Available online at www.elixirpublishers.com (Elixir International Journal)

Information Technology

Elixir Inform. Tech. 77 (2014) 29155-29159

ISSN: 2229-712X

Applying adaptively reduced-order extended Kalman filter in cloud computing for intensifying its security

Masoud Ale Seyyedan
Qods Switching Center, Mashhad, Iran.

ARTICLE INFO

Article history:
Received: 18 July 2014;
Received in revised form:
25 November 2014;
Accepted: 22 December 2014;

Keywords
Kalman filtering,
Cloud computing,
Information technology.

ABSTRACT
This paper organized as such: In the first section authors mention about all aspects of cloud computing – basic definitions, benefits, important features, workability and application, and so on. In the second section of this paper, authors mention Microsoft decisions about applying of cloud computing onto their newest products, such as Azure OS. In the third section of this paper, authors reveal new innovative algorithm for overcoming malfunctions and spyware actions from cloud platforms and resources. In this section, they purpose a new adaptively reduced-order extended Kalman filter for using as an estimator and predictor in cloud platforms. Authors claim that, if this algorithm were implemented on cloud gateways and resources, cloud service providers were able to track and trace lots of spyware and Trojans and also predict and estimate amount of power consumptions.

Introduction

It is general idea today that security for network applications is their very vital feature and property. Its variety includes protection of data, messages, software modules and other resources, privacy of users, reliability, availability and integrity of resources and other properties. In the last 20 – 25 years there are very contributions in the scope of computer networks security: standards, research projects, conference and journal papers and commercial products. Governments, companies, banks and other users of network services invest huge deal of time, effort and budgets installing and using different security products and solutions [10].

Although, in spite of all these activities, on-going efforts and current solutions, it is usual belief that security in today networks and applications is not adequate. We are daily witnessing different problems – infection of computers by malware, distribution of E–mail spam, phishing of Web pages, penetrations by hackers, software bugs, stolen industrial secrets and credit cards, disclosure of sensitive documents, and so on.

All such interests and current problematic situations justify efforts and efforts towards creating effective security solutions for network applications and environments. At the moment, there are two usual approaches to network applications security [7-10]:

- One approach is rely on isolation, that is protecting them by isolating operational environments at their periphery using firewalls, port scanners, intrusion–detection tools, spam and phishing filters, "demilitarized zones", E- mail spam filters, etc. and also using virus/malware scanners, virus signatures, encrypted disk files, etc.
- Another approach is named software security which is rely on methodology to create secure, robust and protected applications, bug–free and not vulnerable to attacks, by using well–established methodology for design of applications, software tools for their development, and testing methodology and environments for their debugging and testing [10].

Although both approaches give some degree of security and protection, current condition in open networks indicates that in basic none of the current approaches is effective and does not produce secure, reliable and protected network applications and cloud environments. This means that current, basically single point–solutions and approaches, reactive to emerging difficulties, have limited scope and effectiveness. This means that the two current approaches, one based on solving individual problems ("point–solutions") in a reactive mode and the other based on conceptual methodology for design and development of secure software, so far have not produced effective results and are not capable to create the ultimate solution – secure and reliable network environment and its applications. So, in the current condition, new approach and new thinking towards inventing strongly and guaranteed secure network environments and applications are needed [8-10].

How cloud computing influences on Microsoft products:

Windows Azure platform (Microsoft, 2011c) provides friendly interfaces to deal with the heart of PaaS in progressive and deploying differently .NET applications start from very simple Hello World applications to distributed relational databases. One of Windows Azure's services is SQL Azure, which deliver DB services from building the DB to deploy it and scale it through Microsoft data centers. SQL Azure consists of relational database services, such as reporting, querying, and data synchronizing. Windows Azure present several features including computing resources, storage, database, Virtual Machines (VMs), access control, Content Delivery Network (CDN), caching, virtual network, service bus, business intelligence, and market place. Windows Azure was built to help developers progress in their application, especially for the developers who build remote data center applications by providing different tools. Windows Azure delivers a platform service which includes operating systems, a set of developer tools, and different levels on network controls to develop, host, scale, and manage developed applications on web and non-web environments (Jadhav et al., 2010). So, students will focus on

Tele:
E-mail addresses: mahdidarbandi@hotmail.com

developing the assignment without any pre-configurations for specific software or hardware [5-10].

