t+1),$$

which is an interference free version of $\tilde{\mathbf{z}}(t+1)$. We use $\mathbf{\Sigma}_{\tilde{z}}$ to denote the covariance matrix of $\tilde{\mathbf{z}}(t+1)$. Hence, $I(\mathbf{u};\mathbf{z}(t),\mathbf{z}(t+1)) \leq I(\mathbf{u};\tilde{\mathbf{z}})$. As the result,

$$R_k \geq [I(\mathbf{u};\tilde{\mathbf{y}}(t)) - I(\mathbf{u};\tilde{\mathbf{z}}(t))]^+. \quad (11)$$

To maximize (11), one should again choose $\mathbf{u}$ to be Gaussian, which allows us to achieve

$$\begin{aligned}R_k &\geq [I(\mathbf{u};\tilde{\mathbf{y}}(t)) - I(\mathbf{u};\tilde{\mathbf{z}}(t))]^+\\ &\geq \max_{\mathbf{\Sigma}_u}\left[\log\det\left(\mathbf{CB\Sigma}_u\mathbf{B}^T\mathbf{C}^T+\mathbf{\Sigma}_{\tilde{n}}\right)-\log\det\left(\mathbf{\Sigma}_{\tilde{n}}\right)\right.\\ &\qquad \left.\log\det\left(\mathbf{DB\Sigma}_u\mathbf{B}^T\mathbf{D}^T+\mathbf{\Sigma}_{\tilde{n}}\right)-\log\det\left(\mathbf{\Sigma}_{\tilde{z}}\right)\right].\end{aligned}$$

IV. Key Establishment

While the bounds derived in Section III predict the fundamental limits of key rates that can be generated, no practical coding schemes have been designed so far to achieve these bounds. In this section, we study practical schemes that allows us to establish keys. For simplicity, we will focus on the special case in which $\mathbf{D}=\mathbf{0}$, i.e., the eavesdropper cannot observe the system state directly. The design of practical schemes for the general $\mathbf{D}$ will be our future work. We first introduce how to convey the common information via the control action. Then, we study how to extract the common random bits from the common information.

A. Information via Control

The essence of key establishment is to establish a common information between the controller and the sensor. This common information is the observation incurred by the action taken by the controller. Since the attacker is unable to sense the system directly ($\mathbf{D}=0$), there is no way for the attacker to obtain this common information, thus the secrete key.

We assume that key establishment happens at time slot t. At this time, the previous key used for the transmission from the sensor to the controller may not be reliable (this is why a new key is needed). The controller takes an action $\mathbf{u}(t)$. At time slot $t+1$, the output vector $\mathbf{y}(t+1)$ is observed at the sensor. Then, the controller and the sensor compute the following vectors separately:

$$\mathbf{s}_c = \mathbf{Bu}(t), \quad (12)$$

which is computed at the controller, and

$$\mathbf{s}_s = \hat{\mathbf{x}}(t+1) - \mathbf{A}\hat{\mathbf{x}}(t), \quad (13)$$

which is computed at the sensor. Note that $\hat{\mathbf{x}}(t)$ is the estimation of the system state $\mathbf{x}(t)$, which can be obtained from Kalman filtering. Obviously, $\mathbf{s}_s$ is an estimation of $\mathbf{s}_c$. If the estimation of the system state is precise and the noise power is small, the vectors $\mathbf{s}_c$ and $\mathbf{s}_s$ should be similar. Then, the sensor and the controller will exchange information to establish the key. Once the key is correctly established, the sensor will use the new key for the transmission.

B. Obtaining Common Randomness

Now, we focus on the key establishment between the sensor and the controller. First, we assume that $K \geq N$, i.e., the dimension of the action vector $\mathbf{u}$ is larger than or equal to that of the system state $\mathbf{x}$. If not, we will truncate both $\mathbf{s}_c$ and $\mathbf{s}_s$ to K dimensions, e.g., omitting the first $N-K$ dimensions. The reason is that, if $K<N$, only K dimensions in $\mathbf{s}_c$ and $\mathbf{s}_s$ are independent and the remaining $N-K$ dimensions do not provide any new information.

