sTrack: Secure Tracking in Community Surveillance

Chun-Te Chu, Jaeyeon Jung, Zicheng Liu, Ratul Mahajan
Microsoft
{chch, jjung, zliu, ratul}@microsoft.com

ABSTRACT
We present sTrack, a system that can track objects across multiple cameras without sharing any visual information between two cameras except whether an object was seen by both. To achieve this challenging privacy goal, we leverage recent advances in secure two-party computation and multi-camera tracking. We derive a new distance metric learning technique that is more suited for secure computation. Compared to the existing methods, our technique has lower complexity in secure computation without sacrificing the tracking accuracy. We implement it using a new Boolean circuit for secure tracking. Experiments using real datasets show that the performance overhead of secure tracking is low, adding only a few seconds over non-private tracking.

Categories and Subject Descriptors
I.4.8 [**Scene Analysis**]: Tracking; K.4.1 [**Public Policy Issues**]: Privacy

Keywords
Tracking; Privacy-preserving; Secure Two-Party Computation; Distance Metric Learning; Human Re-identification

1. INTRODUCTION
In conventional surveillance systems (e.g., those deployed in airports, subways, and corporate buildings), multi-camera tracking is extremely useful to fight crime. For example, when an investigator spots a suspicious person entering a building from one of the surveillance videos, she may want to know where he left the building. If the building has multiple entry-exit points, tracking across cameras is needed. Tremendous progress has been made on multi-camera tracking [3][6][13][23], and commercial systems include this ability [10]. As all cameras have the same owner (e.g., a government agency or a company), current techniques do not isolate the information from different cameras.

However, privacy concerns severely limit the applicability of multi-camera tracking to residential community surveillance. Householders in many communities, including those in high-crime neighborhood, are loath to law enforcement deploying surveillance cameras in areas because of privacy issues [2]. Camera feeds contain many aspects of their lives that are unrelated to crime and are potentially embarrassing [11]. The concerns around sharing camera feeds are unfortunate because processing feeds across them would enable community-wide surveillance to detect suspicious activities and objects across cameras (e.g., a person knocking on several doors, a car cruising the neighborhood), which individual cameras cannot detect. One possibility is for each home to send its feed to a central party (e.g., law enforcement or a corporation), and for this party to run one of the existing surveillance algorithms. However, householders are often unwilling to share their camera data with third-parties [2].

We investigate an alternative, peer-to-peer approach in which cameras communicate to each other to determine if they saw the same object. Our goal is to enable cameras to cooperate in order to track suspicious objects (e.g., unknown people or cars) without revealing any information about the objects to other cameras except matching outcomes. We assume that each family only has access to the video feeds from their own cameras. Each time the camera identifies an object of interest, it checks with peer cameras from other families to determine whether they saw the same object. This determination can be made using a distance function over the object features that individual cameras extract. However, private information may be revealed if the cameras share the raw values of the feature vectors (e.g., it is possible to identify the object using color histogram).

To address this privacy risk, we explore if secure two-party computation (2PC) can be used for privately matching objects. 2PC allows two participants to compute any function over their inputs, without revealing anything but the function output (and what may be inferred from it). Much progress [12][14][15][16] has been made recently towards reducing the overhead of 2PC. In this paper, we show that 2PC can enable high-accuracy, low-overhead multi-camera tracking.

This paper makes the following contributions:

1. We present a new metric learning technique that permits an efficient secure two-party computation with significantly lower complexity. It has low overhead because it refactors the problem such that cameras do the complex portions of the computation (e.g., matrix multiplication) locally and the portions that are computed jointly are simple. Experiments with real datasets show the overhead of our technique is 6 to 200 times lower than the prior approaches [8][23] without sacrificing the accuracy.

Like the prior techniques, distance metric learning is used in many domains such as matching hand-written text, face recognition, and matching biometric features. Thus, although we focus on multi-camera tracking, our technique may enable privacy-preserving matching in other domains as well.

2. We implement our technique in a distributed community surveillance system, presenting the first instance of privacy-preserving peer-to-peer multi-camera tracking system. Experiments show that the overhead of secure tracking is low, adding only a few seconds over non-private tracking.