Furthermore, the Windows Azure platform offers prebuilt sub-programs which often represent reuse functionalities to save the developers' time and let them focus more on their projects. By using Windows Azure, the companies and universities do not need special instruments or infrastructure; all they need is Internet connection. Windows Azure also has more solution which provides a database solution with a user friendly interface. Building a database by Windows Azure is started with invention of a subscription followed by creating the server with the access levels and their passwords. So the databases are created using a .NET framework after connection of the .NET to the created server [1-10].

Adaptively Reduced-Order Extended Kalman Filter:

The reduced-order extended Kalman (ROEK) filter has been introduced by Cane et al. (1996) as a means to decrease the cost of the extended Kalman filter [11 - 14]. It essentially includes in projecting the dynamic of the model onto a low dimensional subspace obtained via an empirical orthogonal functions (EOF) analysis. However, the choice of the dimension of the reduced state space (or the amount of EOFs to be retained) remains a delicate question. In fact, Cane et al. (1996) have been enabled to explain the fact that intensifying the number of EOFs does not improve, and even sometimes worsen, the performance of the ROEK filter [11 - 15]. We guess that it is due to the optimal character of the EOF analysis which is optimal in a time-mean sense only. In this view, we develop a simple effective adaptive scheme to tune, according to the model mode, the dimension of the reduced state space, which would be therefore variable in time.In the ROEK filter, the dimension of the reduced state space (or the amount of retained EOFs) was chosen according to the variability (or inertia) explained by the first few EOFs and also to keep the cost of the filter reasonable [11]. Cane et al. (1996) predicted that the filter would perform better as the number of retained EOFs increase, since the reduced state space generated by the EOFs represents better, in some sense, the variability of the model. However, their numerical experiments reveal a surprising feature: intensifying the number of EOFs does not improve, and even sometimes worsens, the performance of the filter. The same event has also been observed in our numerical experiments. A plausible explanation is that the optimality characteristic of the EOF analysis is true in a time-mean sense only as such an analysis is relying on a long historical run [11]. The last observation motivates us to progress a simple adaptive plan to tune the dimension of the reduced state space. The opinion includes simply in fixing a number of EOFs effective to represent the variability of the system in the stable periods and then add some new EOFs when model instabilities appear in order to represent more completely the local structures of the model. A similar algorithm has been already successfully implemented by Hoteit et al. (2002) to tune the "forgetting factor" and the evolution of the correction basis, for the singular evaluative extended Kalman (SEEK) filter and its variants [11].

The ROEK filter:

We shall adopt the symbolization proposed by Ide et al. (1995). Consider a physical system introduced by [11]:

$$X^t(t_k) = M(t_k, t_{k-1})X^t(t_{k-1}) + \eta_k$$

Where $X^t(t_k)$ introduce the vector representing the true state at time t, $M(s,t)$ is an operator denoting the system transition from time s to time t and η is the system noise vector. At each step t_k, one observes $Y_k^o = H_k X^t(t_k) + \varepsilon_k$

Where H_k denotes observational operator and ε_k is the observational noise [11 - 15]. The noises η_k and ε_k are expected to be independent random vectors with mean zero and covariance matrices Q_k and R_k, respectively.In the linear case, this problem has been entirely solved via the well-known Kalman filter. This filter owns an attractive feature of being recursive. Computation is done on-line once new observations are available. In the nonlinear state, one often linearizes the model around the current estimated state vector, which results to the so called extended Kalman EK filter (see for example Ghil and Manalotte-Rizzoli (1995) for a review) [11 - 15]. Apart from initialization, this filter progressed as succession of predicting and correction steps. Assuming that at a time t_{k-1}, one already has an estimate of the system state, often denotes to as the analysis state vector $X^a(t_{k-1})$, with some analysis error covariance matrix $P^a(t_{k-1})$. The EK filter lets the construction of the next $X^a(t)$ by correcting the predict $X^f(t_k)$ (which is the output of the model starting from $X^a(t_{k-1})$) using the new observation [11]. It also delivers the calculation of the new analysis error covariance matrix $P^a(t_k)$, to reflect the propagation of error from the analysis to the forecasting case (which is introduced by an error covariance matrix $P^f(t_k)$) and the reduction of error achieved by the correction step. The reader can consult Jazwinski (1970) for more information.For simplicity we accept that the reduction operator S is orthogonal so that its pseudo-inverse is equal to S^T. The full case vector X^t is then related to the reduced state vector X_r^t by [11]:

$$X^t = SX_r^t.$$

If this supposition is used in the EK filter, one obtains the equations of the ROEK filter which operates in two stages apart from an initialization stage as the EK filter (see Fukumori and Malanotte-Rizzoli (1995) for more details) [11 - 15].

0- Initialization stage: We option here to an objective analysis, based on the first observation Y_0^o : we take as the initial analysis state vector

$$X^a(t_0) = \overline{X} + SP_r^a(t_0)S^T \mathrm{H}_0^T R_0^{-1}[Y_0^o - H_0\overline{X}]$$

Where:

$$P_r^a(t_0) = [S^T \mathrm{H}_0^T R_0^{-1} \mathrm{H}_0 S]^{-1},$$

$\overline{X}$ is the mean of a sequence of state vectors and H_0 is the gradient of H_0 evaluated at $\overline{X}$. The original analysis error covariance matrix may be taken as [11 - 13]:

$$P^a(t_0) = SP_r^a(t_0)S^T.$$

Note that we have used the first observation for initialization; the procedure actually starts with the next observation [11 - 15].

Forecast stage: One applies the model to calculate the forecast state as:

$$X^f(t_k) = M(t_k, t_{k-1}) X^a(t_{k-1})$$

And assume the covariance matrix of the forecast error in the reduced space $P_r^f(t_k)$ (of dimension $r \times r$) to evolve according to [11]:

$P_r^f(t_k) = [S^T \mathrm{M}(t_k, t_{k-1}) S] P_r^a(t_{k-1}) [S^T \mathrm{M}(t_k, t_{k-1}) S]^T + S^T Q_k S$, Where $\mathrm{M}(t_k, t_{k-1})$ is the gradient of $M(t_k, t_{k-1})$ assessedat $X^a(t_{k-1})$. The predict error covariance matrix is then equal to $P^f(t_k) = S P_r^f(t_k) S^T$.

Correction stage: The correction of the predict state is done according to the formula

$$X^a(t_k) = X^f(t_k) + G_k [Y_k^o - H_k X^f(t_k)]$$

Where G_k is acquired by [11 - 15]:

$$G_k = S P_r^a(t_k) S^T \mathrm{H}_k^T R_k^{-1}$$

H_k is the gradient of H_k assessed at $X^f(t_k)$ and the covariance matrix of the analysis error covariance matrix $P_r^a(t_k)$ is updated from the equation:

$$[P_r^a(t_k)]^{-1} = [P_r^f(t_k)]^{-1} + S^T \mathrm{H}_k^T R_k^{-1} \mathrm{H}_k S.$$

The analysis error covariance matrix is then calculated by [11]:

$$P^a(t_k) = S P_r^a(t_k) S^T.$$

Here, in formula, the representativeness error which should introduce the information that has not been explained by the reduced space defined by S (this error can be conveniently "inserted" into the model error (Cane et al. 1996)) has been neglected. Although, following Pham et al. (1997), we introduce instead the use of a forgetting factor to limit the propagation of this error with time. This algorithm has been adopted as a way to sidestep the difficulty to correctly specify the representativeness error. The formulas for the ROEK filter algorithm with the forgetting factor remain unchanged; expect that the update equation of the analysis error covariance matrix in the reduced space is replaced by [11- 14]:

$$[P_r^a(t_k)]^{-1} = \rho [P_r^f(t_k)]^{-1} + S^T \mathrm{H}_k^T R_k^{-1} \mathrm{H}_k S.$$

Studying the cost of this filter, it mostly comes from the calculation of the evolution equation of the forecast error in the reduced state space. It thus relies on the dimension of the reduced space r because the numerical calculation of $\mathrm{M}(t_k, t_{k-1}) S$ requires $(r+1)$ integrations of the tangent linear model. A reasonably decreased choice of r is then imperative for realistic applications [11 - 12].