1) Quantization: Both the controller and the sensor use the N dimensions of $\mathbf{s}_c$ and $\mathbf{s}_s$ to generate the following bit sequences by quantizing each dimension into B bits:

$$\begin{aligned}\mathbf{s}_i^s &= \left(b_i^s(1),...,b_i^s(B)\right), \qquad i=1,...,N, \quad (14)\\ \mathbf{s}_i^c &= \left(b_i^c(1),...,b_i^c(B)\right), \qquad i=1,...,N, \quad (15)\end{aligned}$$

where, in the superscript, s means sensor and c means controller.

2) Extracting Common Bits: Now we focus on only one pair of sequence, e.g., $\mathbf{s}_i^s$ and $\mathbf{s}_i^c$. The procedure of extracting common bits in other sequence pairs is the same.

We first introduce the approach for extracting common bits from two bit sequences in [15], based on which we will propose a more general method. Using the approach in [15], the controller will first search for all the start positions of excursions with q consecutive identical bits.

When there are more than q consecutive identical bits, the next excursion begins from the $q+1$ bits. For example, if $\mathbf{s}_i^s = 01101101$ (suppose that $B = 8$ and $q = 2$), then the corresponding locations are 2 and 5. Then, the controller sends the locations to the sensor. The sensor will check whether there exist excursions at these locations. For example, if $\mathbf{s}_i^c = 01101001$, the sensor will find that excursion exists at only bit 2 (although there exists another excursion at bit 6, this location was not proposed by the controller). Then, the sensor will send back 2, which means that both the controller and sensor agree on the excursion at location 2. Thus, they achieve the same bit 1. During this procedure, although an eavesdropper can intercept the start location(s) of the excursion(s), it cannot determine whether the corresponding bit is 0 or 1 since this information is never revealed during the information exchange.

Now, we propose our approach for generating the two common bits sequences. First, we observe that the procedure of extracting common bits in [15] (the details are omitted due to the limited page) essentially *determines the locations having the same bit patterns between the controller and the sensor*. The excursion with q bits is one special bit pattern. Moreover, there should exist some ambiguity in the bit pattern. For example, in the above approach, both all-zero excursions and all-one excursions are considered as the same pattern. This ambiguity prevents an eavesdropper from learning any information from the information exchange. For example, if only the all-zero excursion is considered in the bit pattern, the eavesdropper knows that the common bit will be zero.

Based on the above observation, we propose a generalized approach for extracting the common randomness. Notice that the approach in [15] only utilized the patterns of q-excursions, which is still far away from exploiting all the common randomness. In our generalized approach, we first define patterns as follows.

Definition 1: Each pattern is a pair of bit sequences (s_1, s_2), which satisfies $s_1 = \bar{s_2}$, where $\bar{s_2}$ is the bitwise negative of s_2.

Note that the negative requirement $s_1 = \bar{s_2}$ aims at confusing eavesdroppers since an eavesdropper cannot determine the actual bit of the starting bit of the pattern if only the pattern index and the starting location are exchanged between the controller and the sensor.

Based on the definition of patterns, we establish a set of patterns, which is denoted by $\Omega = (\mathbf{p}_1, ..., \mathbf{p}_r)$, where r is the total number of patterns. Note that the number of patterns in Ω and the lengths of the patterns are not specified (at least the pattern cannot be a single bit). One requirement for the patterns is that one pattern cannot be the prefix of any other patterns; otherwise this pattern will be useless in the message exchange. Obviously, the longer a pattern is, the more possibly the two bit sequences at the controller and the sensor corresponding to the pattern are identical. On the other hand, the longer the pattern is, the less possibly the two bit sequences contain the corresponding pattern. It is quite challenging to design the pattern set. We do not discuss it in this paper and only test some arbitrary pattern sets in the numerical simulations.

Now, we assume that we have established a pattern set. The controller checks the patterns in Ω one by one and then sends out a series of messages, each using the following format:

$$\{\text{pattern index, starting location of the pattern}\}. \quad (16)$$

If a bit has been used for one pattern, the corresponding bit sequence (including the bit and the subsequent bits within the pattern) will not be used for other patterns. Note that, in contrast to the message in [15], the pattern indices in the proposed scheme also need to be transmitted since the pattern is default in [15]. Then, the sensor checks all the messages and sends back the pattern indices and starting locations matching its own bit sequence, as well as a parity check (unless there is only one common starting location). The parity check is generated from the bits at these starting locations. The purpose of the parity check is to verify whether the controller and the sensor obtain the same bit sequence for the secret key.