2. BACKGROUND AND RELATED WORK
2.1 Privacy-Preserving Video Surveillance
To our knowledge, our work is the first one focusing on peer-to-peer privacy-preserving tracking over multiple cameras. Much previous effort has focused on privacy concerns with respect to

MM '14, November 03 - 07 2014, Orlando, FL, USA

http://dx.doi.org/10.1145/2647868.2655010

Table 1. Distance computation

	intermediate steps	circuits needed
1	$\mathbf{W}^T\hat{\mathbf{x}}_i$, $\mathbf{W}^T\hat{\mathbf{x}}_j$	(local computation)
2	$\mathbf{W}^T\hat{\mathbf{x}}_i - \mathbf{W}^T\hat{\mathbf{x}}_j$	q subtraction
3	$\| \mathbf{W}^T\hat{\mathbf{x}}_i - \mathbf{W}^T\hat{\mathbf{x}}_j \|^2$	q multiplication $q-1$ addition

the video surveillance without peer-to-peer multiple cameras collaboration [5][18][20]. Other techniques [1][17] are complementary to our effort and cannot be applied to multi-camera tracking in which cameras do not trust each other.

2.2 Secure Computation

Yao's garbled circuit (GC) approach provides a foundation to securely compute a function represented as a Boolean circuit [21]. References [12][14][15] show how these circuits are constructed efficiently. We implement secure matching using Huang et al.'s library [12] with the optimizations proposed by Kolesnikov et al. [14]. The complexity of each secure operation, i.e., the size of the garbled circuit, increases in the order of $O(n^2)$ for multiplication operation, while it increases linearly ($O(n)$) for other operations.

2.3 Object Matching Across Cameras

Matching function computes the distance between two feature vectors extracted from individual cameras to determine similarity. Recent computer vision algorithms employ machine learning approaches to "learn" a distance metric using training data [6][8][23]. We build on these works to show that 1) secure computation on these methods incurs high overhead, and 2) distance computation can be transformed such that its overhead is significantly lowered while accuracy is not impacted.

3. GARBLED CIRCUIT DISTANCE COMPUTATION (GCDC)

Multi-camera tracking involves matching objects seen by different cameras. Two cameras jointly evaluate a distance function $D(\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_j)$ where $\hat{\mathbf{x}}_i$ and $\hat{\mathbf{x}}_j \in \mathbb{R}^r$ are the feature vectors extracted from the objects. If the distance is smaller than a threshold, the objects are deemed the same. The goal of metric learning is to learn a distance function $D(\cdot,\cdot)$ that can well-discriminate same objects and different objects by learning a projection matrix $\mathbf{W} \in \mathbb{R}^{r\times q}$, $q \le r$, which projects feature vectors onto a new space. Normally, Euclidean distance is computed after projection [6][23]. The computation is shown in Table 1. The complexity of the secure computation is determined only by the joint computation, which in this case involves subtraction, addition and multiplication. To avoid expensive multiplication operation, we propose a new distance metric learning technique, garbled circuit distance computation (GCDC), which uses histogram intersection instead of Euclidean distance to reduce the joint computation. Histogram intersection has been shown to be an effective metric in pattern recognition [22]. It is defined as

$$HI(\mathbf{h}_i, \mathbf{h}_j) = \sum_{k=1}^{q} min(h_i^k, h_j^k), \qquad (1)$$

where $\mathbf{h}_i = [h_i^1 \dots h_i^q]^T \in \mathbb{R}_+^q$, $\mathbf{h}_j = [h_j^1 \dots h_j^q]^T \in \mathbb{R}_+^q$ are vectors with nonnegative entries. We define our distance function as:

$$\begin{aligned} D(\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_j) &= -HI(\mathbf{W}^T\hat{\mathbf{x}}_i + T\mathbf{1},\ \mathbf{W}^T\hat{\mathbf{x}}_j + T\mathbf{1}) \\ &= -\sum_{k=1}^{q} min(\mathbf{w}_k{}^{\mathrm{T}}\hat{\mathbf{x}}_i + T, \mathbf{w}_k{}^{\mathrm{T}}\hat{\mathbf{x}}_j + T), \end{aligned} \qquad (2)$$

where $\mathbf{W} = [\mathbf{w}_1 \dots \mathbf{w}_q] \in \mathbb{R}^{r\times q}$, $q \le r$, and $T \ge 0$.