The performance of the ROEK filter highly relies on the representativeness of the reduction operator S. A good choice of S should lead to a large decrease of the dimension of the system and a reduced state space which well represents the variability of the model. Different cases of the operator S have been introduced in the literature such as the use of a coarse resolution (Fukumori 1995), the most dominant singular modes of the tangent linear model and the most dominant Eigenmodes of the analysis error covariance matrix (Cohn and Tolding 1996), etc., which are supported by more or less simplifying assumptions on the dynamics and the characteristic of the model (see De Mey (1997) for a review) [11 - 14].

With the same goal in view, Cane et al. (1996) have adopted a different approach: using empirical orthogonal functions (EOF) analysis, they decreased the state space for the forecast covariance updated to a small set of basic functions, called EOFs, which nonetheless represented all the significant structures that were predicted by the model. More than the implementation cost details, the philosophy of order reduction of Cane et al. (1996) relies on the fact that since one cannot precisely compute the ``true" error covariance matrix, it is useless to try to specify its full description. In numerical applications, this process was shown to lead to a substantial saving without any loss of accuracy compared to the full EK filter (Cane et al. 1996) [11 - 15].

EOF analysis:

This analysis goals at providing a representation as accurately as possible of a sample of state vectors X_1, K, X_N in $\Re^n$ in a low-dimension (denoted r) subspace. For a vector X, let $\tilde{X}$ introduce its orthogonal projection onto a subspace of dimension r spanned by an M-orthogonal basis $S = \{\phi_k\}_{k=1,\mathrm{K},r}$, M being some metric (to be chosen) in the state space, and the constant function [11]:

$$\tilde{X} = \bar{X} + \sum_{k=1}^{r} c_k \phi_k = \bar{X} + S S^T \mathrm{M}(X - \bar{X})$$

Where $\bar{X}$ is the mean of X_1, K, X_N and $c_k = < X - \bar{X}, \phi_k >_{\mathrm{M}} = \phi_k^T \mathrm{M}(X - \bar{X})$. The EOF analysis then includes of minimizing the mean square projection error [11]:

$$e^2 = \frac{1}{N} \sum_{i=1}^{N} \left\| X_i - \tilde{X}_i \right\|_{\mathrm{M}}^2.$$

With respect to all choices of the principle. Here the introduction of a metric M is needed in the case where the state variables are not homogeneous to describe a distance between state vectors independent from unit of measure.

The solution to the above minimization problem is calculated by the first r normalized Eigen vectors $\phi_1, \mathrm{K}, \phi_r$ of the sample covariance matrix P of X_1, K, X_N, namely [11]:

$P = \frac{1}{N} \Omega \Omega^T$ With $\Omega = [X_1 - \bar{X}, \mathrm{K}, X_N - \bar{X}]$.

Relative to the metric M, the eigenvectors being ranked in reducing order of their eigenvalues $\lambda_1, \mathrm{K}, \lambda_r$. With respect to the choice of r, it has been shown that the fraction of variance (or inertia) explained by the first r EOFs is given by [11]:

$$\mathrm{I}(\phi_1, \mathrm{K}, \phi_r) = \frac{\sum_{k=1}^{r} \lambda_k}{\sum_{k=1}^{n} \lambda_k}$$

And thus can be used as a guide for choosing r (this fraction should be close to 1).

In our case, cloud computing platforms, we are interested in representing the variability of the state model around its mean. For this aim, we consider a long historical sequence of model states X_1, K, X_N which can be extracted from a model run. Thus, the matrix P contains the bulk of information on the system variability when N is sufficiently large [11 - 15].