Below is an example to illustrate the procedure.

Example 1: Suppose that the controller and the sensor obtained bit sequences $\mathbf{s}_i^s = 0010110101$ and $\mathbf{s}_i^c = 0010010101$ for one dimension of $\mathbf{Bu}$. We set the pattern set as

$$\Omega = \{\mathbf{p}_1 = (00, 11), \mathbf{p}_2 = (101, 010)\}. \quad (17)$$

Then, the controller will send out the following messages: $\{1,1\}, \{1,5\}, \{2,7\}$. Upon receiving the messages, the sensor checks its own bit sequence and sends back the following messages (suppose that the parity check is a single bit equaling the XOR of all starting bits): $\{1,1\}, \{2,7\}, 1$. Recall that the last quantity 1 is the parity check. Then, both the controller and sensor agree that the common bit sequence is 01.

3) Fusing Common Randomness: After the controller and the sensor complete the message exchange, they first discard the bit sequences with wrong parity checks. The remaining bit sequences are assumed to be identical at the controller and the sensor (although they could be different). Then, they will merge these bit sequences into one bit sequence, which will be used as the shared secret key. For simplicity, we adopt the same approach as that in [15] using the following steps:

1) The bit sequences are merged into the a long bit sequence.
2) The long bit sequence is interleaved using a pseudo-random order.
3) Some bits are XORed to remove possible correlations.

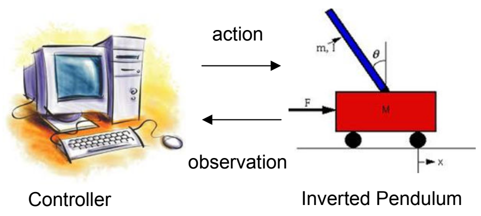

Fig. 2: An illustration of the remote controlled inverted pendulum.

4) Algorithm Summary: The algorithm for generating the secret keys is summarized in Procedure 1. This procedure is called periodically in order to update the secret keys for higher security.

Procedure 1 Procedure of Generating Secret Keys

```
1: The controller takes an action $\mathbf{u}(t)$.
2: The sensor senses the observation and obtain $\mathbf{y}(t)$. It uses Kalman filtering to estimate the system state $\mathbf{x}(t+1)$.
3: The sensor and the controller compute vectors $\mathbf{s}_c$ and $\mathbf{s}_s$, separately.
4: The sensor and the controller use $B$ bits to quantize each dimension of the vectors $\mathbf{s}_c$ and $\mathbf{s}_s$, separately..
5: For each bit sequence obtained from the quantizations, the controller checks the patterns in the predetermined pattern set $\Omega$ and sends the pattern indices and starting locations to the sensor.
6: The sensor sends back the pattern indices and the starting locations matching its own bit sequence, as well as a parity check.
7: Both the controller and sensor fuse the common bits and then generate the shared secret key.
```

V. Experiment

In the experiment, we use a real inverted pendulum to generate the key, which is illustrated in Fig. 2. The system state is a four-dimensional vector, which includes the location, the speed and accelerating speed. The control is a scalar. The system parameters are given as follows:

$$\mathbf{A} = \begin{pmatrix} 0 & 0 & 1.0000 & 0 \\ 0 & 0 & 0 & 1.0000 \\ 0 & 2.2643 & -5.6718 & -0.0073 \\ 0 & 27.8203 & -13.0683 & -0.0896 \end{pmatrix}, \quad (18)$$

and

$$\mathbf{B} = (0, 0, 1.3190, 3.0391)^T . \quad (19)$$

The sensor can sense the system states directly, thus $\mathbf{C} = \mathbf{I}$. The controller uses an LQR control, in which the control action is given by $u = \mathbf{k}\hat{\mathbf{x}}$, where $\hat{\mathbf{x}}$ is the estimated system state and

$$\mathbf{k} = (-14.1421, 63.7780, -14.3342, 8.7250) . \quad (20)$$

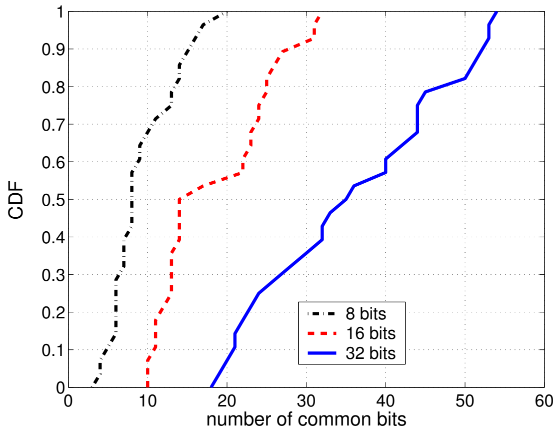

Fig. 3: The CDF of the number of common bits for pattern scheme 1.