Table 2. Our GCDC computation

	intermediate steps	circuits needed
1	$\mathbf{W}^T\hat{\mathbf{x}}_i + T\mathbf{1}$, $\mathbf{W}^T\hat{\mathbf{x}}_j + T\mathbf{1}$	(local computation)
2	min(.)	q min operation
3	$\sum_{k=1}^{q}$min(.)	$(q-1)$ addition

T is a predefined parameter that ensures the non-negativity of $\mathbf{w}_k{}^{\mathrm{T}}\hat{\mathbf{x}}_i + T$ for any feature vector $\hat{\mathbf{x}}_i$. Note that although the value of $D(\cdot,\cdot)$ seems to be negative, we can add a positive scalar constant to make it a valid distance metric, without changing the formulation of our metric learning, so we discard the scalar in the following discussion. The complexity of $D(\cdot,\cdot)$ is shown in Table 2. Note that significant computation is local, and joint computation uses only simple operations (minimum and addition).

Just defining $D(\cdot,\cdot)$ is not enough. For it to be a valid approach, we also need an efficient way to learn $\mathbf{W}$. Our learning approach is motivated by [8]. Let $\mathbb{O}^{\mathrm{p}}$ denote the positive training set consisting of matched feature vector pairs, and $\mathbb{O}^{\mathrm{n}}$ denote the negative training set consisting of unmatched feature vector pairs. The goal is to learn a linear projection matrix, such that the pairs in set $\mathbb{O}^p$ have small distances, and the pairs in set $\mathbb{O}^n$ have large distances. Define a conditional distribution over points $i \ne j$ as

$$p^{\mathbf{W}}(j|i) = \frac{e^{-D(\hat{\mathbf{x}}_i,\hat{\mathbf{x}}_j)}}{\sum_{k\ne i} e^{-D(\hat{\mathbf{x}}_i,\hat{\mathbf{x}}_k)}} \qquad i \ne j. \qquad (3)$$

Ideally, if all the pairs in set $\mathbb{O}^p$ have small distance, and all the pairs in set $\mathbb{O}^n$ have large distances, the distribution would become "bi-level", that is,

$$p_0(j|i) \propto \begin{cases} 1 & if\ \ (\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_j) \in \mathbb{O}^p \\ 0 & if\ \ (\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_j) \in \mathbb{O}^n \end{cases} \qquad (4)$$

Therefore, the first cost function $f_1(\mathbf{W})$ same as [8] is defined as

$$\begin{aligned} f_1(\mathbf{W}) &= \sum_i KL[p_0(j|i) | p^{\mathbf{W}}(j|i)] \\ &= \sum_i \sum_{j\ne i} p_0(j|i) \times log(\frac{p_0(j|i)}{p^{\mathbf{W}}(j|i)}), \end{aligned} \qquad (5)$$

where $KL[\cdot\,|\,\cdot]$ is K-L divergence that measures the distance between two distributions.

We further introduce a regularization term $f_2(\mathbf{W})$ to bound the values of $\mathbf{W}^T\hat{\mathbf{x}}_i$ in such a way that we can always find a nonnegative scalar T to make all the entries in $\mathbf{W}^T\hat{\mathbf{x}}_i + T\mathbf{1}$ nonnegative for all *i*.

$$f_2(\mathbf{W}) = \sum_{k=1}^{q} \mathbf{w}_k{}^T\mathbf{w}_k = Tr(\mathbf{W}^T\mathbf{W}) \qquad (6)$$

Moreover, to satisfy the equality of self-distance, that is,

$$D(\hat{\mathbf{x}}_k, \hat{\mathbf{x}}_k) = D(\hat{\mathbf{x}}_l, \hat{\mathbf{x}}_l) \qquad \forall\ k \ne l \qquad (7)$$

an additional term $f_3(\mathbf{W})$ is added,

$$f_3(\mathbf{W}) = \sum_i (\sum_{k=1}^{q} \mathbf{w}_k{}^T\hat{\mathbf{x}}_i)^2 \qquad (8)$$

The final objective is the sum of the three terms above:

$$J(\mathbf{W}) = f_1(\mathbf{W}) + f_2(\mathbf{W}) + f_3(\mathbf{W}) \qquad (9)$$

The metric learning problem is formulated as finding $\mathbf{W}$ that minimizes the objective function $J(\mathbf{W})$. Gradient descent method is employed to solve the optimization problem. After $\mathbf{W}$ is learned, distance between two objects can be evaluated. There is a match if the value is smaller than a predefined threshold. For more detailed derivation, please refer to [4].