Adaptive tuning of the dimension of the reduced space:

As introduced in the above section, the dimension r of the reduced state space (or the number of EOFs to be retained) was chosen according to the value of the inertia I, providing that the cost of this filter remains reasonable since this cost highly depends on this number. Cane et al. (1996) have observed in their numerical experiments that increasing the number of EOFs does not improve, and even sometimes worsen, the performance of the filter. The same event has been also observed in our experiments. For a reasonable explanation, the optimality property of the EOF analysis is only optimal (at best) in a time-mean sense. In fact, the EOFs analysis is done over a long time period composed of periods in which the system evolves stably and periods in which it evolves unstably. Local perturbations often arise in the concluding period and not in the former, and they are represented by the last EOFs corresponding to the last Eigen values (Cane et al. 1996; Hoteit et al. 2001). Using more EOFs when the system is in a stable state would introduce spurious information which can reduce the performance of the filter. On the other hand, using only a few EOFs would not be sufficient to represent all the local structures of the system during unstable periods. The filter will then remove corrections in these structures and the error may grow. So to achieve better performance of the ROEK filter, our opinion is quite simple and consists in fixing a number of EOFs suitable to the stable period and then adds some EOFs in the unstable periods to represent more completely the model local structures [11]. In other words, the dimension r of the decreased state space will be given one of two values r_1 and r_2 ($r_1 < r_2$) according to the model state [11 - 14].

Such an adaptive plan can be easily implemented in the ROEK filter. In fact, if one introduced by S_1 and S_2 the basis containing the first r_1 and r_2 EOFs, respectively, the algorithm of the ROEK filter is taken to be the same as described previously, using S_1 as a reduction operator when the model is stable and S_2 when model instabilities appear. Although in the transition state: "stable to unstable" and vice-versa, one has to change the above equation. Especially, it is replaced by [11]:

$$P_r^f(t_k) = [S_2^T \mathrm{M}(t_k, t_{k-1}) S_1] P_r^a(t_{k-1}) [S_2^T \mathrm{M}(t_k, t_{k-1}) S_1]^T + S_2^T Q_k S_2$$

Each step the model goes from a stable to an unstable period (i.e. S_2 is used instead of S_1), denotes that the reduced forecast error $e_r^f(t_k) = M(t_k, t_{k-1}) S_1 e_r^a(t_{k-1})$ (where $e_r^a(t_{k-1})$ is the decreased analysis error) is projected onto the subspace generated by S_2, and by:

$$P_r^f(t_k) = [S_1^T \mathrm{M}(t_k, t_{k-1}) S_2] P_r^a(t_{k-1}) [S_1^T \mathrm{M}(t_k, t_{k-1}) S_2]^T + S_1^T Q_k S_1$$

When model instabilities vanish (S_1 is used instead of S_2) [11].

A similar algorithm has been already adopted by Hoteit et al. (2002) and Hoteit et al. (2001) to adapt the forgetting factor and the evolution of the correction basis for the SEEK filter and its variants. To notice the model unstable periods, they suggest tracking the filter's behavior by computing an instantaneous average and a long term average of the prediction error variance, denoted by s_k and l_k respectively. So, if $cs_k \le l_k$ (c is a tuning constant), they assumed steady states have been achieved and considered that the model is in a stable period. In this step we retain $r = r_1$ EOFs. Otherwise, if $cs_k > l_k$, this is asign that the model may be in an unstable period since this would degrade the filter short-term performance (the long-term performance is weakly affected because it is averaged over a long duration). We must then intensify the number of EOFs and thus take $r = r_2$ [11 - 14].

Estimates of s_k and l_k are recursively calculated as in Hoteit et al. (2002):

$$s_k = \alpha s_{k-1} + (1-\alpha) \left\| Y_k^o - H_k X^f(t_k) \right\|^2$$

$$l_k = \beta l_{k-1} + (1-\beta) \left\| Y_k^o - H_k X^f(t_k) \right\|^2$$

Where α and β are constants chosen such as $\beta \le 1$ and $\alpha < \beta$.