We have $\sigma_u^2 = 0.61936$. The covariance matrices of the noise processes are

$$\mathbf{\Sigma}_w = \begin{pmatrix} 0.005 & 0.001 & 0.026 & 0.0594 \\ 0.001 & 0.00086 & 0.002 & 0.0049 \\ 0.026 & 0.002 & 0.509 & 1.23 \\ 0.0594 & 0.0049 & 1.23 & 3.036 \end{pmatrix}, \quad (21)$$

and $\mathbf{\Sigma}_n = \mathbf{0}$

We consider two schemes for the common bit pattern:

- scheme 1: $\Omega = \{(111, 000), (101, 010), (1100, 0011)\}$
- scheme 2: $\Omega = \{(110, 001), (101, 010), (1100, 0011)\}$

We take a sample every 0.1 second and generate one key every 10 samples, i.e., 1 second. The algorithm in Procedure 1 is applied to generate the common bits. We tried different number of bits for quantizing each sample, 8, 16 and 32.

The cumulative distribution function (CDF) of the number of generated common bits per second is shown in Fig. 3. Note that all the generated common bits are correct. We observe that, as the number of bits per sample is increased, the number of common bits is also increased. Note that increasing the number of bits per sample may increase the final number of common bits while it may also incur some redundancy in the final bit sequence, which may impair the security. Moreover, finer quantization may introduce more disparity between the two bit sequences since the same noise can impact more bits due to the smaller quantization step size. Even for the worst case (8 bits per sample), the inverted pendulum can still generate averagely 10 bits per second, which is much more efficient than the key generation using the common wireless channel between the transmitter and receiver [14] (around 1 bit per second for a single antenna system).

The results generated from pattern scheme 2 are shown in Fig. 4. We observe that the performance is similar

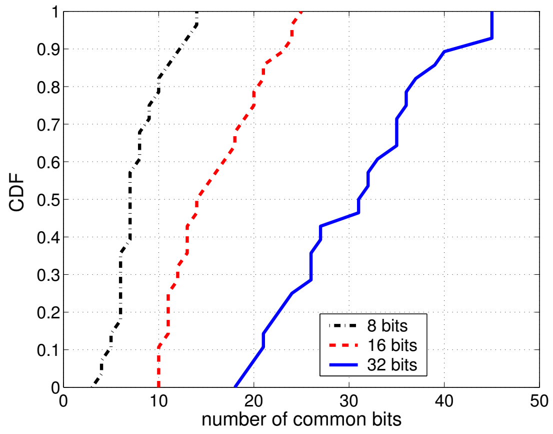

Fig. 4: The CDF of the number of common bits for pattern scheme 2.

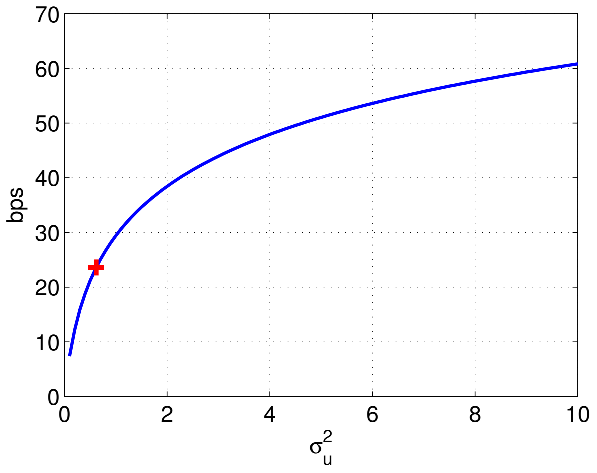

Fig. 5: Theoretic bound (9).

although there is a slight performance degradation which may be due to the randomness.