(a) (b)

Figure 1. Examples of the snapshots in the database. (a) the same person under different cameras. (b) the same car under different cameras.

4. sTrack: DESIGN & IMPLEMENTATION

Our system is developed as an application on top of HomeOS [7]. Each site (i.e., camera) in sTrack is configured with information about its peer sites for object matching. This information includes the peer's location relative to the site and how to contact it (IP address and port) for matching requests. Each site independently tracks objects' information within its own view, including location, entry/exit timestamps, and features.

We employ state of the art computer vision algorithms for single camera tracking, including background subtraction, Kalman filtering, and kernel tracking [4]. Each camera performs tracking as above and stores entry and exit time stamps, tracking history, and feature vectors of all the objects in its own local database. This information is used when the camera performs the evaluation of the matching function between a local object and objects captured by other cameras. As in [23], we divide an object into six horizontal regions. In each region, the color histograms of RGB, YCbCr, and HSV color spaces are extracted. Each channel has 16 bins, and the histograms are concatenated into the feature vector in a 864-dimension feature space. One can, if needed, include more features, e.g., texture, edge features or even biometric features like faces, to enhance accuracy, and our proposed metric learning method can be applied without change.

When an object enters the view of Site A, the feature vector is extracted, and the matching process is initialized against all peer sites. (We currently issue match requests for all objects entering the view; in the future, it could be limited to suspicious objects such as those that have never been seen before or those specified by users.) For these processes, we term the initiator of the process (Site A) as the client and other sites as servers. Each server performs a match against each object that appeared in the server's view within a 10 minute window. The window size can be determined based on the user's interest or the prior knowledge about the topology of the cameras.

We implement our secure computation, including distance function and thresholding, using Huang et al.'s library [12]. The details can be found in [4]. The output (obtained by both parties) is a binary value indicating if the client's object matched against any of the server's objects. No other information is leaked between sites. Matching outcomes allow Site A to learn where the object came from. Match requests can also be issued sometime after the object exits the camera's view, to learn where the object goes. We do not implement this currently and instead rely on incoming match requests from peers to learn this information. Based on the results of the match requests, each site can independently infer the activity pattern of suspicious objects.

5. EXPERIMENTAL RESULTS

5.1 GCDC Accuracy

First, we conduct offline testing by using four real datasets to compare GCDC and other metric learning techniques [8][23] in terms of matching accuracy and secure computation overhead. We use two public datasets, VIPeR [9] and i-LIDS [19], and two of our own datasets, which we call *human dataset* and *car dataset*. Fig. 1 shows some example images. For each dataset, in each trial we randomly select 80 objects (68 for our car dataset) as the training set to learn the parameters in offline training stage, and randomly select 15 objects as the testing set. The two sets do not overlap. We evaluate the matching functions against 15 objects based on the learned parameters. In real settings, a camera may capture objects at a faster (slower) rate if pointed at a busy (quite) street. We use 15 to represent a moderately busy street. We report the average of ten trials for each dataset.

Table 3. Complexity and accuracy summary table

VIPeR dataset	**Non-XOR gates**	**AUC**
PRDC	499,623	0.8515
PRDC_woABS	579	0.7468
MCC	10,298	0.9059
GCDC	1,778	0.9017
iLIDS dataset		
PRDC	499,623	0.8151
PRDC_woABS	579	0.6404
MCC	11,022	0.8697
GCDC	1,902	0.8653
human dataset		
PRDC	379,902	0.8473
PRDC_woABS	2,175	0.7643
MCC	10,009	0.9062
GCDC	1,729	0.9272
car dataset		
PRDC	379,902	0.8877
PRDC_woABS	579	0.8395
MCC	7,831	0.9471
GCDC	1,351	0.9567

We use AUC (Area Under Curve) of Receiver Operational Characteristic (ROC) curve as the accuracy evaluation metric. The complexity of the secure computation is determined by the number of non XOR gates in the garbled circuits [14][15]. Table 3 shows the comparison for different matching functions. PRDC_woABS [23] has the minimum number of non XOR gates for three datasets but has very poor accuracy. We thus do not consider it to be a viable method. Both PRDC [23] and MCC [8] provide accurate matching. GCDC has similarly high accuracy and even outperforms PRDC and MCC for two datasets. Non-XOR gates for GCDC is on average 261x and 6x fewer than PRDC and MCC, respectively. In retrospect, this result is not surprising; those previous approaches were not designed with secure computation in mind. Thus, we conclude that GCDC is promising for secure object matching. Readers can refer to [4] where the details of implementation and analysis are provided.