Hoteit et al. (2002) have used the same algorithm to tune the value of the forgetting factor by giving it one of two values according to the model mode: $\rho_1 \le 1$ when the model is stable and $\rho_2 < \rho_1$ when model unstable states appear. One can then use this adaptive plan together with the one on the number of retained EOFs [11 - 15].

Conclusion:

Contents of this paper can organized in three sections. In the first section of this paper, authors mention about all different aspects of cloud computing. They tell about applications and benefits of such network and explain all different aspects of this technology. In the second section, authors discuss about Microsoft new products – especially Microsoft Azure Operating System – and future decisions of this company based on cloud computing. In the third section of this paper, authors reveal new novel filter, which named as reduced-order extended Kalman filter, the authors claim that, if we use this filter in equipments and instruments of cloud service providers we can estimate and predict about all involved factors in controlling of such network.

References:

[1] David C. Wyld; "the cloudy future of government IT: cloud computing and the public sector around the world", IJWesT, Vol. 1, Num. 1, Jan. 2010.

[2] Jean-Daniel Cryans, Alain April, Alain Abran; "criteria to compare cloud computing with current database technology, R. Dumke et al. (Eds.): IWSM / MetriKon / Mensura 2008, LNCS 5338, pp. 114-126, 2008.

[3] Anil Madhavapeddy, Richard Mortier, Jon Crowcroft, Steven Hand; "multiscale not multicore: efficient heterogeneous cloud computing", published by the British Informatics Society Ltd. Proceedings of ACM-BCS Visions of Computer Science 2010.

[4] Harold C. Lim, Shivnath Babu, Jeffrey S. Chase, Sujay S. Parekh; "automated control in cloud computing: challenges and opportunities", ACDC'09, June 19, Barcelona, Spain.

[5] N. Sainath, S. Muralikrishna, P.V.S. Srinivas; "a framework of cloud computing in the real world"; Advances in Computational Sciences and Technology, ISSN 0973-6107, Vol. 3, Num. 2, (2010), pp. 175-190.

[6] Kyle Chard, Simon Caton, Omer Rana, Kris Bubendorfer; "social cloud: cloud computing in social networks"

[7] G. Bruce Berriman, Eva Deelman, Paul Groth, Gideon Juve; "the application of cloud computing to the creation of image mosaics and management of their provenance",

[8] Roy Campbell, Indranil Gupta, Michael Heath, Steven Y. Ko, Michael Kozuch, Marcel Kunze, Thomas Kwan, Kevin Lai, Hing Yan Lee, Martha Lyons, Dejan Milojicic, David O'Hallaron, Yeng Chai Soh; "open cirrus TM cloud computing testbed: federated data centers for open source systems and services research"

[9] Rajkumar Buyya, Chee Shin Yeo, Srikumar Venugopal, James Broberg, Ivona Brandic; "cloud computing and Emerging IT platforms: Vision, Hype, and Reality for delivering computing as the 5th utility"

[10] Lamia Youseff, Maria Butrico, Dilma Da Silva; "toward a unified ontology of cloud computing"

[11] Daniel A. Menasce, Paul Ngo; "understanding cloud computing: experimentation and capacity planning"; Proc. 2009, Computer Measurement Group Conf. Dallas, TX. Dec. 2009.

[12] Won Kim; "cloud computing: today and tomorrow"; JOT, Vol. 8, No. 1, Jan-Feb 2009.

[13] Richard Chow, philippe Golle, Markus Jakobsson, Elaine Shi, Jessica Staddon, Ryusuke Masuoka, Jesus Molina; "controlling data in the cloud: outsourcing computation without outsourcing control"; CCSW'09, Nov. 13, 2009, Chicago, Illinois, USA.

[14] Bo Peng, Bin Cui, Xiaoming Li; "implementation issues of a cloud computing platform"; Bulletin of the IEEE computer society technical committee on data engineering.

[15] Daniel Nurmi, Rich Wolski, Chris Grzegorczyk, Graziano Obertelli, Sunil Soman, Lamia Youseff, Dmitrii Zagorodnov; "the eucalyptus open-source cloud computing system