Figure 5 shows the theoretical bounds derived in (9) using the same parameters as above for various value of σ_u^2. As shown using '+' mark, the key rate is 23.6 bps when $\sigma_u^2 = 0.61936$. This theoretic prediction is in agree with our experiment results when the quantizer uses 8 or 16 bits, but is smaller than that of the key rate generated using 32-bits quantizer. As discussed above, as the number of bits in quantizer increases, there could be redundance in the bit sequences, as the case in the 32-bits quantizer case. In practice, one can use the theoretic value as a benchmark to decide the number of bits used in quantizer.

VI. Conclusion

In this paper, we have discussed the key generation in the networked control system, which utilizes the common physical information between the sensor and the controller and avoids the troublesome key distribution procedure. We have obtained the theoretical bound for the rate of key generation using the information theoretic argument. We have also proposed a practical algorithm for generating the common secret bit sequence in the key. Experimental results using a real inverted pendulum have shown that the proposed key generation scheme can effectively generate common bit sequence, e.g., generating tens of bits per second.

References

[1] S. Amin, A. A. Cárdenas and S. Sastry, "Safe and secure networked control systems under denial-of-service attacks," *Lecture Notes in Computer Science*, Springer, 2009.

[2] J. Baillieul and P. J. Antsaklis, "Control and communication challenges in networked real-time systems," *Proc. of IEEE*, vol. 95, pp.9–28, Jan. 2007.

[3] C. Bennett and S. Wicker, "Decreased time delay and security enhancement recommendations for AMI smart meter networks," in *Proc. of Innovative Smart Grid Technologies Conference*, 2010.

[4] ISO New England Inc., *Overview of the Smart Grid: Policies, Initiatives and Needs*, Feb. 17, 2009.

[5] A. A. Cárdenas, S. Amin and S. Sastry, "Secure control: Towards survivable cyber-physical systems," in *Proc. of the 28th International Conference on Distributed Computing Systems Workshops*, 2008.

[6] A. A. Cárdenas, S. Amin and S. Sastry, "Research challenges for the security of control systems," in *Proc. of the 3rd Conference on Hot Topics in Security*, 2008.

[7] J. P. Hespanha, P. Naghshtabrizi and Y. Xu, "A survey of recent results in networked control systems," *Proc. of IEEE*, vol. 95, pp.138–162, Jan. 2007.

[8] Z. Li, W. Xu, R. Miller and W. Trappe, "Securing wireless systems via lower layer enforcements," in *Proc. of the 5th ACM workshop on Wireless security*, 2006.

[9] R. Wilson, D. Tse, and R. A. Scholtz, "Channel identification: Secret sharing using reciprocity in ultrawideband channels," *IEEE Trans. Inform. Forensics and Security*, vol. 2, pp. 364–375, Sept. 2007.

[10] L. Lai and H. V. Poor. A unified framework for key agreement over wireless fading channels. In *Proc. IEEE Inform. Theory Workshop*, Taormina, Sicily, Italy, Oct. 2009.

[11] L. Lai, Y. Liang, and H. V. Poor. Key agreement over wireless fading channels with an active attacker. In *Proc. Allerton Conf. on Communication, Control, and Computing*, Monticello, IL, Sept. 2010. To appear.

[12] L. Lai, Y. Liang, and W. Du. Cooperative key generation in wireless networks. In *Proc. IEEE Conf. Computer Communications (Infocom)*, 2011. Submitted.

[13] S. Mathur, W. Trappe, N. Mandayam, C. Ye and A. Reznik, "Radio-telephathy: Extracting a secret key from an unauthenticated wireless channel," in *Proc. of the 14th ACM International Conference on Mobile Computing and Networking (Mobicom)*, 2008.

[14] J. W. Wallace, C. Chen and M. A. Jensen, "Key generation exploiting MIMO channel evolution: Algorithm and theoretical limits," in *Proc. of the 3rd European Conference on Antennas and Propagation*, 2009.

[15] K. Zeng, D. Wu, A. Chan and P. Mohapatra, "Exploiting multiple-antenna diversity for shared key generation in wireless networks," in *Proc. of IEEE Conference on Computer Communications (Infocom)*, 2010.

[16] A. Khisti and G. W. Wornell. Secure transmission with multiple antennas: The MISOME wiretap channel. *IEEE Trans. Inform. Theory*, 2009. To appear.

[17] A. D. Wyner, "The wire-tap channel," *Bell System Technical Journal*, vol. 54, pp. 1355–1387, Oct. 1975.