5.2 sTrack Performance

Unlike the offline testing in Section 5.1 where the well-cropped images are used, here we quantify the end-to-end accuracy of our system which includes inaccuracies due to imperfect object extraction/tracking in each camera's view. For this experiment, we use the video clips from our datasets with total length 47 minutes as the input to our system. The system has two sites, and each site is fed the video gathered by one of the cameras in the datasets. As in the actual system, each site performs independently single camera tracking within its own view and store objects' information in its local database. The sites perform object matching using the parameters learned in offline training. Fig. 2 shows the matching accuracy. Compared to offline testing, we see that there is only a slight loss in accuracy due to errors in within-camera tracking.

We now benchmark the performance of secure matching. For this benchmark, we use two relatively cheap and weak computers

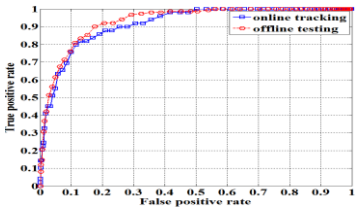

Figure 2. Online tracking accuracy.

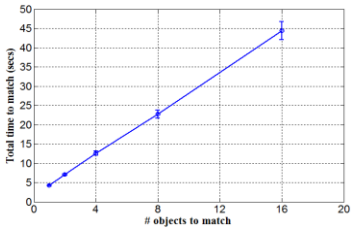

Figure 3. Mean and standard deviation of the time to securely match objects.

on purpose, Windows 8 netbooks with 1 GHz processor and 2 GB memory. We expect that many camera sites will not be equipped with a powerful computer and will instead use small, headless PCs or embedded processors like smart cameras. Using a network emulator, we introduce a round-trip network delay of 100 ms between the netbooks to mimic real-world situations in which sites may not have low-latency network paths between them. We measure the total time from the client issuing the match request to it recovering the results of the match. Thus, this time includes the time for initial handshake in which the client learns about the ports on which the server processes are running and the time to start client and server processes. We conduct ten different trials with each number of objects and plot the mean and standard deviation across those trials. Fig. 3 shows the results as a function of the number of objects against which a match is performed (i.e., the number of objects that the server saw in the specified time window). We see that the time to securely match objects increases linearly with the number of objects. For single-object matches, the time is roughly 4 seconds. For matching 4-16 objects, the total time is roughly 3 seconds per match. Thus, even if as many as 20 objects need to be matched, it will complete in under a minute.

This level of performance, which was obtained on relatively weak computers, can lead to a practical, real-time surveillance system. Consider the following back-of-the-envelope analysis. Let the community be such that one new object enters every X minutes on average, each camera needs to consult N neighbors, and that the time period of interest is Y minutes. In this community, a camera will issue $\frac{N}{X}$ requests per minute for matching, where each request will involve $\frac{Y}{X}$ object matches. Thus, a camera needs to perform $2 \times \frac{Y}{X} \times \frac{N}{X}$ matches per minute. The factor of two is due to the fact that a camera will answer match requests as well. With $X = 5$, $N = 10$, $Y = 10$ in a normal community neighborhood, it results in 8 matches per minute, well within the performance of our system.

6. CONCLUSION

We presented a system, sTrack, that embodies a set of techniques for tracking objects across cameras without leaking any visual information other than the binary matching result. It is based on a new distance metric learning approach that, compared to state-of-art approaches, has 6-200 times lower overhead and similar or better matching accuracy. Experiments show that the performance overhead of private tracking is moderate, taking around 3 seconds even on a netbook. Distance metric learning is a powerful approach for matching in a range of domains (e.g., face recognition, biometric matching, etc); we hope our work will spur privacy-preserving matching in all such domains. For more details about this work, please refer to [4].

7. REFERENCES

[1] S. Avidan and M. Butman, "Blind vision," *ECCV*, May, 2006.

[2] A.J. Brush, J. Jung, R. Mahajan, and F. Martinez, "Digital Neighborhood Watch: Investigating the Sharing of Camera Data amongst Neighbors", *CSCW,* Feb 2013.

[3] C. Chu and J. Hwang, "Fully unsupervised learning of camera link models for tracking humans across nonoverlapping cameras," *IEEE Transactions on Circuits and Systems for Video Technology,* 2014.

[4] C. Chu, J. Jung, Z. Liu and R. Mahajan, "sTrack: Secure Tracking in Community Surveillance," *Microsoft Research, TechReport, MSR-TR-2014-7*, 2014.

[5] Y. Chen, C. Chu, J. Hwang, and J. Yoo, "A privacy-preserving human tracking scheme in centralized cloud based camera network," *IEEE Conf. on Communications*, 2014.

[6] M. Dikmen, E. Akbas, T. S. Huang, and N. Ahuja, "Pedestrian recognition with a learned metric," *ACCV*, 2010.

[7] C. Dixon, R. Mahajan, S. Agarwal, A.J. Brush, B. Lee, S. Saroiu, and P. Bahl, "An operating system for the home," *Proc. NSDI*, 2012.

[8] A. Globerson and R. Roweis, "Metric learning by collapsing classes," *Proc. NIPS*, 2005.

[9] D. Gray, S. Brennan, and H. Tao, "Evaluating Appearance Models for Recognition, Reacquisition, and Tracking," *Proc. IEEE International Workshop on Performance Evaluation for Tracking and Surveillance (PETS)*, October, 2007.

[10] http://www.videosurveillance.com/neighborhoods.asp. Last accessed in Aug 2014.

[11] http://www.sfgate.com/crime/article/Oakland-hills-residents-fight-crime-with-cameras-3687341.php. Last accessed in Aug 2014.

[12] Y. Huang, D. Evans, J. Katz, and L. Malka, "Faster secure two-party computation using garbled circuits," *20th USENIX Security Symposium*, Aug, 2011.

[13] O. Javed, K. Shafique, and M. Shah, "Appearance modeling for tracking in multiple non-overlapping cameras," *CVPR* , 2005.

[14] V. Kolesnikov, A.-R. Sadeghi, and T. Schneider, "Improved Garbled Circuit Building Blocks and Applications to Auctions and Computing Minima", *Proc. CANS* 2009.

[15] V. Kolesnikov and T. Schneider, "Improved garbled circuit: Free XOR gates and applications", *Proc. ICALP*, 2008.

[16] B. Kreuter, a. shelat, and C.-H. Shen, "Billion-Gate Secure Computation with Malicious Adversaries", *Proc. USENIX Security,* 2012.

[17] M. Osadchy, B. Pinkas, A. Jarrous, and B. Moskovich, "SCiFI-A system for secure face identification," *IEEE Symposium on Security and Privacy*, 2010.

[18] A. Senior, S. Pankanti, A. Hampapur, L. Brown, Y.-L. Tian, A. Ekin, J. Connell, C. F. Shu, and M. Lu, "Enabling Video Privacy through Computer Vision", *IEEE Security and Privacy*, 2005.

[19] UK Home Office, "The Image Library for Intelligent Detection Systems (i-LIDS): Multiple Camera Tracking (MCT)," 2008.

[20] M. Upmanyu, A. M. Namboodiri, K. Srinathan, and C.V. Jawahar, "Efficient Privacy Preserving Video Surveillance", *IEEE Proc. Intl. Conf. on Computer Vision (ICCV),* 2009.

[21] A. C. Yao, "How to Generate and Exchange Secrets", *Proc. Foundations of Computer Science*, 1986.

[22] W. Zhang, S. Shan, W. Gao, X. Chen, and H. Zhang, "Local gabor binary pattern histogram sequence (lgbphs): A novel non-statistical model for face representation and recognition," *Proc. ICCV*, 2005.

[23] W. Zheng, S. Gong, and T. Xiang, "Person re-identification by probabilistic relative distance comparison," *Proc. CVPR* , 2011.